III

Learning Local Descriptive Models from Tabular Data

Overview

The learning of flexible and robust local descriptive models, the building blocks for all subsequent contributions, is the central requirement of this work. This book tackles the task of learning local descriptive models from tabular data. These principles are then further extended to learn local descriptive models from structured data (*Book IV*) and combined with new principles to guarantee the statistical significance of these models (*Book V*). As a result, these local descriptive models provide the necessary basis to learn associative classifiers and study their behavior (*Book VI*). As such, in order to understand how the number, positioning, size, coherence and quality of the selected regions affect the performance of both descriptive and classification models: we need to guarantee the *flexibility* and *robustness* of the target learners. Furthermore, *optimality* guarantees need to be present for a reliable quantification of the impact of the selected regions in the performance of descriptive and classification models, as well as *scalability* guarantees to enable the learning from high-dimensional tabular data.

In this context, the biclustering task, aiming to discover subsets of observations exhibiting a coherent pattern over a subset of features (Def.I-1.10), is by definition able to achieve these qualities. In fact, biomedical and social data are characterized by the presence of local regions, whose discovery has been largely studied and motivated in the context of biclustering [310]. However, due to the complexity of the biclustering task, most of the existing biclustering algorithms place restrictions on the coherency, quality and structure of biclusters (preventing the recovery of all relevant regions) and/or rely on greedy or stochastic approaches (associated with suboptimal solutions) [311]. To address this problem, this book proposes a new class of biclustering models. As such, to guarantee the flexibility, optimality, robustness and scalability of this learning task, the following four major requirements need to be satisfied:

- flexible structures of biclustering with guarantees of optimality **[R2.1]**;
- biclusters with flexible (yet meaningful) coherency **[R2.2]**;
- biclusters robust to different types and amount of noise **[R2.3]**;
- scalable searches in flexible settings (loose constraints) **[R2.4]**.

Furthermore, additional requirements are placed to learn local models from network and tabular data [**R2.5**], and to effectively guide the learning in the presence of background knowledge and user expectations [**R2.6**]. These six requirements are depicted in Figure 1 and further decomposed in the provided tables throughout this section.

A new class of biclustering approaches based on principles from pattern mining [310, 578, 501], referred in this work as pattern-based biclustering, is well-positioned to answer the introduced requirements as it is able to exhaustively (yet efficiently) discover flexible structures of biclusters. However, state-of-the-art contributions are dispersed and the potential of their integration remains unclear. *Chapter 1* provides a structured view on pattern-based biclustering. For this purpose, we survey its potentialities and limitations, and make available a set of principles for a guided definition of new pattern-based biclustering approaches that are able to combine existing contributions as well as accommodate new contributions.

Chapter 2 proposes new principles to guarantee the robustness of biclustering solutions to discretization procedures, varying forms and amount of noise, and arbitrary high-levels of missing values. These principles are integrated within a new pattern-based biclustering approach, referred as BicPAM (Biclustering based on PAttern Mining), to allow the discovery of biclusters with robust yet (possibly) parameterizable quality.

Chapter 3 extends the constant assumption of pattern-based biclustering towards more flexible coherencies, including additive and multiplicative coherencies in the presence or absence of symmetries. In this context, we first

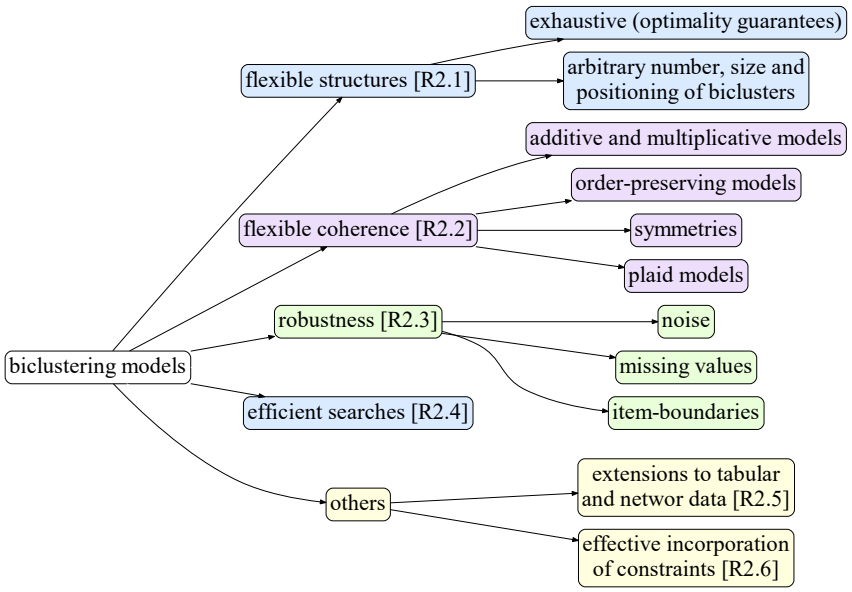

Figure 1: Requirements associated with the target task of learning flexible biclustering models.

motivate their need across biomedical and social data and provide principles for their exhaustive discovery that surpasses the limitations of existing searches. Second, we extend the biclustering task towards tabular data with non-identically distributed features.

Chapter 4 enhances the previous searches with principles to model biclustering solutions with a plaid model. The plaid model considers a cumulative composition of contributions in the areas where the biclusters overlap, thus being required to accommodate meaningful interactions between biclusters in biological and social domains. We propose BiP (Biclustering using Plaid models), an extension of BicPAM for recovering excluded regions due to unaccounted plaid effects. To address prominent restrictions associated with the plaid model, we further propose meaningful relaxations to allow a noise-tolerant composition of contributions and integrate them in BiP.

Chapter 5 tackles the efficiency bottlenecks of pattern-based biclustering searches. First, a new search based on annotated pattern-growth trees is proposed to avoid the use of expensive data structures and procedures required to maintain the sets of observations and features per region of interest. Second, we introduce principles to foster the scalability of the learning methods for very large data while still providing optimality guarantees, including searches in distributed and partitioned data settings or targeting biclusters derived from approximative patterns.

Chapter 6 explores a last form of coherency commonly observed in biomedical and social data domains: the order-preserving coherency. The values of the observations of an order-preserving bicluster induce the same linear ordering across features. This chapter tackles the limitations of existing order-preserving approaches, which either suffer from scalability or flexibility issues and are not able to discover biclusters with symmetries and parameterizable noise-tolerance. For this purpose, we first propose new searches able to seize efficiency gains from the item-indexable properties associated with the order-preserving biclustering task. Second, these searches are enhanced with new principles to guarantee robustness to noise and to accommodate symmetries.

The previous chapters introduce scalable searches for pattern-based biclustering under the assumption that the postprocessing step is accomplished efficiently. However, in the presence of voluminous biclustering solutions, the cost of computing similarities between all pairs of biclusters or testing the inclusion/removal of observations and features per bicluster can be highly expensive. *Chapter 7* provides principles to guarantee the efficiency of postprocessing procedures.

Chapter 8 extends the previous contributions towards network data. We propose BicNET (Biclustering NETworks), an algorithm to discover non-trivial yet coherent modules in weighted graphs with heightened efficiency. First, we motivate the relevance of discovering network modules given by biclusters with flexible coherencies and

tolerance to missing and noisy interactions in order to guarantee a focus on meaningful yet non-necessarily dense modules. Second, we adapt the underlying data structures and searches to seize high efficiency gains from the inherent structural sparsity of network data.

Chapter 9 combines all previous contributions. In this context, we provide the integrative algorithmic solution, analyze its computational complexity, and describe how its behavior can be either dynamically or parametrically customized based on the properties of the input data.

Chapter 10 extends pattern-based biclustering to effectively seize efficiency gains in the presence of constraints. In particular, we illustrate how constraints with succinct, (anti-)monotone and convertible properties can be derived from knowledge repositories and user expectations, and discuss and how they can be effectively used to prune the search space and guarantee a focus on regions of interest.

Index of Requirements and Contributions

Tables 1-10 provide an enumeration of the proposed contributions throughout the chapters of this book.

Table 1: Major contributions for the exhaustive discovery of flexible structures of biclusters (*Chapter* III-1).

R2: Efficient learning of biclustering models with flexible structures, coherency and quality;
R2.1: Discovery of flexible structures of biclusters with optimality guarantees;
C2.1.1a: Formal mapping of biclustering as frequent itemset mining and association rule mining;
C2.1.1b: Principles for declarative biclustering based on constraint-based mining and formal concepts;
C2.1.1c: Formal mapping of biclusterings as structured pattern mining, including sequential pattern mining and graph mining;
C2.1.2: Comparison of pattern representations (with impact on structures), including simple, maximal and closed;
C2.1.3: Preprocessing: principles to dynamically select adequate normalization and discretization procedures, and to handle outliers;
C2.1.4: Survey of alternative (anti-)monotonic for learning directly from real-valued data (with impact on structures);
C2.1.5: Adequate postprocessing options to affect the properties of the target structures;
C2.1.6: Systemic (qualitative and quantitative) comparison of existing pattern-based biclustering algorithms;
C2.1.7: Structured view of pattern-based biclustering: guidelines for the development of new algorithms;
R2.2: Ability to learn biclustering models with varying coherency assumptions;
R2.3: Biclustering solutions robust to noise and parameterizable quality;
R2.4: Efficiency of biclustering searches (optimality guarantees);
R2.5: Extension of original contributions towards tabular and network data;
R2.6: Effective learning in the presence of background knowledge;
R2.7: Sound integration of contributions;

Table 2: Major contributions to guarantee the robustness of biclustering models (*Chapter* III-2).

R2.3: Biclustering solutions robust to noise and parameterizable quality;
R2.3.1: No exposure to the items-boundary problem (discretization drawback);
C2.3.1: Effective principles to handle item-boundaries based on multi-item assignments;
R2.3.2: Efficient discovery of robust biclusters with calibrated type and amount of noise;
C2.3.2.1: Noise-tolerance through the use of association rules with parameterizable confidence and interestingness;
C2.3.2.2: Calibrating coherency strength through mapping options (alphabet, normalization and discretization);
C2.3.2.3: Guaranteeing robustness through adequate closing options;
C2.3.2.3a: Principles for effective merging and filtering of biclusters;
C2.3.2.3b: Principles for effective extension and reduction of biclusters;
R2.3.3: Adequate handling of missing values;
C2.3.3a: Strict handling of missings: removal, replacement as dedicated symbol, and imputation;
C2.3.3b: Relaxed handling of missings: multi-item assignments based on imputed value;

Table 3: Contributions for learning additive, multiplicative and symmetric models (*Chapter* III-3).

R2.2: Ability to learn biclustering models with varying coherency assumptions;
R2.2.1: Learning additive and multiplicative models;
C2.2.1.1: Exhaustive discovery of additive models based on shifting factors per observation or feature;
C2.2.1.2: Exhaustive discovery of multiplicative models based on common multiples;
C2.2.1.3: Principles for pruning the search space and to accommodate noise relaxations;
R2.2.2: Learning plaid models;
R2.2.3: Learning order-preserving models;
R2.2.4: Learning coherent models in the presence symmetries;
C2.2.4a: Exhaustive accommodation symmetries over constant models and adequate pruning of the search space;
C2.2.4b: Extensions for testing symmetries within additive models;
R2.5: Extension of original contributions towards new data contexts;
R2.5.1: Learning from network data;
R2.5.2: Learning from tabular data;
C2.5.2.1: Adequate procedures to discretize and balance feature's domains for the application of pattern-based biclustering;
C2.5.2.2: Principles to address limitations associated imbalanced feature's domains;
C2.5.2.2: Method to find biclusters with mixtures of coherency assumptions;

Table 4: Major contributions for the effective learning of plaid models (*Chapter* III-4).

R2.2.2: Learning plaid models;
R2.2.2.1: Effective learning of complex plaid models;
C2.2.2.1.1: Principles to recover excluded areas due to plaid effects and to exclude noisy areas non-explained by plaid effects;
C2.2.2.1.2: Residue-based heuristic for learning complex plaid models (large sets of interacting biclusters with non-trivial overlaps);
C2.2.2.2.3: Incorporation of previous contributions to allow the discovery of plaid models with non-constant coherencies and not requiring an exhaustive coverage of all data elements;
C2.2.2.2.4: Systematization of possible interactions between biclusters across domains (e.g. is-part-of, depends-on);
R2.2.2.2: Addressing the restrictive exact additive match;
C2.2.4.2.1: Definition and efficient inclusion of meaningful relaxations: a) in-between and b) noise-tolerant matches;
C2.2.4.2.2: Efficient inclusion of advanced composition functions: a) weighted and b) scaling composition functions;

Table 5: Proposed contributions to guarantee the efficiency of pattern-based biclustering (*Chapter* III-5).

R2.4: Efficiency of biclustering searches (optimality guarantees);
R2.4.1: Searches less susceptible to current time-memory bottlenecks;
C2.4.1.1: Comparison of full-pattern mining searches for biclustering;
C2.4.1.2: Extended pattern tree (FP-tree) with transactions annotations (minimum overhead) to tackle costs associated with bitsets;
C2.4.1.3: New pattern-growth algorithm with efficient build and traversal of annotated FP-trees for efficient pattern-based biclustering on dense and high-dimensional data;
C2.4.1.4: Principles to customize search options from data size, dimensionality and regularities;
R2.4.2: Efficient sequential pattern mining in item-indexable data;
R2.4.3: Efficient postprocessing (merging, extension and reduction procedures);
R2.4.4: Scalability of pattern mining searches;
C2.4.4.1: Data partitioning principles with constraint-guided searches to preserve optimality;
C2.4.4.2: Parallelization principles and mining of approximative patterns (with optimality guarantees);

Table 6: Contributions for the robust learning of order-preserving models in the absence and presence of symmetries (*Chapter* III-6).

R2.2.3: Learning order-preserving models;
R2.2.3.1: Efficient and exhaustive discovery of order-preserving models;
C2.2.3.1.1: Pattern-based biclustering with sequential pattern mining over mapped sequential databases from tabular data;
C2.2.3.1.2: Formulation of the new task of SPM over item-indexable databases (sequential database from ordered tabular data);
C2.2.3.1.3: New SPM pattern-growth search based on efficient data-projections and compact data structures;
C2.2.3.1.4: Principles for pruning search space in the presence of user expectations;
R2.2.3.2: Flexible order-preserving models;
C2.2.3.2.1: Methods to mine both monotonically and strictly increasing orders;
C2.2.3.2.2: Methods for the exhaustive discovery of order-preserving models with symmetries;
C2.2.3.2.3: Parameterizable degree of co-occurrences versus precedences according to the target task and data properties;
R2.2.3.3: Robustness of order-preserving models;
C2.2.3.3.1: Inclusion of previous principles to handle varying levels of noise and missings;
C2.2.3.3.2: Compliance of order-preserving searches with preprocessing and postprocessing options;

Table 7: Proposed methods to guarantee the efficiency of merging, extension and reduction procedures (*Chapter* III-7).

R2.4.3: Efficient postprocessing options;
R2.4.3.1: Efficient merging procedures;
- **C2.4.3.1.1**: Mapping merging as the task of discovering maximal circuits in unweighted undirected graphs;
- **C2.4.3.1.2a**: Formulation of frequent itemset mining task with variable support (multi-support FIM);
- **C2.4.3.1.2b**: Mapping merging into multi-support FIM;
- **C2.4.3.1.2c**: Efficient methods for multi-support FIM and implications;
- **C2.4.3.1.3**: Anti-monotonic heuristics for efficient calculus of similarities (with short-circuiting verifications);
- **C2.4.3.1.4**: Pushing merging procedures to the mining step (similarities using operations over tree structures);

R2.4.3.1: Efficient extension and reduction procedures;
- **C2.4.3.1a**: Pattern-based biclustering with varying support and pattern length thresholds to guide extensions and reductions;
- **C2.4.3.1b**: Constraint-based searches over specific regions of interest to guarantee heightened postprocessing efficiency;

Table 8: Contributions for the efficient and robust biclustering of network data to find non-trivial (yet coherent) modules (*Chapter* III-8).

R2.5.1: Learning from network data;
R2.5.1.1: Efficiency of biclustering methods for large-scale networks;
- **C2.5.1.1.1**: Extension of pattern-based biclustering to learn from sparse data;
- **C2.5.1.1.2**: Efficient data structures and principles for adequate space exploration;

R2.5.1.2: Discovery of network modules with non-dense yet meaningful coherency;
- **C2.5.1.2.1**: Incorporation of previous principles to discover modules/biclusters with flexible coherency;
- **C2.5.1.2.1a**: Constant and symmetric models for finding modules with non-necessarily high (yet coherent) interactions;
- **C2.5.1.2.1b**: Plaid models to accommodate weight variations associated with network topology (hubs, and between- and within-pathway interactions;
- **C2.5.1.2.1c**: Order-preserving models for discovering modules with coherent degree of influence between sets of nodes;
- **C2.5.1.2.2**: Extension of principles for different types of networks;
- **C2.5.1.2.2a**: Principles for biclustering homogeneous and heterogeneous networks;
- **C2.5.1.2.2b**: Principles for biclustering networks with quantitative (weights) and qualitative (labels) interactions;

R2.5.1.3: Discovery of robust network modules;
- **C2.5.1.3a**: Principles for biclustering modules robust to missing interactions (tackling fully-interconnectedness restriction);
- **C2.5.1.3b**: Principles for biclustering modules robust to noisy interactions;

Table 9: Consistent integration of previous contributions and exploration of their synergies (*Chapter* III-9).

R2.7: Sound integration of previous contributions;
- **C2.7.1a**: Exploration of efficiency gains from synergies associated with the integrative search for different coherency assumption;
- **C2.7.1b**: Efficiency gains from applying biclustering with varying levels of tolerance to noise, levels of coherency strength and from the combined application of preprocessing and postprocessing steps;
- **C2.7.2**: Consistent algorithmic basis and the analysis of its computational complexity;
- **C2.7.3**: Robust default and dynamically parameterized behavior based on the input data properties;
- **C2.7.4a**: Structured view of the parameters of pattern-based biclustering to customize the coherency, quality and structure of solutions;
- **R2.7.4b**: Declarative interface for the constraint-based definition of the desirable properties of biclustering solutions;
- **C2.7.4c**: Graphical interface for usable parameterization, display of results and soundness checks;
- **C2.7.4d**: Programmatic interface to flexibly control, extend and adapt the biclustering behavior;

R2.5.3: Effective learning from sparse data;
- **C2.5.3**: Principles for biclustering data with an arbitrary-high number of uninformative elements and/or missings;

Table 10: Contributions for biclustering in the presence of background knowledge (*Chapter* III-10).

R2.6: Effective learning in the presence of background knowledge;
R2.6.1: Effective incorporation of constraints with nice properties;
- **C2.6.1.1a**: Structured view on the relevance and properties of constraints with succinct, (anti-)monotonic and convertible properties for biclusters from frequent itemsets for matrix and network data;
- **C2.6.1.1b**: Extensibility of constraints for plaid models from association rules;
- **C2.6.1.1c**: Prefix-monotone constraints and regular expressions for order-preserving models from sequential patterns;
- **C2.6.1.2**: Comparison of efficiency gains from available principles for domain-driven pattern mining;
- **C2.6.1.3a**: Compliance of F2G searches for pattern-based biclustering with CFG principles;
- **C2.6.1.3b**: Compliance of F2G searches for with Bonsai and data-reduction principles;
- **C2.6.1.4**: Extension of IndexSpan searches for with prefix-monotone verifications during database projections;

R2.6.2: Guided instantiation of nice constraints;
- **C2.6.2.1**: Roadmap with illustrative sets of succinct, (anti-)monotonic and convertible across biomedical domains (expression data and biological networks) and social domains;
- **C2.6.2.2**: Formalism and procedures to interpret (parametric) constraints to customize the biclustering behavior with sharp usability;

R2.6.3: Effective learning in the presence of background knowledge;
- **R2.6.3**: Effective learning in the presence of knowledge-driven annotations;
- **C2.6.3**: Guided searches when rows and columns are annotated with labels extracted from knowledge bases and literature;

1

A Structured View on Pattern-based Biclustering

The clustering of data aims to group observations according to their overall values across features. However, in real data contexts, the correlation of a subset of observations (rows) is typically only significant and meaningful for a subset of features (columns) [588]. To address this problem, biclustering can be applied to find sub-matrices (biclusters), subsets of rows exhibiting a coherent pattern across subsets of columns (Def.I-1.10). Illustrating, given a real-valued matrix that captures the expression of a set of genes (rows) across a set of conditions (columns), a bicluster is useful to model a subset of genes participating in cellular processes or pathways of interest that may be only active in a subset of the conditions. Biclustering real-valued matrices is a critical task for a wide-set of biomedical and social contexts [429, 230, 202]. A detailed enumeration of high-dimensional data domains benefiting from local views given by biclustering models was provided in Table I-1.1.

As introduced, flexible biclustering models with optimality guarantees are required to study the relevance of modeling regions with varying properties when learning from high-dimensional data[1]. However, due to the complexity of the biclustering task, most of the existing algorithms: 1) are either based on greedy or stochastic approaches, producing sub-optimal solutions; and 2) commonly place restrictions on the allowed structure, coherency and quality of biclusters, compromising the flexibility of the models [429, 324]. Biclustering involves combinatorial optimization to select and group rows and columns and it is known to be a NP-hard problem (proven by mapping the biclustering task over binary matrices into the problem of finding maximal cliques in weighted bipartite graphs [518]). The problem complexity increases for non-binary data contexts and when elements are allowed to participate in more than one bicluster (non-exclusive structure) and in no bicluster at all (non-exhaustive structure). In this context, the few available exhaustive searches for biclustering place restrictions on the solution space that prevent the flexibility of the biclustering task, including the search for a fixed number of biclusters, non-overlapping structures, and biclusters with simplistic forms of coherency (e.g. given by differential-values or overall constant values) and intolerance to noise [429, 616]. In this context, this chapter aims to understand whether these restrictions can be minimized while still providing guarantees of optimality.

Interestingly, some attempts to perform biclustering based on pattern mining techniques [442, 578, 501], referred in this work as pattern-based biclustering, allow for exhaustive and flexible searches and show solid levels of efficiency [578, 501]. Since pattern mining research is driven by scalability requirements [290], its integration with biclustering defines a new promising direction. In particular, pattern-based biclustering allows the discovery of an arbitrary number, size and positioning of biclusters with flexible coherency and noise-tolerance, being a successful attempt to tackle the restrictions of peer algorithms.

Despite these potentialities, there is a general lack of research on pattern-based biclustering, being absent from recent surveys on biclustering [230, 202, 123, 588]. This may be explained by the fact that many of the existing contributions on pattern-based biclustering remain dispersed [57, 442, 578]. As such, there is still space for new approaches that benefit from the integration of principles provided in the context of existing pattern-based

[1] Recent findings from biomedical and social domains underline the importance of such models to gather unprecedented insights on cellular behavior [303, 578], on the native structure and modular organization of biological and social networks [57], and on high-order polymorphisms [207].

biclustering algorithms as well as from other fields of research. In this context, this chapter aims to fill such space. It provides four major contributions:

- motivates, formalizes and provides a qualitative assessment of the state-of-the-art algorithms for pattern-based biclustering;
- discusses how different patterns (including itemsets, association rules, sequences and formal concepts) can be adequately used to learn biclustering models with varying properties;
- offers a structured view on how to define, parameterize and extend pattern-based biclustering. In particular, we show how to coherently integrate the available (yet dispersed) contributions;
- further surveys pattern mining principles, as well as adequate preprocessing and postprocessing procedures, to guarantee the robustness, flexibility and scalability of pattern-based biclustering.

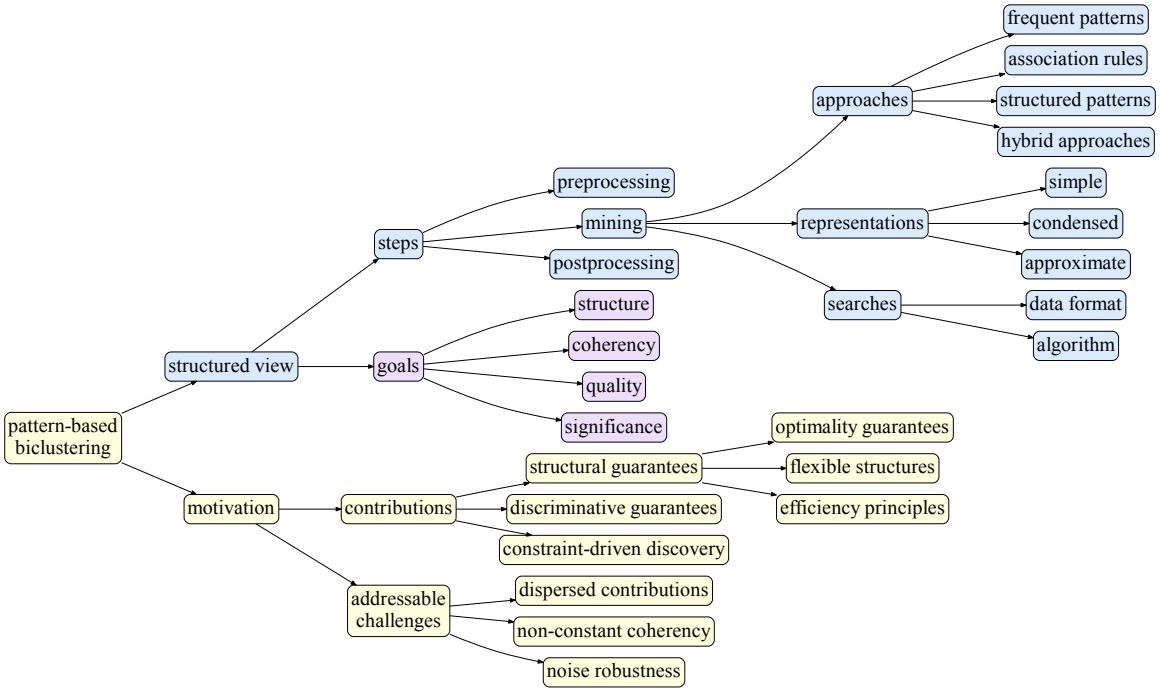

Figure 1.1: Pattern-based biclustering: pattern mining options, contributions and current challenges.

Figure 1.1 provides a graphical view on the covered contents in this chapter. This chapter is organized as follows. *Section 1.1* motivates the need for pattern-based biclustering. *Section 1.2* provides background on biclustering and pattern mining. *Section 1.3* surveys the contributions from existing pattern-based biclustering approaches. *Section 1.4* introduces a consistent set of principles to guide the definition of pattern-based biclustering approaches. In particular, *Sections 1.4.1-1.4.3* cover principles according to three major decision dimensions (mining, mapping and closing), and *Section 1.5* compares the behavior of state-of-the-artpatternM-based biclustering approaches and proposes a set of principles to address their current challenges. Finally, the implications of this work are synthesized.

1.1 On the Relevance of Pattern-based Biclustering

As previously discussed, the challenging combinatorial nature of the biclustering task led to the development of algorithms producing suboptimal models with undesirable restrictions on the allowed structure and homogeneity of biclusters. Interestingly, research in the complementary field of pattern mining (PM) research is associated with exhaustive searches driven by scalability requirements [290] that do not constrain the number and positioning of the discovered regions (patterns). Therefore, its integration with biclustering, referred as pattern-based biclustering, defines a new promising direction. Below, *Sections 1.1.1*, *1.1.2* and *1.1.3* respectively cover the inherent benefits, (addressable) challenges, and relevance across data domains of pattern-based biclustering.

1.1.1 Potentialities of Pattern-based Biclustering

Contributions of pattern-based approaches for biclustering include:

- efficient exhaustive searches. PM algorithms as-is allow for the efficient analysis of large matrices (over 10.000×400 elements). Additional PM principles can be used to foster scalability, including searches in distributed/partitioned data settings or targeting approximate patterns [290, 277];
- biclusters from patterns with parameterizable coherency strength (multiple ranges of values) [500, 509], contrasting with peer approaches to find biclusters with differential values or fixed coherency strength [616];
- flexible structures of biclusters: arbitrary positioning of biclusters without the need to fix the number of biclusters apriori [500, 578];
- inherent orientation to learn constant values on columns, already an improvement over approaches requiring constant values on both columns and rows [310, 303, 311];
- easy extension for labeled data using discriminative pattern mining or classification rules [208, 498].

1.1.2 Challenges of Pattern-based Biclustering

Pattern-based biclustering suffer from seven major drawbacks that need to be addressed to test the target hypothesis of this thesis. These drawbacks are the following:

- dispersed contributions, with the potential of their integration remaining unclear [**R2.1**];
- non-flexible coherence: only consider constant assumptions [**R2.2**];
- absence of principles to deal with noisy and missing values [**R2.3**];
- efficiency bottlenecks associated with the disclosure of patterns' features and observations [**R2.4**];
- non-easily extensible for sparse data and data with non-identically distributed features [**R2.5**];
- lack of principles to effectively incorporate constraints in the presence of background knowledge [**R2.6**];
- absence of tests to assess and guarantee the statistical significance of flexible biclustering models [**R4**].

1.1.3 Applicability

In biological domains, the discovery of biclusters has been largely applied over: 1) expression data to identify co-regulated genes, proteins and metabolites associated with specific functional processes and pathways [429, 230, 202, 310, 303], 2) biological networks to find modules of biological entities with coherent interaction (from matrices mapped from pairwise interactions) [53, 30, 145, 56, 473, 426], 3) structural genome variations to identify correlated groups of mutations (polymorphisms) and copy number variations [207, 324], 4) genome-wide data [662, 640] to find conserved subsequences (alignments), factor binding sites and mutations; and 5) translational [175, 138], chemical [415] and nutritional data [393]. In fact, biclustering can be applied over any dataset where sets of objects (such as molecules or patients) are analyzed against identically distributed features (such as distinct samples or samples on different time points) [310].

In medical domains, biclustering has been mainly applied over physiological data [108, 201, 211] to identify sets of (sliding) features and signal partitions with coherent values across responses, but also over clinical data [302, 122] to group patients with correlated clinical features. However, its applicability over clinical data has been limited since the collected features typically have non-identically distributed domains of values. To surpass this limitation, preprocessing options [309] or the selection of subsets of features with real-valued domains or identical distributions [310] have been proposed.

In social domains, biclustering has been applied over: 1) social networks [257] to either group individuals with correlated activity and intercommunication pattern or group contents based on the accessors actions; 2) financial trading [334] to identify indicators producing similar profitability for specific trading points (buy, hold and sell signals) in the stock market; 3) (e-)commerce data [35] to find hidden browsing patterns from correlated sets of

(web) users, visited (web) pages and operations; and 4) collaborative filtering data [159] to group users who share preferences and behaviorial patterns for a subset of available actions.

Biclustering has been additionally applied over text and unstructured data [38, 171] to identify correlated (web) contents (from matrices counting categories/words across text segments); multimedia data for processing and feature extraction [607, 266]; and over large data for size and/or dimensionality reduction [134, 690].

The enumerated advantages and (addressable) challenges of pattern-based biclustering in previous sections are critical to tackle this large set of applications. The relevance of the listed properties of pattern-based biclustering for these domains is synthesized in Table 1.1.

Property	*Benefit*
Exhaustive scalable searches	Delivery of optimality guarantees for large data such as data from clinical, molecular and social web domains.
Noise robustness	Handling of uncertainty relations observed in social networks [257] and stock markets [335]; artefacts in multivariate physiological data (such as electroencephalograms [205]), experimental errors in molecular arrays [295].
Handling of missing values	Adequate mining of incomplete and/or sparse matrices derived from biological networks, web social contexts, and healthcare data.
Flexible coherency	Constant models for non-differential (yet coherent) functional associations; additive and multiplicative factors to model the distinct responsiveness and experimental bias of biological molecules and physiological signals; symmetries to simultaneously capture activation and repression mechanisms and opposed (yet correlated) regularities associated with trading, tweeting, browsing and (e-)commerce activity; plaid models for overlapping regulatory influence in biological contexts and cumulative effects in social/biological networks [303, 310, 308].
Parameterizable level of coherency	Dynamic definition of the desirable coherency strength for an adequate multi-level analysis of matrices derived from expression data (optimum number of expression levels [442]), scored networks, collaborative filtering data (grading scale), and physiological signals (adequate resolution [201]).
Flexible structures	Possibility to model plaid effects (meaningful overlaps between groups of molecular units, physiological features, patients, web users and transactions) and find regions with varying size and configurations.
Annotated significance	Testing the statistical significance of biclustering solutions (guaranteeing that their coherence does not occur by chance) to further validate their use to support critical decisions, such as medical and financial decisions.
Constraint-driven searches	Discovery of non-trivial biclusters and ability to focus the search on specific biclusters of interest (e.g. specific regulatory behavior, high-order SNPs from genome-wide data, web users with a specific behavior, health records related with particular medical conditions, domain-guidance from background knowledge [207, 321]).
Biclustering-based classification	Support for classification tasks from matrices with a large number of uninformative elements (benefiting from local views), including computer-aided diagnosis, phenotype discrimination and user recommendations [122, 208].

Table 1.1: Benefits of pattern-based biclustering for pattern recognition.

1.2 Background

In this section, we briefly revise biclustering concepts, formalize concepts of pattern mining (*Section 1.2.1*) and describe the paradigmatic behavior of pattern-based biclustering (*Section 1.2.2*).

Biclustering. According to Def.I-1.10, given a matrix with n rows $\mathbf{X}=\{\mathbf{x}_1,..,\mathbf{x}_n\}$ in $\mathbb{R}^m$, and m columns $\mathbf{Y}=\{\mathbf{y}_1,..,\mathbf{y}_m\}$ in $\mathbb{R}^n$, and elements $a_{ij} \in \mathbb{R}$ relating row i and column j: 1) a *bicluster* $\mathbf{B} = (\mathbf{I}, \mathbf{J})$ is a (r, s)-space, a subset of rows and columns, where $r \leq n \wedge s \leq m$, $\mathbf{I} \subseteq \mathbf{X}$ and $\mathbf{J} \subseteq \mathbf{D}$; and 2) the **biclustering** task aims to identify a set of biclusters $\{\mathbf{B}_1,..,\mathbf{B}_p\}$ such that each bicluster $\mathbf{B}_k = (\mathbf{I}_k, \mathbf{J}_k)$ satisfies specific criteria of *homogeneity* and *significance*. An illustrative application of this task is discussed in *Basics I-1.8*.

The *homogeneity* criteria determines the structure, coherency and quality of biclustering solutions. The structure of the outputted biclusters is essentially defined by the number, size and positioning of biclusters. Flexible structures are characterized by an arbitrary-high set of possibly overlapping biclusters (see Fig.II-3.1). The *coherency* of a bicluster is defined by the observed correlation of values (coherency assumption) and by the allowed deviation over expectations (coherency strength). According to Defs.II-3.1 and II-3.2, a bicluster can follow coherency assumptions across its rows, columns or overall elements, where the values can follow constant, additive, multiplicative, symmetric, plaid and order-preserving models [429]. The *quality* of a set of biclusters is defined by the type and amount of accommodated noise. Large biclusters may be found by either fixing a loose coherency strength with a strict quality criteria (intolerance to noise) or by fixing a stricter coherency strength yet allowing a

loose quality criteria (tolerance to noise). Fig.1.2 illustrates biclustering models with varying structure, coherency and quality. Def.1.1 introduces the additional concepts of bicluster pattern and support.

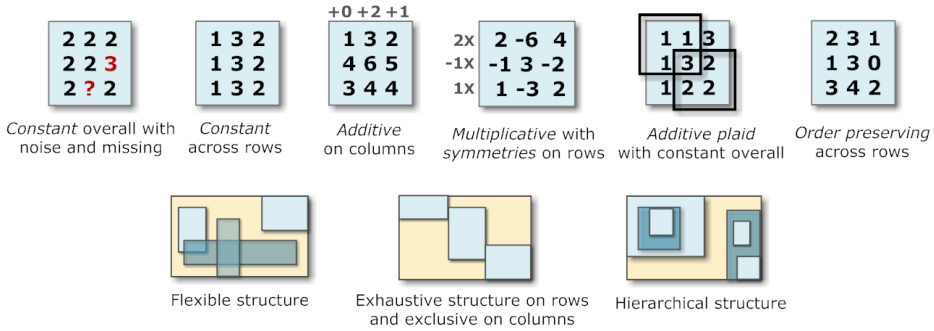

Figure 1.2: Biclustering solutions with varying coherency and quality.

Def. 1.1 The bicluster **pattern** φ_B is the ordered set of expected values in the absence of adjustments (γ=0)and noise (η_{ij}=0): φ_B=$\cup_{j=1}^{m}\{c_j\}$ for coherency across rows (or φ_B=$\cup_{i=1}^{n}\{c_i\}$ for column-based coherency). The bicluster **support** $\sup_B$ is, respectively, the number of rows $|\mathbf{I}|$ or columns $|\mathbf{J}|$ respecting φ_B.

1.2.1 Pattern Mining

Patterns are itemsets, rules, subsequences, or substructures that appear in a dataset with frequency no less than a user-specified threshold. The paradigmatic case of pattern-based biclustering is to rely on frequent itemset mining, the search for biclusters given by itemsets with support (number of rows) and pattern length (number of columns) above minimum thresholds. Association rule mining, sequential pattern mining and graph mining can be alternatively used to discover biclusters [442, 311, 415]. However, their soundness, relevance and applicability is yet poorly understood [310].

Def. 1.2 Let $\mathcal{L}$ be a finite set of items, and P be an itemset $P \subseteq \mathcal{L}$. A *transaction* $\mathbf{x}$ is an itemset with a non-fixed number of items $|P| \in \mathbb{N}$. A **transactional database** A over $\mathcal{L}$ is a finite set of transactions $\{\mathbf{x}_1, .., \mathbf{x}_n\}$, in other way, a multi-set of itemsets with the language $L_{\mathcal{L}} = 2^{\mathcal{L}} \backslash \emptyset$.

Def. 1.3 A transaction P contains P' if $P' \subseteq P$. The *coverage* Φ_P of an itemset P is the set of all transactions in $\mathbf{A}$ in which the itemset P occurs: $\Phi_P = \{\mathbf{x} \in \mathbf{A} \mid P \subseteq \mathbf{x}\}$. The *support* of an itemset P in $\mathbf{A}$, denoted sup_P, can either be absolute, being its coverage size $|\Phi_P|$, or a relative threshold given by $|\Phi_P|/|\mathbf{A}|$.

Def. 1.4 An **association rule** is defined as an implication of the form $P \Rightarrow P'$, where $P, P' \subseteq \mathcal{L}$ and $P \cap P' = \emptyset$. The left-hand side of the rule is named antecedent and the right-hand side consequent. Given a transactional database A: the *support* of a rule, $sup_{P \Rightarrow P'}$, is given by $sup(P \cup P')$, and the *confidence* of a rule, $conf_{P \Rightarrow P'}$, is given by $\frac{sup(P \cup P')}{sup(P)}$.

Confidence reveals the strength of the rule (the conditional probability that a transaction that contains the items in the antecedent also contains the items in consequent). To mine specific rules of interest, other correlation measures have been used to augment the support-confidence framework. Correlation measures including lift, conviction, chi-square, cosine and all-confidence [614]. Studies were also conducted on mining interesting or unexpected rules compared with user's prior knowledge [658].

Def. 1.5 Given a transactional database A and a minimum support and confidence thresholds, θ_1 and θ_2:

- **frequent itemset mining** (FIM) problem consists of computing the set $\{P \mid P \subseteq \mathcal{L}, sup_P \geq \theta\}$;
- **association rule mining** aims to compute $\{(P, P') \mid P \subseteq \mathcal{L}, P' \subseteq \mathcal{L}, sup_{P \Rightarrow P'} \geq \theta, conf_{P \Rightarrow P'} \geq \delta\}$.

A pattern is a frequent itemset or rule satisfying the given thresholds and any other placed constraints over D.

Basics 1.1 Illustrating pattern mining concepts

To illustrate the introduced concepts, consider the transactional database $\mathbf{A}_7$ provided in Table 1.2 with $|\mathcal{L}|$=12. The {b,f} itemset has coverage $\Phi_{\{b,f\}}=\{\mathbf{x}_2, \mathbf{x}_3\}$ and relative support $sup_{\{b,f\}}= |\{t_2, t_3\}|/6=0.(3)$. An illustrative rule in $\mathbf{A}_7$ is $R_1 : \{f, h\}\Rightarrow\{a\}$ with support sup_{R_1}=0.5 and confidence $conf_{R_1}$=0.75. For θ=4 support threshold, the FIM tasks returns $\{\{a\}, \{f\}, \{h\}, \{f, h\}\}$.

Table 1.2: Transactional database ($\mathbf{A}_7$)

$\mathbf{x}_1$	{b,e,g}
$\mathbf{x}_2$	{a,b,c,e,f,h}
$\mathbf{x}_3$	{a,b,d,f,h}
$\mathbf{x}_4$	{d,f,h}
$\mathbf{x}_5$	{a,f,h}
$\mathbf{x}_6$	{a,g}

Consider two itemsets P and P', where $P' \subseteq P$, and a predicate M. M is *monotonic* when $M(P) \Rightarrow M(P')$ and *anti-monotonic* when $\neg M(P') \Rightarrow \neg M(P)$. Pattern mining approaches rely on these properties to guarantee the efficiency of the searches: the support of P is bounded by the support of P' and, if P' is not frequent, then P is also not frequent. The (anti-)monotonic principle is the basis for existing searches that either rely on candidate-generation or pattern growth structures, and either assume horizontal or vertical data formats.

Since FIM proposal [7], multiple extensions have been proposed, including principles to enhance the scalability of pattern miners and to deliver condensed and approximate pattern representations [114, 290].

Pattern mining has been additionally applied over structured datasets, leading to contributions in different fields, including sequential pattern mining [424], graph mining [668] and cube computation [291].

1.2.2 Pattern-based Biclustering

While traditional biclustering approaches rely on flexible merit functions to guide the space exploration, pattern-based approaches require these functions to be defined in terms of support and, eventually, confidence or other interestingness metrics. The application of these functions within (anti-)monotonic searches enables an efficient (yet exhaustive) space search that produces an arbitrarily high number of biclusters within a flexible structure.

Def. 1.6 Let $\mathbf{A}$ be a discrete matrix with values in $\mathcal{L}$, possibly derived a real-valued matrix under a discretization function $\mathbb{R} \rightarrowtail\!\!\!\!\!\twoheadrightarrow \mathcal{L}$. A **discrete bicluster** is a sub-matrix $(I, J) \subseteq A$ with elements $a_{ij} \in \mathcal{L}$ defining a pattern. A discrete bicluster under a *constant* assumption can either follow: an overall orientation where $a_{ij} = k$ and $k \in \mathcal{L}$; a column-based orientation where a_{ij}=k_j and $k_j \in \mathcal{L}$; or a row-based orientation where $a_{ij} = k_i$ and $k_i \in \mathcal{L}$.

Def. 1.7 Given a matrix $\mathbf{A}$ whose elements are the concatenation of the observed values $a_{ij} \in \mathcal{L}$ with their column (or row) indexes. Let Ψ_P of an itemset P in A be its set of indexes and Υ_P be its original items in $\mathcal{L}$. The **pattern-based biclustering model** $\cup_k(I_k, J_k)$ can be derived from a set of frequent itemsets $\cup_k P_k$ by mapping (I_k, J_k)=(Φ_{P_k}, Ψ_{P_k}), to compose biclusters with coherency across rows, or (I_k, J_k)=(Ψ_{P_k}, Φ_{P_k}) for column-coherency, with pattern Υ_P.

Defs.1.6 and 1.7 formalize the simplest pattern-based biclustering model, where coherency assumption is constant, coherency strength δ is given by the number of items (alphabet length $|\mathcal{L}|$), and no noise is tolerated (perfect quality).

In this context, two classes of pattern-based biclustering approaches can be considered: *1)* a first class targeting discrete matrices by using as-is pattern miners, and *2)* a second class targeting numeric matrices by extending methods based on the introduced monotonic (or Apriori) property [7].

The first class of methods rely on an itemization step followed by the application of FIM under a low support threshold. The itemization step maps a real-valued or discrete matrix into an transactional database. For real-valued matrices, normalization and discretization procedures are applied. Then, the discrete value of each element is concatenated with its column index. Each transaction of the target transactional database corresponds to a row with these new values. FIM is then applied over this database to mine frequent patterns for composing biclusters with coherency across rows. The second class of methods relies on variants of the FIM task to learn frequent patterns directly from the real-valued matrix. In this class, the coherency strength is not implicitly defined by the number of items but explicitly fixed using the maximum allowed distance. Biclusters with coherency across columns can be mined using the transpose matrix. Finally, biclusters with coherent values overall can be discovered

by mining one item (or range of values) at a time. Figure 1.3 illustrates how to deliver these different types of biclusters from frequent patterns when considering the constant model.

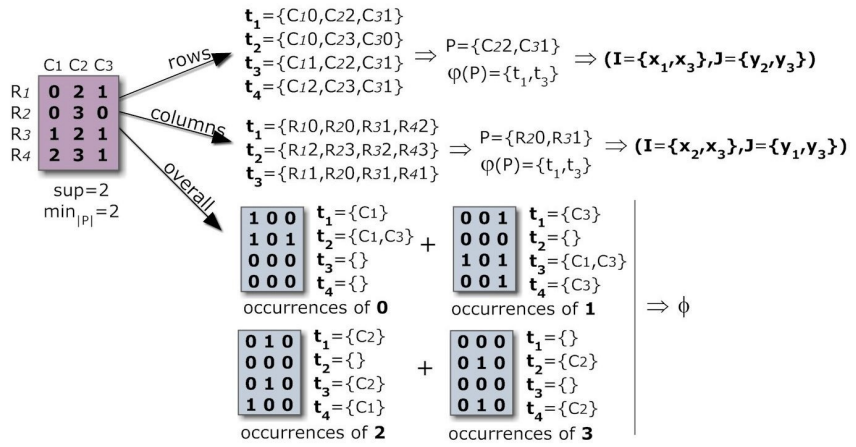

Figure 1.3: Mining biclusters with constant assumptions over itemset matrices.
To discover biclusters with constant values on rows, the input matrix needs to be itemized. Column identifiers are combined with the observed values, and FIM applied under a parameterizable support (θ=2$\wedge$ $|P|\geq$ 2). Constant values on columns can be mined using the transpose matrix. To find biclusters with constant values overall, each item needs to be separately mined. In each iteration, only the elements with the selected item are included in the transactions.

1.3 Related Work

Below we provide a concise survey on biclustering and pattern mining, and overview the state-of-the-art approaches for pattern-based biclustering (excluding the contributions originated in the context of this work).

Biclustering. Following the taxonomy proposed by Madeira and Oliveira [429], Table 1.3 synthesizes the state-of-the-art biclustering approaches according to their learning function and guarantees of optimality. The learning function is essentially defined by the considered merit function and how it is applied. Merit functions can be defined to locally maximize greedy iterative searches [690, 116, 477, 59, 706, 496], to combine row- and column-based clusters [247, 617, 109], to exploit matrices recursively [298], and to stochastically model the target solution [393, 584]. In exhaustive searches, which commonly place restrictions on the solution space, merit functions are the heuristics that guide the space exploration [616, 653]. These searches can either produce one bicluster at a time (which can be masked and the search restarted) [393, 584], a set of biclusters at a time [247, 617, 298] or all biclusters at a time [690, 477, 616, 116]. To preserve conciseness, we do not further expand this overview and redirect the reader to the already extensive number of recent surveys on biclustering [123, 588, 230, 429, 202] for

Domain	Optimality Guarantees
Divide-and-conquer approaches [298, 663, 709] explore the matrix recursively according to a global merit function. Although efficient, the structure of biclusters is restrictive and the initial assumptions can easily lead to the missing of relevant biclusters. *Greedy* iterative approaches [116, 168, 461, 678, 477, 59, 690, 370] rely on the selection, addition and removal of rows and columns until a local merit function is maximized;	*Local optima* (local searches dependent on initial assumptions and convergence behavior)
Two-way clustering approaches use similarity/merit functions to produce clusters on both dimensions of the data matrix. An additional merit function is adopted to combine the clusters to produce subgroups of rows and columns [247, 617, 234, 109]; *Stochastic* approaches [324, 449, 573, 527, 581, 72, 584] assume that data follows a multivariate distribution and learn a parametric model that maximizes a likelihood/merit function under a specific convergence scheme. The model's parameters are used to derive biclusters.	*Distance-based guarantees* as learners rely on approximative views (clustering abstractions or generative models)
Ensemble methods [295] use a merit function to aggregate a large set of biclustering solutions from the iterative application of multiple biclustering approaches.	Dependent on the selected approaches
Exhaustive approaches rely on heuristics based on merit functions to guide the space exploration. The search space is commonly restricted by fixing the output number of biclusters and relying on constrained coherency assumptions [310, 311, 616, 653, 563].	*Global optima*

Table 1.3: Classes of biclustering approaches according to merit-guided searches and optima guarantees.

more details.

Pattern Mining. Similarly, we consider an in-depth overview of pattern mining methods to be out of the scope of this dissertation. Table 1.4 characterizes the three major search variants – Apriori, pattern-growth and vertical searches –, with a focus on state-of-the-art principles to guarantee an heightened effeciency of these searches. This categorization is not only applicable to searches on transactional databases, but also to searches on structured databases [290, 424, 668]. Illustrating, sequential pattern miners can also rely on Apriori, pattern-growth and vertical searches, and find closed and maximal (sequential) patterns [424].

Strategy	Principles	Optimizations [290, 106, 412]	Criticism
Apriori [7]	Monotonicity principle (an itemset is candidate if all its subsets are frequent): (k-1)-itemsets are combined to create new candidate k-itemsets in k scans until no new candidate group can be generated.	Incremental mining; Hashing; Use of bit-sets; Reduced scans; Partitioning and sampling; Dynamic itemset counting;	Inefficient for dense data (density above ~20%).
Pattern Growth [292]	Divide-and-conquer without candidate generation and multiple scans. A frequent-pattern tree is built (from an ordered list of frequent items) and mined (based on prefix paths co-occurring with growing suffix patterns). By using the least frequent items as a suffix, a good selectivity is achieved.	Depth-first tree generation; Alternative trees; Combined bottom-up and top-down traversals; Array-based structures.	Not able to deliver the supporting transactions of a pattern (required for biclustering). Adequate for dense matrices and low supports.
Vertical Projection [701]	Eclat, a representative vertical method, builds the transaction-set for each item and grows the itemsets under a depth-first strategy (similar to FP-growth) by intersecting transaction-sets to avoid multiple scans.	Specialized structures; Memory-efficient diff-sets; Efficient bit-set operations;	Optimized for flattened matrices ($m \ll m$).

Table 1.4: Major search strategies to perform frequent itemset mining

Pattern-based Biclustering. Revisiting the contents from *Section 1.2.2*, two major classes of pattern-based biclustering approaches can be considered depending on whether data is discretized (first class) or not (second class). To our knowledge, BiModule [500], DeBi [578], Bellay's et al. [57] and GenMiner [442] are the state-of-the-art methods for the first class of pattern-based biclustering. BiModule [500, 501] allows a parameterized multi-value itemization of the input matrix to discover constant biclusters derived from (closed) frequent patterns using the LCM miner [642]. DeBi [578] derives biclusters from (maximal) frequent patterns mined over binarized matrices using the MAFIA miner [106], and places key post-processing principles to adjust them in order to guarantee their statistical significance. Bellay's et al. method [57] uses the Apriori miner [7] with additional principles to evaluate the functional coherency of the discovered biclusters against the background noise. GenMiner [442] includes external knowledge within the input matrix to derive biclusters from association rules that relate annotations (external grouping of rows or columns) with clusters derived from (closed) frequent patterns using CLOSE miner [514]. Although other biclustering approaches seize contributions from these previous works [430, 20], we do not categorize them as pattern-based approaches since their core mining task does not rely on pattern mining searches.

The itemization step is optional for the second class of methods [29]. To our knowledge, RAP [509], RCB discovery [29] and ET-bicluster [277] are state-of-the-art methods here. RAP [509] plugs an adapted range-based metric to mine constant biclusters on rows (or columns) using a dedicated Apriori-based search, while RCB discovery targets biclusters with constant values overall [29]. ET-bicluster extends the previous approaches to discover noisy biclusters, although an exhaustive enumeration of biclusters is not guaranteed [277]. Alternative support metrics for Apriori-based searches have been additionally proposed [338, 601, 289].

1.4 Structured View on Pattern-based Biclustering

We propose a structured view on pattern-based biclustering contributions in order to support the design of new learning functions based on the desirable properties of the target biclustering models. This view, illustrated in Figure 1.5, is organized according to a set of dimensions. We rely on state-of-the-art contributions, as well as on new contributions, to identify the major design options and principles per dimension. The placed options and

gathered principles are grouped according to: 1) the stage of their incorporation (see Figure 1.4), and 2) the output (target structure, coherency and quality).

Pattern-based biclustering algorithms are characterized by the presence of three major stages: *mapping* (preprocessing), *mining* (pattern discovery), and *closing* (postprocessing). The core step is the *mining* step, corresponding to the application of the target pattern miners. This step is driven by the chosen PM paradigm, pattern representations and search properties. The *mapping* step (optional for biclustering algorithms able to deal with non-discrete data) is responsible for the itemization of a (real-value) matrix and for other preprocessing options to handle outlier, noisy and missing elements. Finally, the *closing* step is responsible for the postprocessing of the mined patterns to affect the properties of the target biclustering solutions. The options along these stages impact the homogeneity of the biclustering solutions, and thus their coherency, structure and quality.

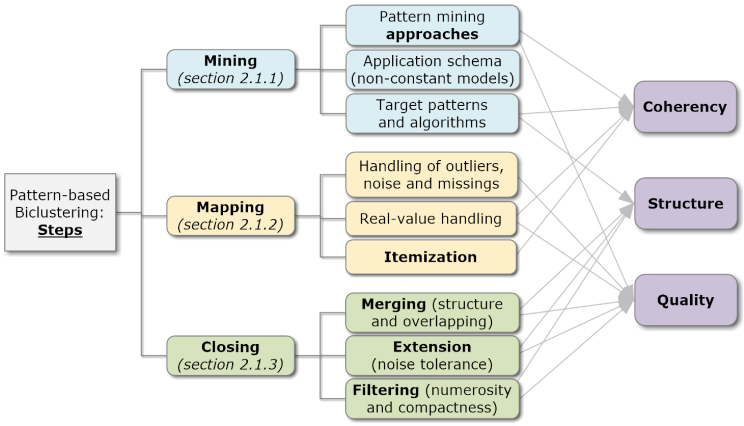

Figure 1.4: Process-view dimensions and their impact on biclustering solutions.

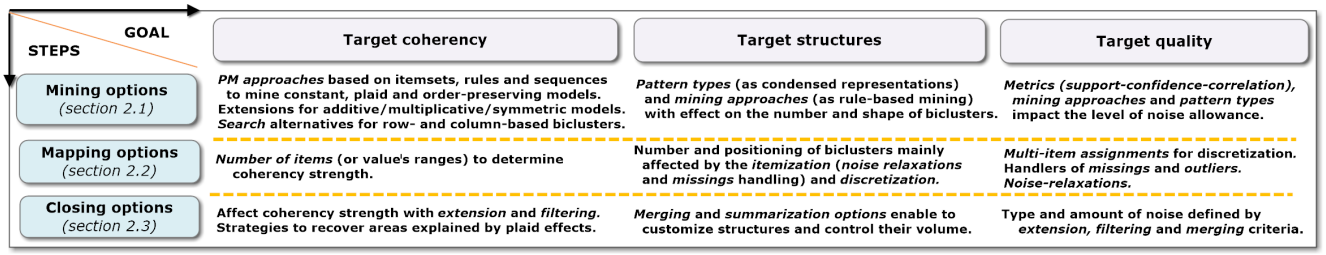

Figure 1.5: Structured view of pattern-based biclustering: illustrative options across the major dimensions.
The groups critical decision dimensions (corresponding to either a row, a column or a cell of the framework) to support the design of pattern-based biclustering approaches. A set of principles for each dimension is illustrated and detailed throughout this work for each biclustering step (mining, mapping and closing) and biclustering goal (defined according to a specific type, structure and quality of biclustering solutions).

We rely on this process-view to describe the dimensions of the proposed framework (Figure 1.5). Accordingly, *Sections 1.4.1*, *1.4.2* and *1.4.3* cover, respectively, the mining, mapping and closing options for pattern-based biclustering and discuss their implications in the behavior on pattern-based approaches.

1.4.1 Mining Options

Understandably, non-constrained settings, where the number of biclusters and their properties are not known apriori, require efficient searches. Pattern mining approaches have been tuned during the last decades to be computationally efficient. Therefore, their adequate use for biclustering is critical and depends essentially on three variables described below: *1)* the chosen pattern mining task, *2)* the target pattern representations, and *3)* the properties of the search.

1.4.1.1 Pattern Mining Task

In what follows, we introduce different pattern mining tasks to perform biclustering: 1) *frequent itemset mining*, 2) *association rule mining*, 3) *structured pattern mining*, 4) *constraint-based mining* (also termed actionable pattern

mining), and 5) *hybrid* approaches. Frequent pattern mining has been amply used as the means to compose biclusters, while the contributions on how to use association rule mining and structured pattern mining have been only loosely discussed and not yet consistently studied and validated [442, 310, 415].

1.4.1.1.1 Frequent Itemset Mining

Within frequent itemset mining (FIM), two major strategies can be considered: *1)* relying on FIM's support metric as-is; or *2)* defining new (anti-)monotonic support metrics for a dedicated search.

FIM-based Biclustering. In Figure 1.3, we illustrated how PM can be applied to find biclusters with constant items overall, on rows and on columns. When ignoring the closing step, the discovered biclusters are directly derived from frequent itemsets. The support threshold defines the minimum number of rows in a bicluster. By decreasing this threshold we are degrading the efficiency of the task, but searching for a broader set of biclusters with smaller sizes. In the context of expression data and biological networks, this is critical since small groups of molecular units can be functionally related. Additionally, the search can allow the pruning of itemsets below a minimum number of columns and above a maximum number of rows and columns.

From the point of view of a transactional database, the FIM-based biclusters are perfect biclusters, that is, they do not allow value-variations in any of its elements. Contrasting, from the point of view of the input real-value matrix, these biclusters tolerate noise as different values may be assigned to the same item. The number of items can be flexibly parameterized to control the strength of coherency. This possibility contrasts with traditional biclustering approaches on discrete matrices, whose behavior depends on a fixed coherency strength [477, 616].

BiModule [500], DeBi [578] and Bellay's et al. [57] are the state-of-the-art FIM-based approaches. Although BiModule [501, 500] allows a parameterizable number of items and support threshold, the presence of noise and the applied itemization procedure often leads to the partitioning of large biclusters into smaller ones (with many of them filtered out as no longer satisfy the support criterion). Contrasting, although DeBi [578] and Bellay's et al. method [57] alleviate this problem by providing postprocessing strategies to improve the functional coherence of the discovered biclusters, they require the input data to be binarized (two items only). All of these FIM-based approaches suffer from the additional risk of assigning two elements with similar real-values to two different items. We refer to this drawback as the **items-boundary problem**.

Range-based Biclustering. In order to mine patterns in real-valued matrices and surpass the items-boundary problem, the notion of support of a pattern can be redefined. As long as the new support metric is (anti-)monotonic, its inclusion within Apriori-based frameworks can be easily handled with efficiency. Patterns are therefore generated using breadth-first level-wise pattern tree. Naturally, as this Apriori-based framework relies on candidate generation procedures (see Table 1.4), time and memory bottlenecks can appear for high-dimensional data with the presence of large coherent regions.

Support metrics to mine ordinal and numeric data have been originally proposed by Han et al. [289], Huang et al. [338] and Steinbach et al. [601]. Calders et al. [115] proposed the use of rank-based measures to score the similarity of sets of numeric attributes within new support metrics by extending τ, Spearman's ρ, and Spearman's Footrule F correlation metrics [361]. Here, efficient algorithms are designed to deal with the ranks of attribute values, but not with the original numeric values. However, these approaches do not capture key properties of real-valued matrices, such as the need to ensure that the values of items in a transaction are within a range to guarantee coherence and distinguish positive from negative values.

RAP [509], RCB [29] and ET-biclustering [277] extend the principles provided by this former research to perform biclustering. RAP [509] uses a customized anti-monotonic range support merit function. An Apriori-based algorithm is proposed to discover constant patterns on rows from numeric matrices without the need for discretization. This has the advantage of avoiding the items-boundary problem (see Figure 1.6). Although this allows the

discovery of patterns sensitive to a particular sign, it constrains the final bicluster to only either have positive or negative values. An alternative, RCB discovery method [29], verifies range constraints on both dimensions (rows and columns) to discover functionally similar biclusters using a monotonic range measure. The Apriori-based method is slightly modify in order to grow homogeneous-squares that are then used to compose rectangles (biclusters). Similarly to RAP, the range metric guarantees that the variation in the magnitude of each bicluster element is lower than a user-specified threshold. Finally, ET-bicluster model [277] revises the previous support metrics for the discovery of noisy biclusters by guaranteeing that each supporting transaction of a target pattern does not exceeds a specific error-threshold. However, the support metric is not anti-monotonic and, therefore, does not guarantee the exhaustive search of all possible patterns, although guarantees can be given.

Despite the fact that range-based approaches hold a promising direction to pattern-based biclustering, three major challenges need to be considered. First, support metrics need to guarantee the (anti-)monotonic property and be carefully parameterized according to the background values and selected normalization procedures. Second, they are not easily supported under PM searches who rely on tree-based structures to enhance scalability. Third, condensed representations for these range-based patterns are non-trivial. This can lead to a large amount of outputted biclusters, and thus high memory complexity. Figure 1.6 provides an illustrative application of this class of enhanced FIM-based approaches against traditional FIM-based approaches.

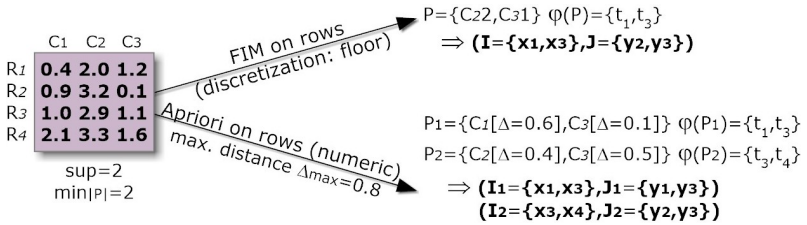

Figure 1.6: Biclustering using FIM over discrete matrices versus range-based searches over numeric matrices. Biclusters discovered with range-based support metrics are less prone to the items-boundary problem.

In labeled datasets, FIM-based and range-based approaches have been extended for the discovery of class-discriminative biclusters – biclusters with significantly higher support for a particular class [208, 602, 498]. These contributions are, however, out of the scope of this book.

1.4.1.1.2 Association Rule Mining

Association rule mining can alternatively be used to compose biclusters [303]. Its core task is the support- and confidence-guided discovery of associations between itemsets [290]. Given a matrix $\mathbf{A}$, a simple association rule either relates two sets of columns ($\mathbf{J}_1 \Rightarrow \mathbf{J}_2$) or, when transposed, two sets of rows ($\mathbf{I}_1 \Rightarrow \mathbf{I}_2$). An illustrative rule from a transposed gene expression matrix is $\{gene_A \downarrow\} \Rightarrow \{gene_B \uparrow, gene_C \uparrow\}$, meaning that when $gene_A$ is under-expressed, it is very likely that genes B and C are over-expressed. An arbitrary high number of states/items can be considered. In this context, when using association rules to compose biclusters, the items on the antecedent and consequent of a rule, as well as the supporting transactions/observations from both sides, are considered to derive biclusters.

Association rules can be used to capture larger biclusters when confidence levels are below 100%, which turn the biclustering task less prone to the problem of coherent regions being partitioned due to the presence of noisy elements. Consider the illustrative rule $R_1 : \{\mathbf{y}_2, \mathbf{y}_3\} \Rightarrow \{\mathbf{y}_4, \mathbf{y}_5\}$ with confidence below 90%. Instead of using the observations that support $\{\mathbf{y}_2, \mathbf{y}_3, \mathbf{y}_4, \mathbf{y}_5\}$ to build the bicluster, one can extend it by considering the observations that support uniquely R_1 antecedent, $\{\mathbf{y}_2, \mathbf{y}_3\}$. Confidence is thus seen as an additional indicator of homogeneity.

Although mining rules has a computational cost, they can be used to perform biclustering in ways beyond the simple frequent itemsets. In particular, to mine specific rules of interest, interestingness metrics (beside the illustrated confidence) can be used to augment the support-confidence framework. Lift metric, $lift(P \Rightarrow P') = sup(P \cup P')/(sup(P) \times sup(P'))$, measures rule's novelty. Conviction metric, $conv(P \Rightarrow P') = (1 - sup(P'))(1 - conf(P \Rightarrow P'))$, measures rule's certainty. Additional metrics, such as χ^2, cosine and all-confidence have been proposed for different

interestingness criteria [614].

In matrices with numerous correlations, the support should be set low, the confidence set high, and constraints incorporated to deal with the explosion of rules from frequent itemsets. However, such criteria can still be insufficient, due to high combinatorial possibilities associated with the exhaustive discovery of association rules.

To address this problem, the rule-based GenMiner approach [442] imposes rules to be non-redundant with minimal antecedent and maximal consequent (minimal non-redundant rule for short) in order to avoid the explosion of rules. Alternatively, association rules can be pruned based on their statistical/biological significance [20] according to hypotheses verified by correlation coefficients (such as Pearson's Product Moment Correlation, Spearman's Rank-order Correlation Coefficient and Kendall's Tau).

Carmona et al. [120] extended simple rules by integrating annotations from semantic sources, knowledge bases and bibliographic databases. Annotations are labels associated with groups of rows or groups of columns. In line with this work, GenMiner [442] integrates the input data with annotations. In particular, it focuses primarily on rules over expression data of the type *annotations* $\Rightarrow$ *expression profiles*, but also extends the association rule mining towards two additional types of associations: *1) expression profiles* $\Rightarrow$ *gene annotations*, meaning that a group of genes with an expression pattern across a set of conditions is likely to have a set of corresponding annotations; and *2)* relations among gene annotations. Illustrative rules include: $annotation_1 \Rightarrow \{c_1 \downarrow, c_2 \uparrow\}$, meaning that a group of genes (with the same annotation) is likely to be under-expressed in condition c_1 and over-expressed in condition c_2. An alternative rule is $\{c_1 \downarrow, c_2 \uparrow\} \Rightarrow annotation_1$, meaning that a group of genes with the expression profile given by c_1 and c_2 is likely to have specific annotations.

In the absence of background knowledge, annotations can be retrieved from the input matrix based on clusters of rows and columns.

Finally, and similarly to support customization, confidence and other interestingness metrics can be customized and plugged within an Apriori-based framework. However, to our knowledge, there are not yet studies and implementations on this type of rule-based approaches for biclustering.

1.4.1.1.3 Structured Pattern Mining

Approaches that target different types of patterns provide alternative search paradigms for biclustering and hold the potential to discover biclusters with specific properties. Major options are:

- *Sequential pattern mining* (SPM) approaches. SPM can be used to mine order-preserving biclusters [308, 311, 415]. A bicluster is order-preserving if there is a permutation of its columns under which the sequence of values in every row is (either monotonically or strictly) increasing. For this aim, the indexes of the elements in the matrix are reordered per row; the ordered set of indexes are mapped into a sequential database; SPM is applied; and the biclusters are mapped from the frequent sequences and their supporting transactions. Figure 6.3 illustrates this possibility;
- *Two-way pattern-based clustering* approaches. This is an interesting direction since the patterns associated with the discovered clusters on each dimension can be used to affect the structure, quality and coherency of the target biclustering model [247, 617, 109];
- *Graph mining* approaches. Real-value matrices can be mapped into weighted bipartite graphs, and thus biclustering can be mapped into the task of finding maximal cliques [432] or other substructures, such as significantly weighted modules [616]. Mining over weighted graphs for the discovery of meaningful biclusters is a direction with growing attention [313, 123, 616];
- *Cube computation* approaches. Cube computation shares similarities with frequent pattern analysis, being well-suited to deal with matrices in $\mathbb{R}^n$ when n>2 [291, 681]. The additional dimensions can be used to:
  - capture additional informative views (such as time points or replicates) [13];
  - model contributions from overlapping areas of biclusters under a plaid model assumption [312];
  - find biclusters' consensus over the matrices in cubes with different pre- and postprocessing criteria.

1.4.1.1.4 Actionable Pattern Mining

An alternative task for pattern-based biclustering is constraint-based pattern mining, also referred as actionable pattern discovery. While FIM-based and rule-based searches aim to achieve high efficiency, constraint-based alternatives use more generic searches and declaratively define flexible pattern constraints to model biclusters. In a constraint-based approach we specify *what* the problem is, rather than outlining *how* a solution should be computed [275, 364, 274]. In this context, Formal Concept Analysis (FCA) can be applied to perform biclustering. A bicluster is a specific formal concept called bi-set. A bi-set must satisfy a local constraint: the column set (or intent) is the maximal set of columns that are true for the supporting set of rows (or extent) [89]. Bi-sets may, additionally, satisfy user-defined constraints.

Def. 1.8 Given a binary matrix $\mathbf{A}$, a bi-set $(\mathbf{I}, \mathbf{J})$, where $\mathbf{I} \subseteq \mathbf{X} \wedge \mathbf{J} \subseteq \mathbf{Y}$, is a *formal concept* if it satisfies the constraint: $\forall_{i \in \mathbf{I}} \forall_{j \in \mathbf{J}} (i, j) \in \mathbf{A} \wedge \forall_{i \in \mathbf{X} \setminus \mathbf{I}} \exists_{j \in J} (i, j) \notin \mathbf{A} \wedge \forall_{j \in \mathbf{Y} \setminus \mathbf{J}} \exists_{i \in \mathbf{I}} (i, j) \notin B$.

Since formal concept analysis [68] is only applied over binary matrices, new procedures need to be applied to guarantee the discovery biclusters with parameterizable coherency strength. For this aim, we propose the discretizion of the input real-value matrix followed by its *denormalization* into larger matrices (see *Basics 1.2*. In the denormalization step, each column is decomposed into $|\mathcal{L}|$ new columns, being $|\mathcal{L}|$ the number of items.

Basics 1.2 Matrix denormalization to apply FCA-based biclustering

Naturally, biclustering on binary matrices is only natively capable of discovering biclusters with differential coherency strength ($|\mathcal{L}|$=2). However, denormalization procedures over discrete matrices with $|\mathcal{L}| > 2$ followed by the subsequent search for dense regions (high ratio of 1s) enables the application of formal concept analysis with parameterizable coherency strength. Figures 1.7 and 1.7 show an illustrative discrete matrix and its respective denormalized form.

Figure 1.7: Discrete matrix $\mathbf{A}_9$ ($|\mathcal{L}|$=3)

	$\mathbf{y}_1$	$\mathbf{y}_2$	$\mathbf{y}_3$
$\mathbf{x}_1$	a	b	a
$\mathbf{x}_2$	c	c	a
$\mathbf{x}_3$	b	b	a

Figure 1.8: Denormalization of $\mathbf{A}_9$ matrix

	$\mathbf{y}_1 a$	$\mathbf{y}_1 b$	$\mathbf{y}_1 c$	$\mathbf{y}_2 a$	$\mathbf{y}_2 b$	$\mathbf{y}_2 c$	$\mathbf{y}_3 a$	$\mathbf{y}_3 b$	$\mathbf{y}_3 c$
$\mathbf{x}_1$	1	0	0	0	1	0	1	0	0
$\mathbf{x}_2$	0	0	1	0	0	1	1	0	0
$\mathbf{x}_3$	0	1	0	0	1	0	1	0	0

Def.1.8 guarantees that a bi-set is a maximal sub-matrix of true values (no zero value inside, at least one zero value outside). However, the discovery of biclusters under this definition suffers from three major drawbacks. First, only perfect bi-sets (intolerant to noise) are discovered. Second, when the size or dimensionality of the input matrix is high, the extraction becomes unfeasible. Finally, even when the computation is tractable, it often results in a large amount of bi-sets.

To address the first problem, noise-tolerant bi-sets, referred as DRBS, can also be considered [89]. These bi-sets can include false elements below a parametrized threshold.

To address the remaining problems, the use of user-defined constraints can be placed to guarantee the efficiency of the searches. Depending on the type of constraints, this option may tackle the scalability limits and, additionally, be used to affect the structure, coherency and quality of biclusters by specifying subjective interestingness criteria. Examples of user-defined constraints include:

- $C_{size} \equiv |\mathbf{I}| > \alpha \wedge |\mathbf{J}| < \alpha$, where $\alpha \in \mathbb{N}^+$;
- $C_{area} \equiv |\mathbf{I}| \times |\mathbf{J}| > \alpha$, where $\alpha \in \mathbb{N}^+$;
- $C_{mean} \equiv \forall_{i \in \mathbf{I}} a_{i\mathbf{J}} > \alpha$, where $\alpha \in \mathbb{R}$;
- $C_{member} \equiv \mathbf{x}_i \in \mathbf{I} \vee \mathbf{y}_j \in \mathbf{J}$;
- $C_{inter} \equiv |\ \mathbf{I} \cap \mathbf{I}'\ | > \alpha \wedge |\ \mathbf{J} \cap \mathbf{J}'\ | < \beta$, where $\mathbf{I}' \subseteq \mathbf{X}$, $\mathbf{J}' \subseteq \mathbf{Y}$.

Since some constraints cannot be processed efficiently, special care must be taken when defining constraints in order to guarantee that they are either succinct, monotonic or anti-monotonic. For instance, the introduced

illustrative size constraint is anti-monotonic on the intent and monotonic on the extent. The use of such constraints is proven to enlarge the applicability of FCA for high-dimensional matrices [89].

Pointers 1.3 Incorporating advanced constraints

A few constraint-based algorithms suited to find local patterns support the definition of more flexible constraints [454, 364, 274]. Some of these constraint include the possibility to not only define *constants* (either values, items, itemsets or specific transactions); but also *variables*, set and numerical *operators* ($\cup, \cap, \setminus, \times, +, -$), as well as *functions* involving one or several terms as $overlapTrans(P, P') = |\Phi_P \cap \Phi_{P'}|$ or $area(P) = |P| \times |\Phi_P|$.

1.4.1.1.5 Hybrid Approaches

Biclustering can rely on multiple types of patterns discovered by different PM tasks. Valid options include the definition of *ensemble* methods either combining: 1) plain and structured patterns, or 2) the output of pattern mining searches parameterized with different support-confidence thresholds. Frequent itemsets can be also used to produce an initial solution, while rules can be posteriorly mined to shape the discovered biclusters by accommodating noise. An alternative ensemble model can be inferred from the iterative parameterization of a PM method with different constraints of interest. To our knowledge, these hybrid possibilities have not been systemically studied in literature.

1.4.1.2 Pattern Representation

Depending on the chosen PM approach, different patterns, such as frequent itemsets, association rules, sequential patterns or structured patterns, can be considered. Each of these patterns can have different representations, being the most common: simple, maximal, closed, pseudo-closed, approximated, rare, top-K, multilevel and erasable [114, 519, 688].

In particular, when targeting association rules, additional representations can be considered as indirect, minimal, non-redundant, approximative, quantitative and sporadic rules [614].

Although an analysis of the impact of using each representation on the biclustering solutions is possible, we consider simple, maximal and closed representations to preserve conciseness. Nevertheless, the provided analysis can be easily extended to study how the remaining representations affect the structure and homogeneity of biclustering models.

Def. 1.9 Given an itemset matrix, a support threshold θ, and the coverage function $\Phi : 2^{\mathcal{L}} \rightarrow 2^{D}$ that maps an itemset P to its set of supporting transactions.

- a *closed itemset* is a frequent itemset that has no superset with the same support ($\forall_{P' \supset P}\ |P'| < |P|$).
- a *maximal itemset* is a frequent itemset with all supersets being infrequent, $\forall_{P' \supset P} |\Phi(P')| < \theta$.

Basics 1.4 Illustrating condensed representations

A frequent itemset is maximal if it is frequent and all supersets are infrequent, while it is closed if it is frequent and there exists no superset with the same support. Given the illustrative transactional database $\mathbf{A}$=$\{\{a, b, h, j\}, \{d, h, j\}, \{c, d, h, j\}\}$, and minimum support threshold θ=2 ($|\Phi_P| \geq 2$) and pattern length $|P| \geq 2$, there is: one maximal frequent itemset ($\{d, h, j\}$) and there are two closed frequent itemsets ($\{d, h, j\}$ and $\{h, j\}$).

The selection of the pattern representation not only impacts the properties of the biclustering models, but also the computational complexity of the biclustering task. Although efficiency gains are observed for condensed representations, the worst-case complexity of pattern miners is similar for simple, maximal and closed representations. Thus, differences on the computational complexity are primarily determined by the post-processing costs, which depend on the number and size of found biclusters.

Maximal itemsets for biclustering, such as those used in DeBi [578], are associated with biclusters with the columns' size maximized. Such flattened biclusters are only of interest when there is an extension step to be performed to include new rows. However, since both vertical and smaller biclusters are lost, this representation

leads to incomplete solutions as they are just a subset of all valid biclusters (frequent itemsets with fewer items but higher support are discarded).

The opposite alternative is the use of all frequent itemsets for biclustering for outputting all the potential combinations of biclusters above the minimum support threshold (number of rows) and itemset length (number of columns). This solution leads to a high number of potentially redundant biclusters (if contained by another bicluster), which can degrade the performance of the mining and closing steps.

Finally, the search for closed itemsets, such as target by the FIM-based BiModule [500] and rule-based GenMiner [442], allows the discovery of overlapping biclusters if a reduction on the number of columns results in a higher number of rows. Closed pattern solutions thus enable the return of all maximal biclusters (Def.1.10). The properties of these three alternative representations are illustrated in Figure 6.4.

Def. 1.10 A **maximal bicluster** is a bicluster that is not included in other biclusters. An **optimal bicluster** is a maximal bicluster whose homogeneity is satisfied globally (exhaustively guaranteed) with regards to a certain criteria.

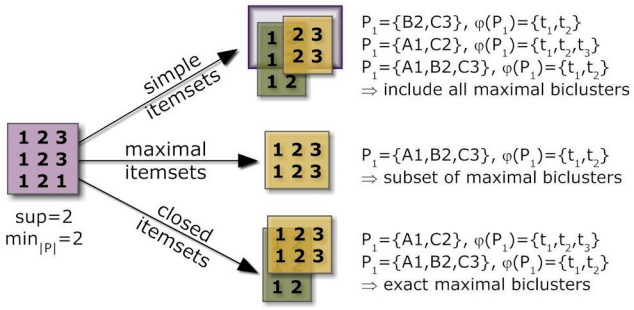

Figure 1.9: Comparison of biclustering solutions using simple, maximal and closed patterns.

1.4.1.3 Algorithmic/Search Options

The choice regarding the search strategy placed by the PM task depends essentially on the data size, data dimensionality, coherency strength and coherency orientation of biclusters (whether across rows or columns).

The choice of whether to use a vertical or an horizontal data format depends essentially on the dimensionality-size ratio and orientation of the target biclusters. To find constant values on the rows or on both dimensions, PM with searches over horizontal data are usually preferred assuming that data size is significantly larger than dimensionality. When the data dimensionality is significantly larger than data size, the search for biclusters with constant items on rows benefits from vertical data formats. If the goal is to find biclusters with constant values on columns, vertical formats are preferrable when $m > n$ since the performance of searches under horizontal format degrades exponentially with an increased number of items. Otherwise, horizontal formats should be target.

The choice of whether to use an Apriori-based, pattern-growth or combined approach, depends on three variables: *1)* the class of pattern-based biclustering (e.g. range-based approaches cannot rely on pattern-growth methods), *2)* the density of the resulting transactional database, and *3)* the ability to retrieve the supporting transaction set for each frequent itemset without degrading the overall efficiency.

When biclusters with constant values overall are targeted, the resulting matrices are sparser (see Figure 1.3) and, therefore, an Apriori strategy is preferred. For denser matrices, pattern-growth strategies are preferable. The density of a transactional database is determined by the assumed coherency strength. Illustrating, an alphabet of 5 items leads to a database with a density of approximately 20%.

Note, however, that the discovery of patterns together with their supporting transactions has been only tackled under Apriori and vertical-based searches by recurring to bitset vectors to capture the supporting transactions per pattern [442, 578, 500]. To our knowledge, there are not yet pattern-growth searches able to deliver the supporting

transactions as they are lost during the construction of frequent pattern tree structures (FP-tree). Unlike Apriori and vertical-based searches, the extension of pattern-growth searches towards this end is not trivial.

An additional key aspect is the chosen implementation. The use of efficient bit-set operations, adequate data structures and further strategies to avoid candidate generation and successive scans to the transactional database are required for a top performance. In Table 1.4, we surveyed state-of-the-art principles to enhance the efficiency of pattern mining searches. Efficient implementations include algorithms to mine closed itemsets under an Apriori search (LCM [642], Charm [702]), vertical search (TD-Close [413], A-Close [513]) or pattern-growth search (FPClose [267]); and to mine maximal itemsets under an Apriori search (MaxMiner [50]), vertical search (Mafia [106]) or pattern-growth search (AFOPT [412]). Similarly, multiple implementation variants can be found to compose association rules [714, 446] and to mine structured patterns [424]. In DeBi [578], BiModule [501] and GenMiner [442] use Mafia [106], LCM [642] and CLOSE [514] implementations, respectively. Range-based variants use Apriori [7]. Additional principles proposed in literature [714, 506, 507] can be seized to guarantee the scalability of the search when mining large biclusters from dense or high-dimensional data settings.

1.4.2 Mapping Options: Preprocessing Input Data

Previous section covered the essential mining options with impact on the structure, coherency and quality of pattern-based biclustering solutions. However, their optimum application requires the input matrices to be correctly normalized and (depending on the pattern-based approach) discretized. Independently from the need for discretization, the problem of defining an adequate coherency strength is identical for range-based approaches (distance thresholds as a function of data domain values) and discrete pattern-based approaches (number of items). This section covers pre-processing options according to normalization, coherency strength selection and discretization decisions.

Normalization Options. Normalization is often applied before biclustering to enhance differences across rows and/or columns and, consequently, to improve the ability to discover biclusters. As such, normalization can be either applied in the context of a row, a column or the overall matrix. Each context leads to different biclusters and is, respectively, suited to find activity patterns on bicluster's columns, rows or on both dimensions.

Souto et al. [165] compared three normalization procedures: z-score, scaling and rank-based procedures over high-dimensional (expression) data using alternative clustering algorithms. A high number of additional methods for preprocessing the input matrix have been reported [615, 305, 430, 115, 500].

Since the properties of these procedure may constrain further itemization options, a common principle is to adopt the zero-mean for the row/column/matrix. The use of zero-mean allows for symmetries and it provides a simple setting for the application of multiple probabilistic distributions. Finally, when assuming the presence of missing and outlier elements, a masking bitmap is sometimes should be considered in order to exclude them from the computation of the mean and dispersion metrics [500].

Coherency Strength. The coherency strength of the target biclustering models is fixed during the mapping step, and strongly impacts both the properties of the resulting models and the computational complexity of the biclustering task. A loose coherency strength is more computationally expensive than a high coherency strength since it is associated with denser transactional databases and larger biclusters. The coherency strength is either defined by the number of items (also referred to as symbols or expression levels) for discrete approaches to pattern-based biclustering and by the maximum distance thresholds for range-based approaches to pattern-based biclustering. A sensitivity analysis on the impact of the coherency strength on the properties of biclustering models was, first, performed in Bidens [430] and BiModule [500].

The coherency strength is essentially dependent on the data domain, user expectations and biclustering goal. Aggregation procedures can be used to estimate the desirable strength [430, 500]. Since coherency strength is

easily parameterized within pattern-based biclustering approaches, the biclustering search can be applied to find explorative solutions and, depending on their properties, new searches can be applied with alternative coherency strength. In fact, we advise the use of iterative searches with varying coherency strengths. These searches (involving mapping and mining steps) should be performed prior to the application of postprocessing procedures due to the relevance of post-filtering for avoiding voluminous solutions with a high number of similar biclusters.

Discretization Options. Although discretization may imply loss of information, it alleviates the noise dilemma [120, 152] and it has been arguably defended as the cost to pay for the target exhaustive searches.. In what follows, we review adequate discretization methods to map a normalized real-value matrix into a transactional database assuming a fixed number of items (given by previous principles).

Figure 1.10 illustrates how simple discretization options can lead to different solutions. The itemization (concatenation of the item with the column-index) implies that the resulting number of items is at most $m \times |\mathcal{L}|$, being $|\mathcal{L}|$ the alphabet length specified by the user.

The use of fixed ranges (potentially equal sized intervals between the observed maximum and minimum) is the simplest discretization option, but it usually leads to an accentuated weak distribution of items and it is prone to the items-boundary problem. The first problem can be corrected using a percentage-based method for the depth partitioning of items that leads to intervals containing approximately the same number of elements.

Alternatively, distributions combine the properties of the previous solutions. In Figure 1.10, a Gaussian distribution is able to minimize the loss of potentially relevant biclusters. By finding multiple suitable curves (for each row or column) or one suitable overall curve to approximate the matrix, one can either use threshold methods [120, 152] or compute the statistical cutoff points to create equally-distributed areas. Statistical tests should be used to adequately select the underlying probability function. In the presence of matrices with multimodal distributions, more expedite methods based on a mixture of distributions must be considered. Nordi [442] is a Gaussian-based method used in GenMiner [442] that statistically detects outliers (using the Grubbs method), applies normality tests (using QQ-plot and Lilliefors) to transform the initial row distributions into a "more" Normal distribution, and computes cutoff thresholds using the z-score methodology. In this case, three conditions (over-expressed, $\frac{a_{ij}-\mu_j}{\sigma_j} \leq \alpha$, under-expressed, $\frac{a_{ij}-\mu_j}{\sigma_j} \leq -\alpha$, or unexpressed) are obtained by computing the cutoff thresholds under a given confidence degree α, and the outliers are integrated in a final step.

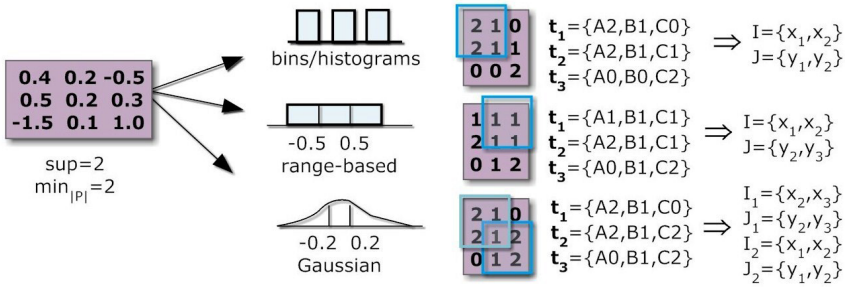

Figure 1.10: Comparison of alternative discretization options by addressing their impact on the itemization and biclustering solutions with constant values on columns.

More advanced discretization options that aim to deal with the items-boundary problem include:

- Adaptive discretization based on dynamic threshold selection policy [533];
- Statistical methods to detect differentially activity of elements as the basis to create partitions [152] (commonly adopted as a binarization method);
- Distance-based subspace clustering models [411] to flexibly partition the values while preserving meaningful and significant clusters;
- Fuzzification approaches where a continuous domain is partition into fuzzy sets, provided to be more robust to noise when compared with other simple binning techniques [422];

- Supervised discretization methods [212] (when descriptive labels per row or column are present or computed using clustering methods), where a row or column is partitioned into a number of disjoint intervals in such a way that the entropy of the partition is minimal.

1.4.3 Closing Options: Postprocessing Biclustering Solutions

Pattern-based biclustering approaches produce exhaustive solutions with flexible structures (arbitrary number and positioning of biclusters). Nevertheless, two key challenges are associated with such solutions: high intolerance to noise and propensity to deliver an intractable high number of valid biclusters. Part of these questions were answered recurring to the mapping options by selecting an adequate coherency strength and (possibly) discretization procedure able to minimize the items-boundary problem. However, they are insufficient to tackle the challenges of the noise dilemma. On one hand, a too restrictive noise tolerance, leads to many the partitioning of true biclusters in many small sized biclusters. On the other hand, high tolerance to noise, commonly occurring by binarizing data or through the use of rule-based approaches under a relaxed level of confidence, leads to inclusion of false positive rows and columns per bicluster. To handle these challenges we propose the definition of desirable homogeneity criteria for the target solutions, which is then satisfied according to four major postprocessing steps (merging, filtering, reduction and extension), described below.

1.4.3.1 Merging Options

Merging biclusters may serve two goals: noise allowance (to avoid solutions composed uniquely of small biclusters) and overall biclustering structure manipulation. The first goal is driven by the observation that when two biclusters share a significant area it is probable that their merging composes a larger bicluster still respecting some homogeneity criteria. Commonly, such decomposition is related to the items-boundary problem or with a missing value.

The simplest criterion to allow the merging is either to rely on the overlapping area (as a percentage of the smaller bicluster), to compute the overall noisy percentage after the merging, or both. More expressive homogeneity criteria (potentially relying on the real-values provided by the input matrix) can be additionally formulated.

Although principles have been proposed to foster the efficiency of merging procedures [680, 686], they are not scalable as they require a heavy combinatorial pairing of biclusters. Bellay et al. [57] proposed a Markov Clustering (MCL) algorithm to both summarize biclustering solutions and allow for the creation of larger biclusters from subsets of biclusters. An alternative strategy is to rely on the concept of colossal patterns (from subsets of smaller patterns) [714]. Recently, Henriques et. al [308] proposed three distinct merging procedures showing heightened efficiency.

1.4.3.2 Filtering Options

Filtering is required to guarantee the dissimilarity of biclusters, removing biclusters partially contained in larger biclusters. The existence of biclusters included in larger biclusters is a necessary result of the merging options and it is a common problem when adopting mining approaches that do not rely on condensed pattern representations. To efficiently filter biclusters, existing approaches typically start with the larger biclusters and compare them with remaining biclusters. When a set of overlapping biclusters show large overlapping areas, only the one being scored as the most interesting (e.g. based on size, quality or domain relevance) is preserved.

Both DeBi [578] and BiModule [500] provide alternative heuristics to efficiently perform this type of filtering. DeBi [578] starts with the largest bicluster and compares it to the other biclusters, filtering those whose overlap with the largest bicluster exceeds L% (typically 50%) of the size of the smaller bicluster. BiModule [500] filters out small biclusters in two steps. First, biclusters are sorted according to the score $a_{\mathbf{IJ}} \times log_2|\mathbf{I}| \times log_2|\mathbf{J}|$. Second, biclusters whose cells overlap by more than 25% with a bicluster with higher priority (based on domain relevance)

are removed. The work by Bellay et al. [57] separates biclusters that represent biological phenomena from false discoveries (emerging from the background data distributions) using randomized data scores.

1.4.3.3 Extension Options

High coherency strength and intolerance to noise are often associated with biclustering solutions with high homogeneity but weak interest. Two optional and non-exclusive strategies can be used to extend the such biclustering solutions so that the resulting solution still satisfies some pre-defined homogeneity criteria. A first strategy consists on the use of statistical tests to include rows or columns from each bicluster. An illustrative statistical test is to assess the strength between key columns of a bicluster and a new row using Fisher's exact test for independence on a contingency table [578]. This guarantees that each row in the bicluster shows a statistical difference between the columns in the bicluster and the columns not in the bicluster, leading to more functionally coherent biclusters. Due to the high number of tests, heuristics need to be applied in order to perform these tests on a subset of candidate rows and columns respecting already some interestingness criteria according to a similarity metric.

Illustrating, DeBi [578] uses statistical tests to extend biclusters obtained from binary matrices by association strength of new rows (columns) by testing their independence on a contingency table. A row $\mathbf{x}_i$ is added to the bicluster's row-set $\mathbf{I}$ if its p-value p_i is lower than a threshold α[2]. DeBi proposes a way to reduce the computational effort of computing the p-value for every row (column) not belonging to a target bicluster by using the monotonicity property of the hypergeometric distribution: cut-off values yielding $p_i < \alpha$ are pre-computed. Only these rows (columns) are seen as candidates for extension. The quality of biclusters is, then, measured using the negative sum of the p-values of the included rows, $-\Sigma_{\mathbf{x}_i \notin \mathbf{I}} log(p_i)$, and corrected by the bicluster size.

A second strategy is to rely on traditional merit functions for further (greedy) extensions over pattern-based biclusters. In other way, the obtain pattern-based biclustering solution is given as input for a greedy biclustering search for iterative extensions over original solution/seed.

The use of simple thresholds, statistical tests, or merit functions to verify/guide extensions that still preserve certain homogeneity criteria can be either computed from the discrete matrix (item matchings) or, more interestingly, using numeric distances from the original real-valued matrix. The problem associated with these extension procedures is their prohibitive computational complexity.

1.4.3.4 Reduction Options

Reduction procedures are needed to exclude rows or columns from a particular bicluster in order to increase its homogeneity due to high tolerance to noise or loose coherency strength. This is usually the case when a low number of items is considered, leading to highly noise-tolerant biclusters. For this purpose, and similarly to extension options, we can rely on statistical tests on each row and column of a particular bicluster to identify removals [578], as well as use existing greedy-iterative approaches to maximize a merit function until a parameterizable reduction in size is verified [134].

1.4.3.5 Alternatives to Previous Closing Options

Alternative postprocessing options besides merging, filtering, extension and reduction can be considered to guarantee the robustness of large solutions, including:

- summarization techniques based on simple and hierarchical clustering methods or on the definition of similarity measures to compare biclusters [75];
- user-driven formal constraints and querying expressions [89, 90];
- co-clustering for exclusively partition both dimensions to select representative biclusters [172];

[2] Since statistical tests can be highly dependent on the significance criteria, DeBi dynamically performs a sensitive analysis over a score function to fix α. The optimized threshold was shown to be smaller for a large number of columns (rows), which limits the number of rows (columns) accepted into a bicluster.

- pre- and post-pruning techniques (including item-based constraints and discrimination metrics) [457];
- patterns based on half-spaces (as quantitative rules) in which external sources of information are used as a filtering basis [244];
- verification techniques based on metrics computed using external data sources as, for instance, term enrichment (in gene expression data) to affect the addition-removal of columns-rows per bicluster.

1.5 Comparison of Pattern-based Biclustering Approaches

In what follows, we provide a synthesis of the benefits and challenges of using pattern-based biclustering approaches together with principles on how to tackle existing challenges. Table 1.5 discusses the properties of the major classes of pattern-based biclustering in general, and of each surveyed approach in particular.

Approach	*Major Benefits*	*Challenges*	*Principles to tackle Challenges*
Pattern-based Biclustering	- Exhaustive searches; - Flexible structures; - Parameterizable coherency strength (noise-tolerance); - Extensible for multi-class data settings;	1) Deterioration of efficiency levels for very large data (in the absence of PM scalability principles); 2) Not prepared as-is to capture additive, multiplicative, order-preserving and plaid coherence; 3) High number of similar biclusters; 4) Need to fix thresholds.	1) Data partitioning methods; PM in distributed settings; approximated patterns [290, 684]; 2) Iterative data mappings to mine advanced types of biclusters; merging of biclusters sensitive to overlaps; 3) Adequate data structures; filtering options; 4) Converging thresholds; data-driven estimation.
Range-based Support Biclustering	- Range-based support addresses the items-boundary problem; - Easy extension of Apriori methods to seize efficiency gains when dealing with multiple distances (support thresholds).	1) Need to separate positive and negative values to guarantee monotonicity leading to biclusters without symmetries; 2) Dedicated Apriori-based methods do not allow the direct use of PM scalability principles.	1) Since combination of signs violates (anti-)monotonic property, there is only the option for merging biclusters with different signs but identical supporting transactions; 2) Dedicated extensions to deal with tree structures (for dense datasets), data partitions, and distributed settings.
DeBi *(see pattern-based)*	Complete and statistical rigorous post-processing options with parameterizable significance.	Binarization of data; Efficiency deterioration from post-processing procedures; Maximal patterns only (loss of a large number of relevant biclusters).	Discovery of closed patterns (removes the need for exhaustive extension procedures); Multi-level discretization.
BiModule *(see pattern-based)*	Multi-level discretization with removal of outliers.	No merging-extension options for handling noise and discovering larger biclusters.	Inclusion of post-processing options.
GenMiner *(see pattern-based)*	Robust biclusters from association rules (non-perfect confidence levels); Ability to integrate background knowledge.	Requires annotations from knowledge bases; Non-parameterized levels of expression.	Retrieval of annotations directly from data (in the absence of knowledge bases); Rule discovery in the absence of annotations; More flexible preprocessing options.
RAP *(see pattern- & range-based)*	Parameterizable distances with no propensity to discretization problems.	Constant coherency is hardly extensible; Not able to accommodate noise.	Inclusion of post-processing options (merging and extension).
RCB Discovery *(see pattern- & range-based)*	Able to incorporate symmetries.	Restrictive coherence (excluding differences across columns); Discovery from joint squares is a combinatorial problem that impacts efficiency [29].	Combined results with biclustering solutions from peer approaches; Revised computational methods to provide guarantees of efficiency.
ET-Bicluster *(see pattern- & range-based)*	Discovery of biclusters with parameterizable levels of noise.	Not able to guarantee exhaustive solution since error-based thresholds violate the (anti-)monotonic search property.	Relaxed thresholds (together with a post-filtering) to avoid loosing biclusters of interest; Bounding optimality guarantees.

Table 1.5: Comparison of the two major classes of pattern-based biclustering approaches: benefits, challenges and possible improvements of state-of-the-art pattern-based biclustering approaches.

Understandably, different applications may be optimally answered by different pattern-based biclustering approaches. BiModule and RAP are default options for settings where meaningful biclusters can only be found using multiple coherency levels, which is often the case with scored biological/social networks, expression data and physiological data [310, 500, 509]. DeBi is critical for the analysis of large Boolean datasets, such as the ones derived from genomic structural variations or non-weighted biological/social networks [578]. GenMiner's ability to incorporate external knowledge is relevant for biomedical contexts [442]. The constant overall assumption of RCB is critical to efficiently mine biclusters disclosing a specific behavior or preference in (web) social data and collaborative filtering data [29]. The noise-tolerance of ET-biclusters is relevant to deal with experimental errors and individual variations of molecular, psycho-physiological and clinical data [277].

1.6 Summary of Contributions and Implications

This chapter provided a structured view on pattern-based biclustering by surveying contributions from the fields of pattern mining, biclustering and multivariate data analysis. Existing pattern-based approaches for biclustering perform exhaustive searches under relaxed conditions (flexible structures of biclusters with parameterizable properties) with heightened efficiency. In this context, we listed the current opportunities and challenges of pattern-based biclustering and discussed its relevance for a wide-set of applications.

We studied alternative design options to guide the definition of pattern-based biclustering approaches:

- mining options, including different pattern mining tasks (including frequent itemset mining, association rule mining, sequential PM, actionable/constraint-based PM and structured PM) with impact on the solutions; decisions associated with the selection and calibration of support-confidence-correlation metrics; pattern representations (simple, condensed and approximate); and algorithmic decisions dependent on data size, dimensionality and coherency orientation;
- preprocessing options, including normalization and discretization options and initial strategies to deal with the items-boundary problem and noisy elements;
- postprocessing options, including merging, filtering, extension and reduction procedures and their criteria thresholds (homogeneity and (dis)similarity conditions, overlapping degree, and statistical significance levels) to guarantee an adequate tolerance to noise without the need to adapt the core task.

A set of principles was proposed based on the understanding of how the previous options affect the performance of the target approaches and the structure, coherency and quality of the target solutions. However, turning all of these options available to the user may difficult the use pattern-based biclustering approaches. As such, we provided a framework that allows for an easily parameterizable environment for their design and use, where the behavior can be dynamically defined according to the properties of the input data and output solution. Finally, we provided a qualitative comparison of the state-of-the-art pattern-based biclustering approaches.

Future Work. Following this comprehensive chapter, new research is required to embrace critical new directions, including: 1) development of integrative pattern-based biclustering approaches satisfying the proposed principles; 2) further investigation of new principles to guarantee robustness to noise and missing values; 3) extension of PM searches for constant biclusters towards more flexible coherency assumptions; 4) revision of PM searches to seize efficiency gains from the specificities of the biclustering task and address time-and-memory bottlenecks; 5) integration of principles from domain-driven PM to incorporate constraints in pattern-based biclustering when background knowledge is available; 6) definition of statistical tests to effectively assess the significance of biclusters; and 7) design effective classifiers based on discriminative biclusters.

2

Biclustering Robust to Noise, Missings and Discretization Problems

Guaranteeing the robustness of biclustering models is essential since real data is characterized by the presence of different forms and amount of noise. Noise can be associated with inaccuracies from (experimental) data collection, stratum biases on the observed sample, applied pre-processing techniques, subjectivity of features, among other sources of error variability. As a result, elements within a relevant region (true bicluster) can show slight deviations against the expected value, as well as large value deviations due the presence of artefacts, plaid effects or other forms of noise.

In the absence of closing options, the majority of existing approaches to pattern-based biclustering are only prepared to deliver biclusters with no tolerance to noise (all elements respect a given coherency strength). Existing research on biclustering further suffers from the three following drawbacks. First, biclustering from discrete data is susceptible to the items-boundary problem [**R2.3.1**]. The items-boundary problem is related with the risk of assigning two elements with similar real-values to two different items. In this context, although range-based approaches for pattern-based biclustering are not prone to this problem, their dependency on Apriori-based searches is associated with efficiency bottlenecks for high-dimensional data.

Second, there is a lack of an integrative view on how to deal with different forms of noise [**R2.3.2**]. Particular care should be placed to avoid polarization towards one of the two poles associated with the noise dilemma: too restrictive tolerance to noise (associated with many small sized biclusters) or heightened tolerance to noise (associate with low-quality biclusters).

Third, biclustering tasks are not able to effectively handle arbitrary-high levels of missings [**R2.3.3**].

This chapter aims to target these requirements by proposing principles for their satisfaction. Also, in order to guarantee their applicability for problems that require the delivery of dedicated structures of biclusters, we target the additional requirement of customizing biclustering structures from robust solutions. Furthermore, we propose a new biclustering approach, BicPAM (**Bic**lustering based on **PA**ttern **M**ining), to integrate these principles and the disperse efforts towards pattern-based biclustering surveyed in the previous chapter. As a result, BicPAM provides the distinct possibility of discovering exhaustive solutions of biclusters with parameterizable quality and underlying structures, and dynamically adapts its behavior to mine data with varying levels of missing values and noise. In this context, five major contributions are provided:

- extensions to discretization procedures without susceptibility to the problem of item-boundaries;
- principles to guarantee an adequate identification and tolerance to noise;
- principles to handle arbitrary-high missing data according to different relaxation levels;
- principles to compose customized biclustering structures of interest;
- integrative pattern-based approach that consistently combines disperse contributions on pattern-based biclustering with the previous principles.

Empirical evidence shows the robustness of BicPAM and its superiority against peer pattern-based approaches and state-of-the-art biclustering approaches. Results on synthetic data show that BicPAM is able to successfully recover planted biclusters with varying levels of missing values and noise. Also, its application over biological data

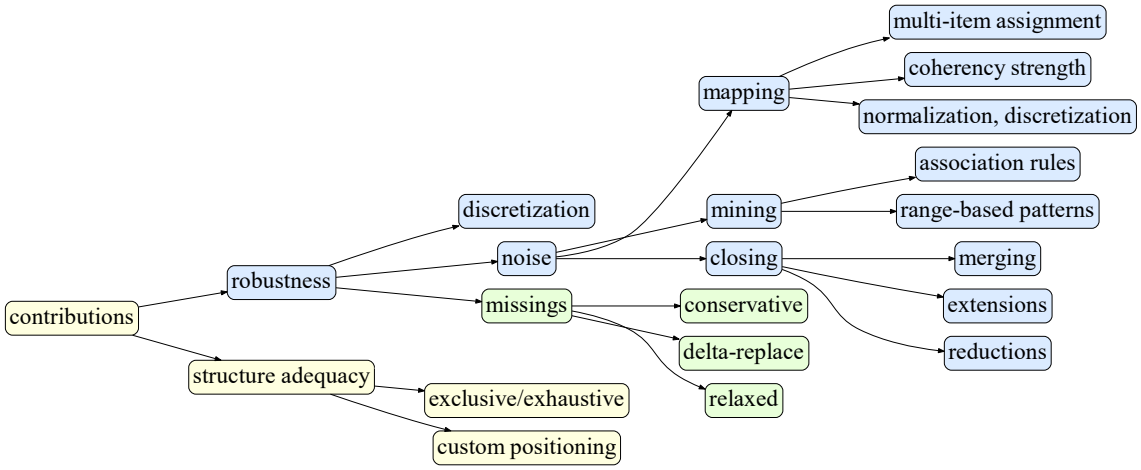

Figure 2.1: Contributions to guarantee the robustness of (pattern-based) biclustering solutions and customize structures.

leads to unique solutions with heightened biological relevance.

Figure 2.1 provides a structured view on the contributions of this chapter. Accordingly, this chapter is organized as follows. *Section 2.1* provides a brief view on how pattern-based biclustering behavior can be used to affect the quality of the models. *Sections 2.2, 2.3* and *2.4* propose the contributions to, respectively, guarantee robustness to discretization problems [R3.1], different forms of noise [R3.2], and high levels of missings [R3.3]. *Section 2.6* uses the structured view proposed in the previous chapter in order to derive an integrative pattern-based biclustering approach. *Section 2.7* assesses the performance of BicPAM on synthetic and real data, and discusses its impact.

2.1 Affecting the Quality

The homogeneity criteria of a biclustering model can be used to determine the quality of individual biclusters, the overall solution, or both. Illustrating, given a parameterizable percentage of noisy elements, this percentage can be respected by every bicluster or for the overall solution (to allow some biclusters to deviate from quality expectations). Note that the task of handling noise (guaranteeing quality) should not be confused with the task of selecting an adequate coherency strength. Coherency strength is a property associated with the expected correlation of values in a given bicluster, which is given by the number of items in discrete settings and by allowed maximum distances in real-valued settings. In fact, only a subset of biclustering approaches rely on parameterizable coherency strength, including pattern-based approaches [310, 500, 509, 29, 277] and approaches based on residue-based merit functions with parameterizable threhsolds [134, 690, 689]. For this set of approaches, the quality of a bicluster is given by the overall deviation of values of elements from the allowed numeric interval. For the remaining set of approaches, the quality of a given bicluster is given by the observed deviations of values of elements from their expect value.

According to the principles surveyed in the previous chapter, the quality of a pattern-based biclustering solution can be easily affected through:

- preprocessing options. Normalization, selection of items and (optional) discretization procedures highly affect the quality of biclustering solutions, as they can address the noise dilemma and determine the propensity of solutions to the items-boundary problem;
- mining options. This includes the possibility to rely range-based support metrics tolerant to noise, to seacr association rules with varying confidence levels to adjust the tolerance to noise, and to mine approximate patterns;
- postprocessing options. This includes merging, filtering, extension and reduction of biclusters guided by the allowed percentage of noisy elements, dissimilarity conditions and desirable homogeneity.

2.2 Robustness to Discretization Problems

In order to define pattern-based biclustering approaches not susceptible to discretization problems range-based support metrics can be used. However, due to the dependency of range-based approaches on candidate generation procedures, they can show both time-and-memory bottlenecks for high-dimensional data. To avoid this problem, the surveyed list with advanced discretization procedures to minimize the items-boundary problem in previous chapter (Section 1.4.2) can be used for this purpose. Yet, these principles still do not guarantee a total avoidance of the problem. In this context, we propose a new simplistic and effective strategy: multi-item assignments. Accordingly, below we briefly overview the properties of range-based support metrics and then describe the multi-item assignments procedure.

Range-based Support. Range-based support metrics applied in the context of dedicated Apriori-based searches were originally proposed by Han et al. in the context of Min-Apriori [289], an algorithm to deal with ordinal items. Steinbach et al. [601] introduced a framework to generalize this notion of support towards association analysis to continuous-based patterns. An alternative support metric [338] has been proposed to mine hyperclique patterns (perfect constant biclusters) over numeric matrices. Calders et al. [115] proposed three distinct rank-based measures to score the similarity of sets of numeric features. More recently, RAP algorithm [509] was proposed under a support metric with a sign-coherence constraint, enforcing that a transaction can only contribute to the support of a pattern if the values of all the items in it have the same sign. Given a real-valued matrix with $\mathbf{J} \subseteq \mathbf{Y}$, the support is defined as $sup(\mathbf{J}) = \Sigma_{\mathbf{x}_i \in \mathbf{X}} S(\mathbf{x}_i, \mathbf{J})$, with:

$$S(\mathbf{x}_i, \mathbf{J}) = \begin{cases} min_{\mathbf{y}_j \in \mathbf{J}} |a_{ij}| & \text{if } (\max_j a_{ij} - \min_j a_{ij}) \leq \sigma \min_j |a_{ij}| \wedge (\forall_j a_{ij} > 0 \vee \forall_j a_{ij} < 0) \\ 0 & \text{otherwise.} \end{cases} \quad (2.1)$$

Alternative range-based support metrics were proposed with RCB and ET-biclustering methods [29, 57] to, respectively, verify range constraints on both dimensions (constant values overall) and guarantee noise-tolerant counts.

Range-based support metrics need to be (anti-)monotonic in order to seize efficiency gains while offering guarantees of optimality. With the exception of the support metric from ET-biclustering, all remaining metrics are anti-monotonic. To illustrate this property, consider a support metric that verifies if the minimum of values associated with a set of elements is within a pre-specified threshold. This metric is anti-monotonic since the minimum absolute value of an element can only decrease if another item is added to it (q.e.d.).

Despite the relevance of range-based metrics, not only efficiency bottlenecks associated with candidate generation procedures become visible, but also specific principles to enhance the scalability of PM searches, as well as the possibility to discover condensed representations.

Multi-items assignment. A unique advantage of PM searches is their ability to learn from transactions (mapped from the rows of a discretized matrix) with an arbitrary number of items. This gives the possibility to assign multiple items to a single element of the input matrix. As such, elements with values near a boundary of discretization (or cut-off point) can be assigned to two items associated with the closest ranges of values. Under this simplistic mapping procedure, pattern-based biclustering can effectively address the items-boundary problem. *Basics 2.1* provides an illustrative application of this strategy, showing how sound solutions can be derived from transactional databases with a non-fixed number of items.

Basics 2.1 Multi-items assignments

To illustrate the inherent simplicity and soundness associated with multi-item assignments consider c_1-conditional observations of the real-valued data matrix $\mathbf{A}_1$ provided in Figure I-1.3, and a Gaussian discretization procedures with five items $\mathcal{L}$={a,b,c,d,e} corresponding to the following cut-off points: {a:[-4,-2[,b:[-2,-0.4[,c:[-0.4,0.4],d:]0.4,2],e:]2,4]}. Figures 2.2 and 2.3 show respectively the discretized matrix and the mapped transactional database with multiple items for the elements with values near discretization bounds. The application of frequent itemset mining over this transactional database (with minimum support θ=2) delivers $\mathbf{B}$=($\mathbf{I}$=$\{\mathbf{x}_2, \mathbf{x}_4\}$, $\mathbf{J}$=$\{\mathbf{y}_2, \mathbf{y}_3, \mathbf{y}_5, \mathbf{y}_7\}$) bicluster. Understandably, in the absence of multi-items assignment, only smaller biclusters, such as ($\mathbf{I}$=$\{\mathbf{x}_2, \mathbf{x}_4\}$, $\mathbf{J}$=$\{\mathbf{y}_3, \mathbf{y}_5\}$), could be discovered due to the inability to deal with noisy elements from an inaccurate discretization.

Figure 2.2: Discretization of matrix $\mathbf{A}_1$ using $|\mathcal{L}|$=5 and Gaussian cut-off points (items near boundary in bold)

	$\mathbf{y}_1$	$\mathbf{y}_2$	$\mathbf{y}_3$	$\mathbf{y}_4$	$\mathbf{y}_5$	$\mathbf{y}_6$	$\mathbf{y}_7$
$\mathbf{x}_1$	e	b	**e**	e	b	d	d
$\mathbf{x}_2$	d	**a**	e	c	a	c	**d**
$\mathbf{x}_3$	d	a	**d**	a	a	c	c
$\mathbf{x}_4$	a	**b**	e	**d**	a	b	**e**

Figure 2.3: Mapped transactional database with multi-items assignment (newly assigned items in bold)

	transactions
$\mathbf{x}_1$	$\{y_1e, y_2b, \mathbf{y_3d}, \mathbf{y_3e}, y_4e, y_5b, y_6d, y_7d\}$
$\mathbf{x}_2$	$\{y_1d, \mathbf{y_2a}, \mathbf{y_2b}, y_3e, y_4c, y_5a, y_6c, \mathbf{y_7d}, \mathbf{y_7e}\}$
$\mathbf{x}_3$	$\{y_1d, y_2a, \mathbf{y_3d}, \mathbf{y_3e}, y_4a, y_5a, y_6c, y_7c\}$
$\mathbf{x}_4$	$\{y_1a, \mathbf{y_2a}, \mathbf{y_2b}, y_3e, \mathbf{y_4d}, \mathbf{y_4e}, y_5a, y_6b, \mathbf{y_7d}, \mathbf{y_7e}\}$

2.3 Robustness to Noise

In the general research field of biclustering, some contributions have been proposed to calibrate the tolerated type and amount of noise per bicluster. The primary means to affect quality is to adjust the homogeneity criteria (Def.I-1.10) by controlling the merit function and its error-thresholds. In the particular case of pattern-based biclustering, preprocessing, mining and postprocessing options can be used to affect quality (*Section 2.1*). Consider the two following major forms of noise associated with a bicluster: 1) presence of a compact amount of highly noisy elements, 2) presence of a (possibly large) amount of elements with slight deviations on the expected values. Below, we describe strategies to handle these two forms of noise.

Association rules with parameterizable confidence levels, as well as *closing* options are used to guarantee the modeling/recovery of true positive elements and the exclusion of false positive elements. *Multi-item assignments* and *item-aggregation* strategies are used to guarantee the allowance of (possibly high) number of deviating elements. Naturally, all of these strategies can be used to minimize each one of these forms of noise. This division was suggested to better differentiate their nature. As we already discussed multi-item assignments in the previous section and closing options in the previous chapter, below we give further details on robust rule-based biclustering and describe item-aggregation strategy.

Rule-based Biclustering. An additional option to pattern-based biclustering is to derive biclusters from association rules. When using association rules to compose biclusters, the items on the antecedent and consequent of a rule, as well as the supporting transactions from both sides are considered, to derive each bicluster. Thus, association rules can be used to capture a form of local noise when confidence levels are below 100%, as illustrated in Figure 2.4. Confidence is thus seen as the homogeneity criterion to control the amount of allowed noise, and therefore the means to avoid the introduced noise dillema.

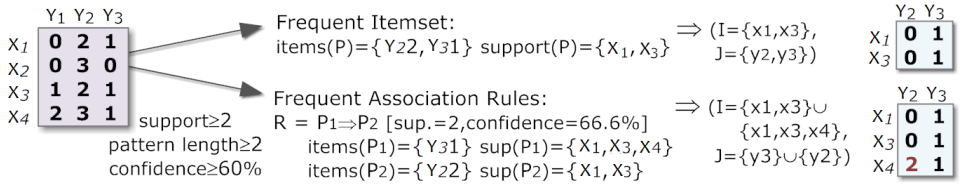

Figure 2.4: Discovering biclusters from association rules: comparing noise-intolerant biclusters from frequent itemsets vs. noise-tolerant biclusters from association rules.

To discover biclusters with varying quality, other interestingness metrics can be used to augment the support-confidence framework, including lift, conviction, χ^2, cosine and all-confidence [303], as well as criteria based on the statistical or domain-driven significance [20] using statistical tests on the correlation coefficients, such as Pearson's Product Moment, Spearman's Rank-order and Kendall's Tau. A unique property of noise-tolerant association rules is the fact the allowed noisy elements are coherently located within the bicluster. Therefore, their use is appropriate to deal with specific sources of variability such as plaid effects [303].

Item Aggregation. An alternative key direction for pattern-based biclustering is to consider multiple levels of tolerance to noise by hierarchically joining contiguous (items are viewed as being ordinal and no longer nominal).

Each joining procedures implies a reduction on the number of items and therefore one application is typically sufficient. Understandably, the level of noise should be maintained by each bicluster, so that closing steps can be applied yet still respecting the desirable quality criteria of the target biclustering model. Optimizations to this strategy can be made by collapsing items only for regions of the matrix where the presence of biclusters is scarcer.

A comparison of this strategy with other alternative for handling noise is provided in Figure 2.5.

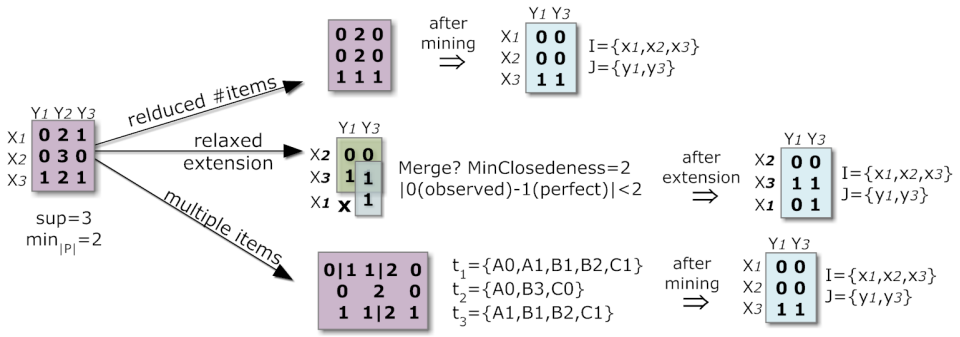

Figure 2.5: Strategies to deal with noise-relaxations.

2.4 Robustness to Missing Values

A third concern with impact on the robustness of biclustering solutions appears for matrices with an arbitrary-high number of missing elements. This is common across biomedical data domains. Although multiple imputation methods have been proposed [631, 183, 301] to alleviate this problem, they can introduce additional noise and undesirably affect the homogeneity of the output models. A non-treated (or erroneously imputed) missing value may result in the loss of a critical row and of a column for one or more biclusters, rapidly affecting solutions.

We propose four different strategies with varying relaxations to tackle this problem: *1)* imputation, *2)* removal, *3)* handling as a special value, and *4)* noise-tolerant imputation. First, a large portion of literature has been dedicated to define effective imputation methods based on the observed data regularities [631, 183, 301]. In fact, part of these contributions reside on the estimation of missing values based on biclustering [160]. The analysis of imputation methods is, however, out of the scope of this thesis. As discussed, although imputation can alleviate the problem, it is insufficient for its minimization, preventing the delivery of optimality guarantees.

Second, the removal of the missing value by mapping the matrix into a transactional database where transactions may have less than m items when the associated observation has one or more missing values. Understandably, such option is not supported by other biclustering approaches. For this aim, they either remove the containing row or column (usually the dimension with smaller size). Understandably, the performance in these contexts rapidly degrades for a low-to-medium number of missings, which contrasts with the alternative simplistic option of removing the single element. However, this second strategy does not also minimize the risk of excluding relevant elements (false negatives). Nevertheless, it is useful to compare solutions learned in the absence of missing values (baseline scenario) against more robust solutions.

Third, and also for exploratory purposes, a missing value can be replaced by a special item. This allows the identification of coherent biclusters of missing values [160] and of their inclusion in biclustering models when they consistently appearance on a column across a subset of rows.

Finally, we propose the replacement of the missing value by a parameterizable number of items (or ranges of values) around its value-estimation according to a level of relaxation (possibly) defined by the user. The lowest constrained setting (*relaxed*) replaces the missing item by all other adopted items. Similarly to multi-items assignments, this results in a database with transactions with varying size. The medium constrained setting (δ-*replace*) considers multiple items around its value-estimation. If the difference between the estimated value and the centroid-value of a discretization range is less than δ, then the item assigned to the range is added. Based

on empirical evidence, we suggest the value estimation according to the observed values for the nearest neighbor observations, as well as a lower bound of two items and an upper bound of three assignable items. This strategy, illustrated in Figure 2.6, guarantees robustness against missings.

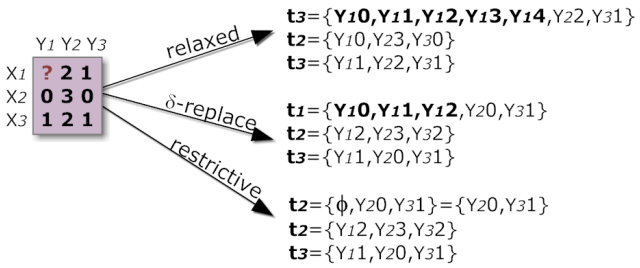

Figure 2.6: Proposed mapping methods to handle missings: relaxed, conservative (*δ-replace*) and restrictive alternatives to imputation. A missing value is represented as ?.

2.5 Customizing Biclustering Structures

Up to this point, we introduced several principles to adequately parameterize pattern-based biclustering approaches and to further guarantee their robustness. These principles can be used to affect the underlying biclustering structure for specific purposes. Illustrating, consider that we want that every observation belongs to at least one bicluster for the purpose of unsupervised categorization/labeling of observations. Understandably, pattern-based structures cannot offer that guarantee and therefore new principles need to be considered.

A biclustering structure can be classified based on whether it is exclusive and/or exhaustive on rows and/or columns, and on whether it places non-overlapping and/or shape-specific restrictions. A structure is exclusive on rows (columns) when a row (column) is not assigned to more than one bicluster, $\cap_{k=1}^{p} I_k = \emptyset$ ($\cap_{k=1}^{p} J_k = \emptyset$); and exhaustive on rows (columns) if each belongs to at least one bicluster, $\cup_{k=1}^{p} I_k = X$ ($\cup_{k=1}^{p} J_k = Y$). The exhaustive criterion of a structure should not be confused with the exhaustive criterion of a search. Besides the non-overlapping restriction ($\cap_{k=1}^{p}(I_k, J_k) = \emptyset$), other restrictions, such as chessboard structures and hierarchical division of the matrix accordingly to incrementally smaller biclusters (totally contained in a larger bicluster) have been proposed [429]. In Figure 1.2, some of these structures were illustrated.

Pattern-based structures are non-exhaustive, non-exclusive and allow for overlaps. The flexibility of pattern-based solutions offer an adequate basis to compose additional structures of biclusters. Nevertheless, the task of composing different structures has been poorly addressed in literature and rather seen as the byproduct of the considered biclustering algorithm [429]. Below, we briefly present principles to customize biclustering structures.

If an *exhaustive* structure (either overall, across rows or across columns) is targeted [429, 298], two alternative strategies can be followed. First, biclusters can be incrementally extended and merged following, for instance, a hierarchical criterion based on the proximity and the area of biclusters, until all the rows or/and columns of the matrix are covered. An alternative strategy is to extract exhaustive structures of clusters from both dimensions of the input matrix, and use them to guide the extension of the discovered biclusters. Exhaustive structures are relevant in social domains to guarantee that, for instance, users in a social network and collaborative filtering context are associated with at least one bicluster that is indicative of their activity and preferences (profile). When the extraction of a single profile per observation is sufficient or desirable, exclusive structures can be considered. To produce an *exclusive* structure (either overall, across rows $\Sigma_k a_{ik} = 1$ or across columns $\Sigma_k a_{kj} = 1$) [429, 617], biclusters can be iteratively merged to reduce the overlap across one or both dimensions and, additionally, the shared rows (columns) filtered according to a specific criterion (as size or noise level) until exclusivity is guaranteed. Finally, the closing options can be alternatively applied to produce alternative structures, such as tree-based and hierarchical-based structures surveyed in [429]. A key property of these principles is that their application is independent from the core behavior (mining step) of pattern-based approaches.

2.6 BicPAM Algorithm

To integrate the proposed principles retrieved and easily support new the incorporation of new principles, we propose a new pattern-based biclustering approach referred as BicPAM (Biclustering based on PAttern Mining). BicPAM is an ordered composition of mapping, mining (pattern discovery), and closing decisions, as Figure 2.7 illustrates. In what follows, *Section 2.6.1* lists the implemented procedures along these steps, and *Section 2.6.2* describes newly proposed extensions to guarantee that BicPAM can satisfy minimum expectations, such as a minimum number of (dissimilar) biclusters.

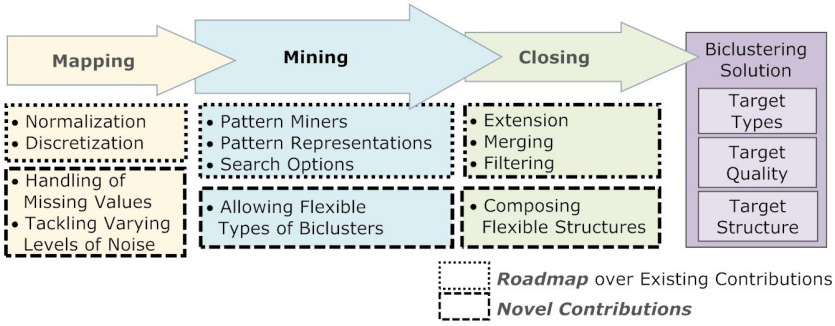

Figure 2.7: BicPAM's Methodology. BicPAM relies on three steps that determine the type, quality and structure of the biclustering solutions. Within each step, we make available principles based on existing contributions, and accommodate new contributions to deal with noise, flexible coherency assumptions, and customized biclustering structures.

2.6.1 Integrating Contributions

BicPAM uses frequent closed itemsets to accomplish the mining step. Range-based approaches are only selected for small-to-medium datasets. In the presence of domain knowledge (such as functional groups of genes or dependencies on conditions), BicPAM relies on association rules to compose biclustering solutions. The set of simple and maximal frequent patterns are also made available in BicPAM for user selection depending on the desirable structure and on the postprocessing needs. Illustrating, simple patterns increase the computational complexity of closing options, although their output can be used to guide the extension-and-reduction of biclusters and the customization of new structures. Finally, BicPAM makes available Apriori-based [9], pattern-growth [292] and vertical searches [701], whose selection depends on the properties of data and of the target models (see previous chapter). To support these mining options, BicPAM implements F2G [315] (algorithm proposed by the authors described in *Chapter 5*), Charm [702] (with the possibility to mine closed and maximal patterns using either bitsets or diffsets), AprioriTID [9], Eclat [701] and TNR [228]. BicPAM can also be parameterized with structured pattern mining since it additionally supports Bide+ [654], Clasp [259], CM-SPADE [222], PrefixSpan [523] and TNS [227].

For preprocessing, BicPAM makes available zero-mean normalization procedures (with the possibility to mask outliers and missing values) and distinct discretization procedures: fixed ranges, equal depth partitioning and statistical cut-off points of distributions. For this latter discretization option, BicPAM relies on statistical test to fit adequate distributions and relies on principles provided by SAX tool [407] in order to compute cut-off points from equally probable areas under the distribution curve.

For postprocessing, BicPAM replicates the merging principles provided by DeBi [578], filtering principles provided by BiModule [500], and makes available simple statistical tests to guide the extension and reduction of biclusters based on the observed gains for the extended mean-squared residue [690] (the homogeneity indicator). In order to increase the effectiveness of these options, while tackle efficiency bottlenecks it makes available new merging, extension and reduction procedures proposed by the authors and described in *Chapter 7*.

Finally, the robustness principles non-addressed by these mapping, mining and closing principles are also integrated in BicPAM. The assignment of multi-items and the handling of missings is done at the preprocessing stage (although it impacts the subsequent behavior). Finally, the recalibration of coherency strength is done through the iterative application of preprocessing and mining steps followed by a single postprocessing stage.

2.6.2 Stopping Criteria

A problem associated with existing pattern-based biclustering approaches is the fact that their application can either result in a very high number of (possibly similar) biclusters respecting a minimum support threshold, or, alternatively, not able to find biclusters. These situations can easily happen when the minimum support and pattern length thresholds are inadequately selected. However, their selection is not always as it does not only depends on the data size, dimensionality and global regularities but also by the presence of highly coherent regions. In this context, two major requirements need to be addressed. First, the adequate selection of the inputted thresholds for pattern-based biclustering. Second, that the learning task is not jeopardize by a large and highly coherent region, but it is able to adequately explore different regions of the data space and guarantee an adequate coverage.

To address this problem, BicPAM is iteratively applied with a decreasing support threshold until a stopping criteria is achieved [310]. BicPAM makes available distinct stopping criteria, including a minimum coverage of the elements in the input matrix by the discovered biclusters or, alternatively, an approximate number of biclusters. The coverage criterion allows an adequate exploration of the input matrix, and its minimum threshold can be dynamically derived from the properties of the input matrix (by default $0.1 \times n \times m$). The number of biclusters can be either tested after or prior to postprocessing and is commonly driven by user expectations (by default >5). Both criteria are tested after filtering biclusters non-satisfying structural constraints, such as a minimum pattern length.

2.7 Results and Discussion

In this section, we experimentally assess the performance of BicPAM and its relevance for real data analysis. The results were collected and analyzed in four steps. *Section 2.7.1* compares the performance of BicPAM against state-of-the-art biclustering approaches. *Section 2.7.2* assesses the proposed principles to guarantee robust models over synthetic data with varying properties. The biological relevance of BicPAM's outputs is analyzed in *Section 2.7.3*. BicPAM was implemented in Java (JVM version 1.6.0-24). The following experiments were run in an Intel Core i5 2.30GHz with 6GB of RAM.

Data Settings. Synthetic data was generated according to the properties proposed in *Chapter II-3*. In particular, and accordingly to Table II-3.1, we varied the size of the matrices, number and shape of biclusters (using Uniform distributions), and their coherency strength. To test robustness, we planted varying amounts of noisy elements and missing values, added noise factors to create slight-to-medium deviations on the observed values, and allowed for overlapping regions. To assess performance, 40 matrices were instantiated with background values following different distributions.

Evaluation Metrics. According to principles proposed in *Chapter II-2* (*Book II*), we rely on distinct performance views to measure the precision, coverage and overall similarity of the found biclusters $\mathcal{B}$, against hidden biclusters $\mathcal{H}$ – $MS(\mathcal{H},\mathcal{B})$, $MS(\mathcal{B},\mathcal{H})$, and $FC(\mathcal{B},\mathcal{H})$. In the presence of real data, term-enrichment tests are applied to assess their domain relevance.

Biclustering Approaches. We selected 10 state-of-the-art biclustering approaches: FABIA with sparse prior Equation [324], ISA [342], Bexpa [532], Cheng and Church (CC) [134], OPSM [59], OP-Clustering [415], BicSPAM [311] (*Chapter 6*), Samba [616], xMotifs [477] and BCPlaid [638]; and additional 3 pattern-based biclustering approaches: BiModule [500], DeBi [578] and RAP [509]. We used the following software: R packages fabia[1] and biclust[2] (to run BCPlaid), BicAT [45] (to run OPSM, ISA, CC and xMotifs), BicPAM[3] (to run BicPAM and additionally implemented pattern-based biclustering approaches: BiModule, DeBi and RAP), BicSPAM[4], BiP[5], (Evo-)Bexpa

[1] http://www.bioinf.jku.at/software/fabia/fabia.html
[2] http://cran.r-project.org/web/packages/biclust
[3] web.ist.utl.pt/rmch/software/bicpam
[4] web.ist.utl.pt/rmch/software/bicspam
[5] web.ist.utl.pt/rmch/software/bip

[532] and Expander[6] (to run SAMBA). The specified number of biclusters for FABIA, Bexpa, ISA, CC and xMotifs (number of starting points) was the number of hidden biclusters plus 10%: $|\mathcal{H}|\times 1.1$. Note that this required specification can be used to guide the search space exploration against other biclustering approaches, and optimistically bias FC levels. The default number of iterations for the OPSM method was varied from 10 to 200 iterations. BicPAM was parameterized with default options: closed pattern representations, three distinct levels of coherency strength (given by $|\mathcal{L}| \in \{3,5,7\}$), iterative searches with decreasing coherency strength and stopped when the discovered biclusters covered a minimum area of the input matrix (>5%$\times|X|\times|Y|$), simple merging option (70% overlap) and filtering of biclusters overlapping with a larger bicluster on more than 70% of its elements. Additionally, two items were assigned to values near item-boundaries, leading to an increase in the size of transactions of 8-11%. The remaining methods were applied with default parameterizations.

2.7.1 Comparison of Biclustering Approaches in Synthetic Data

Biclusters with Varying Structures and Coherency Strength. Figure 2.8 compares the performance of BicPAMS algorithms over the synthetic data settings described in Table II-3.1, assuming a fixed coherency strength ($\delta=\frac{1}{5}$) and post-imputation of noise on the generated values (up to 15% of the input range of values). For this coherency strength, results show the important role of using exhaustive searches able to find biclusters with non-differential coherencies. This explains the better performance of pattern-based biclustering approaches in terms of MS($\mathcal{B},\mathcal{H}$) score (completeness/coverage) and also MS($\mathcal{H},\mathcal{B}$) score (correctness/precision). The exhaustive nature of BicPAM searches and its ability to rely on multiple discretization levels without risk of introducing noise (by assignment multiple items for values near ranges-boundaries) turns BicPAM the best performer. In particular, the discovered biclusters by BicPAM are nearly perfectly described by the hidden biclusters (highest MS($\mathcal{B},\mathcal{H}$)) and the all of the hidden biclusters can be mapped into a discovered bicluster (highest MS($\mathcal{H},\mathcal{B}$)). The performance of peer pattern-based methods is penalized when noise handlers and merging procedures are not consider. This results in the exclusion of certain rows due to the inability to model elements a_{ij} participating in overlapping regions or with a high degree of planted noise, as well as in the partitioning of biclusters, leading to a high number of smaller biclusters, which degrades the *FC* levels. Although the gains on the effectiveness of pattern-based methods come at a cost on their efficiency, they are still able to perform exhaustive searches in useful time for computationally complex settings. As FABIA, ISA and OPSM are primarily tuned to discover non-constant coherencies, high degrees of noise are associated with the discovered biclusters, penalizing their matching scores. Illustrating, the FC levels of OPSM are strongly penalized since OPSM tends to output a large number of biclusters with distinct shapes than the planted ones. Although all approaches are scalable for medium-sized matrices, efficiency deterioration is faster for OPSM, BicPAM and CC.

Figure 2.9 assesses the ability of pattern-based biclustering algorithms to discover planted biclusters with varying coherency strength (ranging from δ=0.1 to δ=0.5). With an increasing loose coherency strength, the probability of background values to form a non-planted bicluster is higher. For this reason, we introduced a weighting criteria ν to affect the average number of rows and columns, where ν=1 for $\delta=\frac{1}{5}$, ν=0.8 for $\delta=\frac{1}{10}$, ν=1.2 for $\delta=\frac{1}{4}$, ν=1.5 for $\delta=\frac{1}{3}$ and ν=2 for $\delta=\frac{1}{2}$. This is easily accomplished with BiGen by affecting the Uniform distribution of rows and columns with U($\nu\times$a,$\nu\times$b). Again, this analysis also confirms the superiority of pattern-based biclustering algorithms available in BicPAMS to guarantee noise robustness are employed. Understandably, the biclustering approaches better prepared to discover differential coherencies, such as SAMBA, perform better with a loose coherency strength. Similarly, approaches that tend to discover biclusters with arbitrary-high levels of noise, such as CC, are also less prone to errors when considering loose coherency strength. Contrasting with these sets of approaches, the performance of the remaining approaches is either preserved or even improved with an increased coherency strength. Understandability, a loose coherency strength (corresponding to a smaller number of symbols

[6] http://acgt.cs.tau.ac.il/expander

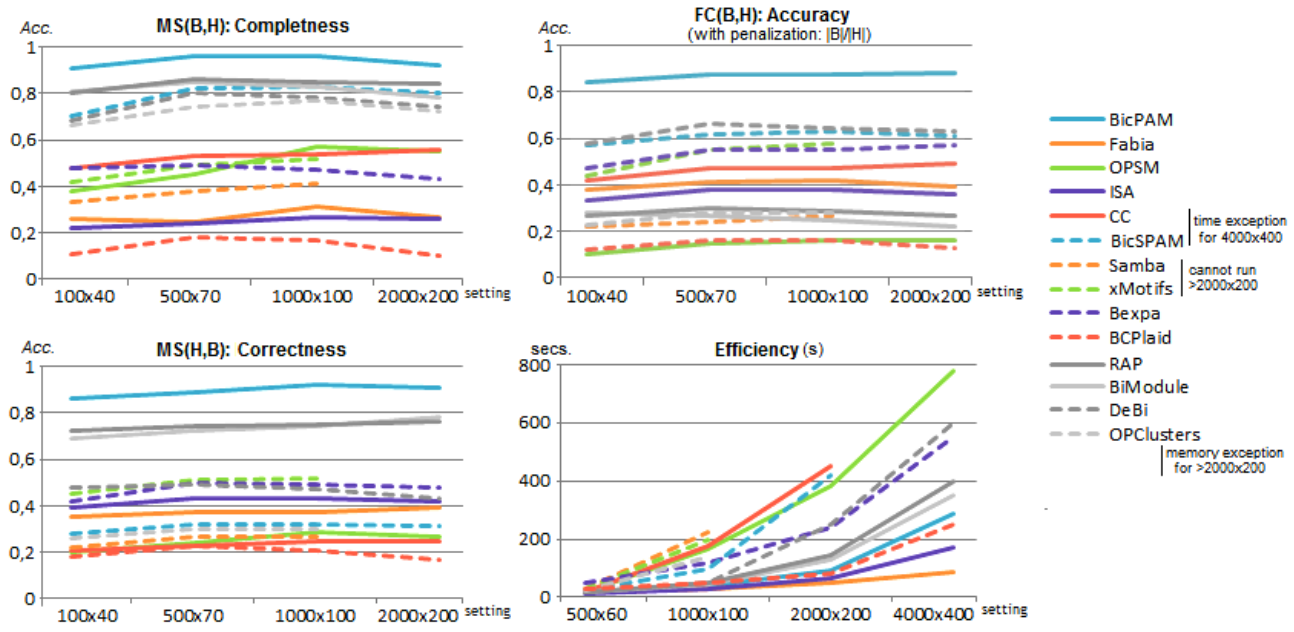

Figure 2.8: Comparing the performance of state-of-the-art biclustering approaches on data settings with varying properties and constant coherencies with fixed strength.

in symbolic settings) turns the matrix denser, decreasing the efficiency of the analyzed biclustering methods.

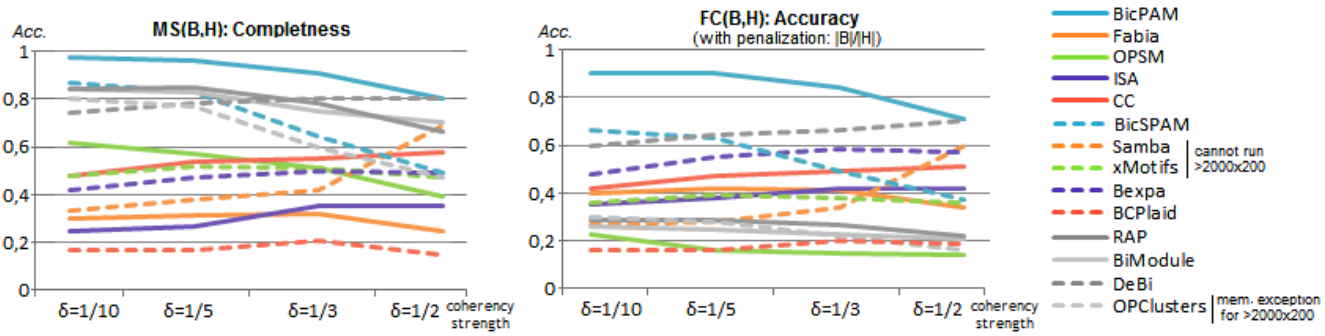

Figure 2.9: Performance of biclustering approaches on data with planted constant biclusters with varying coherency strength.

Biclusters with Varying Forms and Degree of Noise. In order to test the robustness of pattern-based biclustering methods, we generated data with: *1)* parameterizable percentage of noisy elements, and *2)* deviations on the generated values according to different distributions. For this analysis, illustrated in Figure 2.10, we considered the 1000×100 data setting with: *1)* a varying percentage of noisy elements (from 0 to 10%), and *2)* a varying degree of deviations on the background values given by Uniform distributions ranging from U(-$\frac{1}{2}\delta$,$\frac{1}{2}\delta$) to U(-2δ,2δ). The collected results support the superiority of BicPAM as it implements the listed principles to handle noisy elements, such as effective postprocessing procedures, and value-deviations, such as multi-item assignments. Generally, we can observe that, although biclustering approaches are able to deal with small portions of noise, their performance naturally degrades with an increasing level of noise. In pattern-based biclustering methods (BicPAM, BicSPAM, BiModule, RAP, DeBi), the penalization is greater for planted deviations on overall values than for planted noisy elements since these methods rely on postprocessing options well-prepared to recover noisy elements. Contrasting, the performance of the remaining biclustering methods tend to be more affected by the presence of elements with arbitrary-high levels of noise in the biclusters, and less penalized by deviations on the values since these deviations have a less impact on the homogeneity of the biclusters.

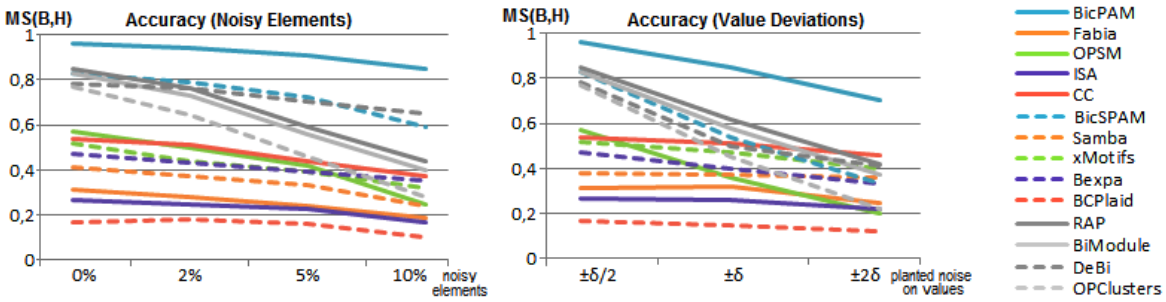

Figure 2.10: Performance of biclustering approaches over data with varying number of noisy elements and degree of deviations on the values from the expected coherency.

2.7.2 BicPAM Performance in Synthetic Data

In this section we start by studying the efficiency limits of BicPAM. Then we assess the ability of BicPAM to discover different types of biclusters for data with varying regularities. Finally, we go further on understanding the impact of using different strategies related with the mining, mapping and closing steps.

Efficiency Limits. To show the boundaries on BicPAM efficiency we considered matrices with high-dimensionality given by m=10.000 rows (magnitude of the human genome). The results are provided in Figure 2.11. We varied the number of rows, the coherency strength ($|\mathcal{L}| \in \{5,7\}$) and the impact of accommodating multi-item assignments and postprocessing options (merging and filtering procedures described in *Chapter III-7*) to guarantee robust solutions. We planted 10 biclusters to occupy 5% of the area of the generated matrices and used Charm algorithm [702], an efficient pattern miner to deliver closed patterns (maximal biclusters). Generally, we observe that BicPAM is able to discover constant biclusters for matrices with over 500 rows for the given high-dimensionality. Understandably the number of items has strong impact in efficiency as it determines the density of the correspondent transactional database and, therefore, the complexity of the mining step. Note, additionally, that the extensively studied scalability principles based on extensions over pattern mining methods – parallelization, distribution, streaming and error-bounding principles [290] – can be included in the mining step of BicPAM to guarantee its scalability over harder data settings. The efficiency gains from the use of such principles is out of this chapter's scope (see *Chapter III-5*).

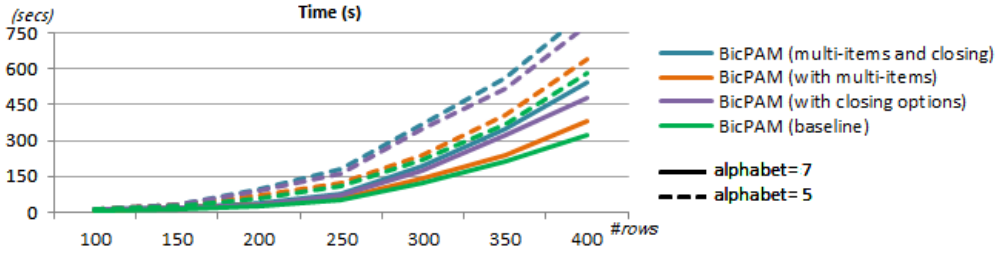

Figure 2.11: Efficiency bounds of BicPAM for 10.000 features/columns (magnitude of the human genome), with the disclosure of the impact of multi-item assignments and postprocessing options.

Handling Noise. We planted additional levels of *noise* to evaluate the closing options. This was performed by changing the values of specific elements by a randomly distant value (distance >25% of the domain range). The percentage of noisy elements was varied from 0 to 10%. We used the 1000×100 setting, Charm and a total of 10 items.

Figure 2.12a describes the impact of alternative strategies to *extend* biclusters. When no noise is planted, merging-based strategies are able to achieve slightly higher matching scores since they can cover elements originally missed due to discretization errors or by the allowed overlapping among planted biclusters. When increasing the planted noise, the presence of extension options is critical to maintain interesting accuracy levels. Both the inclusion of new rows and columns (recurring to statistical tests or by lowering the support of pattern miners) and

the merging of the resulting biclusters are able to maintain match scores above 90% (20 percentage points higher than the baseline option).

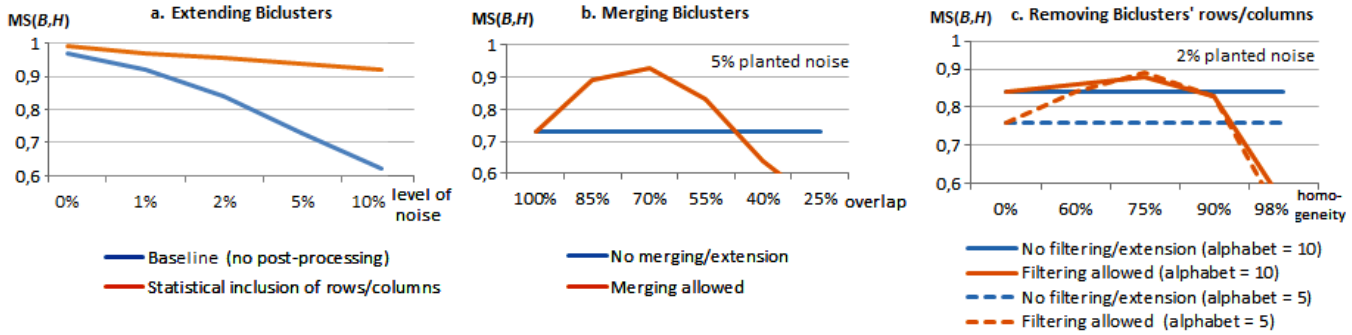

Figure 2.12: Impact of postprocessing procedures for the 1000×100 setting with planted noise:
a) Impact of extension procedures for biclustering data with varying levels of planted noise.
b) Impact of merging procedures when considering varying overlapping thresholds for data with 5% of planted noise.
c) Impact of filtering procedures when considering varying homogeneity criteria for data with 2% of planted noise.

Figure 2.12b illustrates the impact of *merging* biclusters with large overlapping areas assuming a level of planted noise of 5%. The baseline case corresponds to an overlapping area of 100%. When relaxing the overlapping criteria, $MS(\mathcal{B},\mathcal{H})$ (and also $MS(\mathcal{H},\mathcal{B})$) increases, as the merging step allows for the recovery of missing rows and columns. However, this improvement in behavior is only observable until a certain overlapping threshold (near 70% for this experimental setting). Match scoring decreases below this threshold. A correct identification of the optimum threshold can lead to significant gains (near 15 percentage points for this experimental setting).

Finally, the use of *reduction* procedures can also lead to an enhanced ability to recover the planted biclusters. Although the reduction of biclusters with weak homogeneity impacts accuracy, this analysis targets the removal of rows and columns (on each bicluster) that do not satisfy a specific homogeneity threshold. Figure 2.12c illustrates the impact of removing potentially false rows and columns assuming a level of planted noise of 2%. The impact is only significant when considering a low-to-medium number of items, since for these cases filtering is able to correct the errors related with the large ranges of values per item that lead to false biclusters. Similarly to the merging option, an increase in the matching score is observed when compared to the baseline case (an homogeneity degree of 0%) up to 75%, given by 1–*MSR* [134]. From this upper threshold the match scores decrease since the homogeneity criteria becomes too restrictive.

Handling Missing Data. In order to assess the impact of the proposed mapping strategies to handle *missing values* (Figure 2.6), we randomly removed a varying number of elements from the generated matrices for the 1000×100 setting. Figure 2.13 illustrates how the performance of BicPAM (using Charm and 10-item discretization) varies with a percentage of missings ranging from 0 to 10% (that is, from 0 to 10.000 elements). Note that 10% is already considered a very critical number of missings that may compromise the ability to retrieve the true biclusters. We observe that this problem can be mitigated recurring to the proposed BicPAM missing handlers.

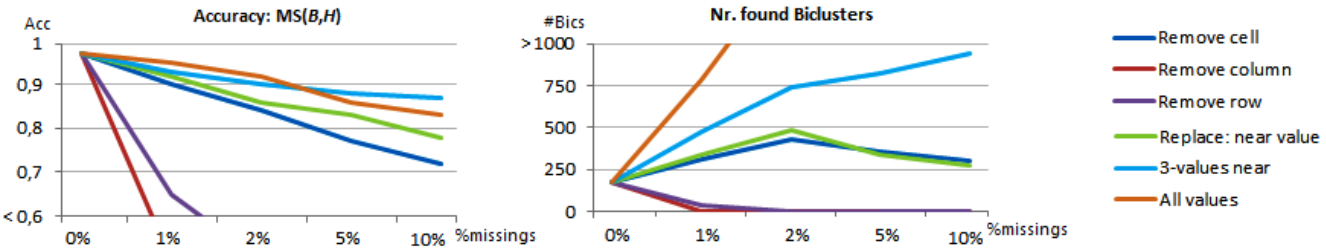

Figure 2.13: Comparing the handling of missings for data with varying levels of noise.

When analyzing the results in Figure 2.13, three observations can be retrieved. First, $MS(\mathcal{B},\mathcal{H})$ under the baseline strategy (remove the missings) significantly decreases from 97% to near 70% when the percentage of missings

reaches 10%. Although this solution is easily implemented in BicPAM (removing an element from respective transactions), the majority of existing biclustering algorithms only allow for removals on the columns or the rows where a missing occurs (impracticable even in the presence of a few missings as illustrated). Second, the ability to retrieve the planted biclusters increases when considering the nearest 2-3 values against the strategies that consider the closest value only or all the possible values (relaxed strategy). This is justified by two factors: *1)* when estimating more than one value for a missing, there is an increased chance to recover the original value and, therefore, of not damaging a planted bicluster; *2)* when considering all the possible values for a missing, there is an increased amount of noise that is added and can lead to the emergence of false biclusters. Third, although inserting multiple values to replace a missing is an attractive option in terms of accuracy, its efficiency is penalized as the itemized matrix becomes denser (consistent with the number of discovered biclusters). Still, when considering only the closest 2-3 values, scalability is maintained for levels of noise up to 10%.

2.7.3 Results in Real Data

To assess the performance of BicPAM in real data, we compared the biological significance of BicPAM's solutions against state-of-the-art biclustering solutions using three distinct gene expression datasets[7,8]: *1) dlblc* dataset (660 genes, 180 conditions) to study responses to chemotherapy [560], *2) hughes* dataset (6300 genes, 300 conditions) to characterize nucleosome occupancy [395], and *3) gasch* dataset (6152 genes, 176 conditions) to measure Yeast responses to environmental stimuli [239]. For the *gasch* dataset, we considered the multiple time points per condition and averaged the replicates of the steady state. The missing values were not removed since BicPAM can cope with them. For the state-of-the-art biclustering approaches, we maintained the parameterizations used in the previous section. In particular, pattern-based approaches were parameterized with multiple levels of expression ($|\mathcal{L}|\in\{4..7\}$). The selected closing options were: merging (70% overlap); relaxed merging (55% overlap) with reduction on rows; and tight merging (90% overlap) with extensions on rows that appear in another bicluster sharing a minimum 50% of the conditions.

According to principles introduced in *Chapter II-2*, the biological relevance of the biclusters from the different biclustering solutions was obtained accessing the over-representation of biological ontology terms using hypergeometric tests. A bicluster is considered significant if it shows enrichment in one or more of the "biological process" terms by having a (Bonferroni corrected) p-value below 0.05. In what follows, we analyze the gathered results according to their functional enrichment and transcriptional regulation.

Functional Enrichment. Table 2.1 provides a compact view on the biological significance of the compared approaches. BicPAM is able to discover the largest number of (non-similar) biclusters with significantly enriched terms for each dataset. The analysis of these terms against the significant terms found in other biclustering solutions shows the completeness of BicPAM's solutions (as they cover the majority of the gathered biological functions per dataset), together with the exclusivity and relevance of BicPAM solutions (as they model biclusters with significantly enriched GO-terms that are not discovered by the remaining approaches). Although peer pattern-based solutions also find a large number of biclusters with significantly enriched terms, these terms have lower significance. This is partly due to the fact that these approaches do not provide noise-correction procedures to adequately handle the item-boundaries problem and different forms of noise. Remaining biclustering solutions are associated with incomplete sets of GO-terms since their algorithms are not able to deliver flexible structures with tolerance to noise. Moreover, some of these approaches are neither able to discover biclusters with varying coherency strength nor postprocess the raw biclustering solutions. Still, some of the compared approaches were able to deliver a few small biclusters whose terms are more significant than those found with BicPAM. Subsequent analyzes (Tables 2.2-2.4) provide further empirical evidence for the relevance, completeness and exclusivity of BicPAM solutions.

[7] http://www.bioinf.jku.at/software/fabia/gene_expression.html
[8] http://chemogenomics.stanford.edu/supplements/03nuc/datasets.html

Dataset	*Approach*	♯*Bics*	*Avg.*♯*Genes* ×♯*Conds*	♯*Bics sig. enriched*	*Coverage and exclusivity of enriched GO terms*
dlblc (human genome)	BicPAM	56	83×7	43 (77%)	Highest number of exclusively enriched terms (partial list in Table 4).
	BiModule	322	62×4	79 (25%)	Absence of closing options leads to redundant and less significant terms.
	DeBi	31	73×6	21 (68%)	Loss of relevant terms due to the inability to discover all maximal biclusters.
	CC	10	41×33	5 (50%)	Exclusive bicluster related with circulatory & cardiovascular system development.
	ISA	72	23×8	8 (11%)	Exclusive bicluster for extracellular structure organization and heparin binding.
	Plaid	3	12×49	1 (33%)	Majority of genes modeled in a single background bicluster with general terms.
	Fabia	10	79×35	6 (60%)	Small bicluster with superior enrichment of antigen binding functions.
	Bexpa	10	16×87	2 (20%)	Small sets of genes supported by large number of conditions.
	Samba	100	17×6	18 (18%)	Dedicated terms for antigen processing, peptide cross-linking and disassembly.
	OPSM	12	128×5	5 (42%)	High variance of ♯genes and ♯conditions; some of the biclusters with low ♯genes (coherency across high ♯conditions) have exclusive significantly enriched terms.
hughes (yeast genome)	BicPAM	47	360×7	38 (81%)	Exclusive enriched terms due to flexible coherency and post-processing criteria.
	BiModule	219	285×4	43 (20%)	Terms with lower sig. than terms from noise-tolerant BicPAM solutions.
	DeBi	28	317×7	21 (75%)	Terms observed across very small sets of conditions (≤5) are not enriched.
	CC	10	228×58	6 (60%)	GO terms covered by BicPAM constant biclusters.
	ISA	8	120×4	5 (63%)	Small biclusters with exclusive significance GO terms: spindle pole and karyogamy.
	Plaid	8	78×39	3 (38%)	One bicluster with higher significance for fungal-type cell wall assembly.
	Fabia	10	210×49	5 (50%)	Higher significance observed for actin cortical patch and oxidoreductase GO-terms.
	Bexpa	72	42×49	1 (10%)	Low number of enriched terms (probably due to the low ♯genes per bicluster).
	Samba	120	18×9	11 (9%)	Enriched terms covered by pattern-based biclustering solutions.
	OPSM	6	531×4	3 (50%)	Exclusive bicluster for the negative regulation of metabolic processes.
gasch (yeast genome)	BicPAM	149	411×8	123 (83%)	Large diversity of highly significant GO-terms (partial list in Table 4).
	BiModule	653	287×4	159 (24%)	Large but incomplete set of GO-terms as it excludes non-constant biclusters.
	DeBi	82	310×6	61 (74%)	Significance of terms slightly differ than BicPAM due to the handling of noise.
	CC	10	203×79	7 (70%)	Enriched terms appear in BicPAM solutions with higher significance.
	ISA	23	292×22	18 (78%)	Enriched terms covered by pattern-based biclustering solutions.
	Plaid	6	48×12	3 (50%)	Biclusters (apart from background layer) with lower enrichments than peers.
	Fabia	10	310×41	8 (80%)	Bicluster with higher sig. for specific proteasome complexes.
	Bexpa	10	63×29	3 (33%)	The few biclusters with deviation in size (higher ♯genes) are significant.
	OPSM	16	212×8	11 (69%)	One bicluster with higher significance for pre-ribosome functions.

Table 2.1: Comparing the biological relevance and novelty of different biclustering solutions.

Table 2.2 shows the number of biologically significant biclusters found by BicPAM when using closing strategies. In this analysis, a bicluster is considered to be highly significant if it has at least one enriched term with a corrected p-value below 0.01. To complement this analysis, Table 2.3 lists some of the most significant biological processes associated with these enriched terms for each data setting[9].

Dataset	Closing option	♯Bics	Avg. Area	♯Filtered bics	♯Sig. bics	♯Highly sig. bics
dlblc	merging	4803	81×7	28	22	5
	relaxed merging + reductions	980	83×9	24	19	3
	tight merging + extensions	7652	79×6	27	25	2
hughes	merge	6311	432×6	36	19	12
	relaxed merging + reductions	1259	492×7	22	12	8
	tight merging + extensions	9210	398×5	39	22	11
gasch	merge	27031	392×8	89	66	12
	relaxed merging + reductions	2177	486×11	67	49	11
	tight merging + extensions	52123	367×7	92	79	9

Table 2.2: Summary on the biological relevance of BicPAM's biclusters.

Transcriptional Regulation To complement the results on functional enrichment, we analyzed the highly enriched transcription factors (TFs) using the TFCONES database [394] (human genome) and Yeastract database [620] (yeast genome) using a corrected hyper-geometric statistical test.

The TFs associated with BicPAM's biclusters for the human and yeast genome are provided in Table 2.4. In this analysis we retrieved the TFs that are more *representative* – high coverage of the genes in the biclusters – and *significant* – high functional enrichment (p-value<1E-3) – for each one of the twenty five distinct biclusters disclosed in Table 2.3 associated with the *dlblc* and *gasch* dataset. In line with the goal of the these experiments [560, 239], we observe that the identified TFs are either directly or indirectly related with the responses to chemotherapy

[9]List of genes available in http://web.ist.utl.pt/rmch/software/bicpam/

Dataset	ID	Terms	Bicluster with best p-value	♯Genes
dlblc	Dl1	translational elongation; cytosolic part; translational initiation	4.49E-5	81
	Dl2	Golgi apparatus; MHC protein complex	5.40E-5	83
	Dl3	defense response; receptor activity; single organism signaling; vacuole; cell communication	4.91-5	162
	Dl4	immune response; response to interferon-gamma	1.06E-4	58
	Dl5	immune system process	1.27E-4	52
	Dl6	response to interferon-gamma; cellular response to chemical stimulus; response to cytokine stimulus	0.001	60
	Dl7	membrane-enclosed lumen; cell division; cell cycle process	2.92E-12	81
	Dl8	small molecule binding; catalytic activity; cell cycle process	6.14E-8	108
hughes	H1	mitochondrion organization; organellar ribosome; mitochondrial matrix; mitochondrial translation	2.70E-39	416
	H2	cell periphery; cell wall constituent; oxidoreductase activity; cell wall organization; sexual sporulation	1.73E-4	370
	H3	ribonucleoprotein complex biogenesis; nucleus	3.61E-30	426
	H4	cellular amino acid metabolic/biosynthetic process; carboxylic acid metabolic/biosynthetic process	1.3E-25	581
	H5	organonitrogen compound metabolic process; sulfur compound metabolic process	1.62E-4	504
	H6	macromolecular complex; intracell. non-membrane-bounded organelle; membrane-enclosed lumen	4.80E-14	512
gasch	G1	nitrogen compound metabolic proc.; carboxylic/organic amino acid processes; structural cytoskeleton	1.84E-16	434
	G2	cellular carbohydrate metabolic process; cytoplasm	2.01E-7	265
	G3	generation of precursor metabolites and energy; tricarboxylic acid cycle	1.16E-14	954
	G4	endomembrane system; retrotransposon nucleocapsid; pore; viral procapsid maturation	4.34E-6	102
	G5	nucleolus; ncRNA metabolic process	1.03E-61	611
	G6	intracell. non-membrane-bounded organelle; structural molecule activity	5.33E-76	293
	G7	cytosolic part; ribosomal subunit	1.61E-88	460
	G8	membrane-enclosed lumen; nuclear lumen; intracell. organelle lumen	1.17E-47	263
	G9	mitochondrion organization; mitochondrial part; cytoplasmic part; protein complex biogenesis	2.06E-26	592
	G10	cellular response to oxidative stress; generation of precursor metabolites and energy	2.37E-4	296
	G11	binding; nuclear part; preribosome	2.87E-11	508
	G12	cellular process involved in reproduction	0.001	435
	G13	macromolecular complex; cell part; structural molecule activity	6.05E-29	1442
	G14	vacuolar transport; chromosome	5.09E-7	606
	G15	regulation of cellular (macromolecule) biosynthetic process; protein modification process	2.28E-13	1019
	G16	organic substance catabolic process; carbohydrate metabolic process; cytoplasm	1.02E-15	648
	G17	ribonucleoprotein complex biogenesis (general)	1.08E-94	784

Table 2.3: Terms highly enriched in BicPAM's biclusters

(human) [564, 394] and stress conditions (yeast) [135, 620]. This analysis thus further supports the domain-relevance and adequacy of BicPAM.

Dataset	*Bic.ID* (Table 2.3)	*Highly enriched TFs*
dlblc	Dl1	BCL11A, LZTS1, GTF2I, HCLS1, HDAC1, MBD4, MEF2B, NCOA3, STAT6
	Dl2	ANP32A, HCLS1, IRF1, MNDA, NCOA1, RUNX3, STAT1, TRIM22, TRIP10
	Dl3	BCL3, TRIM22, ANP32A, ARID5B, CEBPB, CREG1, IRF1, PFDN5, STAT1
	Dl4	ANP32A, IRF1, NCOA1, STAT1, TRIM22
	Dl5	CREG1, IRF1, TRIM22, ANP32A, STAT1
	Dl6	ANP32A, IRF1, NCOA1, STAT1, TRIM22
	Dl7	BCL6, BCL6B, HIF1A, ILF2, POU2AF1, SERTAD1, TCF3
	Dl8	DR1, DRAP1, HIF1A, ILF2, NCOA3, SERTAD1, TMF1, ZNFN1A1
gasch	G1	Gcn4p, Sfp1p, Ace2p, Tec1p, Ste12p, Ash1p
	G2	Sfp1p, Msn2p, Bas1p, Tec1p, Sok2p, Abf1p, Ash1p, Cst6p
	G3	Sfp1p, Tec1p, Ste12p, Msn2p, Bas1p, Sok2p, Msn4p, Gcn4p
	G4	Snf6p, Tec1p, Ste12p, Rap1p, Sin4p, Abf1p, Snf2p, Ash1p
	G5	Sfp1p, Ace2p, Cst6p, Tup1p, Msn2p, Spt10p, Spt20p
	G6	Hsf1p, Spt23p, Mga2p, Sfp1p, Spt10p, Msn2p, Gcr1p, Gcn4p
	G7	Sfp1p, Swi5p, Tup1p, Spt10p, Spt20p, Gcr1p, Sin3p, Mga2p
	G8	Sfp1p, Swi5p, Cst6p, Tup1p, Spt20p, Ash1p, Spt10p
	G9	Yap1p, Ace2p, Sfp1p, Msn2p, Ash1p, Msn4p, Abf1p
	G10	Sfp1p, Msn2p, Msn4p, Cst6p, Abf1p, Sok2p, Bas1p
	G11	Snf6p, Tup1p, Snf2p, Cst6p, Sin4p, Rap1p, Swi3p, Hap2p
	G12	Yap1p, Tec1p, Msn2p, Msn4p, Ste12p, Sok2p
	G13	Snf6p, Tup1p, Abf1p, Snf2p, Cst6p, Sin4p
	G14	Sfp1p, Tec1p, Ste12p, Bas1p, Sok2p, Yrm1p
	G15	Ace2p, Sfp1p, Tec1p, Ste12p, Ash1p, Bas1p, Gcn4p, Sok2p
	G16	Cin5p, Gcn4p, Msn4p, Sfp1p, Msn2p, Tec1p, Ste12p, Sok2p
	G17	Sfp1p, Ace2p, Cst6p, Snf6p, Rap1p, Tup1p, Spt10p, Swi5p

Table 2.4: Analysis of TFs of the putative regulatory modules given by the BicPAM's biclusters provided in Table III-2.3 for the human genome (*dlblc* dataset) and the yeast genome (*gasch* dataset).

Consider the enriched TFs provided in Table 2.4 for a sample set with 8 distinct biclusters found by BicPAM in the *dlblc* dataset. Different groups of TFs were identified, each associated with a specific chemotherapy outcome. Some of the TFs acting as putative tumor suppressors include: ANP32A, LZTS1 (protein-coding silenced in rapidly metastasizing and metastatic tumor cells), RUNX3 (protein that binds to the core site of leukemia virus, also frequently silenced in cancer), HCLS1 (antigen receptor signaling deletion in lymphoid cells), IRF1 (protein that stimulates

immune responses and regulates tumor cell differentiation), HIF1A (gene responsible for tumor angiogenesis and pathophysiology of ischemic disease), HDAC1 (complex interacting with retinoblastoma tumor-suppressor proteins), TCF3 (protein regulating lymphopoiesis as its deletion is associated with lymphoblastic and acute leukima malignancies) [564, 394]. Other TFs dedicated to regulate cell proliferation include the STAT families, CREG1, MEF2B, ARID5B, and BCL3 [564]. We also observed the B-cell lymphoma protein (BCL6 and its paralog coding gene BCL6B) and other leukemia-related disease genes involved in lymphoma pathogenes, such as BCL11A [394]. Complementarily, immune responses are associated with TRIM22 antiviral proteins, CEBPB, NFATC2 complex, and GTF2I for activating immunoglobulin heavy-chain transcription upon B-lymphocyte activation [564].

Finally, consider the enriched TFs provided in Table 2.4 for a sample set with 17 distinct biclusters found by BicPAM in the *gasch* dataset. Since a large number of enriched TFs was identified, Table 2.4 only provides an illustrative set containing TFs regulating over 50% of the genes associated with each bicluster. Although the enriched TFs regulate very distinct processes (see Table 2.3), most TFs are activated in stress conditions, namely: Yap1p, Cin5p and Hap2p during oxidative stress [135]; Gcn4p, Msn2p and Msn4p during amino acid starvation [135]; Hsf1p during variable heat shock elements including hyperthermia [135]; Sfp1p during DNA damage [620]; and Spt23p and Mga2p during cooling [480]. The stress conditions are associated with invasive growth (regulated by Tec1p, Ste12p, Ash1p and Sok2p), and with the need for chromatin remodeling (regulated by Snf6p, Snf2p, Spt20p, Tup1p and Swi3p) and DNA repair (regulated for instance by Abf1p and Spt10p) [135, 620].

2.8 Summary of Contributions and Implications

In this chapter, a consistent set of principles was proposed to enhance pattern-based biclustering and guarantee their robustness to varying forms and amount of noise. We proposed the use of multi-item assignments to address the items-boundary problem and the drawbacks of existing imputation methods. The possibility to assign multiple items to a single element in the matrix was shown to be easily compliant with pattern-bassed biclustering without significantly affecting the computational complexity of the searches. We also showed the possibility of addressing the noise dilemma by adequately parameterizing pattern-based biclustering algorithms through mining options (search approximative patterns and association rules with varying confidence levels), mapping options (normalization, discretization and aggregation decisions), and closing options (minimum/maximum similarity for merging/filtering and desirable homogeneity for extension/reduction).

As a result, robust solutions of biclusters are learned under these contributions. Principles were further provided to customize the properties of the structure of biclusters in order to be able to map and answer a wider-range of problems using the biclustering task.

Finally, we proposed BicPAM to integrate dispersed contributions on pattern-based biclustering, and accommodate principles for handling noise, missings and discretization drawbacks. As a result, BicPAM provides exhaustive searches with flexible stopping criteria and guarantees of robustness. BicPAM relies on dynamic parameterizations for a tuned performance and adequate tolerance to noise across dependent on the input dataset. Nevertheless, the user can still flexibly parameterize BicPAM to customize the desirable solution's properties without the need to adapt BicPAM's behavior.

Results over synthetic and real data confirm the superiority of BicPAM against peers and its inherent ability to provide exhaustive yet efficient searches for biclusters with parameterizable quality. In particular, we demonstrate BicPAM's ability to deal with varying coherence strength and structures of biclusters, as well as its robustness to discretization problems (assessed against planted deviations on the values) and to varying amounts of noisy and missing elements. Biological analyzes from the application of BicPAM over expression data reveal its unique ability to find complete, meaningful and non-trivial biclusters with heightened biological relevance that could not be discovered by other biclustering algorithms. This analysis was further corroborated by their transcriptional regulation.

3

Additive, Multiplicative and Symmetric Models for Tabular Data

The proposed approaches for pattern-based biclustering are prepared to discover biclusters with constant values on rows (or columns). This is already an improvement in terms of coherency flexibility against biclusters with constant values overall. However, this form of coherency does not allow for the critical occurrence of shifting (or additive) and scaling (or multiplicative) factors on rows (or columns). Illustrating, two genes may be regulated in the same conditions but show different expression levels explained by a shifting or scaling factor, which can be associated with: the structural responsiveness of a gene, or the bias introduced by the applied measurement and normalization of a gene. These factors are also critical to analyze physiological and clinical data in order to handle the structural differences across individuals [122]. In social applications, additive and multiplicative models are relevant to model social interactions with coherent behavior but differing in the extent of popularity and frequency of actions, and to group subjects with identical variation of preferences for browsing, commercing and collaborative filtering [257].

Furthermore, the allowance for symmetries over a given coherency is critical to adequately model local regularities from biological and social data contexts. Similarly, pattern-based biclustering as-is does not support this form of coherency. In biological contexts, symmetries are key to simultaneously capture activation and repression mechanisms within biological processes associated with biclusters in transcriptomic, proteomic or metabolic data [429]. In social contexts, symmetries are used to capture opposed (yet correlated) regularities associated with trading, tweeting, browsing and (e-)commerce activity [311].

An additional problem associated with biclustering models with either constant or more flexible coherency is whether they can be applied to tabular data contexts. Tabular data contexts are characterized by a set of observations with values for a set of features with (possibly): 1) categoric or numeric domains, and 2) non-identical value distributions. Typically, medical data observations are a combination of nominal, ordinal and numeric features. Social and biological data observations are also commonly annotated with categoric features associated with subject's behavioral, demographic, psychological and physiological/health profile. Furthermore, the orientation of the majority of existing biclustering algorithms towards real-valued matrices with identically distributed columns (rows), together with the absence of principles in literature for their extension towards more complex data domains, stress the need to answer this problem.

This chapter addresses these three problems. As a result, four major contributions are provided. First, BicPAM algorithm is extended to exhaustively learn additive and multiplicative models, and enhanced with pruning strategies to guarantee the efficiency of the resulting searches. Second, BicPAM is further enhanced to combine constant and non-constant coherency assumptions with symmetries. Third, a set of principles are proposed to guarantee the applicability of biclustering algorithms in general, and BicPAM in particular, over tabular data. In particular, a novel view on mixtures of constant and non-constant coherencies is proposed when learning from data domains with numeric and categoric features. Finally, empirical evidence from real data is used to demonstrate the unique properties of BicPAM and its role to retrieve meaningful, significant and non-trivial modules.

The importance of these contributions is shown experimentally. Results show BicPAM's superiority against its

peers, its ability to retrieve unique types of biclusters of interest and to efficiently deliver exhaustive solutions.

Figure 3.1 illustrates the contents covered in this chapter. *Section 3.1* surveys major related work and its limitations. *Section 3.2* addresses these limitations by extending BicPAM to consider more flexible coherency assumptions (*Section 3.2.1* and *3.2.2*) and to be applicable over tabular data (*Section 3.2.3*). *Section 3.3* discusses major results from assessing the proposed contributions on synthetic and real data. Finally, we draw major conclusions and implications from this work.

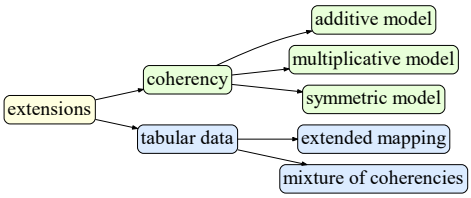

Figure 3.1: Pattern-based biclustering of additive, multiplicative, symmetric and mixture models.

3.1 Limitations of Related Work

3.1.1 Additive and Multiplicative Models

Some of the existing biclustering approaches claim to be able model additive biclusters include residue-based approaches [134, 690, 689], ISA [342], pClusters [653] and ITWC [617]), while others are positioned as being more prone to discover multiplicative biclusters such as Fabia [324]. Some attempts unify the seemingly incompatible additive and multiplicative models, such as approaches based on the matching of linear geometries in hyper-spaces [235], evolutionary computation with customizable pattern-based objectives [532], swarm intelligence [161], among others [14, 694, 133].

However, the surveyed approaches suffer from two drawbacks. First, they either place restrictions on structure, coherency strength and quality of biclustering models. Second, and more critically, empirical evidence from the application of these approaches [310] show that they have a high propensity to discover constant biclusters with arbitrary-high levels of noise, instead of planted biclusters with strict additive and multiplicative coherency [311]. In fact, most of them assume that a constant model on rows (columns) captures shifting and scaling factors, when this is not true. As such, there are not yet solid contributions to robustly model(iclusters with delineate additive and multiplicative coherency. *Basics 3.1* illustrates the relevance of the concept of shifting/scaling factors used in our dissertation and confront it with the concept implicitly considered by the majority of the surveyed approaches.

Basics 3.1 Additive/multiplicative coherency: contesting existing definitions

In Figure 3.2 we illustrate a bicluster with additive coherency on rows, and respectively distinguish between value differences on the observed pattern from value differences associated with shifting factors. Most of recent research defending to model shifts (or scales) are in fact simply modeling pattern differences (empirical evidence collected in results). The combined differences on the patterns with shifting (or scaling) factors are required across data domains, including to model: genes with different responsiveness, individual differences on physiological responses, distinct use of the rating scales for collaborative filtering, and different degree of user activity/involvement in social networks.

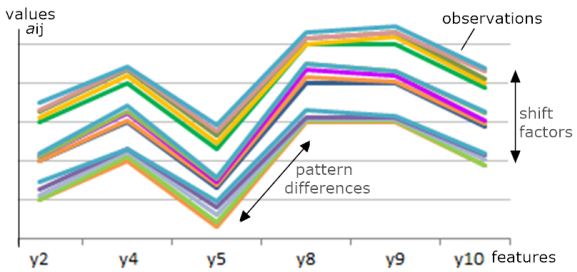

Figure 3.2: Illustrative bicluster with additive coherency on rows: differences associated with the pattern vs. shifts.

3.1.2 Symmetric Models

A critical, yet less studied, coherency variant is the symmetric model. Biclusters with the symmetric assumption are also referred as biclusters with sign-changes in literature [429]. Despite the existing contributions to find constant

biclusters with symmetries [616, 429, 380], they suffer from two major drawbacks. First, some approaches claim to be able to find symmetries by preserving zero-mean during preprocessing stage and by performing biclustering on the absolute values of the preprocessed matrix. Understandably, the discovery of biclusters in such matrices can easily lead to inconsistencies. Illustrating, given {2,2,3} and {-2,2,3} observations. Although these observations are neither constant nor symmetric, they would be considered coherent since the second observation would interpreted as {2,2,3}.

Second, to our knowledge there are no biclustering algorithms allowing the discovery of non-constant coherencies with symmetric behavior. For instance, $\mathbf{x}_1$={1,-2,2} and $\mathbf{x}_2$={-2,3,-3} observations cannot be explained by any of the previous coherencies, yet are approximated by an additive model (with γ_1=1 and γ_2=2) with a symmetry on the second observation. Similarly, the combination of symmetries with plaid and order-preserving models leads to functional coherent regions with relevance across different data domains (*Chapters III-4* and *III-6*).

3.1.3 Biclustering Tabular Data

Despite the relevance of existing principles for biclustering high-dimensional data, they have been proposed in the context of data domains with numeric and identically distributed features. This prevents the application of biclustering algorithms over a wide-set of clinical and social data domains, commonly characterized by a combination of ordinal and nominal features associated, for instance, with individuals profile and qualitative views of psycho-physiological measurements, preferences or (web) activity. In this context, only few attempts for applying biclustering on tabular data contexts have been proposed [309, 443, 181], all being dependent on domain-specific preprocessing procedures. In this context, the major challenges associated with this task are four-fold: learning from (numeric) features non-identically distributed, ordinal features, categoric features with imbalanced cardinality, and data domains with mixtures of numeric and categoric features.

3.2 Solution

BicPAM is extended below to use more flexible coherency assumptions and be applicable over tabular data.

3.2.1 Additive and Multiplicative Models

An *additive model* defines a set of biclusters, each with values $a_{ij} = c_j + \gamma_i + \eta_{ij}$, where c_j is the expected value for column $\mathbf{y}_i \in \mathbf{I}$, γ_i is the shifting factor on row $\mathbf{x}_j \in \mathbf{J}$ and η_{ij} is the structural noise. A *multiplicative model* is considered when biclusters are better approximated by $a_{ij} = c_j \times \gamma_j + \eta_{ij}$, where $c_j = logc'_j$ and $\gamma_i = log\beta'_i$ (scaling factor). Consider $\mathcal{R}$ to be a finite set of numeric values, and that the result from additive and multiplicative operators on $\mathcal{R}$ has a bijective correspondence to a set of items $\mathcal{L}$. A discrete bicluster with additive or multiplicative coherency assumes $c_j \in \mathcal{R}$ and $\gamma_i \in \mathcal{R}$. We propose two pattern-based strategies for the discovery of these models. The first strategy is to use local normalization procedures to correct row- or column-based differences and then to map the task into the search for constant biclusters.

The second strategy, the default BicPAM option, is to iteratively perform adjustments based on the observed values on each column (row) for coherency on rows (columns). This guarantees that all the adjustments needed to compose these biclusters are considered. Therefore, the selected pattern miner is applied either m (or n) times, leading to a higher computational complexity. Figure 3.3 illustrates this strategy.

An additive adjustment over a target column $\mathbf{y}_j$ can be computed by adding for each element on the row $\mathbf{x}_i$ the difference between the maximum of the column and the observed value on the row $max(\mathbf{y}_j) - a_{ij}$.

A multiplicative adjustment over a target column y_j can be computed by adding, for each element on the row $\mathbf{x}_i$, the least-common-multiple between the maximum of the column and the discretized value $lcm(max(\mathbf{y}_j), a_{ij})$. The resulting number of items under an additive assumption is in the worst case the double of the number of items initially considered. The number of final items under a multiplicative model is the size of the least-common-multiple

combinations across the initial items. As illustrated in Figure 3.3, a distance-based δ-error can be considered to gather close items in the multiplicative model due to the lower probability of finding coherent biclusters as a consequence of the resulting large number of items.

To enhance the performance of these searches, adjustments per column can be preserved in order to detect and remove repeated adjustments. Additionally, heuristics based on the similarity of a candidate adjustment against previous adjustments can be made to promote further efficiency gains.

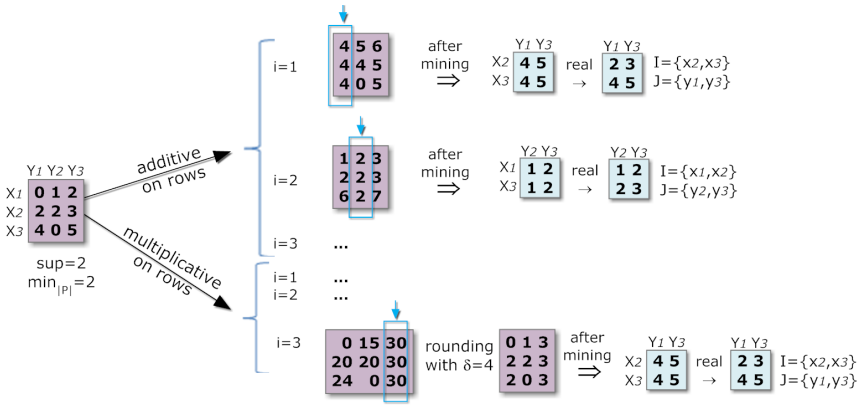

Figure 3.3: Pattern-based discovery of biclusters under additive and multiplicative assumptions.

3.2.2 Symmetric Models

A *symmetric model* assumes biclusters $(\mathbf{I}, \mathbf{J})$ accomodate symmetries on rows (columns) with elements being described by $c_j \times c_i \times a_{ij}$ where $c_j{=}1 \wedge c_i \in \{\text{-}1,1\}$ when a_{ij} has coherency on rows ($c_j \in \{\text{-}1, 1\} \wedge c_i{=}1$ for column-based coherency). Similarly to previous models, iterative adjustments can be performed per column, each one associated with the (possible) correction of the sign across rows (columns) in order to guarantee consistency of signals for a given column (row). The top example in Figure 6.6 illustrates this strategy. Additional efficiency can be achieved by stopping the search when all the sign combinations have been achieved. Assuming an exhaustive search, the worst case complexity requires the application of a pattern miner $min(m, \binom{n}{2})$ times for row-based coherency ($min(n, \binom{m}{2})$ times for column-based coherency). Nevertheless, the proposed heuristics for previous models are also applicable for symmetric models.

For the purpose of finding biclusters with symmetries, the normalization should satisfy the zero-mean criterion, and post-filtering should be considered to remove potential duplicates resulting from repetitions of alignments for subsets of rows (or columns). Additionally, if the number of considered items is odd, there is one item being its own symmetric that must be specially handled. This may imply two adjustments per column, which may slightly penalize efficiency levels.

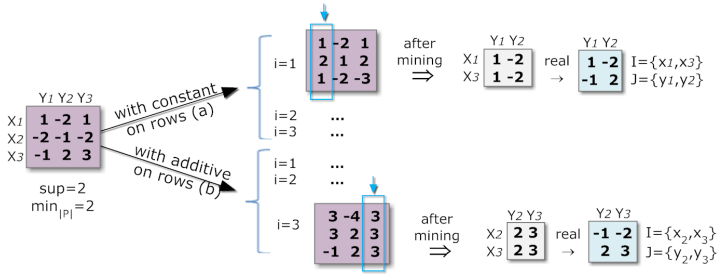

Figure 3.4: Pattern-based discovery of biclusters with symmetries for constant and non-constant biclusters.

The combination of this strategy with the search for biclusters under an additive or multiplicative model can be expensive ($m{\times}m$ times iterations). Therefore, BicPAM makes available an additional option to combine the use of the sign and of the additive or multiplicative adjustments together for every column (or row). This model (combined sign and coherent model) is not equivalent to the previous model (sign plus coherent model), since it

3.3 Results and Discussion

In this section, we experimentally assess the performance of BicPAM with flexible coherency assumptions and its relevance for real data analysis. *Section 3.3.1* analyzes the BicPAM's behavior against state-of-the-art biclustering approaches in synthetic data. The biological relevance of the flexible solutions learned by BicPAM are analyzed in *Section 3.3.2*. Finally, *Section 3.3.3* provides a synthetic view on the gains of BicPAM against baseline pattern-based biclustering. Experiments were run in an Intel Core i5 2.30GHz with 6GB of RAM.

BicPAM output include constant, additive, multiplicative and symmetric biclusters, discovered under different closing options. Finally, *Section 4.4* goes further on comparing BicPAM and its pattern-based peers. Below, we describe the evaluation metrics and data settings used.

Data Settings. We preserved the properties of generated data settings used in previous chapter, with the exception that the coherency assumption associated with the generated biclusters now follow additive, multiplicative and symmetric models and integrative models (solution where biclusters may follow different coherency assumptions). Additionally, to further guarantee an unbiased assessment (independence from the proposed data generator), we also make use of a second set of synthetic datasets generated by Hochreiter et al. [324]. These datasets simulate specific characteristics of expression data, such as heavy tail properties, using three settings: multiplicative models and additive models under signals according to $N(\pm 2, 0.5^2)$ and $N(\pm 4, 0.5^2)$ distributions[1]. Results are computed from 100 data instances per setting, each with 1000 genes, 100 conditions and 10 planted biclusters.

Other Settings We also maintained the evaluation metrics, as well as the set of biclustering algorithms from previous chapter and their parameterizations. From these algorithms: 5 biclustering approaches are prepared to discover biclusters with either constant or differential values (Samba [616], xMotifs [477], BiModule [500], DeBi [578] and RAP [509]), 5 approaches claim to be able to discover multiplicative and/or additive biclusters (FABIA with sparse prior Equation [324], ISA [342], Bexpa [532], CC [134] and BicPAM [310]), 3 approaches are prepared to find order-preserving biclusters (OPSM [59], OP-Clustering [415] and a variant of BicPAM referred as BicSPAM proposed in *Chapter III-6*), and the remaining approach is able to model plaid structures (BCPlaid [638]).

3.3.1 Results in Synthetic Data

For assessing the ability of BicPAM to recover planted biclusters with non-constant coherencies, we maintained the experimental data settings described in previous chapter and varied coherency assumption of the planted biclusters, considering size adjustments on the planted biclusters to guarantee their significance. The number of rows and columns for additive, multiplicative and symmetric models are 1.2 times the size of constant biclusters. Figures 3.6 and 3.7 illustrate the performance of the compared methods for datasets with additive and multiplicative regularities and fixed coherency strength (δ=0.2). In order to promote the readability of these charts, we excluded the performance of the approaches not prepared to discover biclusters under these assumptions. Results from these figures show the superior performance of BicPAM. Superiority is shown both in terms of MS($\mathcal{B}, \mathcal{H}$), that is, the majority of the discovered biclusters are well described by the hidden biclusters (correctness), and MS($\mathcal{H}, \mathcal{B}$), that is, the majority of the hidden biclusters can be mapped into a discovered bicluster (completeness). Yet, these levels are slightly worse than the comparable levels for the discovery of constant coherencies due to the higher probability of background values to form a non-planted additive and multiplicative bicluster. Again these gains come with a cost in efficiency. In particular, the search for additive models is the most computationally expensive. The multiplicative model is more efficient since it relies on data transformations associated with sparser matrices. Fabia is a competitive option for the discovery of additive and multiplicative biclusters. However, it is not prepared to deal with overlapping areas and accommodates high levels of noise since it not able to approximate the

[1] http://www.bioinf.jku.at/software/fabia/benchmark.html

parameterized coherency strength, leading to biclusters with a larger number of false positive rows. Although ISA is tunned to discover biclusters with gradual changes on values, its scoring schema based on self-consistency property is not well suited to discover biclusters with delineated additive and multiplicative factors. Although OPSM and OP-Clustering can be used to discover biclusters with additive and multiplicative factors, the discovered biclusters include many non-planted rows and columns as they seek to model order-preserving coherencies. Although all the five compared approaches are efficient for medium-sized matrices, efficiency deterioration is faster for OPSM, BicPAM (under non-constant assumptions) and Bexpa.

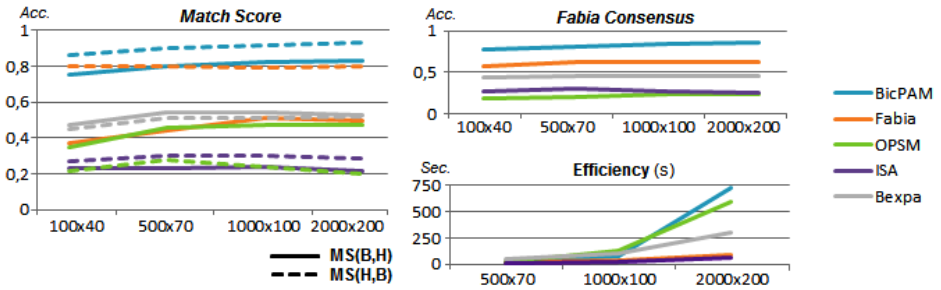

Figure 3.6: Comparing the performance of biclustering approaches for the recovery of planted biclusters with additive coherency assumption (Jaccard-based scores, Fabia consensus and efficiency).

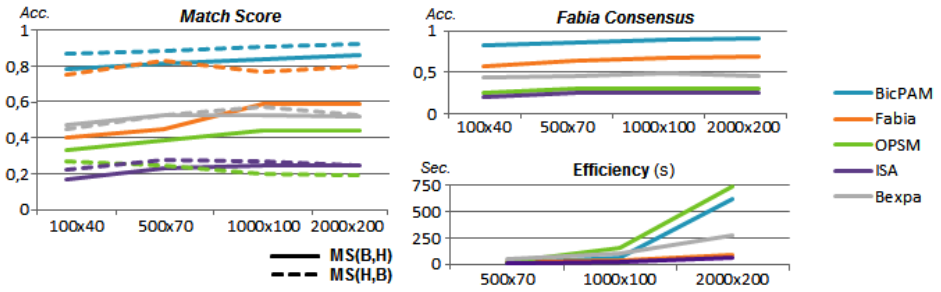

Figure 3.7: Comparing the performance of biclustering approaches for the recovery of planted biclusters with multiplicative coherency assumption.

The ability of these approaches to retrieve the hidden biclusters using FABIA data settings is synthesized in Figure 3.8. According to the gathered $FC(\mathcal{B},\mathcal{H})$ levels, BicPAM is the best performer for biclusters following additive models with different signal properties (Wilcoxon-test at 0.01%) and, together with FABIA, the best option for multiplicative models. With regards to both MS($\mathcal{B},\mathcal{H}$) and MS($\mathcal{H},\mathcal{B}$) scores, BicPAM shows superiority across all settings. The exhaustive nature of BicPAM searches and the ability to rely on multiple alphabets (coherency strength) without risk of introducing noise (by assignment multiple items for values near ranges-boundaries) support these observations. FABIA is a competitive non-exhaustive alternative, sensitive to the planted noise. Nevertheless, it requires prior knowledge regarding the number of biclusters. Since ISA is tunned to discover biclusters with gradual changes on values, its scoring schema to find modules with self-consistency is not well suited to discover biclusters modeled by additive signals. Plaid is able to locally identify additive factors. Understandably, the set of approaches not able to discover biclusters with scaling and shifting factors is considerably less effective. The accuracy levels of OPSM are strongly penalized since OPSM outputs a large number of biclusters with varying sizes (including biclusters with either small number of genes or conditions). The average efficiency levels of BicPAM show its ability to perform exhaustive searches in useful time for computationally complex settings.

A closer look to how the behavior of BicPAM in the absence of enhancements (only a simple pruning of small biclusters and postfiltering of similar biclusters) varies with the desirable coherency strength is provided in Figure 3.9. Although the observed FC levels are considerably good, they are penalized by: the exclusion of rows due to the planted noise, allowed overlapping among planted biclusters, and the number of discovered biclusters (typically higher than the number of planted biclusters). A looser coherency strength (smaller number of items)

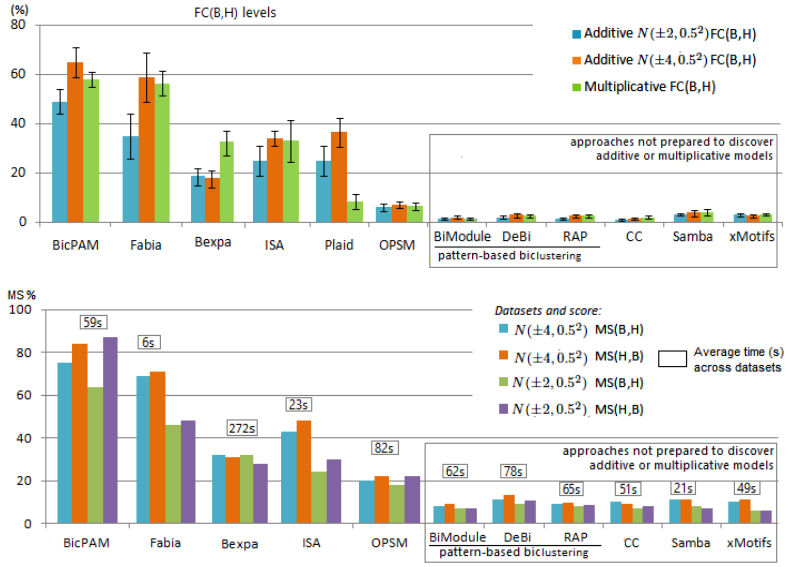

Figure 3.8: FC and Jaccard-based scores across biclustering approaches using FABIA datasets.

turns the underlying transactional database denser, decreasing the efficiency bounds of BicPAM. When considering the additive and multiplicative assumptions, FC scores slightly decrease against a constant assumption due to the higher probability of background values to form a non-planted additive bicluster. This problem can, however, be corrected through closing options. Complementarily, the depicted $MS(\mathcal{B},\mathcal{H})$ levels are higher than FC levels due to the absence of penalizations associated with differences on the number of outputted versus planted biclusters. In particular, $MS(\mathcal{B},\mathcal{H})$ levels for medium- to-large datasets are, respectively, above 95%, 91% and 87% for constant, additive and multiplicative assumptions. $MS(\mathcal{H},\mathcal{B})$ levels were excluded from this figure as they were consistently over 95% across settings. Although a naïve search for biclusters under these assumptions would cost as much as $|\mathbf{Y}|$ times as the search for constant biclusters, BicPAM shows heightened efficiency performance by relying on the proposed principles to prune the search space. In particular, BicPAM under a multiplicative assumption is structurally more efficient than its additive peer since the number of spurious biclusters is considerably low due to the broader range of items observed within each iteration, which leads to sparser matrices.

A detailed look to BicPAM's performance when considering 7 items and default noise handling, merging and filtering options, is provided in Table 3.1. The results are organized according to bicluster type, matrix size (and structure of planted biclusters) and underlying distribution of background values. The slightly worse performance when the input values are generated by a Gaussian distribution is associated with an increased difficulty of recovering the planted local regularities after data normalization. We found MS($\mathcal{B},\mathcal{H}$) to be lower than MS($\mathcal{H},\mathcal{B}$) since the exhaustive nature of BicPAM leads to at least one found bicluster with a direct correspondence to each hidden bicluster.

		100x30		500x60		1000x100		2000x200	
Metric	Coherency	Normal	Uniform	Normal	Uniform	Normal	Uniform	Normal	Uniform
FC	Constant	0.862±0.017	0.930±0.014	0.884±0.018	0.956±0.007	0.909±0.017	0.949±0.006	0.907±0.014	0.948±0.011
	Additive	0.782±0.021	0.831±0.008	0.834±0.014	0.888±0.007	0.845±0.018	0.897±0.007	0.827±0.015	0.887±0.006
	Multiplicative	0.762±0.028	0.794±0.013	0.790±0.019	0.825±0.014	0.785±0.020	0.840±0.011	0.767±0.020	0.819±0.015
MS($\mathcal{B},\mathcal{H}$)	Constant	0.923±0.018	0.974±0.007	0.931±0.012	0.968±0.005	0.935±0.010	0.984±0.005	0.944±0.011	0.987±0.008
	Additive	0.895±0.017	0.945±0.006	0.925±0.012	0.963±0.003	0.913±0.008	0.981±0.007	0.917±0.011	0.974±0.006
	Multiplicative	0.902±0.019	0.958±0.014	0.906±0.015	0.953±0.009	0.910±0.015	0.941±0.008	0.886±0.019	0.948±0.010
MS($\mathcal{H},\mathcal{B}$)	Constant	0.956±0.013	0.984±0.006	0.960±0.007	0.981±0.004	0.961±0.004	0.996±0.002	0.957±0.009	0.993±0.002
	Additive	0.955±0.012	0.997±0.001	0.959±0.006	0.997±0.002	0.955±0.004	0.995±0.002	0.957±0.007	0.995±0.003
	Multiplicative	0.937±0.015	0.966±0.008	0.924±0.012	0.968±0.008	0.923±0.010	0.963±0.009	0.927±0.013	0.974±0.007

Table 3.1: FC and MS levels of BicPAM in different settings (mean and variance from 20 datasets).

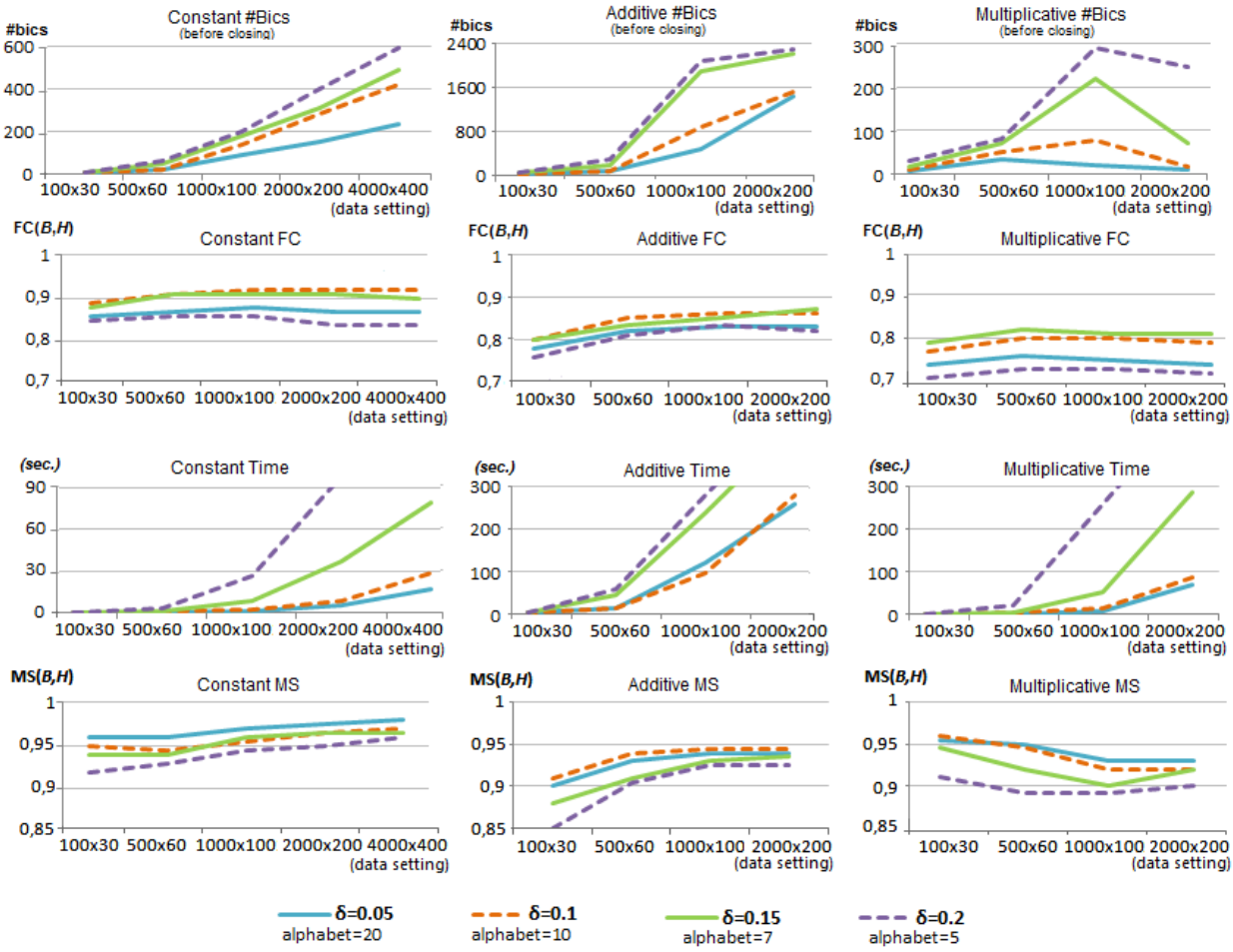

Figure 3.9: Performance of BicPAM (size, accuracy and efficiency levels of the learned solutions) under constant, additive and multiplicative assumptions in the absence of robustness principles for varying coherency strength.

3.3.2 Results in Real Data

To show the relevance of discovering biclusters with non-constant coherencies on real data domains, we rely on expression data[2,3]: *1) dlblc* dataset (660 genes, 180 conditions) to study responses to chemotherapy [560], *2) hughes* dataset (6300 genes, 300 conditions) to characterize nucleosome occupancy [395], and *3) gasch* dataset (6152 genes, 176 conditions) to measure Yeast responses to environmental stimuli [239]. BicPAM was parameterized with multiple levels of expression ($|\mathcal{L}|\in\{3..6\}$) and multiple closing options. The biological relevance of results was assessed based on functionally enriched terms (see *Chapter II-2*).

Figure 3.10 illustrates four biclusters present in the *gasch* dataset. These biclusters are associated with the coherent responses of Yeast genes to heat shock at different time points. BicPAM's behavior is particularly favorable for the discovery of these biclusters, contrasting with other biclustering approaches. In particular, we illustrate the combination of constant models with symmetries, multiplicative models, additive models with several levels of expression, and additive models with symmetries. The analysis of these biclusters shows the relevance of combining multiple levels of expression ($|\mathcal{L}| \geq 5$) with noise relaxations for the discovery of meaningful biclusters, and supports the importance of allowing sign-changes to capture activation and repression mechanisms in regulatory processes.

Table 3.2 describes an additional set of found pattern-based biclusters over expression data with biological significance. Such biclusters could hardly be discovered by peer biclustering methods, since many of them include conditions with multiple degrees of expression (varying coherency strength) and non-constant profiles ($\mathbf{B}_1$, $\mathbf{B}_3$, $\mathbf{B}_6$,

[2]http://www.bioinf.jku.at/software/fabia/gene_expression.html
[3]http://chemogenomics.stanford.edu/supplements/03nuc/datasets.html

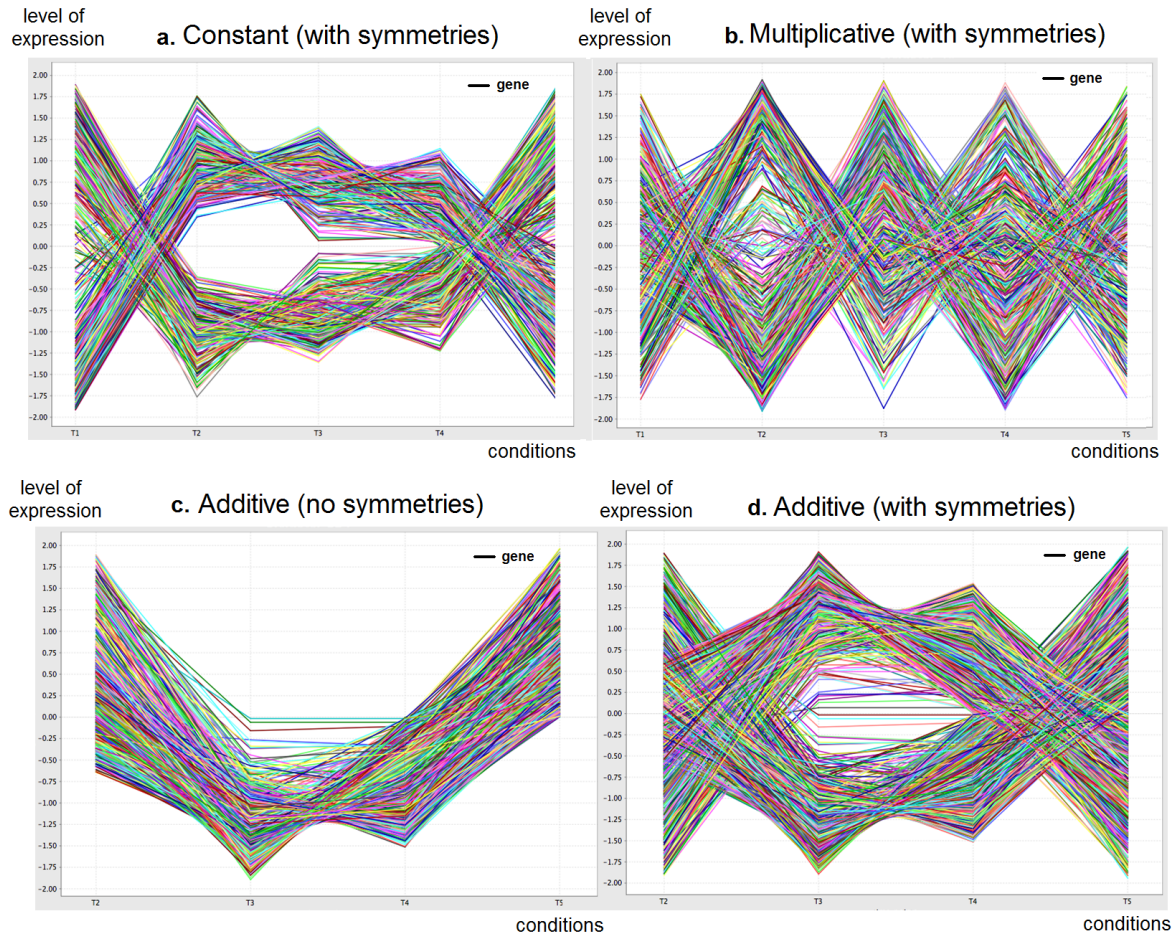

Figure 3.10: Biclusters extracted from *gasch* dataset with constant models (a), multiplicative models (b) and additive models (d) allowing for symmetries.

$\mathbf{B}_8$). All of these biclusters have heightened biological significance as observed by the number of highly enriched terms after Bonferroni correction. Interestingly, we also observe that different closing options lead to biclusters with different shapes, even when the number of items is the same ($\mathbf{B}_4$ and $\mathbf{B}_5$).

Dataset	ID	Pattern	Items	Closing Options
dlblc	B1	FAABFFF/AFFEAAA	A-F	Merging with tight overlapping
	B2	AAABCA	A-C	Extensions allowed (with tight merging)
	B3	AAA/../EEE	A-E	Reducing with high homogeneity
hughes	B4	CCCACC/../EEECEE	A-E	Merging allowed
	B5	CCDCBCBCC	A-E	Merging with relaxed overlapping
	B6	AAAAA/../G..G	A-G	Merging with tight overlapping
gasch	B7	AAAGGGA	A-G	Merging with tight overlapping
	B8	AAABACCCAA	A-E	Merging allowed

ID	Type	♯Genes	♯Conds	♯p−values <0.01	♯p−values [0.01,0.05]	Best p-value
B1	symmetric	85	7	41	21	1.97E-10
B2	constant	153	8	9	1	2.27E-12
B3	multiplicative	119	5	5	18	4.12E-8
B4	additive	589	6	12	7	1.31E-25
B5	constant	654	10	16	4	1.31E-17
B6	additive	476	6	12	10	1.92E-6
B7	constant	483	7	57	10	1.24E-81
B8	additive	521	10	17	5	4.57E-12

Table 3.2: Illustrative set of biclusters with different properties and heightened biological relevance (p-values after Bonferroni correction).

Although a detailed biological analysis is out of the scope of this paper, we provide a brief analysis for one

bicluster per dataset. The bicluster identified in Tables 3.2 and 3.3 as $\mathbf{B}_1$, with 85 human genes with coherent expression across 7 samples, was discovered in *dlblc* using 6 levels of expression (under a Gaussian discretization). The majority of these genes showed very low expression (A) on 2 samples, low expression (B) on 1 sample and very high expression (F) on 4 samples. 3 genes from this sert of genes show coherent symmetric behavior (over-expressed in 3 samples and under-expressed in 4 samples). Over 40 GO terms were highly enriched, with the top set of terms being related with immune defense responses (e.g. immune system process, regulation of immune system process) and signaling functions associated to immunomodulating agents, such as cytokine. Significant terms related with Golgi and with the formation of membrane-bound compartments imply their critical roles during the induction of innate immune responses after chemotherapy [674]. Similar biclusters are not discovered when the number of expression levels is decreased or when noise relaxations are not included, thus motivating the need for BicPAM. The illustrative biclusters, found in *hughes* and *gasch* datasets, concern genes from Saccharomyces cerevisiae analyzed in the context of studying nucleosome occupancy and responses to different stress conditions, respectively. The enriched terms of the additive bicluster $\mathbf{B}_4$ are associated with the formation of carboxylic acid and organonitrogen compounds, with optimum enrichment levels found in the presence of moderate noise-tolerance. Bicluster $\mathbf{B}_7$ captures genes with coherent expression across multiple time points from three different heat shocks (shocks from 17, 21 and 25°C). The analysis of GO terms shows functions related with the ribonucleoprotein complex (p-value 1.24E-81), associated with the reassembly and protection of small particles during heat stress responses [92]. Interestingly, other biclusters found in *gasch* are able to capture coherent levels of expression across different stimuli. An example is the bicluster $\mathbf{B}_8$ that integrates conditions related with nitrogen depletion, heat stress and diauxic shift. $\mathbf{B}_8$ has 521 genes, coherent additive levels of expression across 10 conditions, and over 10 highly significant enriched terms.

ID	Dataset	Top 4 GO Terms (p-value)
B1	*dlblc*	immune response (2.32E-10); immune system process, defense response (<1E-6); cytokine-mediated signaling pathway (1.33E-7); Golgi apparatus (1.19E-7).
B4	*hughes*	carboxylic acid biosynthetic process (1.3E-25) and metabolic process (6.12E-16); organonitrogen compound biosynthetic process (2.23E-18) and metabolic process (2.71E-13).
B7	*gasch*	ribonucleoprotein biogenesis and assembly (1.24E-81); cytosolic part (1.22E-57); intracell. non-membrane-bounded organelle (1.31E-65); ncRNA metabolic process (1.82E-52).

Table 3.3: Enriched GO terms of three illustrative BicPAM biclusters

Consider the illustrative biclusters provided in Table 3.3. Some of the enriched transcription factors regulating the genes in bicluster $\mathbf{B}_1$ (associated with immune system responses in the human genome) include: HCLS1 gene that plays a key role in regulating clonal expansion and deletion in lymphoid cells [564], IRF1 protein that acts as a tumor suppressor and plays a role not only in antagonism of tumor cell growth but also in stimulating an immune response against tumor cells [564], and TRIM22 antiviral protein involved in cell innate immunity [394]. Other highly enriched TFs that regulate proliferation and transformation (tumor supressors) are ANP32A and RUNX3 [564]. The TFs regulating the genes in bicluster $\mathbf{B}_4$ have p-values below 1E-15 after correction, each regulating from 50% to 95% of the genes in bicluster. They are associated with regulatory functions consistent with the enriched terms. Some of these TFs include histidine biosyntehsis (Bas1p), amino acid biosynthesis (Gcn4p), cyclic AMP receptor protein regulation (Sok2p) and other TFs related with the regulation of carboxylic acid and organonitrogen compounds [135]. Consider now bicluster $\mathbf{B}_7$ from *gasch*. Some of the enriched TFs include Sfp1p, Mga2p, Ace2p, Tup1p, Spt10p and Swi5p (p-values below 1E-15), each regulating 55%-97% of the genes in the bicluster. These factors are known to be involved in stress responses as they regulate cooling and oxygen levels (Mga2p), repair cellular damage (Sfp1p and Spt10p), remodel chromatin (Tup1p) and regulate cell wall protection (Swi5p and Ace2p) [480, 185, 135]. Finally, consider bicluster $\mathbf{B}_8$, whose genes coherently regulate heat, nitrogen depletion and diauxic shifts. Sfp1p, Bas1p, Ste12p and Tec1p were the most significant TFs in

this bicluster (p-values<1E-7). Sfp1p controls expression of ribosome biogenesis genes in response to stress and DNA-damage response [135]. Bas1p regulates gene expression for biosynthesis pathways such as pathways related with histidine metabolism, which responds to environmental stimuli (e.g. nitrogen) affecting pH calibration [135]. Finally, Ste12p and Tec1p act together to regulate genes related with invasive growth, whose production is expected under such stress conditions [135].

3.3.3 Comparison of Pattern-based Biclustering Approaches

The gathered results in this and previous chapter provide substantial empirical evidence for the superiority of BicPAM's performance against peer pattern-based methods, such as BiModule, DeBi and RAP. First, Figures 3.8 and 3.9 show the unique ability of BicPAM to discover non-constant biclusters (>50 percentage points in MS and FC against BiModule, DeBi and RAP). Second, Figures III-2.9 and III-2.8 shows improvements in the discovery of constant biclusters related with BicPAM's principles, such as its ability to deal with the items-boundary problem. These results are also supported by BicPAM's ability to combine solutions discovered with effective closing procedures, varying coherency strength and closed pattern representations (all maximal biclusters), contrasting with some pattern-based peers. Third, Figures 2.10, 2.12 and 2.13 show significant performance improvements of BicPAM against peers due to BicPAM's exclusive ability to deal with arbitrary-high levels of noise and missing values. Finally, the superiority of biological relevance of BicPAM's solutions against the peer pattern-based solutions is supported by Table 2.2 and subsequent analyzes. In particular, we show that BicPAM's solutions are able recover biclusters discovered by peer methods and, additionally, discover unique and biologically meaningful biclusters (Tables III-2.3 and 3.2) such as the four illustrative biclusters in Figure 3.10.

3.4 Summary of Contributions and Implications

BicPAM goes beyond the constant assumption made by existing pattern-based approaches, extending the biclustering searches to more flexible coherency assumptions. Iterative adjustments based on differences and of the least common divisor between the values of a given column (or row) are applied in the matrix in order to identify shifting and scaling factors. The removal of these factors in the matrix allows the discovery of additive and multiplicative models. Similarly, combinatorial sign-adjustments across rows (or columns) are used to model symmetries and integrate them with previous models. Pruning strategies were considered to avoid redundant calculus and reduce the computational complexity of these searches. Finally, BicPAM was extended to tabular data contexts possibly characterized by a combination of nominal, ordinal and numeric features non-identically distributed.

The contributions in this chapter are the first attempt to deliver flexible structures of biclusters with delineate shifts and scales, combine shifts with symmetries and guarantee their discovery from tabular data without the need to rely on knowledge-driven preprocessing procedures.

Results on synthetic data show BicPAM's ability to accurately recover non-constant biclusters over high-dimensional data. BicPAM's performance superiority was observed both against peer pattern-based approaches and state-of-the-art biclustering algorithms, supporting its heightened flexibility, optimality and robustness to noise. The application of BicPAM with flexible coherency assumptions on expression data revealed meaningful and biologically significant biclusters that were not able to be discovered by other biclustering approaches.

4

Flexible Plaid Models: Unraveling Meaningful Interactions between Biclusters

Biclustering with a plaid assumption is required to model complex biclustering structures that assume a composition from multiple biclusters on the areas where their rows and columns overlap. The resulting biclustering models, referred as plaid models, are essential to explore mutual influence between regions within a given data space (see Figure II-3.5). Consequently, plaid models are a powerful tool to search for an interpretable structure withing biological and social data contexts. The plaid assumption is necessary to model overlapping transcriptional activity in biological contexts and consider cumulative effects in the interactions from social and biological networks [393, 312]. The plaid model can be also applied to study regulatory cascades from stimuli-elicited biomedical data (such as drug responses [40]) and trading operations from non-trivial behavior of the stock market.

Illustrating, genes can participate in multiple biological processes at a time and thus their expression can be seen as a composition of the contributions from the active processes. In this context, modeling biclusters from expression arrays or RNAseq technologies should ideally assume that the expression level is a composition of the activity levels associated with each active process. As a result, the plaid model can disclose meaningful interactions between putative transcriptional modules associated with the found biclusters.

Despite the biological relevance of these biclusters, few biclustering algorithms consider the plaid model. Lazzeroni and Owen [393] proposed the first biclustering method based on a plaid model, which was later extended [573, 113, 270, 575, 449]. Despite the relevance of this pioneer method, and of the sequent extensions, in providing solutions that address the challenging combinatorial nature of this task, these approaches suffer from four major drawbacks. First, they describe plaid models as an exact additive model of overlapping contributions. However, gene expression may not increase linearly with the involved contributions. Illustrating, although two processes may contribute to an increase on the levels of expression of one gene, the expression of this gene may not be given by their exact sum when they are simultaneously active. Instead, the expression level is expected to increase but not proportionally, and thus can be more realistically described by a more relaxed range of levels of expression. Second, they require all the observations (elements in the dataset) to fit the plaid model, imposing a modular decomposition of the overall data according to a set of contributing biclusters (or biological processes). Understandably, this principle can degrade the learning of plaid models when some of the observations do not follow a plaid assumption or when genes that do not participate in a bicluster have differentiated levels of expression (and thus cannot be modeled by a null background layer). Third, they place restrictive assumptions on the types and structures of biclusters, typically modeling biclusters with constant or additive coherencies and constraining their number and size. Finally, these models are either based on generative or greedy algorithms that do not offer guarantees of the maximality and completeness of biclustering solutions.

In this chapter, we propose BiP (Biclustering using Plaid models), a new biclustering algorithm to address these four shortcomings. For this aim, BiP is able to recover excluded areas due to unaccounted plaid effects and detect noisy areas non-explained by a plaid assumption, thus producing an explanatory modular decomposition of the solution space. In this way, is able to seize contributions from state-of-the-art biclustering approaches by using their solutions as the starting basis for a plaid model. Moreover, it provides relaxations for allowing values in

overlapping areas to change according to more realistic assumptions (weighted and noise-tolerant composition of contributions) besides the common additive constraint. To our knowledge, this is the first approach to model non-constant biclusters with non-additive plaid contributions, allow a relaxed and more realist validation of overlapping effects, and deliver an exhaustive and flexible structure of biclusters.

The contributions of this work for the data mining and bioinformatics communities are seven-fold:

- first algorithm able to consider flexible functions to compute the combined effect of plaid contributions, including weighted and multiplicative functions;
- robust principles to learn plaid models based on relaxations that are able to comply with varying levels of noise and to approximate the activity contributions from overlapping processes/biclusters in a more realistic and (biologically) meaningful way;
- strategies to deliver flexible solutions: structures with an arbitrary number and positioning of biclusters under a plaid assumption, as well as biclusters with parameterizable levels of expression and varying size;
- possibility to adjust biclustering solutions produced by state-of-the-art algorithms (recovery of excluded areas due to unaccounted plaid contributions and removal of included areas unexplained by a plaid assumption);
- initial empirical evidence that the enumerated contributions can be used to unravel significant and non-trivial functional dependencies between biological processes in human and yeast;
- principles to systematize relations among biological processes and social communities, such as *is-part-of* and *depends-on*, based on the proposed plaid model's extensions;
- tool for generating datasets following plaid models, where the structure, quality and type of biclusters can be flexibly controlled, including the overlapping degree and the composition of contributions.

Experimental results on both synthetic and real datasets demonstrate the superior flexibility and robustness of BiP models, as well as their extensibility to retrieve non-constant biclusters with a non-additive composition of contributions. BiP application over expression data is able to model complex transcriptional activity with biological significance. We also show that these models provide an explanatory decomposition of overlapping areas that can be used to unravel meaningful and non-trivial interactions among biological processes.

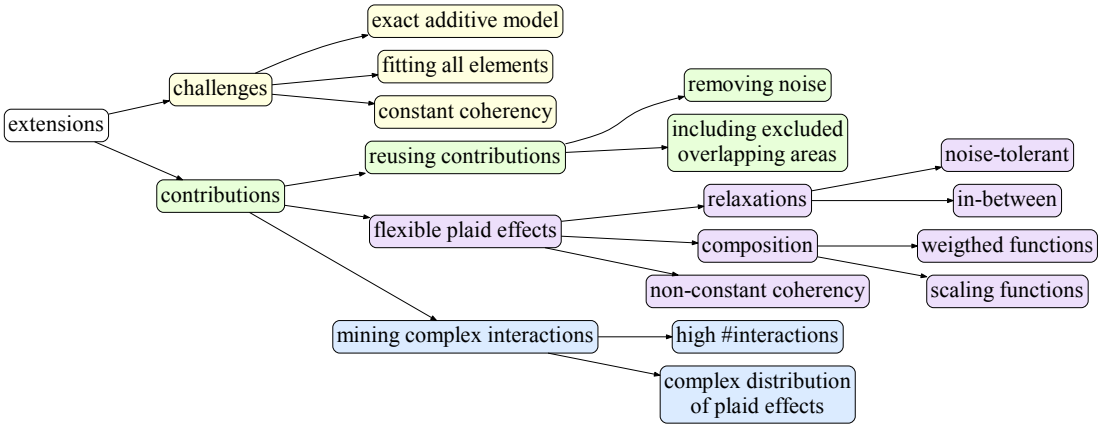

Figure 4.1: Challenges and the three major sets of contributions to effectively learn plaid models.

Figure 4.1 synthesizes the discussed contents throughout this chapter, providing a view on the major challenges and proposed contributions for learning plaid models. This chapter is organized accordingly. *Section 4.1* provides the background on biclustering with plaid models. *Section 4.2* introduces the proposed BiP algorithm. *Section 4.3* assesses the performance of BiP on synthetic data and the biological relevance of the learned plaid models over real data. Finally, the contributions and implications of this work are synthesized.

4.1 Related Work

The types of biclusters supported by Def.II-3.1 do not consider interactions among biclusters. In particular, they do not allow contributions from overlapping biclusters where an element a_{ij} in the data matrix is seen as a sum of the contributions from different biclusters to whom the row i and the column j belong. Lazzeroni and Owen [393] addressed this limitation by introducing the plaid model where the value of an element in the data matrix is viewed as a sum of terms called layers or biclusters. In the plaid model, all the elements a_{ij} in the input data matrix are described by a linear function of layers.

Def. 4.1 A biclustering solution following a *plaid model* is a composition of K biclusters (layers), where θ_{ijk} specifies the contribution of each bicluster according to the following model of the matrix A:

$$a_{ij} = \mu_0 + \sum_{k=0}^{K} \theta_{ijk}\rho_{ik}\kappa_{jk} \;\; | \; \theta_{ijk} = \mu_k + \alpha_{ik} + \beta_{jk} + \eta_{ijk}, \tag{4.1}$$

where ρ_{ik} and κ_{jk} are binary values defining, respectively, the membership of row i and column j in bicluster k.

According to (4.1), each element a_{ij} is defined as a sum of additive models, each representing the contribution of the bicluster $B_k = (I_k, J_k)$ to the value of a_{ij} in case $i \in I_k$ and $j \in J_k$. In particular, ρ and κ variables are matrices of size $n \times k$ and $m \times k$ defining binary membership variables, where $n = |X|$, $m = |Y|$ and $k = |\mathcal{B}|$. Here $\rho_{ik}=1$ iff row (gene) i belongs to bicluster k, and $\kappa_{jk}=1$ iff column (condition) j belongs to bicluster k. Each bicluster k is specified by which of the variables $\rho_{\bullet k}$ and $\kappa_{\bullet k}$ are equal to 1. $\mu_0 \in \mathbb{R}$ defines the expected background values of the dataset, that is, a bicluster to which all the elements belong. Figure 4.2 illustrates an additive plaid model from constant biclusters. However, the notation θ_{ijk} makes this model powerful enough to identify other types of biclusters by using θ_{ijk} to represent either additive $\mu_k + \alpha_{ik} + \beta_{jk}$ or multiplicative $\mu_k\alpha_{ik}\beta_{jk}$ biclusters. A greedy algorithm is also proposed in [393] to learn these layers by minimizing a quadratic error function [1] (assuming a noise factor η_{ijk} approximately Normal). This algorithm discovers one bicluster at a time by iteratively subtracting the respective layer from the (expression) data. A maximum number of biclusters and minimum size of biclusters are considered as stopping criteria.

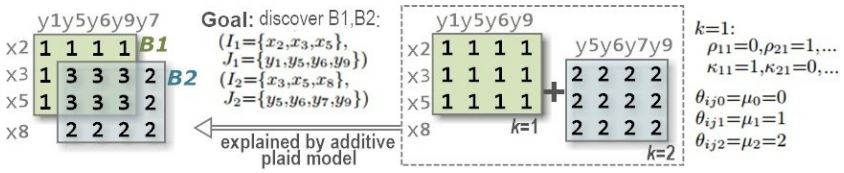

Figure 4.2: Additive plaid model of 2 constant biclusters.

Segal et al. [573, 575] also assumed an identical plaid model where the input matrix is seen as a sum of terms called processes (biclusters). They introduced an extra degree of freedom by considering that each process is generated by a Gaussian distribution with a variance, σ_k^2, that depends (only) on the bicluster index k (assuming that the process $k=0$ models the data variability that is not particular to any bicluster). Although this imposes the use of constant biclusters, it can accommodate for varying degrees of variability[2]. Constrasting with the original greedy algorithm [393], this algorithm learns the whole biclustering solution (the K processes) at a time based on the expectation-maximization algorithm. This is a more robust solution since groups of rows/columns are iteratively updated as the biclusters become more refined. The authors provide empirical evidence of this effect.

Meeds et al. [449] introduced a different generative algorithm using a binary matrix factorization model that approximates the input matrix as a non-parametric Bayesian probabilistic model with binary latent variables

[1] $\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m}(\hat{a}_{ij} - \theta_{ij0} - \sum_{k=1}^{K}\theta_{ijk}\rho_{ik}\kappa_{jk})^2$

[2] $\sum_{k=1}^{K}\frac{(\hat{a}_{ij} - \theta_{ijk}\kappa_{jk})^2}{\sigma_k^2}$

(biclusters). The model explains how each row and column, when seen as a composition of multiple binary latent variables, can approximate complex joint distributions on a variety of data.

In line with these generative models [573, 449], Caldas and Kaski proposed an alternative Bayesian biclustering approach [113] that relies on a collapsed Gibbs sampler to infer the posterior distribution of a bicluster's membership that indicates which rows i and columns j belong to each bicluster (latent variable) k. Gibbs sampling is shown to be an effective option to derive deterministic solutions. A similar Bayesian biclustering model was also proposed by Gu and Liu [270].

Recently, Wilderjans et al. [673] proposed an algorithmic variant for learning plaid models using alternating least squares algorithms, and Mankad and Michailidis [434] extended the plaid model to find subspaces of interest in a collection of matrices (triclusters).

Madeira and Oliveira [429] proposed two extensions over the plaid model that, to our knowledge, were not yet experimentally evaluated. The first extension is to consider that the value of a given element a_{ij} is given by a non-additive composition of the contributions of the different biclusters to whom the row i and the column j belong, such as multiplicative composition (where $a_{ij} = \prod_{k=0}^{K} \theta_{ijk}\rho_{ik}\kappa_{jk}$). It is also possible to combine non-constant types of biclusters, θ_{ijk}, with these functions for composing contributions. Some of these variants are illustrated in Figure 4.3. The second extension is to further assume that rows and columns can exhibit different degrees of variability (4.2):

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \frac{(\hat{a}_{ij} - \theta_{ij0} - \sum_{k=1}^{K} \theta_{ijk}\rho_{ik}\kappa_{jk})^2}{2(\sigma_i^2 + \sigma_j^2 + \sigma_A^2)} \tag{4.2}$$

where σ_i^2, σ_j^2 and σ_A^2 are, respectively, the variance of i row, j column and of the overall observations.

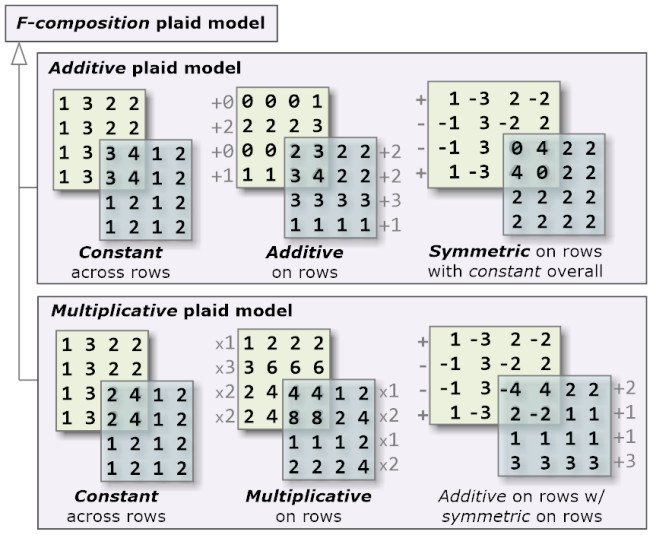

Figure 4.3: Additive and multiplicative composition of plaid contributions from different types of biclusters.

Complementary to research on plaid models, Truong et. [632] and Mahfouz and Ismail [431] propose biclustering algorithms with a focus on overlapping areas between biclusters. Other biclustering algorithms have also been proposed to accommodate overlaps [460]. Still, these works do not assume a plaid effect from the contributions in the areas where biclusters overlap.

Learning from Flexible Structures of Biclusters. Flexible structures of biclusters provide an adequate basis to investigate meaningful overlapping effects for learning plaid models. In particular, pattern-based biclustering solutions [310, 29, 500, 311] are well-positioned to achieve this goal due to three additional properties. First, they guarantee exhaustive space searches, being able to enumerate all maximal biclusters (Def.1.10). Second, they offer an easily parameterizable environment where constraints, such as the desirable coherency strength and the minimum number of columns and rows of a bicluster, can be incorporated. Finally, they provide principles to guar-

antee their robustness to noise [310, 578] and adequately affect the properties of biclusters through postprocessing options [308, 310, 578]. Nevertheless, alternative flexible structures of biclusters discovered by other biclustering approaches, such as CCC-biclustering [427], can be also considered to affect the learning of the target plaid models.

Existing searches for flexible structures of biclusters either: 1) neglect regions explained by plaid effects, leading to biclusters with incorrectly excluded rows and columns (false negatives), or 2) tolerate high levels of noise for the accommodation of these effects, yet they also accommodate non-coherent regions leading to biclusters with incorrectly included rows and columns (false positives).

4.2 Solution

In this section, we incrementally introduce BiP, an algorithm for the discovery of flexible plaid models. First, we show how existing biclustering solutions can be adjusted to recover excluded areas due to plaid effects and remove noisy areas that do not satisfy plaid assumptions. Second, we show how these adjustments can be performed using more realistic criteria to verify the overlapping contributions. In particular, we propose a set of relaxations and advanced composition functions to deal with noise and with the non-linear, scaling and residual plaid effects, and illustrate their relevance for modeling overlapping transcriptional activity from expression data. Third, we refine the proposed solution to deal with complex interactions between an arbitrary number of biclusters. Finally, we show how the proposed solution can be used to model different relations between biological and social modules.

4.2.1 Plaid Models from Exhaustive Biclustering Solutions

This section proposes a set of principles to adjust existing biclustering solutions. As surveyed, complete structures of (maximal) biclusters provide a good starting basis for composing more advanced models, where relations between biclusters (associated with overlapping areas where the observed values are given by a composition of the involved contributions) can be posteriorly included. Despite the relevance of these overlapping areas to model interactions among biological processes and social communities according to a plaid assumption, they are either excluded from these biclustering solutions or accommodated with additional noisy elements due to the absence of appropriate checks.

4.2.1.1 Initial Three-step Methodology

An initial three-step methodology is proposed to recover neglected plaid effects in order to compose plaid models. First, merging or extension operations are used to enlarge biclusters. The accommodated elements in the resulting bicluster B that have values that deviate from the expected pattern φ_B are signaled as noisy elements. When the majority of these elements are coherently signaled as noisy, there is a high chance that the area defined by these elements has some structural significance that should be further investigated. Second, and after the noisy areas have been extracted, the expected contributions from the biclusters with whom they are associated are computed. Finally, the contributions from these noisy areas are tested to check whether they are explained by a plaid model. If the noise associated with a bicluster is explained by a plaid effect, the bicluster is allowed to appear in the delivered solution, otherwise it is excluded. Variants of this behavior can be considered when only a part of the potentially overlapping elements fit the plaid model.

Three major options can be taken in the first step to produce biclusters tolerant to local noise:

- *noise relaxations*, such as multi-item assignments and aggregations of items (see *Chapter 2*), can be used to adapt biclustering tasks to produce more noise-tolerant structures of biclusters;
- *extension* options allow the inclusion of new rows or columns over noise-intolerant biclusters as long as either: their homogeneity is preserved or only a fraction of the elements of each new row or column deviate from the expected values. Two non-exclusive strategies can be used to implement this option: *1)* rely on greedy

biclustering methods (and of their merit functions) for incrementally extend biclusters, and *2)* statistically test new rows or columns. Some of the state-of-the-art methods to perform extensions were proposed within DeBi algorithm [578] and by Bellay et al. [57];

- *merging* options can be used to compose larger biclusters when the candidate-set of biclusters shares a significant common area. This composition is often a result of locally non-coherent values associated with plaid contributions. The simplest criterion to test merging options is to either rely on the overlapping area (as a percentage of the smaller bicluster) or to test the resulting homogeneity after merging biclusters (see [308] for a survey on efficient methods for the discovery of candidate-sets of biclusters for merging);

The locality of noise within each bicluster needs to be assessed for the first two options. This can be efficiently performed by modeling the noisy elements as a set of bitset vectors, and by either performing intersection operations or testing the number of required biclusters to cover the noisy elements. Contrasting with these options, in merging, the locality of noise is guaranteed by default. For simplicity sake, we use merging as the means to compose plaid models.

In the second step, the noisy areas in the biclusters, here referred as *candidate blocks*, are extracted and their φ_B patterns (Def.III-1.1) computed from the expected values of the original bicluster. Illustrating, consider the bicluster (I=$\{x_1, x_2, x_3, x_4\}$, J=$\{y_1, y_2, y_3, y_4\}$) with a_{1J}=a_{2J}=$\{1,2,2,1\}$, a_{3J}=$\{3,4,2,1\}$, a_{4J}=$\{4,3,2,1\}$, the noisy subspace ($\{x_3, x_4\}$, $\{y_1, y_2\}$) is seen as a candidate block with φ_B=$\{1,2\}$. As biclustering solutions are naturally prone to noise, a minimum area threshold should be considered to increase the probability that a candidate block is in fact participating in the plaid model and not appearing by chance.

In the final step, a matrix with a composition (e.g. sum) of the contributions from the elements of the candidate blocks is computed and tested against the original matrix. Figure 4.4 illustrates how these steps can be applied to identify additive contributions from two overlapping biclusters.

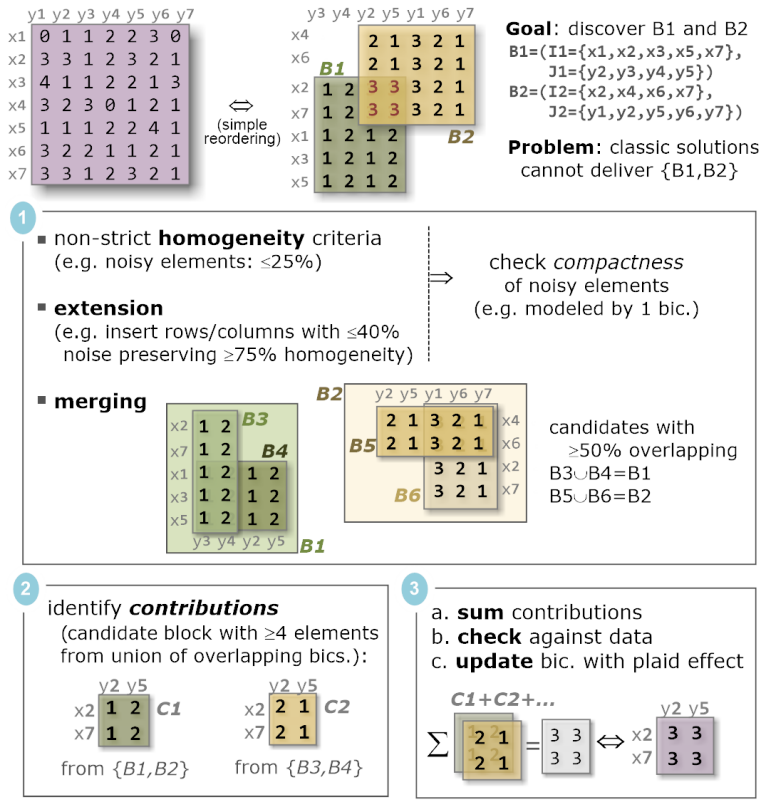

Figure 4.4: Learning plaid models from constant biclusters from local noisy areas (candidate biclusters) by computing their expected contributions and checking their additive composition.

4.2.1.2 Extending vs. Reducing Biclustering Solutions

The properties of the biclustering solutions delivered by the selected biclustering approach determine whether these solutions are predominantly extended or reduced to satisfy plaid assumptions. Illustrating, the biclusters discovered by pattern-based approaches do not accommodate noisy elements (biclusters fully respect the φ_B pattern [500] or alternative range-based criteria [509]) and therefore are predominantly extended for the recovery of plaid effects. However, these biclusters can still be reduced if the areas where the original biclusters overlap are not described by cumulative transcriptional effects.

We propose an alternative behavior to this class of pattern-based approaches by deriving noise-tolerant biclusters from associations rules. As covered in *Chapter III-2*, association rules can be used to capture a form of local noise when confidence levels are below 100% (Figure III-2.4). Confidence and other measures of interestingness are thus seen as the homogeneity criterion to control the amount of allowed noise. Additionally, principles to avoid the explosion of rules are critical, including the search for minimal non-redundant rules or pruning of rule based on their statistical/biological significance [20].

Understandably, the first step of the introduced methodology is optional for such noise-tolerant solutions. Interestingly, the allowed noisy elements are coherently located within the bicluster. Thus, there is no need to guarantee their compactness and there is a high chance that the area defined by these elements has structural significance. In this context, the solutions produced by this alternative class of pattern-based biclustering approaches are more prone to be reduced than extended, since the allowed noisy areas within each bicluster may not satisfy a plaid assumption. These introduced classes of approaches are illustrative of the importance, simplicity and diverse behavior of the proposed strategies to adjust biclustering solutions.

4.2.1.3 Remaining Challenges

A major drawback associated with the introduced solution is that it requires the sum of contributions from elements of candidate blocks to exact match the observed values in order to consider these elements to be part of a plaid model. This is an unrealistic assumption due to two major reasons. First, the additive matching assumption has low probability of occurrence when elements from multiple candidate blocks are involved. Second, some of the retrieved candidate blocks may not participate in the plaid model (false positives) and additional candidate blocks that participate in the plaid model may not be retrieved (false negatives). These two problems are tackled respectively in *Section 3.2* and *Section 3.3*.

4.2.2 Flexible Composition of Biclusters

Despite the relevance of assuming an additive composition of biclusters' contributions to guide the learning of plaid models, this assumption is only meaningful in the presence of certain relaxations. For illustration purposes, we use the task of learning plaid models from gene expression data as a motivational basis for the contents introduced in this section. In this context, consider the presence of a specific condition where three biological processes (biclusters) are active, each causing the up-regulation of a common subset of genes. Consider three additional conditions where each of these processes is uniquely active. Although we may observe that the expression level of the shared genes is greater in the first condition than in the remaining three conditions, in real settings their expression level in the first condition will hardly match the sum of the expression levels associated with the last conditions. A context where this phenomena is observed is when a gene shows a highly differentiated expression in the context of a single process. In this context, the activation of additional biological processes that also activate this gene will not make a significant difference over its already highly differentiated expression. Concluding, although existing plaid models are adequate in assuming plaid effects from overlapping areas, the additive expectation is rarely observed in biological contexts. Instead, we expect a non-linear cumulative effect in the expression of genes

that are active in multiple processes. To tackle this challenge, we propose both a set of relaxations for checking plaid effects and new functions (beyond the additive assumption) to compose plaid models.

4.2.2.1 Noise-tolerant and *In-Between* Relaxations

The introduced validation of plaid effects can be replaced by noise-tolerant matchings and meaningful relaxations, such as validating whether the overlapping biclusters contribute to the increase of the expected expression levels of the shared genes up to the levels given by an additive model. Relaxations thus establish the maximum distance between a similarity metric applied between the observed values and expected values. Additionally, a relaxation can be seen as a combined set of constraints. The properties and integration of the selected constraints is of critical relevance as they can lead to structurally different plaid models.

The use of relaxations only affects the checking stage and thus can be easily applied. Although relaxations can be easily implemented within the proposed methodology, they can hardly be placed within peer biclustering approaches that produce plaid models since modeling these relaxations in the underlying optimization equation (4.1) is non-trivial.

In this work, we propose two major sets of relaxation options. First, *noise-tolerant* relaxations allow approximate matchings. This type of relaxations can be used to guarantee the acceptance of either slight or large deviations from the expected composed transcriptional activity. These deviations can be assessed based on the amount of candidate elements that do not satisfy the plaid assumption, on minimum distance thresholds for testing candidate elements (either individually or the sum of deviations), among others.

Second, *in-between* relaxations guarantee that overlapping biological processes have impact in the expression of a gene but in a non-linear (possibly residual) way. This is accomplished by verifying that the observed values from the candidate elements are higher than each individual contribution and lower than their sum. This last relaxation accounts for non-cumulative effects from the involved contributions, more in line with biological contexts. Figure 4.5 shows the impact of using some of these relaxations on the recovery of planted biclusters.

4.2.2.2 Non-Additive Plaid Models

An additional option is to consider alternative functions to compose contributions from overlapping biclusters. These functions can be applied in both the absence and presence of relaxations. The adaptation of these functions affects only the third step of the proposed methodology, where the matrix of expected values is computed from candidate contributions, and thus it can be easily implemented.

Possible functions of interest for composing contributions associated with plaid models include:

- weighted functions: $\sum_{k=0}^{K} \rho(\nu_{ij})\theta_{ijk}\rho_{ik}\kappa_{jk}$,

  where ν_{ij} specifies the biclusters that include the a_{ij} element and $\rho(\nu_{ij})$ defines the weigth score based on the properties of these biclusters. Weights can be easily assigned based on the number and values of the overlapping biclusters, or more expeditely defined based on the expected cumulative effect a (biological) process will have on another process. In expression data analysis, these weights can be alternatively determined by the expression profiles of the involved sets of genes or derived from background knowledge (retrieved from knowledge bases, literature and ontologies);
- scaling functions: $\prod_{k=0}^{K} \rho(\nu_{ij})\theta_{ijk}\rho_{ik}\kappa_{jk}$.

  The simplest scaling function is the non-weighted product of contributions. Weighted scaling functions are appropriate to model a variety of biological contexts, such as the ones that rely on the assumption that each contribution has a catalyzing effect on the expression of a gene. In these biological contexts, normalizing the input data is critical to calibrate the distribution of values in the matrix below and above 1 (respectively associated with repression and activated regulatory mechanisms).

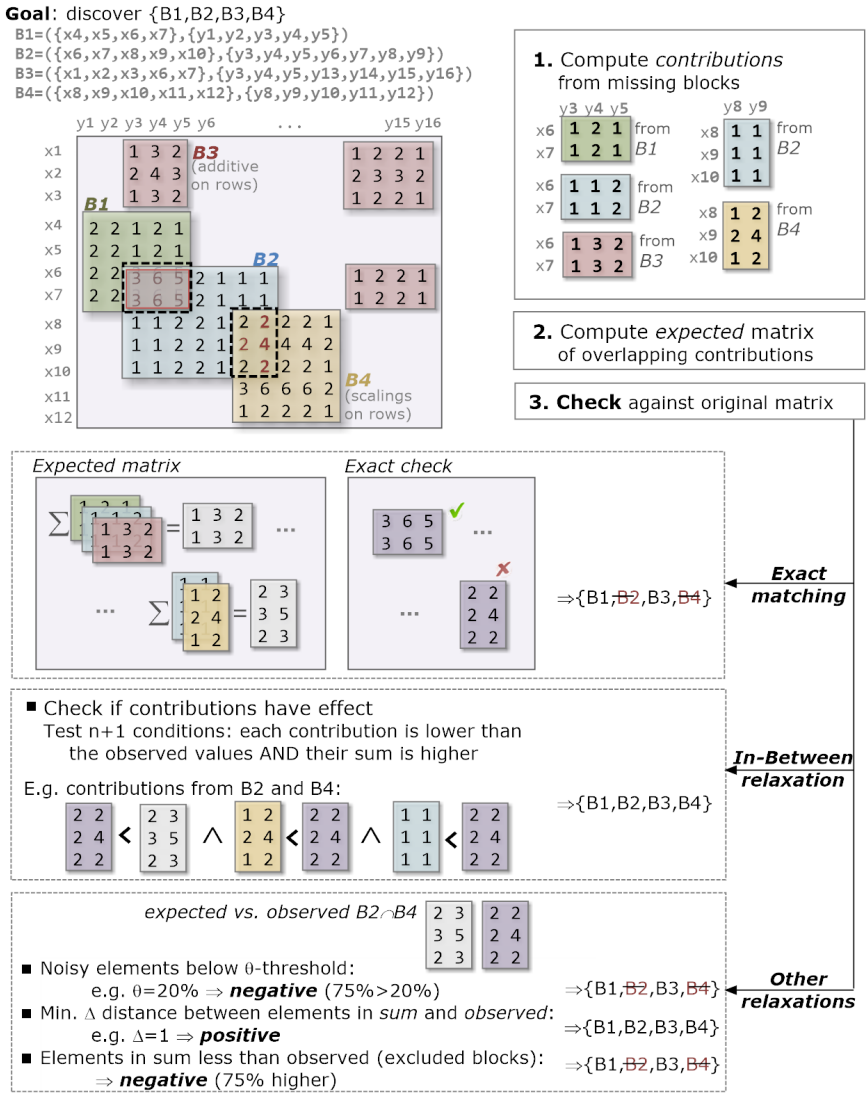

Figure 4.5: Impact of using (biologically) meaningful relaxations for the recovery of (hidden) biclusters described by a plaid assumption with non-exact matchings.

4.2.2.3 Flexible Types of Biclusters

Complementarily to the use of relaxations and advanced composition functions, it is essential to define the expected coherencies of the discovered biclusters. Figure 4.3 illustrates how different types of biclusters can be combined with alternative functions to compose overlapping contributions. Although according to the original formulation (4.1), θ_{ijk} allows for both constant and additive coherencies, the existing algorithms have been tuned to learn plaid models from biclusters with constant assumption [573]. However, non-constant assumptions are relevant across different data domains (*Chapter III-3*). Illustrating, a biological process that regulates two genes may activate their expression levels in different degrees or even activate one gene and repress the other. This behavior supports the need to adapt the learning of plaid models to support the discovery of additive, multiplicative and symmetric coherencies. Illustrating, additive and multiplicative coherencies are useful to handle distinct amplitudes of gene-activation in the context of a single biological process and the discovery of symmetric coherencies is useful to simultaneously capture activation and repression mechanisms.

Interestingly, *Chapter III-3* shows that pattern-based biclustering is well-prepared (under specific search variants) to support the enumerated coherencies [310, 311]. As long as BiP relies on such flexible biclustering solutions, it feasibly supports multiple types of biclusters.

4.2.3 Complex Interactions

The initially proposed methodology is not able to effectively learn complex plaid models. Complex plaid models are characterized by arbitrarily large sets of interacting biological processes with non-trivial overlapping aspects. An illustrative example is to consider that the set of interacting processes overlap simultaneously for a particular set of genes and conditions, while specific subsets of processes locally show pairwise overlaps on different sets of genes or conditions (see Figure II-3.5). In order to be able to learn plaid models with such complex interactions, we use an heuristic that incrementally adjusts the plaid model. Similarly to the adjusting schema proposed by Segal et al. [573], this heuristic is used to iteratively remove contributions from candidate blocks and to identify whether the already discovered biclusters can be further extended or not. When a bicluster is extended, further candidate blocks can be discovered and the area of some of the already discovered candidate blocks can be updated. These steps are performed for all candidate blocks, from larger to smaller until all have been covered.

These steps increase the ability to deal with complex interactions between biological processes as they rely on the assumption that by removing overlapping layers, the residual values become closer to the underlying unstructured noise. Figure 4.6 provides an illustration of this behavior for a scenario with complex interactions, where iterations are shown to be necessary for the retrieval of the original biclusters.

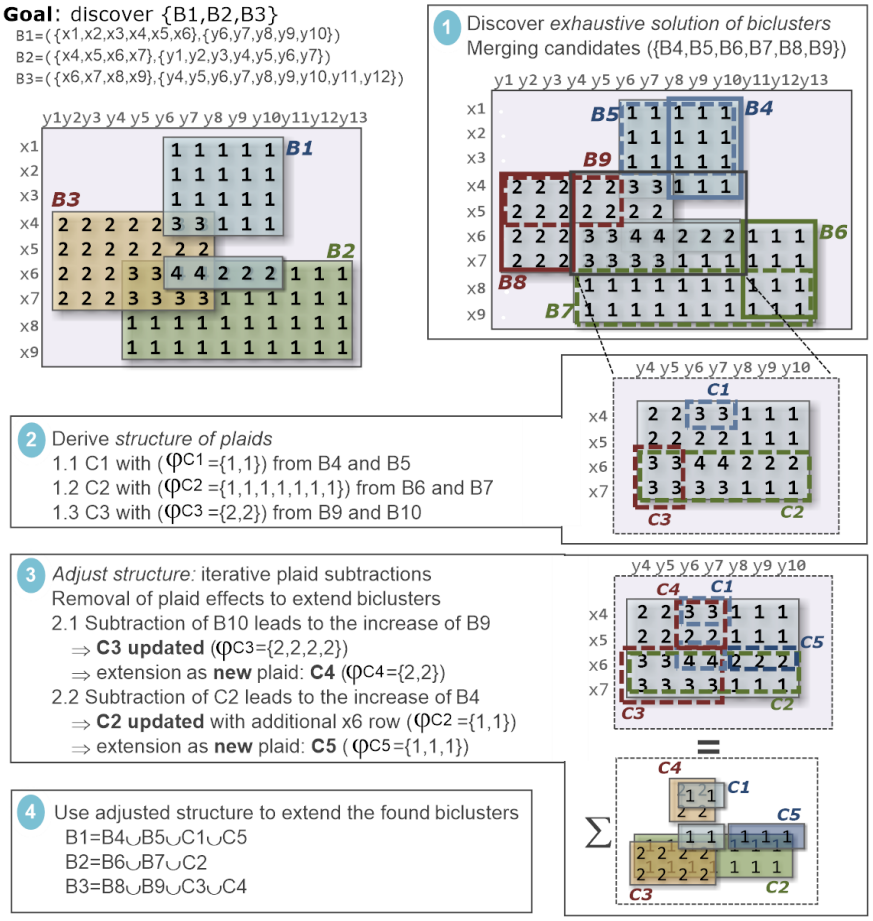

Figure 4.6: Illustrative application of the iterative adjustments over the initial structure of biclusters to deal with complex plaid models.

The introduced principles are coherently integrated in BiP, the proposed algorithm to learn robust and flexible plaid models. The core behavior of BiP is formally described in Algorithm III-3.

4.2.4 Modeling Dependencies between Biological Processes and Social Communities

A direct implication from the proposed variants of plaid models is that they can be used to further investigate meaningful dependencies between biclusters in real data contexts. These dependencies can take multiple forms

Algorithm 3: BiP core algorithmic steps.

Input: A
Output: $\mathcal{B}$
$\mathcal{B} \leftarrow$ getCompleteSetOfMaximalBics(A); *//e.g.* BicPAM(A)
repeat
  $S \leftarrow \emptyset$;
  $A^{plaid} \leftarrow$ zeros(A);
  foreach $(B_H, B_V) \in candidatePairs(\mathcal{B})$ **do**
    if $|B_H \cap B_V| > minOverlap$ **then**
      //candidate block from horizontal B_H and vertical B_V bic.:
      $C \leftarrow (I_H \setminus I_V, J_V \setminus J_H)$;
      if $C \neq \emptyset \wedge |C| > minArea$ **then**
        //recovering C pattern from mode of B_V columns:
        $\varphi_C \leftarrow \cup_{j \in J_V \setminus J_H} \{mode(\cup_{i \in I_H} \{a_{ij}\})\}$;
        //update C rows with the contributing pattern:
        foreach $a_i J \in C$ **do** $a_i J \leftarrow \varphi_C$;
        //add C to plaid matrix using f composition:
        $A^{plaid} \leftarrow f(A^{plaid}, C)$;
      $S \leftarrow S \cup \{B_H, B_V, C\}$;
  $A^{old} \leftarrow A$;
  foreach $\{B_H, B_V, C\} \in S$ **do**
    valid $\leftarrow$ true;
    foreach $(x_i, y_j) \in C$ **do**
      valid $\leftarrow$ valid $\cap$ relaxationCheck(a_{ij}, a_{ij}^{plaid})
    if valid /*merge bics and remove contributions*/ **then**
      $\mathcal{B} \leftarrow (\mathcal{B} \setminus \{B_H, B_V\}) \cup \{(I_H \cup I_V, J_H \cup J_V)\}$;
      $A \leftarrow f^{-1}(A, C)$; *//remove contributions using f^{-1}*
  $\mathcal{B} \leftarrow$ updateAndExtendStructure(A,$\mathcal{B}$);
until $A^{old} = A$;

ranging from simple relations, such as *is-part-of* or *exchanges-with*, to more complex relations characterized by a set of cascades of responses.

In social data contexts, is-part-of and exchanges-with are meaningful dependencies between social communities biclustered, for instance, from social network data. In biological data contexts, the proposed plaid models can be used to characterize regulatory interactions between biological processes given by biclusters from expression data. Figure 4.7 illustrates two biological processes with dependencies described by a is-part-of relation. The flexibility of the proposed plaid models allows the systematic study of biological relations. Additionally, the nature of interactions among groups of biological processes can be further characterized based on the satisfied relaxations and on the most explanatory functions for the composition of overlapping transcriptional activity.

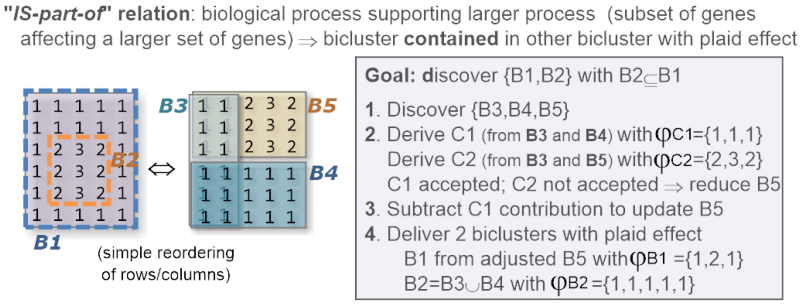

Figure 4.7: Illustrative discovery of *is-part-of* relation between two interacting biological processes.

An in-depth analysis of different biological and social relations is, however, out of the scope of this manuscript.

4.3 Results and Discussion

This section presents and discusses the results from experimentally assessing BiP's performance. First, we study the accuracy and efficiency of BiP over synthetic data with varying plaid structures against peer plaid approaches and state-of-the-art biclustering algorithms. Finally, we present preliminary results on the biological relevance of BiP

models (under alternative relaxations). BiP was implemented in Java (JVM version 1.6.0-24) and the experiments were performed using an Intel Core i5 2.30GHz with 6GB of RAM.

Datasets. To study BiP's performance, the synthetic data settings used in previous chapters are here further parameterized to incorporate plaid effects. The plaid structure was essentially controlled using the following four variables (further details provided in *Chapter II-3*):

- overlapping degree among subsets of biclusters (as a percentage of the area of the biclusters), θ;
- average number of interacting biclusters, δ. To illustrate, consider a solution of 20 biclusters, from which subgroups of δ=5 biclusters have interactions with each other but not with biclusters outside the group. When this number matches the total number of biclusters, all processes influence each other forming a complex model of interactions;
- distribution of the overlapping areas between the δ biclusters, ϕ, where ϕ=100% means that the observed plaid effects are a composition of the contributions from all the δ biclusters and 0% means that plaid effects are mainly derived from pairwise overlaps;
- function for composing contributions, f, average weight per additional contribution, ν, and allowed noise on the composed value, ϵ. Illustrating, considering ν=0.8 and an element with contributions $\{\mu_1, \mu_2, \mu_3\}$ from three biclusters, means that the expected value for the element is between $\sum_{i=1}^{3} \nu\mu_i \pm \epsilon$ and $\sum_{i=1}^{3} \mu_i \pm \epsilon$.

Complementarily, we use two real datasets: *dlblc* dataset (660 genes, 180 conditions) with human responses to chemotherapy [18], and *gasch* dataset (6152 genes, 176 conditions) with Yeast responses to environmental stimuli [239], known to be characterized by the presence of biclusters with non-trivial coherencies [324, 59].

4.3.1 Results in Synthetic Data

To generating the complete structure of maximal biclusters for BiP, we used BicPAM [310] parameterized with the F2G search method (to optimally handle high-dimensional data) [315] and a non-fixed support (pattern support incrementally decreases by 10% until the output solution has over 200 maximal biclusters or an area coverage of the input matrix over 10%). BiP relies on an approximated validation of plaid effects by considering a maximum distance between expected and observed elements of 10% (one contiguous item). We compared BiP against BCPlaid, an improved implementation of the original plaid models [393] provided by Turner et. al [638] (available in the biclust package for R) using the best results from a sensitive analysis on its parameters. Similarly to other existing plaid models [573, 113, 270], BCPlaid aims to learn the model by minimizing the (4.1) error.

In Figure 4.8 we compare BiP's performance (in the absence and presence of iterative adjustments) against baseline BicPAM solutions and BCPlaid. Four major observations can be retrieved. First, results confirm the high accuracy of BiP in terms of MS($\mathcal{B}, \mathcal{H}$), that is, the discovered biclusters are well described by the planted plaid model (correctness), and MS($\mathcal{H}, \mathcal{B}$) score, that is, the planted biclusters can be mapped into a discovered bicluster (completeness). In the absence of iterative adjustments, BiP's performance slightly degrades for larger matrices since a higher number of biclusters is involved in additive overlaps (increased complexity of the plaid models). Second, BCPlaid's solutions are not competitive for two major reasons: they only guarantee local optima and thus a high number of rows and columns is lost per bicluster, and they require background values in the matrix to be approximately explained by a bicluster. Third, score penalization in BicPAM's solutions is associated with the loss of overlapping areas associated with plaid effects. This results in a higher number of smaller biclusters that penalizes the match scores. Finally, although BiP is less efficient than BCPlaid, we observe that the recovery of plaid effects by BiP does not affect the scalability of the searches. This is due to the fact that the three steps within each iteration are efficient, each new iteration is significantly more efficient than the previous, and the need for new corrections rapidly converges (few iterations).

Varying Extent and Complexity of Interactions. In order to assess BiP's accuracy over data contexts following

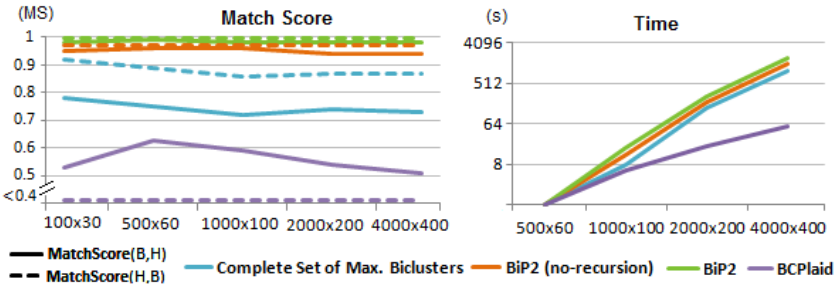

Figure 4.8: Comparing BiP's accuracy and efficiency when considering plaid effects for varying data settings.

plaid models with varying properties, we fixed the 1000×100 setting and varied two parameters. First, we varied the overlapping degree θ to affect the extent of plaid effects. Second, for a fixed number of interacting biclusters δ=5 (two groups), we varied the distribution of overlapped areas ϕ among the δ biclusters to affect the complexity of the plaid model. The match scores for these settings are provided in Figure 4.9. Similarly to the previous analysis, BCPlaid is not a competitive option for the tested parameters. Two additional observations are retrieved. First, the differences in performance between BiP and BicPAM (non-extended solution of biclusters) increase for high degrees of overlapping since plaid effects are associated with larger areas. Second, the differences in performance between BiP and BiP without recursion increase with a lower distribution of overlaps, since the iterative removal of plaid effects from the background values becomes critical to deal with the higher complexity of these datasets.

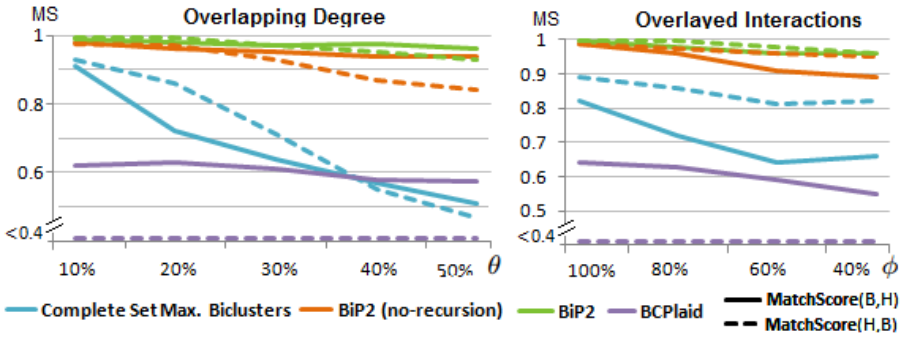

Figure 4.9: Assessing BiP's ability to learn plaid models with varying overlapping θ-extent and ϕ complexity.

Meaningful Relaxations. With the goal of studying BiP's performance in data contexts with several plaid effects, we varied the incremental weight per additional contribution, ν, and increased the associated noise, ϵ. For clarity sake, BCPlaid scores were excluded as they were not competitive. Figure 4.10 shows how the use of two alternative BiP relaxations – in-between validation and approximated matching (allowing sum of the expected values to differ up to 10% from the sum of the observed values) – are able to deal with these plaid effects.

Three major observations can be retrieved. First, BiP with exact matchings should be avoid as it is not able to model plaid effects with associated variability. Second, although the approximated matchings are useful to deal with the variability of noise associated with additive plaid models, they are not able to deal with other meaningful contexts, where new interactions have partial contributions on the overlapping area. This observation is confirmed by the higher number and smaller area of the discovered biclusters with a decrease in ν. Finally, although the in-between relaxation is the most adequate option for highly flexible plaid effects, it can lead to a deterioration of the match scores for a non-weighted sum of contributions. This happens since this relaxation tends to include in the solution space both excluded elements due to plaid effects and non-meaningful elements. This observation is confirmed by the larger areas of the discovered biclusters in comparison with the average area of the planted biclusters.

Non-additive Composition Functions. We tested the BiP's performance when dealing with matrices described by multiplicative plaid models and additive plaid models where the weight of each contribution is weighted by

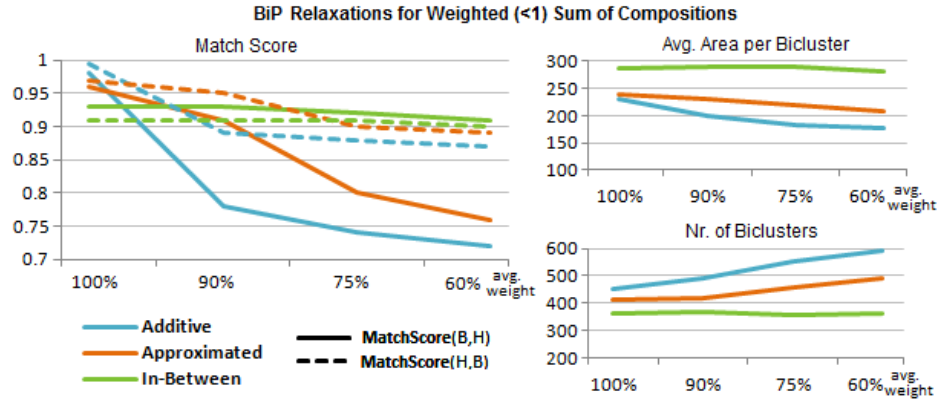

Figure 4.10: Impact of relaxations for learning plaid models in contexts with ε-noisy and ν-weighted plaid effects.

the number of elements in the involved biclusters (as a percentage of the larger contributing bicluster). Figure 4.11 illustrates the impact of considering non-additive composition functions within BiP using an approximated matching criterion (10% noise factor). The observed match scores show that the ability to change the composition function associated with the plaid model is critical to comply with alternative ways to compose contributions from overlapping biclusters.

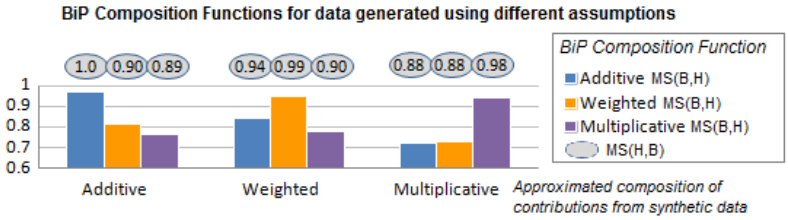

Figure 4.11: Impact of considering non-additive composition functions in data contexts with varying plaid effects.

Biclusters with Non-Constant Coherency. Although the existing biclustering approaches with a plaid assumption use general formulations that allow the discovery of non-constant biclusters, their implementations only support constant biclusters. This section shows that when using BicPAM (allows the discovery of non-constant biclusters [310]), we can easily recover excluded plaid effects, not only from constant biclusters but also from additive and multiplicative biclusters. For this analysis, we planted biclusters with additive coherencies, multiplicative coherencies, and a mix of all coherencies. Figure 4.12 illustrates BiP's performance for different data settings. Again, match scores from BCPlaid were excluded as their scores were significantly lower for constant and additive coherencies and nearly zero for the remaining coherencies. Two additional observations can be retrieved. First, the observed decrease in the match scores associated with non-constant biclusters is related with the higher probability of background values to form a non-planted bicluster. This undesirable effect is amplified in the presence of multiple types of biclusters. Finally, efficiency is penalized when non-constant biclusters are consider due to the additional complexity associated with their discovery.

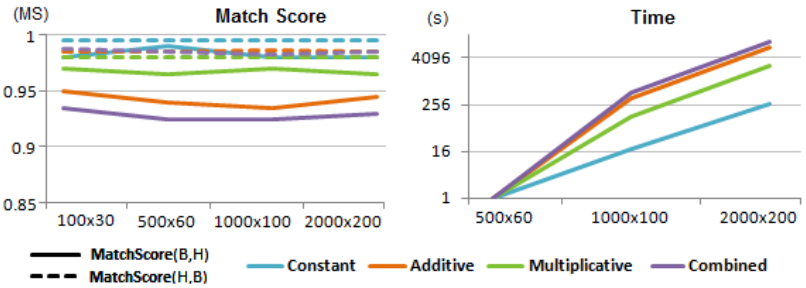

Figure 4.12: BiP's performance in datasets described by plaid models with non-constant coherencies.

Extending vs. Reducing Biclustering Solutions. In order to understand the impact of adjusting solutions that,

contrasting with BicPAM's solutions, can accommodate arbitrarily high levels of noise, we extended BicPAM to derive biclusters from association rules. Figure 4.13 compares BiP's performance to refine solutions that tolerate low-to-high levels of noise by varying the rules' confidence threshold (see Figure III-2.4). This analysis shows that low levels of noise-tolerance slightly degrades the accuracy as the biclusters associated with high-confident rules only accommodate a portion of the overall plaid effects. Low-confident rules can also slightly degrade performance as the signaled candidate blocks contain a mix of true and noisy elements. As such, although these solutions can accurately learn the plaid model, they tend to be associated with a higher number of redundant (imperfect) biclusters.

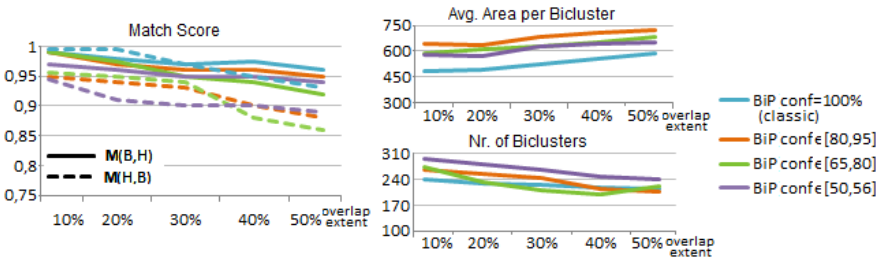

Figure 4.13: BiP's ability to adjust biclustering solutions with varying level of noise-tolerance/size using association rule mining with parameterizable confidence.

Comparison with Non-Plaid Models. Eight state-of-the-art biclustering approaches were used: FABIA [324], Bexpa [532], ISA [342], BCPlaid [638], OPSM [59], CC [134], Samba [616], and xMotifs [477] (parameterizations discussed in *Chapter III-2*). We also considered the non-adjusted solutions produced by BicPAM [310] and the proposed variant based on association rules. Figure 4.14 compares the ability of these state-of-the-art approaches to discover planted biclusters with constant coherencies for data settings with varying size, ϵ=0.1 noise factor, and an average of 20% overlapping areas described by an additive composition with ν=0.8 (adjusted weight per additional layer) with δ=0.3K interdependent biclusters following a ϕ=0.8 distribution of interactions. Results confirm BiP's superior performance both in terms of MS($\mathcal{B},\mathcal{H}$) (correctness) and MS($\mathcal{H},\mathcal{B}$) (completeness), as it provides exhaustive and flexible searches able to learn plaid models with complex interdependencies. The performance of the compared approaches is penalized by their inability to check plaid effects (even when described by pairwise interactions) and to select an adequate number of expression levels, often resulting in large and noisy biclusters.

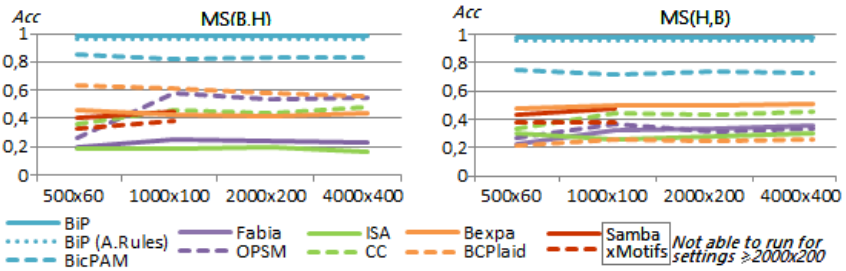

Figure 4.14: Comparing BiP with state-of-the-art biclustering methods.

4.3.2 Results in Real Data

The *biological relevance* of BiP's solutions was statistically assessed by assessing each bicluster against Gene Ontology (GO) terms using GoToolBox [440] to test their over-representation (functional enrichment). We assessed solutions delivered by BiP with: 10% noise-tolerant checks, BicPAM searches (excluded plaid effects) and the in-between relaxation. To compose BiP's solutions, biclusters with a small number of conditions and coherency across rows were not discarded due to their small area or overlapping degree with other biclusters. Instead, these biclusters were used to guide the discovery of relevant areas with plaid effects. Note that, since the discovery of these

biclusters with a small number of conditions is part of the biclustering method, BiP is able to exhaustively mine these datasets in few minutes.

An initial view on the biological relevance of BiP's solutions is synthesized in Table 4.1 for $|L|=7$ levels of expression and merging option. From this analysis we observe that the correct recovery of excluded elements from biclusters due to plaid effects leads to solutions with structural differences and a higher number of significantly enriched terms.

Data	*Approach*	*Avg.♯Genes ×♯Conds*	*♯Sig.Bics (no plaid)*	*♯Sig.Bics (plaid)*
gasch	BiP In-between	392×7	54	61
gasch	BiP Approximated	378×7	54	58
gasch	BiP f-Multiplicative	383×6	54	56
dlbcl	BiP In-between	82×6	17	19
dlbcl	BiP f-Weighted	81×6	17	19

Table 4.1: Biological relevance of BiP's models based on the number of biclusters with significantly enriched terms.

Table 4.2 shows basic properties of BiP's biclusters with terms only enriched under a plaid model. We can observe that such biclusters could not be discovered by existing plaid models provided by peer algorithms. This was possible due to the flexibility associated with the allowed types of biclusters, alphabet length (number of expression levels) and target relaxation criteria. Understandably, the search for biclusters with biological properties of interest can be guided by parameterizing these variables as well as the composition function. This supports the adaptability of plaid models to comply with the specificities of different data contexts.

ID	*Data*	$\|L\|$	*Type*	*Composition*	*♯genes ×♯conds*	*♯terms p<0.01 (candidates)*	*♯terms p<0.01 (plaid)*
B10	gasch	7	Additive	In-Between	559×6	51	58
B11	gasch	6	Constant	Approximated	482×9	5	6
B12	dlblc	5	Constant	In-Between	142×5	8	10
B13	dlblc	5	Additive	Approximated	118×5	5	11

Table 4.2: Illustrative biclusters with GO terms significantly enriched under a relaxed plaid assumption.

Table 4.3 characterizes illustrative sets of overlapping transcriptional modules found in the *dlblc* and *gasch* datasets together with the significantly enriched terms per module (bicluster). The *gasch* dataset defines several groups of conditions, each group either measuring responses over time (4-to-12 time points) or alternative stimuli (5-to-9 conditions). The proposed plaid models can be applied across all the groups of conditions to retrieve, for instance, biclusters with meaningful plaid effects associated with specific processes that may be active in one group of conditions but absent in other. The statistical results from this analysis are provided in Tables 4.1 and 4.2. Contrasting, for simplicity sake, Table 4.3 shows the application of the plaid model within each group of conditions for the discovery of biclusters with plaid effects associated with: 1) specific time points corresponding to time periods when another processes are still active, or 2) variants of a particular stimuli that may trigger new processes that overlap with the default responses.

The results gathered in Table 4.3 provide further empirical evidence for the relevance of considering relaxed plaid models to unravel non-trivial functional dependencies. Although an in-depth biological analysis of these sets of biclusters is out of the scope of this paper, we briefly hypothesize the nature of the functional interactions per set. For this purpose, we used the modular decomposition of gene regulation activity provided by the target plaid models. Complementarily, we used GOrilla [194] and Yeastract [620] tools to explore the hierarchy of the enriched terms per bicluster in order to identify shared parental terms (specialization relations) or cross-term interactions (dependency relations) between the enriched terms.

The set S_0 from *dlblc* contains 2 biclusters, $\mathbf{B}_0$ and $\mathbf{B}_1$, discovered using 6 expression levels (3 up-regulated and 3 down-regulated). The elements of these biclusters show moderate activation ($\forall_{(i,j)\in\mathbf{B}_0\oplus\mathbf{B}_1} a_{ij} \in \{1,2\}$) and moderate-to-high activation in the areas where they overlap ($\forall_{(i,j)\in\mathbf{B}_0\cap\mathbf{B}_1} a_{ij}\in\{2,3,4\}$). The inclusion of the excluded

Data	*Genome*	*Set ID*	*Biclusters*	♯*shared Genes*	♯*shared Conds*	♯*Levels of Expression*	*Relaxation*
dlblc	human	S0	{B0,B1}	16	4	6	Approximate
		S1	{B2,B3,B4}	{54,59}	{3,3}	8	In-Between
gasch	yeast	S2	{B5,B6,B7}	{87,73,78}	{3,3,3}	7	In-Between
		S3	{B8}	–	–	7	Approximate
		S4	{B9,B10}	58	3	8	In-Between
		S5	{B11,B12}	132	4	8	f-Weighted

Bic ID	*Pattern* φ_B	*Overlapping expression*	♯*Genes*	♯*Terms* $<10^{-3}$	*Notes*
B0	U1U2U2U2	{U2,U3}	32	12	*dependency* interaction
B1	U2U2U1U1	{U2,U3}	39	7	
B2	U–U–U–	{U4}	54	27	*is-part-of* relation with B2⊆B4 and B3⊆B4
B3	U–U–U–	{U3,U4}	59	25	
B4	U2U3U3U4U3	{U3,U4}	81	33	
B5	U1U2U2U2	{U2,U3}	145	9	pairwise *dependency* interaction
B6	U2U2U2U2	{U2,U3}	218	17	
B7	U2U2U2U1	{U2,U3}	185	33	
B8	D1D1D1D1	{D2,D3}	191	22	*extended* bicluster
B9	D2D2D2D2	{D2,D3}	121	14	*dependency* interaction
B10	D2D2D2D2	{D2,D3}	131	26	
B11	D–D–D–	{D2,D3}	127	14	*is-part-of* relation with B11⊆B12
B12	D1D1D1D1D1	{D2,D3}	197	21	

Table 4.3: Properties of the 6 illustrative sets of overlapping biclusters.

overlapping areas for each bicluster leads to significantly higher enrichments for their 10 top GO terms. The genes and conditions on the overlapping area can reveal how their processes are related, here – geometric change in DNA conformation ($\mathbf{B}_1$) and cellular metabolic processes involving organophosphate and compounds containing nucleobase or purine ($\mathbf{B}_0$) – seem to be associated with specific chemotherapy outcomes. The set S_1 defines three biclusters according to the is-part-of relation ($\mathbf{B}_2$ and $\mathbf{B}_3$ are contained in larger $\mathbf{B}_4$) with baseline activity $a_{(i,j)\in\mathbf{B}_2\oplus\mathbf{B}_3\oplus\mathbf{B}_4}\in\{2,3,4\}$ and activity in the overlapping areas $a_{(i,j)\in\mathbf{B}_2\cup\mathbf{B}_4}\in\{3,4\}$. $\mathbf{B}_2$ covers processes related with the positive regulation of the immune system, while $\mathbf{B}_3$ gathers genes that model the response to external biotic stimulus. $\mathbf{B}_4$ aggregates these biclusters, thus covering with higher significance more general terms related with the immune system process and signal transduction, naturally associated with chemoterapy responses.

The sets $S_2 - S_5$ from *gasch* were computed from groups of conditions related with Yeast response to (intense and soft) heat and cold. Biclusters with high differential expression impacted the resulting sets (e.g. $\mathbf{B}_8$ and $\mathbf{B}_{12}$ extensions were largely possible due to the recovery of areas from overlappings with such biclusters). However, we decided to exclude them from this analysis as the decomposition of their plaid effects is less interesting. The set S_2 contains 3 biclusters with pairwise interactions, $\mathbf{B}_5$, $\mathbf{B}_6$ and $\mathbf{B}_7$, each capturing a set of dedicated terms, but also sharing some terms (with the majority of the involved genes appearing in plaid areas) related with histone modifications and cytoskeleton-related processes such as cell wall chitin biosynthesis. Bicluster $\mathbf{B}_8$ from S_3 was extended from smaller biclusters (originally decomposed due to the presence of possible plaid areas). While some of these original (non-plaid) biclusters capture processes related with lipid functions such as the transport of ceramide and regulation of fatty acid biosynthesis, the resulting larger bicluster increases the significance of the terms and additionally captures microtubule polymerization for cytoskeleton resistance to the inputted shock and telomere maintenance as a response to the risk of damage. The set S_4 contains two biclusters with down-regulated genes: $\mathbf{B}_9$ with conditions spanning the initial time points and $\mathbf{B}_{10}$ spanning the latter time points. $\mathbf{B}_9$ and $\mathbf{B}_{10}$ intersect in the middle of the timeline, where the observed down-regulated expression is more accentuated. Finally, S_5 captures a 'is-part-of' relation between two biclusters. Interestingly, while the contained bicluster captures specific processes such as processes related with the anaphase of mitotic cells, the container bicluster includes some missing genes that participate in those processes (thus increasing the significance of their terms) and models the regulation of mRNA export from nucleus in response to heat stress.

4.4 Summary of Contributions and Implications

In this chapter, a new algorithm, BiP, was proposed to address the challenges associated with the task of biclustering data under a plaid assumption. For this aim, we first analyzed the major limitations of existing methods and contested their exact additive assumption to compose contributions from overlapping biclusters.

BiP is able to surpass the flexibility problems associated with existing approaches by starting the space exploration from a complete structure of maximal biclusters with constant, additive and/or multiplicative coherencies. Three principles are then used within BiP to enable the recovery of excluded areas due to plaid effects. First, local noise is used to collect candidate contributions for overlapping areas with plaid effects. These areas correspond to interactions among active biological processes associated with overlapping transcriptional activity. Second, the collected candidate contributions are tested under different relaxation against the observed values to verify if they are described by plaid effects. Multiple ways of composing contributions are proposed for a better modeling of complex biological interactions from different biclusters. Finally, the candidate areas associated with plaid effects are used within an iterative procedure to incrementally adjust the biclustering structures in order to be able to model complex web of functional interactions.

Results from an extensive set of synthetic datasets with planted biclusters with complex interactions confirm the BiP's ability to efficiently model the original biclusters. Results in both synthetic and real data also show that BiP can surpass the existing drawbacks of state-of-the-art approaches, namely BiP is able to retrieve an arbitrary number of biclusters, flexibly combine multiple types of biclusters, and show superior robustness against solutions with non-relaxed plaid constraints.

To our knowledge, this is the first biclustering approach that is able to support meaningful relaxations and weight contributions from overlapping areas thus offering a more realistic modeling of the interactivity between biological and social modules. Additionally, contributions from state-of-the-art biclustering algorithms can be easily accommodated within pattern-based biclustering searches to improve the initial structure of biclusters and thus the resulting plaid models.

Finally, we showed that the proposed plaid models are well-positioned for more in-depth transcriptional studies that aim to describe both simple and complex interactions between sets of transcriptional modules associated with related biological processes.

4.4.1 Future work

We highlight four major directions for future work. First, we aim to further assess the fit of different relaxations when modeling biological, medical and social data in order to provide a roadmap of data contexts where each relaxation is more likely to model the underlying regulatory dependencies.

Second, we aim to integrate contributions from constraint-based pattern mining in BiP in order to support knowledge-guided learning of plaid models. This extension opens doors for using implications from the learning of generative plaid models as constraints that can be provided as input to deterministic plaid models in BiP, thus combining the benefits of both existing generative views and proposed deterministic views.

Third, we plan to further explore the significance of weighted contributions in order to study the effects that a given biological process may have on another process. We expect also to assess the impact of using scaling functions over data contexts to study the effect of different stimuli on gene expression by assuming that the processes that become active under specific conditions have a catalyzing effect on the expression of the involved genes.

Finally, a last critical direction is to use the learned plaid models to systemically detail the nature and properties of the interactions between biological processes and social groups from the point of view of their regulatory/behavioral activity. This allows not only the identification and categorization of the interactions between functional modules, such as is-part-of or exchanges-with, but also provides insights to the understanding of complex regulatory/behavioral cascades of responses.

5

Scalable Pattern-based Biclustering

The modeling of regions of interest in (high-dimensional) data from frequent patterns has been increasingly accomplished in biomedical and social data settings, through the use of (but not limited to) pattern-based biclustering [578]. However, unlike pattern mining tasks, whose focus is placed on retrieving relevant patterns, a region/bicluster is additionally identified by the set of features associated with the pattern (subset of columns) and by its supporting observations (subset of rows). The task of jointly mining the frequent pattern, P, column indexes, Ψ_P, and supporting transactions, Φ_P, is referred as *full-pattern mining*. Illustrating, in gene expression analysis the interest is not so centered on the frequent expression profiles (pattern), but mainly on identifying the group of genes and conditions that support those expression profiles. In this data domain, full-pattern mining has been used in the past for biclustering [500, 578], gene association analysis [20] and integrative genomic studies [421].

The discovery of full-patterns can be naively given by a post-discovery of the transactions supporting a given pattern. However, this option becomes rapidly expensive for large outputs and prevents further opportunities, such as the accommodation of succinct constraints on the space of transactions. To address these observations, the search for full-patterns has been enabled recurring to the use of bitset vectors (or similar structures). Apriori and vertical-based searches for pattern mining can easily rely on these structures to deliver full-patterns. However, bitset vectors offer efficiency bottlenecks in terms of memory and time (expensive intersections) for high-dimensional and/or dense datasets.

Furthermore, the problem of guaranteeing the scalability of biclustering for data with arbitrary-high size and dimensionality has been poorly studied.

To address these problems, this chapter proposes two major contributions. First, we propose a new full-pattern miner, referred as F2G (Frequent Full-pattern Growth), surpassing the existing bottlenecks. This is done by extending pattern-growth searches using revised frequent-pattern trees (FP-trees) with annotated transactions. We provide guarantees the extended trees have minimum memory overhead, and that the storage, traversal and retrieval of transactions is done with heightened efficiency. Second, we survey scalability principles from research in pattern mining and show their compliance with the proposed search.

Results confirm the relevance of the F2G algorithm for dense data, high-dimensional data, data with large biclusters and full-pattern mining searches with low support thresholds. The performance improvements against peer searches on both synthetic and real data are measured, as well as its impact on pattern-based biclustering.

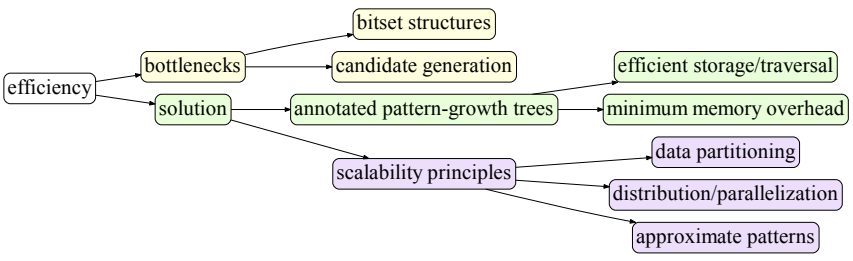

Figure 5.1: Proposed principles to address efficiency bottlenecks state-of-the-art miners and to guarantee the scalability of the searches for hard settings.

Figure 5.1 provides a structured view of the contents covered in this chapter. *Section 5.1* formalizes the full-pattern mining task, as well as the contributions and limitations from existing research for its answer. The proposed F2G algorithm is described in *Section 5.2*. The gathered results from assessing F2G against state-of-the-art alternatives are synthesized in *Section 5.3*. Finally, the major conclusions and implications are drawn.

5.1 Background

Recovering Defs.III-1.2 and III-1.3, we defined a transactional database $\mathbf{A}$ as a set of transactions with items from an alphabet $\mathcal{L}$, and the specific case where a pattern P is given by an itemset (possibly) contained in some transactions. The pattern length is given by the number of items, $|P|$, and the pattern support by the number of transactions containing P (coverage), $|\Phi_P|$. In the context of pattern-based biclustering, we further introduced two concepts: 1) the indexation operator ψ_P (Def.III-1.7), which defines the set of features associated with a given pattern, and 2) the true pattern Υ_P (Def.III-1.7), which defines the bicluster's pattern, $\varphi_B = \Upsilon_P$. Accordingly, Def.5.1 formalizes the task of full-pattern mining. Although Def.5.1 presents the paradigmatic case where full-patterns are derived from frequent itemsets found in a transactional database, this definition can be extended for rules and structured patterns from more structured databases, such as sequential databases.

Def. 5.1 Given a transactional database, $\mathbf{A}$, and minimum support and pattern length thresholds, θ_1 and θ_2, the ***full-pattern mining*** task consists of computing the set: $\{(P, \Phi_P, \psi_P, \Upsilon_P) \mid P \subseteq \mathcal{L}, |\Phi_P| \geq \theta_1, |P| \geq \theta_2\}$.

Basics 5.1 Illustrating full-pattern mining concepts

For an illustrative transactional database $\mathbf{A}$=$\{(\mathbf{x}_1, \{y_1a, y_2c, y_3b\}), (\mathbf{x}_2, \{y_1a, y_2b, y_3c\}), (\mathbf{x}_3, \{y_1a, y_2c, y_3a\})\}$ with $|\mathcal{L}|$=3. For a minimum support θ_1=2 and no inputted minimum pattern length, the full-pattern mining task over $\mathbf{A}$ returns $\{(\{y_1a\}, \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}, \{\mathbf{y}_1\}, \{\mathbf{a}\}), (\{y_1a, y_2c\}, \{\mathbf{x}_1, \mathbf{x}_3\}, \{\mathbf{y}_1, \mathbf{y}_1\}, \{\mathbf{a,c}\})\}$.

5.1.1 Motivation

Applications. Besides the essential utility of discovering full-patterns to perform biclustering, full-pattern mining has been additionally motivated for data indexing, association analysis, clustering, network analysis, among other tasks [315]. The combined retrieval of a pattern and its supporting transactions is commonly relevant for web content mining, web usage mining and collaborative filtering, since the discovery of patterns is only meaningful when assessed against users' profile [197, 600, 538]. Additional applications include: high performance indexing mechanisms dependent on filtering criteria; mining block correlations in storage systems where space partitions have different requirements [687, 402]; and a wide-set of data contexts where the knowledge of transactions is used to assess specific properties of interest, including communication networks, remote sensors and online transactions in the financial market or retail industry [315].

The Problem. The naïve option to perform full-pattern mining is to apply a pattern mining search followed by a search to collect the supporting transactions for a given pattern. However, such option suffers from two major drawbacks. First, the post-collection of transactions implies a traversal of the transactional database multiple times (depending on the number and similarity of patterns), each time with matches performed on every transaction. Understandably, in the presence of a large number of patterns with a (possibly) high number of items (expensive matches), the computational complexity of this task can exceed the complexity of mining frequent patterns. In particular, if the computational complexity of full-pattern mining is bounded by this task, the incorporation of scalability principles for pattern mining searches are useless. Second, separating the disclosure of transactions from the core mining task prevents the use of (possibly) relevant constraints with nice properties. Constraints with nice properties can be used to prune the search space. An illustrative example are succinct constraints, which may require specific transactions of interest (such as a gene or individual) to support the outputted set of patterns.

Chapter 10 further expands on transaction-oriented constraints and motivated their essential role for learning tasks with domain knowledge.

5.1.2 Limitations of Related Work

The first proposal to full-pattern mining, referred as AprioriTID [9], combines the Apriori search with bitset vectors to track patterns coverage (transaction-sets). Illustrative implementations include LCM and CLOSE, used respectively by BiModule [500] and GenMiner [442] biclustering methods. They differ regarding the pruning methods applied over the itemset lattice of frequent candidates. Apriori-based methods generally suffer from the costs associated with the generation of a huge number of candidates for low support and/or pattern length thresholds, a common requirement in biomedical tasks [500].

A second approach is the use of vertical-based methods, mostly through the extension of Eclat [701] and Carpenter [506]. Since vertical-based methods rely on intersection operations over transaction-sets to generate candidates, they require structures (such as bitset vectors or diffsets) to maintain the coverage per pattern. MAFIA [106] is an illustrative implementation used by DeBi [578]. Vertical-based methods are not competitive with horizontal-based methods for datasets with either a high number of observations (when searching for patterns with coherency across rows) or high number of features (when searching for patterns with coherency across columns) [655].

In fact, both AprioriTID and vertical-based approaches suffer from the fact that, when the bitset cardinality becomes large, not only these structures consume a significant amount of memory, but also the intersection gets computationally costly. This is a critical problem when considering searches with low support and pattern length thresholds, lengthy patterns, and dense data.

Although a third class of pattern mining searches, referred as pattern-growth (see Table III-1.4), were shown to be able to better handle hard learning settings [292, 290], they have not yet been adapted for the discovery of full-patterns. In this context, a straightforward extension would be the annotation of each node on the underlying tree structure of pattern-growth searches (FP-tree) with its supporting transactions. However, this option leads to an impracticable additional memory complexity.

5.2 Solution: Handling Efficiency Bottlenecks

The study of efficient full-pattern mining alternatives is of critical importance for the analysis of data domains surveyed in Table I-1.1 since they are characterized by: 1) high-dimensionality, 2) the presence of (possibly) high number of biclusters, and 3) loose coherency strength commonly associated with dense data structures. As such, we propose F2G algorithm to seize efficiency gains in these settings (*Section 5.2.1*), and extend F2G to further incorporate scalability principles from related work (*Section 5.2.2*).

5.2.1 Full Frequent-Pattern Growth

In this section, we propose a variant of the frequent-pattern growth (FP-Growth) searches to deliver full-patterns with heightened efficiency. This algorithm is referred as *F2G*, *F*requent *F*ull-pattern *G*rowth. F2G does neither need to maintain bitset vectors for every frequent candidate nor rely on expensive candidate generation.

Similarly to the original FP-Growth method [292], F2G relies on a compact tree structure (FP-tree), which is recursively mined to enumerate all frequent patterns. Patterns are generated by concatenating the pattern suffixes with the frequent patterns discovered from conditional FP-trees where suffixes are removed. Suffixes are composed according to an ascending frequency order to prune the search space.

Constrasting, in F2G, the transactions are optimally stored since they appear at most once in the FP-tree. Thus, unlikely the original method, the transaction-IDs are not lost at the very first scan. Algorithm 4 describes F2G. The required adaptations, when taking FP-growth algorithm [292] as the basis of comparison, are highlighted in bold.

To illustrate the applied changes consider the revised (conditional) FP-trees provided in Figure 5.2, obtained from the application of Algorithm 4 over the following transactional database:
$\mathbf{A}=\{(\mathbf{x}_0,\{A,C,D,F\}),(\mathbf{x}_1,\{B,D,E,F\}),(\mathbf{x}_2,\{B,D,F\}),(\mathbf{x}_3,\{A,B,E\}),(\mathbf{x}_4,\{A,F\}),(\mathbf{x}_5,\{A,B,D,E,F\})\}$.

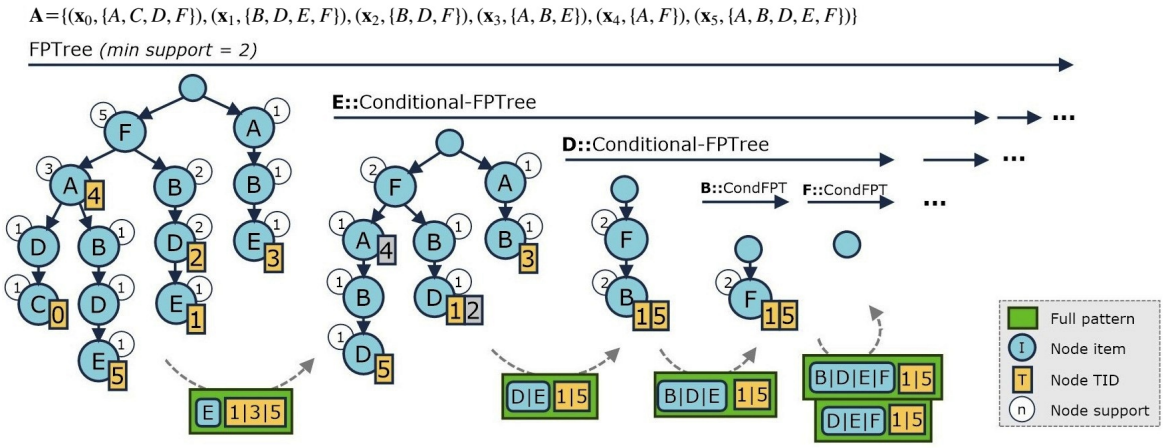

Figure 5.2: Illustrative behavior of F2G

The adaptations can be synthesized in three steps. First, the transaction-IDs are stored in the leaf node of an itemset path in the FP-tree without redundancy to guarantee optimal memory usage. To achieve this, the insertion of transactions in the FP-tree, addTransaction (*line 7*), is adapted so that the identifier can be added to the item-node with least frequency. Considering the {F,A,D,C} path from the original FP-Tree in Figure 7.3, the identifiers $\mathbf{x}_0$ and $\mathbf{x}_4$ are placed in the item-nodes *D* and *A*.

Second, before removing an infrequent node from a path (*line 16*), we need to exchange the transaction-IDs from the infrequent node to its parent (*line 17-19*). In this way the IDs assigned to the target suffixes rise from leafs to the root as long as shorter conditional FP-trees are built. Illustrating, since item *C* ($|\Phi_C|$=1) in the FP-tree from Figure 2 is infrequent (θ=2), $\mathbf{x}_0$ is propagated upwards.

Third, we recursively add the transaction-IDs assigned to the parental nodes of a frequent itemset in order to compute a full-pattern (*line 25*). For instance, before constructing the *D*-conditional FP-tree illustrated in Figure 5.2, the transactions on the nodes ($\mathbf{x}_1$ and $\mathbf{x}_5$) are added to the pattern composed by the *D* item and the {*E*} suffix. Also, with the addition of a full-pattern, the identifiers need to be copied to the parental nodes (*line 30*). For the illustrative case, $\mathbf{x}_1$ and $\mathbf{x}_5$ are propagated towards its parent nodes during the construction of the *E*, *D* and *B* conditional FP-trees. The computational time added by these extensions is residual when compared with the construction of conditional trees (*lines 29, 35*).

Algorithm 5 describes the methods related with the (conditional) FP-tree construction. The computational impact of changing the methods addTransaction (*line 11*) and addPrefixPath (*line 19*) is residual.

Note, additionally, that there is an additional key change to guarantee the validity of our proposal: the order of items in the header list of the conditional FP-trees should preserve the order observed in the original FP-tree (*line 21*). This is due to the fact that a potential reordering of least-frequent items (different from the original ordering) may no longer preserve transactions at the end of itemset paths. To illustrate this constraint, consider the *E* conditional FP-tree from Figure 5.2. If we had opted to not maintain the order of the item nodes, *B* would appear below the root node (since it becomes the item with higher support, $|\Phi_B|$=3), causing transaction $\mathbf{x}_5$ to be relocated. This constraint has a residual impact on the performance. On one hand it has a slight deteriorate effect, since the items in the tree may not be locally order by frequency. On the other hand, this constraint seize efficiency gains associated with the building of conditional FP-trees.

Algorithm 4: F2G Algorithm

```
   Output: FrequentFullPattern[] fullPatterns
1  Method: runFullPatternGrowthDiscovery
   Input: Transaction[] data, double support
2  Map<Int,Int> mapSup ← getItemsFrequency(data);
3  data ← removeInfrequentItems(data,mapSup);
4  data ← sortItemsets(data); //sort items in desc. freq. order
5  FPTree tree;
6  foreach Transaction trans : data do
7  |   tree.addTransaction(trans.itemset,trans.id);
8  tree.createHeaderList(mapSup);
9  F2G(tree, ∅, mapSup);

10 Method: F2G
   Input: FPTree tree, Itemset α, Map<Int,Int> mapSup
11 pruning(tree, α, mapSup); //FP-BONSAI optimization
12 if tree.hasSinglePath() then addAllCombForPath(tree.path, α);
13 else FPGrowthMultiplePaths(tree, α, mapSup);

14 Method: FPGrowthMultiplePaths
   Input: FPTree tree, Itemset α, Map<Int,Int> mapSup
15 foreach Int item : tree.headerList /*items in reverse order*/ do
16 |   if mapSup[item] < relativeMinsup then
17 |   |   foreach Node node : tree.getItemNodes(item) do
18 |   |   |   node.parent.trans←node.parent.trans ∪ node.trans;
19 |   |   |   node.trans = ∅;
20 |   |   continue;
21 |   β.values ← α ∪ item;
22 |   β.support ← min(α.support,mapSup[item]);
23 |   foreach Node node : tree.mapItemNodes.get(item) do
24 |   |   node.parent.trans ← node.parent.trans ∪ node.trans;
25 |   |   β.trans ← β.trans ∪ node.trans;
26 |   fullPatterns.add(β);
27 |   Path[] prefixPaths; //β cond. base (prefixes co-occuring with suffix pattern)
28 |   foreach Node node: tree.getItemNodes(item) do
29 |   |   Path path = node.getParentsUntilRoot();
30 |   |   path.trans ← node.trans;
31 |   |   prefixPaths.add(path);
32 |   Map<Int,Int> mapβSup ← getItemsSup(prefixPaths);
33 |   FPTree βtree; //β conditional FP-Tree
34 |   foreach Path path : prefixPaths do
35 |   |   βtree.addPrefixPath(path, mapβSup, θ);
36 |   βtree.createHeaderList(mapβSup, tree.headerList);
37 |   if βtree.hasNodes() then F2G(βtree, β, mapβSup);

38 Method: addAllCombForPath //recursively adds path nodes with prefix
   Input: Path path, Itemset α
39 Node node ← path.retrieveFirst();
40 β.items ← α ∪ node.item;
41 β.support ← node.counter;
42 β.trans ← node.trans;
43 fullPatterns.add(β);
44 if path.hasMoreNodes() then
45 |   addAllCombForPath(path, α);
46 |   addAllCombForPath(path, β);
```

5.2.2 Scalable Full-Pattern Mining

Over the past decades, multiple principles have been proposed to enhance the efficiency of pattern mining searches. Improvements to the performance of Apriori approaches (compliant with the bitset variant) include hashing techniques [510], partitioning techniques [566], sampling approaches [627], dynamic itemset counting [98], incremental mining [136], and level-wise approaches [242]. Vertical approaches have been extended with: the search

Algorithm 5: FP-Tree Construction Methods

```
1  Method: addTransaction
   Input: Itemset itemset, int tid /*transaction ID*/
2  Node node ← root;
3  foreach Int item : itemset.getItems() do
4      if node.hasChild(item) then
5          Node newNode ← createNode(item, node /*parent*/);
6          node ← newNode;
7      else node ← node.getChild(item) ;
8      if item==itemset.last() then node.trans ← node.trans ∪ tid;

9  Method: addPrefixPath
   Input: Path path, Map<Int,Int> mapSup, Int θ /*support*/
10 Node transNode;
11 foreach Node node : path.nodes() /*backward order*/ do
12     if mapSup.get(node.item) < θ then continue;
13     ... /* code for adding a path to a FP-Tree */
14     transNode ← node;
15 transNode.trans ← transNode.trans ∪ path.getTransactions();

16 Method: createHeaderList
   Input: Map<Int,Int> mapSup, Int[] headerListSuper
17 headerList ← getItemNodes().sortByIndexIn(headerListSuper);
```

for approximate patterns (colossal patterns) discovered by fusing small pattern fragments [714], data-driven parameterizations [507], top-down search strategies to make full use of the pruning power and to minimize the database scans [413], among others [148, 147, 447]. Improvements for pattern-growth approaches include depth-first searches [4], the use of hyper-structures [522], combined top-down and bottom-up traversal of trees [412], and array-based implementations of prefix-trees [268].

The demonstration that full-pattern mining is compliant with all the listed options is out of the scope of our work. Nevertheless, we focus on specific efficiency-enhancing principles and show that F2G can integrate these principles, further promoting its scalability. In particular, we briefly motivate how F2G can comply with the use of: 1) alternative pattern representations, 2) data partitioning procedures and 3) parallelization principles.

5.2.3 Condensed and Approximative Patterns

Depending on the pattern mining task, different patterns (simple, associations, sequences) can be considered, and each pattern can be classified according to whether it is condensed, pseudo-closed, rare, top-K, multilevel, erasable, and so forth [114, 519, 688]. The proposed BicPAM can be parameterized with full-pattern mining searches under these representations in order to achieve efficiency gains.

To illustrate F2G compliance with some of these pattern representations, let us start with two condensed representations: maximal (frequent pattern with infrequent supersets), and closed (frequent pattern with no supersets with the same support). These representations have been supported in full-pattern miners with AprioriTID searches (using LCM [642], A-Close [513], Charm [702], MaxMiner [50]) and vertical searches (using TD-Close [413], Mafia [106]), yet they are not supported by the as-is implementation of F2G. Existing extensions to FP-Trees for the discovery of condensed patterns include CLOSET+ [655], AFOPT [412], FPMax and FPClose [267]. In particular, FPMax and FPClose [267] use variants of conditional FP-trees, referred as M/CFI-trees (Maximal/Closed Frequent Itemset trees), which are optimally constructed to keep track of maximal/closed patterns. Since these trees preserve the item order of the original FP-tree header, the discovery of maximal/closed full-patterns can easily follow the extensions proposed in the F2G method. An alternative effective extension for the sake of mining condensed patterns is to rely on hybrid tree-projections [655] to detect and remove unpromising patterns as early as possible.

In the presence of very large data, the question of whether is it possible to derive a complete set of (condensed)

patterns efficiently can be replaced whether is it possible to derive a compact but high-quality set of useful patterns. Lossy condensed pattern representations, also referred as approximate patterns, can be used for this aim [290]. Pattern-growth searches have been also proposed for this end [290].

The problem of the majority of these representations is the loss of optimality guarantees, which is a necessary condition for the ultimate goal of this dissertation. Therefore, we suggest the parameterization of BicPAM and peer (pattern-based) biclustering algorithms with this option only when maximum-distances to optimal solutions are provided. In particular, BicPAM supports the use of Carpenter [506] and Cobbler [507] searches to compose large-sized biclusters and to learn from large-scale data.

5.2.4 Distributed Settings and Data Partitioning Principles

Multiple parallelization and distribution principles have been largely research for different pattern mining searches [511, 136, 8, 703]. In particular, many contributions have been proposed for the extension of pattern-growth searches for their effective parallelization [290]. Many of these principles assume that originally learned FP-tree is globally accessible, while the creation of conditional FP-trees and their traversal is parallelized and therefore assigned to dedicated processors [124, 699]. Under this principle, the compliance of F2G is easily demonstrated since the information regarding the annotated transactions is not lost or partitioned. Additional principles can be further incorporated to guarantee its application over distributed settings [124, 290].

Alternatively to the parallelization of pattern mining searches, data partitioning principles can be applied [352, 290]. The basic principle of data partitioning is to divide the dataset in a set of data partitions, apply pattern mining searches within each data partition (local patterns), and infer global patterns from the previously applied searches. Data partitioning has been mainly applied horizontally (by grouping observations), but it can be also applied vertically (by grouping features). The soundness of F2G with data partitioning principles is easily shown based on the two following simplistic steps. First, F2G is applied for each partition in order to retrieve sets of full-patterns. Second, global patterns are identified and the sets of supporting transactions across partitions are merged.

Understandably, by parameterizing pattern-based biclustering with pattern mining searches incorporating the previously proposed principles, the scalability of the pattern mining task is guaranteed in the presence of efficient closing procedures. The guarantees of efficiency from closing procedures are studied in *Chapter III-7*.

5.3 Results and Discussion

In this section, we first compare the performance of state-of-the-art implementations of Bitset Apriori and Eclat[1] with F2G on synthetic and real datasets. Second, we study improvements of pattern-based biclustering by parameterizing BicPAM with F2G. F2G was implemented in Java v1.6.0-24 and the experiments were computed using an Intel Core i5 2.30GHz with 6GB of RAM.

5.3.1 Results on Synthetic Data

In order to test the pattern mining searches, we maintained the properties of the generated data settings described in Table II-3.1, originally proposed for biclustering. Accordingly, patterns with different shapes (varying number of transactions and items) were planted, mimicking the local regularities commonly observed in expression data.

Contrasting, instead of generating real-valued datasets, discrete datasets with varying density were generate. The data density, the average percentage of total items per transaction, was varied from 5% (dense data given by an alphabet of 20 items per feature) to 33% (highly dense data given by an alphabet of 3 items per feature). Note that the considered coherency strength is agreeably seen as dense due to the fact that pattern mining is usually

[1] http://www.philippe-fournier-viger.com/spmf/

applied over transactions with short length from an alphabet of hundreds-to-thousands items. Below, we assess F2G in two steps. First, the density is fixed as 20%, and changes in performance are assessed for data with varying size and dimensionality. Second, the 1000×100 data setting is fixed and the density is varied from 5% to 33%.

In Figure 5.3 we assess the computational overhead of extending FP-Growth to perform full-pattern mining. F2G adds only a residual time and memory computational cost. The impact of sorting items in the header table is minor. Contrasting with F2G, the naive way of attaching and gathering transactions on the FP-tree nodes (extended FP-growth) strongly penalizes the performance.

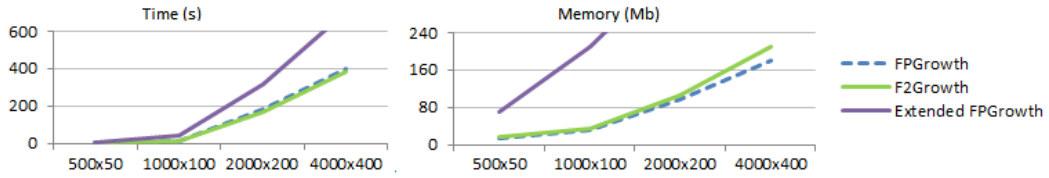

Figure 5.3: Efficiency of F2G and FPGrowth methods for datasets with varying size.

Figure 5.4 compares the performance of efficient implementations of the major full-pattern miners for different data settings in terms of time and memory. F2G shows significant gains in efficiency against Bitset Apriori and Eclat. The Apriori performance is penalized due to the need to generate a large number of candidate patterns when the number and length of frequent patterns is considerably high. The high-dimensionality of the datasets penalizes both the time and memory efficiency of Eclat due to the costs associated with the underlying bitset structures.

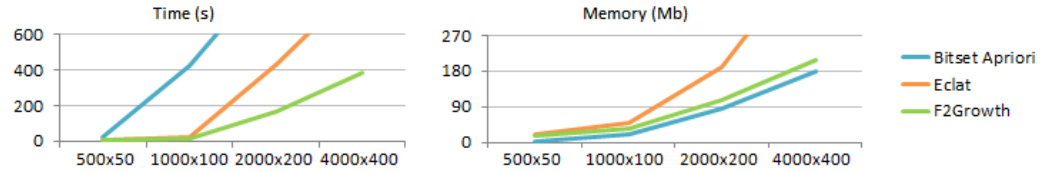

Figure 5.4: Efficiency of full-pattern mining methods for datasets with varying properties.

To further compare the performance of full-pattern mining methods, we fixed the 1000×100 experimental setting and varied the level of density/sparsity. This analysis is illustrated in Figure 5.5. Two main observations can be retrieved. First, F2G is the method more able to deal with dense datasets. Second, the use of horizontal searches for full-pattern mining should be preferred over vertical searches for the generated settings in terms of memory usage.

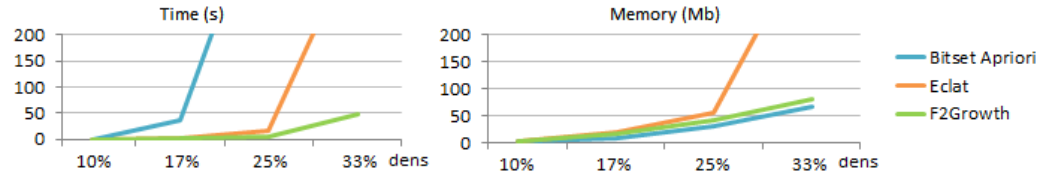

Figure 5.5: Efficiency of full-pattern mining methods for datasets with varying density.

5.3.2 Results on Real Data

To assess the performance of the target approaches in real data, we used gene expression datasets from BIGs repository[2], as well as medical datasets from the UCI repository after the preprocessing described in CP4IM project[3].

In particular, Figure 5.7 illustrates results for the yeast dataset for varying support thresholds. The same relative behavior can be observed across the remaining gene expression datasets from the BIGs. These datasets were discretized (assuming a Gaussian distribution of values and a target density of 10%) and, finally, column indexes

[2]Data available: http://www.upo.es/eps/bigs/datasets.html
[3]Data available: http://dtai.cs.kuleuven.be/CP4IM/datasets/

were concatenated with items, as illustrated in Figure III-1.3. Previous observations remain valid. F2G is the most efficient option in terms of time and, along with Apriori, a competitive choice for efficient memory usage.

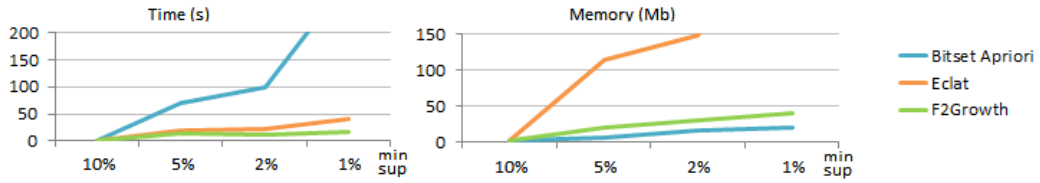

Figure 5.6: Efficiency of full-pattern mining methods for the yeast (2884×17) microarray

Complementarily, Figure 5.6 presents the performance of full-pattern miners on the UCI's primary-tumor dataset. Similarly, F2G offers the best compromise in terms of time-and-memory efficiency. The same relative behavior can be also observed across other medical datasets in UCI, such as lymph and heart-cleveland datasets.

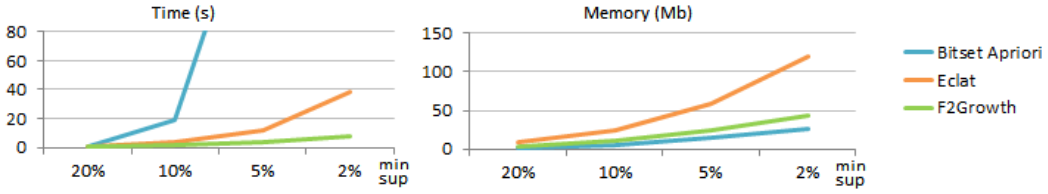

Figure 5.7: Efficiency of full-pattern mining methods for the primary-tumor dataset

5.3.3 Pattern-based Biclustering with F2G

Figure 5.8 assesses the impact of the full-pattern miner in the efficiency of BicPAM using the 1000×100 setting with δ=0.1 density/coherency strength and non-condensed pattern representations. F2G and Eclat are the most competitive choices when modeling regions given by lengthy patterns (commonly discovered under small support), while Apriori should be only chosen only for small patterns (commonly discovered under high support levels). In particular, F2G is the best performer for settings with numerous and lengthy patterns (supports near and below 1%), common for modeling biclusters with coherency across columns for data with high dimensiolaity (size).

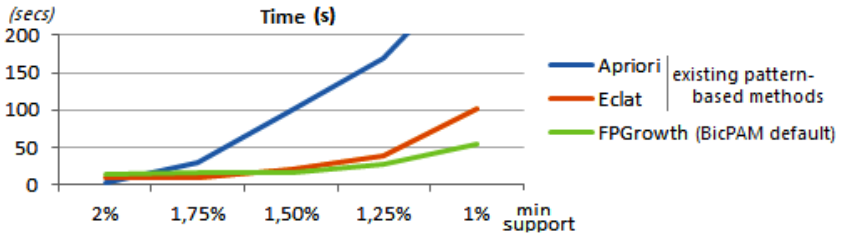

Figure 5.8: Comparison of BicPAM with different pattern mining searches for the 1000x100 setting.

The impact of choosing alternative pattern representations (simple, closed, maximal) in efficiency and MS levels is presented in Figure 5.9. For this assessment we used three distinct methods: F2G [292] to output simple and closed patterns, Charm [702] to output closed patterns and CharmMFI [702] to output maximal patterns. Similarly, we considered the 1000×100 setting and δ=0.1. Three main observations can be retrieved. First, the use of maximal patterns for biclustering should be avoided as it gives preference to biclusters with a large number of columns and discards biclusters with a subset of these columns even when they have a larger number of supporting rows. Understandably, this penalizes the $MS(\mathcal{H},\mathcal{B})$ levels. $MS(\mathcal{B},\mathcal{H})$ scores are not so affected as each maximal bicluster is covered by a planted bicluster. Second, the use of simple patterns for biclustering can degrade the $MS(\mathcal{B},\mathcal{H})$ in comparison with closed patterns. This score penalizes the discovery of biclusters contained in larger planted biclusters, even when the discovered biclusters have a heightened homogeneity. Third, the search for closed and maximal patterns is slightly more efficient than the search for simple patterns as a result of additional

pruning procedures. These observations support the use of closed patterns, corresponding to the search for maximal biclusters (Def.III-1.10).

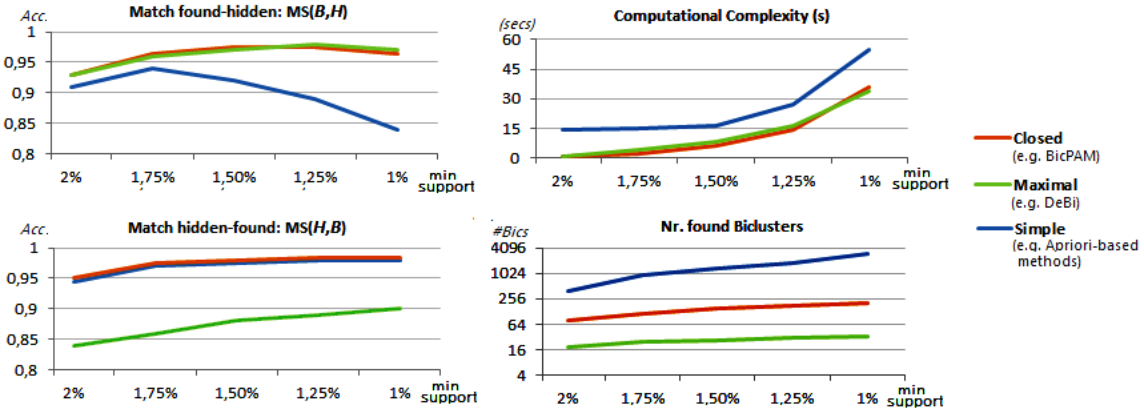

Figure 5.9: Impact of choosing alternative pattern representations over the 1000x100 data setting.

5.4 Summary of Contributions and Implications

This chapter motivated the task of full-pattern mining for the learning of local descriptive models across domains. The behavior of existing full-pattern miners is compared. To tackle their inefficiencies, we propose the F2G algorithm, a pattern growth algorithm that discloses the supporting transactions per pattern with minimum space overhead and heightened efficiency. For this purpose, F2G relies on FP-tree variants that store transaction-IDs without redundancy and on efficient propagation procedures.

Furthermore, we show that F2G can easily accommodate principles from existing research to deliver condensed full-pattern representations or to discover full-patterns in distributed settings.

Results on both synthetic and real datasets show the superior performance of the proposed method. In particular, F2G delivers a distinctive performance both for dense and large data and for inputted low support thresholds, common settings when learning from biomedical data. To the best of our knowledge, this is the first attempt to compare full-pattern mining algorithms.

6

Flexible and Robust Order-Preserving Biclustering

Throughout this book, pattern-based biclustering has been proposed as the means for learning flexible and robust local descriptive models according to constant, additive, multiplicative, symmetric and plaid coherency assumptions. Despite the relevance of these assumptions, certain regions across real data domains follow an alternative form of coherency assumption: order-preserving. A bicluster is order-preserving if there is a permutation of its columns under which the sequence of values in every row is (either monotonically or strictly) increasing. The task of learning order-preserving biclusters was originally proposed by Ben-Dor et al. [59] to find genes co-expressed within a temporal progression (such as stages of a disease or drug response) [59]. Yet, its range of applications goes beyond data contexts where time is absent. In particular, they have been also largely applied in static biological data contexts where gene expression or protein/metabolite concentrations coherently vary across samples [311]. Illustrating, detecting relative changes in the expression/concentration across conditions is typically indicative of functional regulatory behavior, surpassing the need to rely on the exact values that are usually noise-susceptible. As such, order-preserving leads to more inclusive solutions associated with larger and less noise-susceptible biclusters. The order-preserving assumption is also key to: discover coherent orders of preferences from collaborative filtering data [311]; support planning and scheduling tasks [308]; analyze chemical data [415]; and find sets of nodes in a (social and biological) networks with an preserving degree of influence on another set of nodes [313].

Despite the relevance of the pioneer approach to find order-preserving biclusters (OPSM) [59] and of its extensions [697, 209], this class of greedy approaches are only able to deliver approximate solutions (no guarantees of optimality), are biased towards larger biclusters, and place restrictions on the biclustering structures including the non-overlapping condition. Contrasting, a few exhaustive approaches have been proposed to identify the submatrices with largest areas that still respect the ordering constraints [415, 416]. Still, they are not robust to noise (partitioning of large biclusters in multiple smaller biclusters) and their efficiency strongly deteriorates for matrices with more than 50 rows/columns for an order-preserving coherency on columns/rows.

Additionally, the existing order-preserving approaches impose a monotonic ordering of values that does not allow for symmetries [59, 415]. However, their presence is critical for modeling meaningful regularities. Illustrating, for transcriptional activity analysis, regulatory and co-regulatory mechanisms are strongly correlated and, consequently, an increase in expression for some genes is sometimes accompanied by a decrease in expression for other genes.

To address these problems, this chapter extends pattern-based biclustering to perform flexible and noise-tolerant order-preserving biclustering. For this aim, pattern-based biclustering is parameterized with sequential pattern mining (SPM). Strategies are proposed to allow for symmetries and guarantee robustness to arbitrary-high levels of noise. These contributions are integrated within a new algorithm, referred as BicSPAM (Biclustering based on Sequential PAttern Mining).

Despite the inherent simplicity and power of these contributions, efficiency bottlenecks associated with the application of SPM appear in the presence of high-dimensional data and data with large order-preserving biclusters. Nevertheless, the mapped sequential databases from tabular datasets show unique properties of interest: item-

indexable properties. An item-indexable sequential database does not allow item repetitions per sequence. In this context, we additionally propose a new sequential pattern mining method, referred as IndexSpan, which is able to mine sequential patterns over item-indexable databases with heightened efficiency in comparison with the existing alternatives.

Accordingly, the major contributions made available in this chapter include:

[*Efficiency*]

- Principles to efficiently mine sequential databases with item-indexable properties, including minimal data-projections and compact data structures;
- Effective pruning of the search space in the presence of expectations on the minimum pattern length;
- New SPM algorithm (IndexSPAN) combining previous principles for efficient order-preserving biclustering;

[*Flexibility and Robustness*]

- Integration of the order-preserving coherency with symmetries and proposal of methods for its discovery;
- Parameterizable degree of co-occurrences versus precedences with impact on the coherency;
- Methods for the learning of flexible structures of biclusters with either strict or monotonic orderings;
- Strategies for the discovery of biclusters robust to aribitrary-high levels of noise and missings;

A structured view on how to combine these contributions for pattern-based biclustering is provided. To the best of our knowledge, IndexSpan is the first SPM algorithm able to seize efficiency gains from item-indexable properties, and BicSPAM algorithm is the first attempt to model order-preserving biclusters with symmetries and robust to varying levels of noise. Figure 6.1 provides a comprehensive taxonomy with the major and proposed contributions of this chapter.

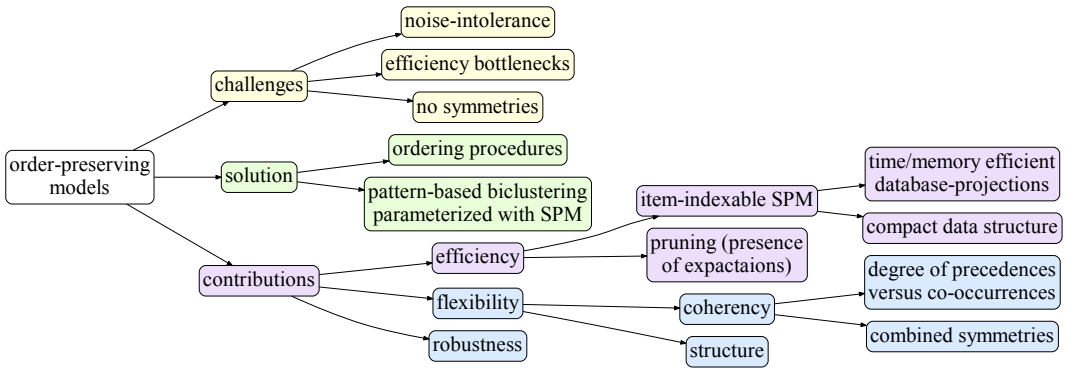

Figure 6.1: Major challenges and proposed contributions to guarantee the efficiency, flexibility and robustness of order-preserving biclustering.

The superior performance of IndexSpan against SPM peers is demonstrated on both synthetic and real datasets, and its relevance for multiple applications is discussed. Further empirical evidence shows the superior flexibility, robustness and effectiveness of BicSPAM. In particular, results on synthetic data confirm that BicSPAM is able to incorporate symmetries and is robust to (planted) noise, while results on relevance data confirm the relevance of these principles to gather putative modules with higher (biological) significance.

This chapter is organized as follows.*Section 6.1* provides background on order-preserving biclustering and item-indexable SPM, and lists the major applications of these tasks. *Section 6.2* proposes BicSPAM and extends its core behavior to satisfy three major requirements: scalability (*Sections 6.3.1* and *6.3.2*) flexibility (*Section 6.3.3*) and robustness (*Section 6.3.4*). *Section 6.4* provides an exhaustive assessment of BicSPAM on synthetic and real datasets. Finally, a summary of the contributions and implications of this work are provided.

6.1 Background

Follows an introduction to order-preserving biclustering and sequential pattern mining with item-indexable properties. The covered concepts provide the foundations for the target task of learning order-preserving models. Finally, we provide a case-by-case view of the most prominent applications of this task.

6.1.1 Order-preserving Biclustering

According to Def.II-3.2, a bicluster $(\mathbf{I}, \mathbf{J})$ follows an order-preserving coherency assumption when a subset of columns $\mathbf{J}$ (rows) induces a π linear ordering whose permutation is respected on a set of supporting rows $\mathbf{I}$ (columns). An order-preserving model defines a collection of order-preserving biclusters satisfying some criteria of homogeneity and (possibly) statistical significance.

Order-preserving biclusters can emulate the majority of the previously introduced coherency assumptions, leading to more inclusive solutions as illustrated in Figure 6.2. This offers a less noise-susceptible setting to recover larger biclusters, possibly embedding constant, additive and multiplicative models.

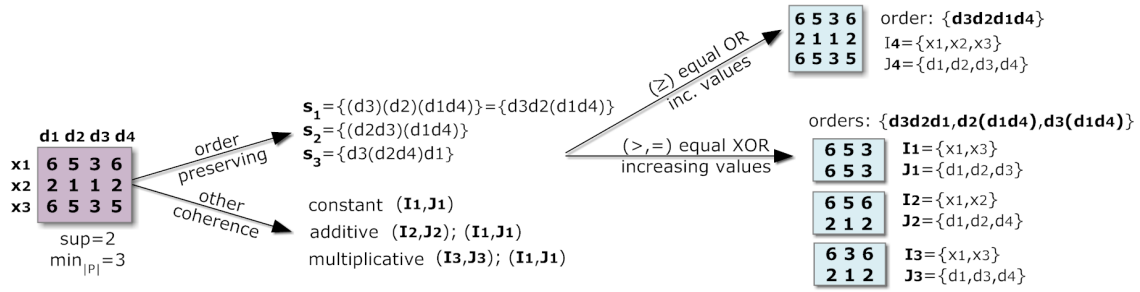

Figure 6.2: Inherent flexibility of order-preserving biclustering solutions and their major variants.
Order-preserving biclusters capture flexible coherency patterns from frequent orderings, covering constant, additive and multiplicative assumptions. They can either be mined across rows or columns, and follow the or behavior (not differentiating between increasing and preservation of values within a permutation of columns/rows) or the more specific xor behavior (requiring either increasing or preservation of values within a permutation of columns/rows).

Although absent from literature, two major types of linear orderings can be conceived: strictly increasing (xor behavior) and monotonically increasing (or behavior). Figure 6.2 illustrates how these different assumptions can lead to structurally different order-preserving models. In particular, when considering biclusters with monotonically increasing values, the permutation π=$\{y_3, y_2, y_4, y_1\}$ in Figure 6.2 becomes supported by all rows $\{x_1, x_2, x_3\}$.

6.1.2 Item-Indexable Sequential Pattern Mining

Revisiting the universe of discourse, let an item be an element from an ordered set $\mathcal{L}$, and an itemset P be a set of non-repeated items, $P \subseteq \mathcal{L}$. A *sequence* s is an ordered set of itemsets. A *sequential database* is a set of sequences.

Def. 6.1 Let $|P|$ be the length of an itemset, and $|s|$ be the length of a sequence defines as the total number of item occurrences: $\Sigma_j |s^j|$ (where $|s^j|$ is the length of the j^{th} itemset). An *item-indexable sequence* is a sequence without repeated items, $(\cup_{j=1}^{|s|} s^j)=\emptyset$. An *item-indexable sequence database* is a set of item-indexable sequences, $\mathbf{A} = \{s_1, .., s_n\}$.

Basics 6.1 Item-indexable sequential databases from tabular data

Tabular data can be mapped as a set of item-indexable sequences by ordering the column (row) indexes per row (column) according to the observed values. The analysis of item-indexable sequences allows the discovery of frequent precedences and co-occurrences associated with order-preserving biclusters. Illustrating, consider a tabular dataset given two genes with the following levels of expression across four conditions: respectively $\mathbf{x}_1$={-1,-1,2,0} and $\mathbf{x}_2$={0,0,2,-1}. The levels of expression for both $\mathbf{x}_1$ and $\mathbf{x}_2$ are preserved (co-occurrence) on the first and second conditions and coherently varied (precedence) on the third condition. These illustrative observations can be mapped as an item-indexable sequential database – $\{(\mathbf{y}_1\mathbf{y}_2)\mathbf{y}_4\mathbf{y}_3, \mathbf{y}_4(\mathbf{y}_1\mathbf{y}_2)\mathbf{y}_3\}$ – and then analyzed using sequential pattern mining. As a result, $(\mathbf{y}_1\mathbf{y}_2)\mathbf{y}_4$ and $\mathbf{y}_4\mathbf{y}_3$ are identified as sequential patterns, associated with two distinct order-preserving biclusters $\mathbf{B}_1$=$(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_4\})$ and $\mathbf{B}_2$=$(\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{y}_3, \mathbf{y}_4\})$.

Let a sequence $a=<a_1...a_n>$ be a subsequence of $b=<b_1...b_m>$ ($a \sqsubseteq b$), if $\exists_{1\leq i_1<..<i_n\leq m}$: $a_1 \subseteq b_{i_1}$,.., $a_n \subseteq b_{i_n}$. The coverage Φ_s of a sequence s w.r.t. to a sequential database $\mathbf{A}$ is the set of all sequences in $\mathbf{A}$ for which s is subsequence: $\Phi_s = \{s' \in D \mid s \sqsubseteq s'\}$. The support of a sequence s in $\mathbf{A}$ is its coverage size $|\Phi_s|$. These concepts were illustrated in *Basics I-1.10*.

A sequence is *maximal* with respect to a set of sequences, if it is not contained in any of them. Illustrating, $s_1=<\{a\},\{be\}>=a(be)$ is contained in $s_2=(ad)c(bce)$ and is maximal w.r.t. $\mathbf{A}=\{ae,(ab)e\}$.

Given a set of sequences D and some user-specified minimum support threshold θ, a sequence $s \in D$ is *frequent* when contained in at least θ sequences. The **sequential pattern mining** (SPM) problem consists of computing the set of frequent sequences, $\{s \mid sup_s \geq \theta\}$.

Def. 6.2 Given a set of item-indexable sequences $\mathbf{A}$, a minimum support threshold θ_1, and a minimum sequence length θ_2. The target task of **SPM over item-indexable sequences** consists of computing the set of all maximal sequences with minimum θ_2 items and contained in at least θ_1 sequences. In other words:

$$\{s \mid |\Phi_s| \geq \theta_1 \wedge |s| \geq \theta_2 \wedge maximal(s)\} \tag{6.1}$$

The set of maximal frequent sequences for the illustrative sequential database, $\mathbf{A}=\{(bc)a(abc)d, cad(acd), a(ac)c\}$, under the support threshold $\theta_1=3$ and no minimum sequence length is $\{a(ac), cc\}$.

This formalization allows the definition of new methods able to explore the properties of item-indexable sequences. The resulting sequential patterns, referred as *item-indexable sequential patterns*, preserve the consistency of item ordering constraints since they do not allow for item duplicates.

Although not existing SPM methods do not explore item-indexable properties, they are able to rely on (anti-)monotonic searches to efficiently find sequential patterns. Consider two sequences s and s', where $s' \sqsubseteq s$, and a predicate M. M is *monotonic* when $M(s) \Rightarrow M(s')$ and M is *anti-monotonic* when $\neg M(s') \Rightarrow \neg M(s)$. As such, if a sequence s' is not frequent then all subsequences s are also not frequent. Since SPM problem proposal [10], multiple extensions and applications have been proposed, including enhanced searches (variants of Apriori, pattern-growth and vertical searches) and multiple pattern representations [290, 531].

Def. 6.3 Given an (item-indexable) sequential database and a minimum support threshold θ: a frequent sequence s is a sequence with $|\Phi_s| \geq \theta$; a *closed* frequent sequence is a frequent sequence not contained in sequences with same support, $(\forall_{s' \sqsupset s} |s'| < |s|)$; and a *maximal* frequent sequence is a frequent sequence not contained in frequent sequences, $\forall_{s' \sqsupset s} |\Phi_{s'}| < \theta$.

Often considered representations include condensed (including maximal and closed), pseudo-closed, approximate, rare, top-K, multilevel and erasable [531, 290]. Accordingly Def.6.3, given the sequential database $\mathbf{A}=\{(bc)a(abc)d, (ac),$ support $\theta=3$ and constraint $|s| \geq 2$, there are 2 maximal patterns ($\{a(ac), cc\}$), 3 closed patterns ($\{a(ac), (ac), cc\}$) and 5 simple patterns ($\{a(ac), aa, ac, (ac), cc\}$).

6.1.3 Applications

Although order-preserving biclustering has been mainly applied for the analysis of gene expression and chemical data, a wide set of additional problems can also be solved when mapped as an order-preserving biclustering task. As such, Table 6.1 describes its major applications. Note that the described settings have been either poorly studied or not yet tackled in literature. We thus expect an increasing attention towards this end in the upcoming years.

6.2 Related Work

6.2.1 Sequential Pattern Mining

Current SPM methods can be classified according to two axes: search space exploration (Apriori-based, pattern-growth and/or early-pruning [424]) and data orientation (horizontal or vertical). *Apriori*-based algorithms [598]

Domain	*Description*
Analysis of preferences	Consider the analysis of ratings of interest. Increasingly, the selection of videos, booking of hotels, shopping of items, meals' experiences, among many other activities, are accompanied by ratings on large-scale platforms (e.g. IMDB, booking.com, Amazon etc.) associated with high-dimensional collaborative filtering data. It is, therefore, of high-interest to understand local coherent orders of preferences for specific populations/segments of customers. When these orderings satisfy some frequency criteria they can be seen as statistically significant evidence to either support recommendations or for the analysis of user profiles. Illustrating, the following ratings across four movies: $\mathbf{x}_1$={5,5,3,4} and $\mathbf{x}_2$={4,4,1,2} respect the same ordering regularities.
Analysis of biological and social networks	Biological/social networks capture the weight of interactions between groups of molecules/individuals (nodes). A node is thus defined by a set of values characterizing the weights of the relations with all remaining nodes. Consider that the local ordering $\mathbf{x}_{31} < \mathbf{x}_2 = \mathbf{x}_{202} < ... < \mathbf{x}_{88}$ is observed for a subset of nodes $\{\mathbf{x}_3, \mathbf{x}_{28}, ..., \mathbf{x}_{43}\}$. This means that the relative degree of influence from $\{\mathbf{x}_{31} < ... < \mathbf{x}_{88}\}$ nodes is preserved across $\{\mathbf{x}_3, \mathbf{x}_{28}, ..., \mathbf{x}_{43}\}$ nodes. A detailed view on the meaning and relevance of order-preserving coherencies in network data is provided in *Chapter III-8*.
Expression data analysis	In this paradigmatic problem, an order-preserving bicluster defines a subset of genes with similar variations on their levels of expression across certain conditions (either time-points, methods, stimuli, environmental contexts, tissues, organs or individuals). This is critical since different genes can have different default baselines of expression and degree of responsiveness when regulated, and thus an analysis of their absolute values is misleading.
Planning and scheduling	Scheduling is a key problem to: 1) prioritize computational tasks by processors, 2) support the planning of operational tasks by people in private and public sectors, and 3) analyze decisions made, for instance, by a population of robots or computational methods. Frequent co-occurrences (selected tasks for a particular day) and precedences (tasks for different days) are discovered under item-indexable constraints (no repeated items since all tasks have unique identifiers).
Analysis of recommendations	Similarly to the analysis of preferences, the analysis of: 1) the quality of services, 2) the performance of people (increasingly accomplished in the both public and private sectors), and 3) surveys; can reveal frequent orderings pinpointing coherent and potentially relevant answers. The relevance of the order-preserving assumption in these data contexts comes from the fact that different subjects may use the answering scale in different/subjective ways, being the variation of values more indicative than the absolute value.
Analysis of education data	Frequent orderings on the observed grades and other performance indicators of students across courses can be mined to identify common behaviors and, thus, support teaching and school management activities. A simplistic illustration is to analysze item-indexable sequences, each capturing the finished courses per semester along a student's academic path. In this example, order-preserving coherencies reveal relevant precedences on the course's offerings for different segments of students.
Others	The discovery of meaningful variations across features independently of their absolute values is also relevant across other biological domains, including clinical, chemical and mutagenesis data analysis [59, 311]; and social domains, including shopping and traveling behavior, text mining and financial transactions [711, 293].

Table 6.1: Applications of item-indexable sequential pattern mining and order-preserving biclustering.

use the anti-monotonic property to generate candidate sequences in a breadth-first manner using multiple database scans. To overcome the computational complexity of maintaining the support count for each sequence generated, the use of bitmaps or direct comparison have been proposed [137]. *Pattern-growth* methods [31, 524, 523] avoid the Apriori-based costs from candidate generation by building an expressive representation of the database and by generating as few candidate sequences as possible using a traversal recursion to grow the already mined frequent sequences. PrefixSpan [523], one of the most efficient options to SPM, recursively constructs patterns by growing their prefix and by maintaining their corresponding postfix subsequences into projected databases. This guarantees a narrowed search space and avoids the generation of candidates since it only counts the frequency of local sequences. The major cost resides on the construction of projected databases. Finally, *early-pruning* methods [695, 137] emerged more recently in the literature. They adopt position induction to prune candidate sequences very early in the mining process and to avoid support counting as much as possible. These algorithms usually employ a table to track the last positions of each item in the sequence to evaluate whether the item can be appended to a given prefix. Methods based on pattern-growth structures and early-pruning principle offer a good the best performance for the majority of biological data settings. Complementarily, vertical projections for the SPM task are only competitive with the alternatives for very flattened matrices ($m \gg n$). In high-dimensional data settings, vertical projections should be only target for (order-preserving) coherency on the rows.

Despite the relevance of these searches they do not explore efficiency gains from mining sequences with item-indexable properties. Thus, their application is often associated with efficiency bottlenecks due to: 1) the computationally hard nature of the order-preserving biclustering task, 2) the high density of item-indexable sequences derived from tabular data, and 3) the often very high number of (item-indexable) sequential patterns discovered under low support thresholds [416].

Although there are principles for the discovery of colossal patterns based on the fusion of smaller itemsets [714], these approximations suffer from noise and, to our knowledge, have not been extended for the SPM task.

Additionally, despite former attempts to discover sequential patterns in item-indexable sequential databases using OPC-Trees [415, 416], these trees show a significant memory overhead that turn their application impracticable for large databases.

Finally, despite the relevance of inputting a minimum number of items per sequential pattern, the surveyed SPM methods are not natively prepared to effectively seize efficiency gains in the presence of this constraint. This input is commonly desirable to guarantee the domain-driven relevance and statistical significance of the discovered patterns. Illustrating, frequent ratings or variations associated with biological and social activity are only meaningful above a minimum pattern length.

6.2.2 Order-Preserving Biclustering

There are two major types of approaches for modeling order-preserving biclusters: greedy and exhaustive. Exhaustive approaches aim to identify the largest submatrices where the set of rows are the maximum sets that support a linear order of values across the set of columns [415]. Contrasting, greedy approaches, such as the pioneer OPSM [59], rely on a merit function to guide the composition of incrementally larger/smaller biclusters. Existing order-preserving approaches also vary with regards to the number of output biclusters – either parameterized (existing greedy approaches) or undefined (existing exhaustive approaches) – and to the number of search iterations – either one bicluster at a time (existing greedy approaches) or all biclusters at a time (existing exhaustive approaches).

Greedy approaches are guided by the upper-bound probability that a random data matrix contains a linear permutation of columns with more rows supporting it. Multiple extensions have been proposed over the original OPSM method [59], including: the OPSM-RM method [697] to discover order-preserving biclusters from multiple matrices obtained from replicated experiments; the *p*OPSM method [209] to model uncertain data with continuous distributions based on a probabilistic to which a row belongs to bicluster; and the MinOPSM method [323] that implements a variant of the order-preserving task. Despite the relevance of this class of greedy approaches, they suffer from three major drawbacks: *1)* delivery of approximate solutions without optimality guarantees; *2)* biases towards larger biclusters; and *3)* restrictions on the biclustering structure, including the output of a fixed number of non-overlapping biclusters.

Exhaustive approaches aim to identify the submatrices with largest areas that still respect the ordering constraints [415]. *u*-Clustering (also known as OP-Clustering) [415, 416] delivers solutions that overcome the flexibility issues of greedy approaches. Still, their adoption presents three major challenges: *1)* efficiency strongly deteriorates for matrices with more than 50 rows; *2)* noisy values lead to the partition of large biclusters in multiple smaller biclusters since they search for perfect orderings; and *3)* output of a large number of uninformative biclusters due to the use of non-condensed pattern representations.

The surveyed approaches also impose a monotonic ordering of values that does not allow for symmetries [59, 415]. Also, they are neither able to handle an arbitrary-high number of missings nor able to fix adequate levels of tolerance to noise. Finally, the properties of these order-preserving models are not parameterizable. In sum, existing order-preserving approaches suffer from flexibility, robustness, and efficiency issues.

6.3 Solution: BicSPAM

To tackle the scalability, flexibility and robustness issues of existing order-preserving approaches, we propose BicSPAM (Biclustering from Sequential PAttern Mining). Accordingly to the framework for pattern-based biclustering proposed in *Chapter III-1*, these qualities are achieved through the consistent application of adequate strategies along the mapping (pre-processing), mining, and closing (post-processing) steps.

The *mapping* step is responsible to compose an item-indexable sequential database $\mathbf{A}$ from either a real-value or discrete matrix with $X=\{\mathbf{x}_1,..,\mathbf{x}_n\}$ rows/observations and $Y=\{\mathbf{y}_1,..,\mathbf{y}_m\}$ is a set columns. For this aim, the column indexes are linearly ordered for each observation according to their absolute or discretized values. Each observation

is seen as a sequence of items that correspond to column indexes. When two columns have equal values, they are seen as *co-occurrences*, while when their values differ they are treated as *precedences*. Consider the illustrative row $\mathbf{x}_2=\{\mathbf{y}_1=0, \mathbf{y}_2=2, \mathbf{y}_3=0\}$ in Figure 6.3, $\mathbf{y}_1$ and $\mathbf{y}_3$ co-occur, while $\mathbf{y}_1$ precedes $\mathbf{y}_3$.

The *mining* step, corresponds to the application of sequential pattern miners for the discovery of order-preserving biclusters. An order-preserving bicluster is thus associate with a sequential pattern capturing frequent precedences and co-occurrences across columns. A variant of this mining step was initially proposed by Liu et al. [415, 416]. The support threshold defines the minimum number of rows (columns) in the bicluster for a coherency with orientation on rows (columns). In the context of expression data analysis, a low support is critical since significant co-expression patterns can occur for small groups of genes and/or conditions. Similarly, the remaining data domains introduced in Table I-1.1 benefit from low supports. Complementarily, the minimum pattern length (number of items in the sequential pattern), defining the minimum number of columns (rows) in the outputted biclusters, can be parameterized to pruning sequences with a number of items below that threshold. In this context, an order-preserving bicluster can be derived from a frequent sequence s according to Def.6.4. Accordingly, the mapping and mining steps can be easily adapted for an order-preserving assumption on columns by transposing both the input matrix and the derived biclusters.

Def. 6.4 Given a matrix $\mathbf{A}$ and a minimum support threshold θ_1 and pattern length θ_2, a set of *order-preserving biclusters* $\cup_k B_k$ where $\mathbf{B}_k = (\mathbf{I}_k, \mathbf{J}_k)$ can be derived from a set of *frequent sequences* $\cup_k s^k$ (where $|\Phi_{s^k}| > \theta_1 \wedge |s^k| > \theta_2$) by: 1) mapping $(\mathbf{I}_k, \mathbf{J}_k) = (\Phi_{s^k}, \{s_i^k \mid i = 1..|s^k|\})$ to compose order-preserving biclusters with coherency on rows, or by 2) $(\mathbf{I}_k, \mathbf{J}_k) = (\{s_i^k \mid i = 1..|s^k|\}, \Phi_{s^k})$ from $\mathbf{A}^T$ to compose order-preserving biclusters with coherency on columns.

Finally, closing options can be applied to post-process the delivered biclustering models. Figure 6.3 illustrates how order-preserving biclustering can be solved using SPM.

Similarly to FIM-based biclustering, SPM-based biclustering can either rely on SPM methods as-is or target more dedicated searches by adapting their support metric (merit function) and use it within the Apriori-based SPM framework [289, 29]. However, to our knowledge, there are not yet contributions on this second direction.

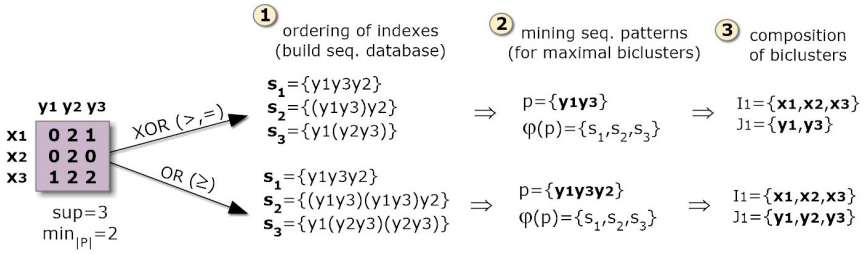

Figure 6.3: Mining order-preserving biclusters in real or itemized matrices.

To discover order-preserving biclusters the first step is to order the column indexes according to real or discretized values and map them to itemset sequences based on the observed ordering (precedences and co-occurrences). In particular, when targeting the or behavior, co-occurrences are propagated n times, being n the number of items co-occuring. Illustrating, $x_2=\{y_1=0, y_2=2, y_3=0\}$ is mapped as $(y_1y_3)y_2$ sequence according to the xor behavior and as $(y_1y_3)(y_1y_3)y_2$ under the or behavior. Second, a SPM method is applied over the set of sequences to extract the set of sequential patterns. Finally, biclusters are derived from the set of items and supporting transactions for each sequential pattern.

Basics 6.2 Impact of alternative sequential pattern representations

Similarly to BicPAM, the use of condensed pattern representations can largely impact the properties of the BicSPAM solutions, as illustrated in Figure 6.4. Biclustering solutions derived from simple sequential pattern representations include all combinations of biclusters above a minimum support threshold (number of rows) and pattern length (number of columns). The adoption of maximal sequential patterns can lead to the loss of biclusters with a moderate number of columns but with a high number of rows since frequent sequences with fewer items but with a higher support are discard. Finally, approaches that use closed sequential patterns are the ones capable of returning all the maximal biclusters, the set of biclusters that are not totally included in another bicluster.

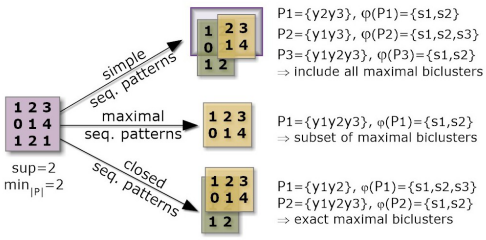

Figure 6.4: Comparing biclustering solutions using simple, closed and maximal sequential patterns.

As illustrated in Figure 6.5, principles along these steps are proposed to guarantee that BicSPAM surpasses the drawbacks associated with existing order-preserving models. Accordingly, *Sections 6.3.1* to *6.3.4* propose principles to guarantee the efficiency, flexibility and robustness of the searches.

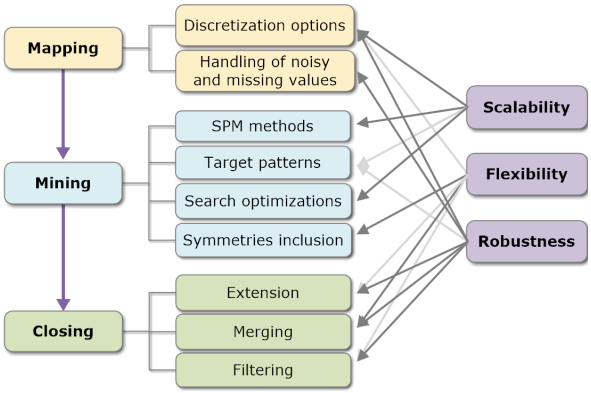

Figure 6.5: BicSPAM methodology: major dimensions.

6.3.1 Efficiency: Item-Indexable Sequential Pattern Mining

As surveyed in *Section 6.2*, although a large number of SPM methods has been proposed, they are neither able to explore efficiency gains from sequences without repeated items (item-indexable sequences) nor able to effectively prune the search space in the presence of a minimum pattern length threshold. As such, they show efficiency bottlenecks when applied over highly dense sequences, such as the orderings derived from tabular data targeted by this work. Therefore, in order to avoid the drawbacks of existing SPM methods, we propose the IndexSpan algorithm, an extension of PrefixSpan to discover sequential patterns with heightened efficiency from item-indexable sequential databases.

IndexSpan explores some of the prominent strategies that promote the efficiency of the SPM methods [424], including: mechanisms to reduce the support counting; narrowing of the search space; optimally sized data structure representations of the sequential database; and early pruning of candidate sequences. In particular, IndexSpan guarantees a search space as small as possible and relies on a narrow search exploration using depth-first searches.

IndexSpan extends PrefixSpan [523] in order to incorporate additional efficiency gains from three principles. First, IndexSpan relies on an easily indexable and compacted version of the original sequence database. Second, it uses faster and memory-efficient database projections. A projected database only maintains a list with the IDs of the active sequences. Finally, IndexSpan relies on early-pruning techniques. Algorithm 6 describes IndexSpan.

IndexSpan considers the three following structural adaptations over the PrefixSpan algorithm. First, it maintains a simple matrix in memory that maintains the index of each item per row. This matrix is constructed at the very beginning (*lines 2-5*) and the original database is removed.

Additionally, for sparse sequential databases, this matrix can be replaced by a vector of hash tables to optimize memory usage. Considering an illustrative sparse database with $|\mathcal{L}|=20$, $\mathbf{A}=\{a(cp), am, (ac)\}$. Instead of relying on a 20×3 matrix to track the indexes [[0 -1 1 ...][...][...]], it is enough to monitor the positions of the available items [h_1{a:0,c:1,p:1}$,h_2${a:0,m:1}$,h_3${a:0,c:0}] to obtain the indexes (e.g. h_2(m)=1).

These data structures support position induction. The idea behind is simple: if an item's last/start position precedes the current prefix/postfix position, the item can no longer appear before/after the current prefix.

Second, a projected database can be constructed with heightened efficiency by avoiding the need to update and maintain postfixes. A projected database simply maintains the identifiers of the supporting sequences of a specific prefix.

To know if a sequence is still frequent when an item is added over a specific prefix, there is only the need to compare its index against the index of the previous item as well as their lexical order for the case where the index

Algorithm 6: IndexSpan

Input: sequential database $\mathbf{A}$, minimum support θ_1, minimum sequence length δ
Output: set of sequential patterns S
Note: α is a sequence, $\mathbf{A}_\alpha$ is the α-projected database
($\mathbf{A}_\alpha$ simply maintains a reference to the current sequences)

```
1  mainMethod() begin
2      foreach sequence s in D /*add array of item indexes per sequence*/ do
3          foreach item c do
4              s.indexes[c] ← position(s,c);
5      α.items ← φ; α.trans ← φ;
6      indexSpan(α,A);

7  indexSpan(α,A_α) begin
8      foreach frequent item c in A_α do
9          β.items ← α.items ∪ c; //co-occurrence (c is added to the last α itemset)
10         γ.items ← α.items · c; //α precedes c (c is inserted as a new itemset)
11         //pruning and fast gathering of supporting transactions (for efficient data projection)
12         foreach sequence s in A_α do
13             currentIndex ← s.indexes[c];
14             upperIndex ← s.indexes[α_n] /*α_n is the last item*/;
15             if leftPositions(currentIndex)≥ θ_2 − |α| /*pruning*/ then
16                 if currentIndex > upperIndex then
17                     γ.trans ← γ.trans ∪ s.ID;
18                 else
19                     if currentIndex=upperIndex ∧ c>α_n then β.trans ← β.trans∪s.ID;
20         if sup_β(A_α) ≥ θ_1 then
21             S ← S ∪ {β};
22             A_β ← fastProjection(β,A_α);
23             indexSpan(β,A_β);
24         if sup_γ(A_α) ≥ θ_1 then
25             S ← S ∪ {γ};
26             A_γ ← fastProjection(γ,A_α);
27             indexSpan(γ,A_γ);

28 fastProjection(β,A_α) begin
29     foreach sequence s in A_α do
30         currentIndex ← s.indexes[β_n];
31         upperIndex ← s.indexes[β_{n−1}];
32         if leftPositions(currentIndex)≥ θ_2 − |α| /*pruning*/ then
33             if currentIndex > upperIndex then
34                 A_β ← A_β ∪ s;
35             else
36                 if currentIndex=upperIndex ∧ c > α_n then A_β ← A_β ∪ s;
37     return A_β;
```

is the same (i.e. the new item co-occurs with the last items of the pattern). In this way, database projections, the most expensive step of PrefixSpan both in terms of time and memory, are handled with heightened efficiency. The proposed projection method is described in Algorithm 6, *lines 12-19* and *28-37*.

Finally, the input minimum number of items per sequential pattern, δ, can be used to prune the search as early as possible. If the number of items of the current prefix ($|\alpha|$) plus the items of a postfix s_α (computed based on the current and last index positions) is less than δ, then the sequence identifier related with the s_α postfix can be removed from the projected database since all the patterns supported by s will have a number of items below the inputted threshold.

For an optimal pruning, this assessment is performed before item indexes comparisons, which occurs in two distinct moments during the prefixSpan recursion (Algorithm 6 *lines 15* and *32*).

The efficiency gains from fast database projections and early pruning techniques, combine with the absence of memory overhead, turn IndexSpan highly attractive in comparison with peer methods based on OPC-Trees. These

peer methods [415, 416] rely on a breadth search with high memory complexity $\Theta(n \times m^2)$ that do not scale for medium-to-large datasets (even in the presence of pruning techniques).

6.3.2 Efficiency: Additional Principles

Understandably, optimal and flexible solutions where the number and positioning of biclusters are not previously fixed require efficient searches. Below, we discuss five groups of principles to foster the scalability of BicSPAM for large datasets.

BicSPAM makes use of IndexSpan as the default mining search. Contrasting with existing approaches, closed sequential patterns (maximal biclusters) is the default option in BicSPAM. Since IndexSpan relies on a prefix-growth search, it can easily accommodate principles from existing research to deliver condensed pattern representations, such as CloSpan [687]. As such, IndexSpan was extended to provide efficient checks for the delivery of condensed representations. For an optimal learning order-preserving models with coherency on rows (columns) from data with a high number of features (observations) and $m \gg n$, BicSPAM makes also available the SPADE [700] algorithm. Nevertheless, empirical evidence pinpoints the superiority of IndexSpan in these contexts due to the density of sequential databases mapped from tabular data and the gains from light projections [311]. In order to guarantee the possibility for users to select additional state-of-the-art SPM methods that may achieve optimal performance on data with certain regularities, BicSPAM also makes available CloSpan [687] and BIDEPlus [654], both optimized for the discovery condensed sequential patterns.

In the absence of optimality criteria, BicSPAM makes also available of two SPM algorithms – TSP [639] and TKS [223] – to mine the top-k sequential patterns. State-of-the-art searches for top-k sequential patterns are able to scale significantly better than traditional peers. Although they required a fixed number of biclusters as input, they are able to guarantee dissimilarity of the outputted sequential patterns, and thus adequately explore different regions from the input matrix.

An alternative principle to foster efficiency is to either prefer the use of the original real-values or of a lengthy discretization alphabet to define the orderings in the sequential database. This guarantees a high number of precedences among column indexes (and low number of co-occurrences), leading to smaller sequential patterns. Contrasting, although discretization with a low number of items is critical to guarantee more noise-tolerant solutions, this option can degrade efficiency due to the exponential increase of frequent sequential patterns (in number or size). To create a compromise between noise and efficiency, BicSPAM allows an arbitrary number of items and provides medium-to-high number of items as the default option ($|\Sigma| \approx m/5$). In this context, extending and merging biclusters discovered using a high number of items can be applied to guarantee efficiency while preserving the quality of solutions. A second strategy is to increase the minimum support threshold (under a relaxed discretization more robust to noise) to promote an heightened SPM efficiency and the later application of reduction procedures to remove biclusters' rows and columns in order to intensify their homogeneity. As such, and similarly to BicPAM, BicSPAM makes available extension, merging and filtering methods.

BicSPAM makes also available two critical extensions to the introduced IndexSpan algorithm (natively prepared to effectively discover sequential patterns from item-indexable sequences and prune the search space in the presence of a minimum pattern length). First, the discovered closed frequent sequences are represented within a compact tree structure, where the supporting transactions are annotated using principles proposed for full-pattern discovery [315]. Second, parameters from closing options are pushed to mining step. Illustrating, overlapping criteria for merging biclusters can be efficiently checked based on the properties of the tree, which significantly removes the complexity associated with computing similarities between all pairs of biclusters. This extension is carefully covered in *Chapter III-7*.

Finally, many of the principles proposed in the last decade to guarantee the scalability of SPM methods can be easily applied with IndexSpan. These principles include: 1) data partitioning principles (inter- and intra-

sequence), 2) principles for the application of SPM methods for multiprocessing and/or distributed settings, and 3) the delivery of approximated sequential patterns (discovered under specific performance guarantees) [290, 424]. Illustrating, since IndexSpan relies on a prefix-growth search, the scalability principles used in MapReduce [540] can be implemented in order to relax its efficiency boundaries.

6.3.3 Flexibility: Affecting the Properties of Order-preserving Models

BicSPAM relies on flexible searches (no need to fix the number of biclusters apriori), delivers flexible structures of biclusters (arbitrary size and positioning) and allows for a flexible parameterization of its behavior (if a user opts not to use the dynamically learned parameters from data).

Similarly to BicPAM, closing procedures can be applied within BicSPAM for the goal of customizing biclustering structures. Illustrating, extension and merging procedures can be adopted to produce exhaustive structures (either overall, across rows or across columns), while reduction, merging and filtering and can be used to guarantee exclusive structures (either overall, across rows or across columns). As such, the composition of alternative structures is performed with sharp usability since there is no need to change the core mapping and mining steps.

In order to further guarantee the flexibility of the target BicSPAM approaches, *Section 6.3.3.1* extends the mining step to support the discovery of order-preserving biclusters with symmetries, and *Section 6.3.3.2* shows how to further affect the order-preserving coherency by controlling the degree of precedences and co-occurrences.

6.3.3.1 Order-Preserving with Symmetries

Two rows from an order-preserving bicluster may show monotonic orderings of values but differing in sign (either increasing or decreasing coherently with the sign of values). Despite the relevance of combining symmetries with order-preserving models, BicSPAM is, to our knowledge, the first algorithm supporting this possibility.

An (order-preserving) bicluster $\mathbf{B}$ with *symmetries*, has either symmetries on rows $\hat{a}_{ij} = c_i \times a_{ij}$ or columns $\hat{a}_{ij} = c_j \times a_{ij}$, where $c_i \in \{-1, 1\}$ ($c_j \in \{-1, 1\}$) is the symmetry factor and a_{ij} is an element contained in $\mathbf{B}$. For the purpose of finding biclusters with symmetries, normalization should satisfy the zero-mean criterion when applied. Additionally, when the discretization is applied and the number of considered items is odd, there is one item being its own symmetric, which must be specially handled.

Similarly to the principles introduced in *Section III-3.2*, the proposed method to combine order-preserving and symmetric models is based on iterative sign corrections. If the goal is to find an order-preserving coherency on the rows, then there is one iteration for each column $\mathbf{y}_j$. Within each iteration j, each row $\mathbf{x}_i$ is either multiplied by a 1 or -1 factor in order to guarantee that the observed values for the $\mathbf{y}_j$ column have the same sign. After the correction of the sign for each row, mining and closing steps are applied, the discovered biclusters are added to the solution set, and the method proceeds with the next iteration (column $\mathbf{y}_{j+1}$). Figure 6.6 illustrates this strategy. According to the illustrated setting, $\mathbf{x}_2$ and $\mathbf{x}_3$ vectors need to be multiplied by the -1 symmetric factor in the context of $\mathbf{y}_1$ column in order to guarantee the consistency of signs. SPM is then applied over this revised matrix. Although the alignment of signs can be applied for every column $\mathbf{y}_j$, efficiency gains can be achieved by stopping the search when all the sign combinations have been achieved. Filtering is a critical post-processing step to remove potential duplicates resulting from the repetition of coincident alignments.

6.3.3.2 Degree of Co-occurrences versus Precedences

When the original numeric values are ordered without any form of normalization and discretization, the biclusters delivered by the application of SPM methods do not allow for ordering mismatches (independently of how small the variations across values are). If discretization with an ordinal alphabet is applied, the number of co-occurrences per sequence increases. In this case, the outputted biclusters are naturally more robust as their coherency is not based on exact variations across values but on approximate variations. For this reason, BicSPAM discretizes by

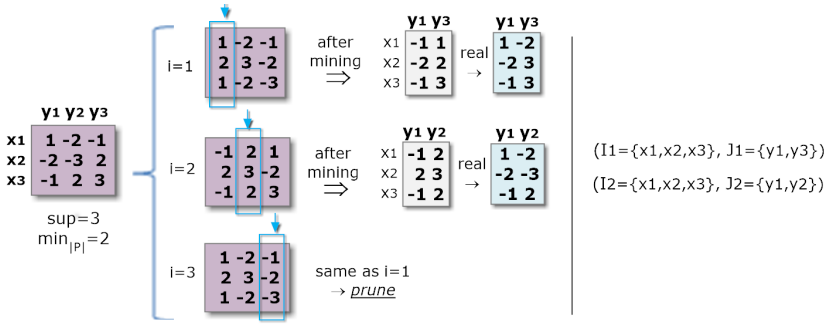

Figure 6.6: Discovery of order-preserving biclusters with symmetries.

default the input data. In this context, the number of items in the considered alphabet can be used to affect the coherency. Increasing the number of items decreases the number of co-occurrences and, therefore, reduces the tolerance to elements with closer values but mismatched orders. As illustrated in *Basics 6.3*, the selected procedures to normalize and discretize data also determine the distribution of co-occurrences and precedences per sequence and, consequently, the final coherency.

Basics 6.3 Preprocessing sequential databases

Similarly to BicPAM, BicSPAM allows for the application of multiple normalization and discretization procedures on the rows, columns or overall matrix before orderings. The choices depend essentially on the data regularities (statistical tests can be used to dynamically select the procedures) and on the coherency orientation. When assuming the presence of missing and outlier elements, a masking bitmap is considered for a more accurate application of these procedures [500]. BicSPAM makes available range-based, equal-depth partitioning and Gaussian cut-off points methods for discretization (default option), illustrated in Figure 6.7 (a variant of Figure III-1.10). The use fixed ranges is the simplest option, but can lead to an accentuated weak distribution of items, and thus it is highly prone to the items-boundary problem. Percentage-based method minimize this problem by using a depth partitioning of items. Alternatively, the cut-off points of equally distributed areas from alternative density probability functions (as Gaussian or Poisson with $\lambda \geq 3$) can be considered to minimize the loss of relevant biclusters.

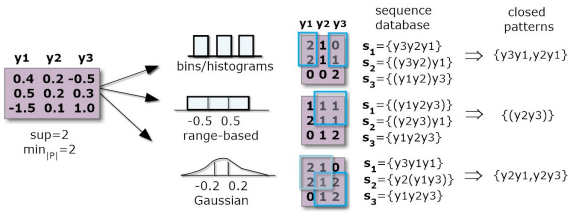

Figure 6.7: Comparing BicSPAM's discretization options.

6.3.4 Robustness: Affecting Quality of Order-Preserving Models

To guarantee the robustness of BicSPAM to discretization, noise and missing values, we discuss below how some of the strategies proposed in the context of *Chapter III-2* can be extended for the order-preserving model.

Robustness to Discretization. On one hand, the use of discretization with ordinal alphabet was motivated as important to increase the tolerance to ordering mismatches due to the proximity of values. On the other hand, discretization is associated with the item-boundaries drawback. The item-boundaries drawback is particularly critical for order-preserving models due to the fact that discretization is commonly applied using lengthier alphabets. Understandably, the number of items in the order-preserving model is no longer used to determine the coherency strength, but to decrease the probability of mismatched orders occurring due to the natural variability associated with the observed values ($\eta_{ij} \neq 0$).

In order to benefit from discretization while avoiding the item-boundaries problem, the introduced multi-item assignments' strategy can be applied. For this purpose, multiple items can be assigned to a single element in the matrix (prior to the composition of the sequential database) based on the observed distances to the upper and lower cut-off points. Different levels of relaxation can be considered based on the parameterizable number of items assignable to each position. When lengthier alphabets of discretization are considered ($|\mathcal{L}|$), we advise the assignment of one-to-three items per element based on the distance of the observed value to the interval centroids.

In this context, item-indexable properties are only applicable for the rows without multiple items assigned.

Robustness to Noise. In order to guarantee that SPM-based biclustering can handle varying amount and types of noise, five strategies can be considered. First, merging, filtering, extension and reduction procedures can be either extended or applied as-is for post-processing order-preserving models. *Section III-1.4.3* provides an in-depth description of strategies that can be used to guarantee their effective and efficient application.

Second, discretization alphabets with different lengths can be used according to one of two options. The first option is to apply the mapping-and-mining steps independently for each alphabet, and then post-process a solution given by all of the discovered biclusters. In this context, the observed overlapping from similar biclusters discovered with different alphabets is then used to guide merging, filtering, extension and reduction options. Figure 6.8a illustrates how noisy-tolerant biclusters can discovered by reducing the number of items of the discretization alphabet. The second option is to use the different alphabets to generate lengthier sequences with possibly repeated items at different positions. Although this option avoids the application of the mining step multiple times, it prevents the exploration of the efficiency gains from item-indexable properties as the associated databases no longer satisfy these properties.

Third, the closing options can be applied using tests based on statistical test sensitive to the closeness of original or discretized values. Figure 6.8b shows that the use of a lower support leads to ($\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$,$\{\mathbf{y}_1, \mathbf{y}_2\}$) and ($\{\mathbf{x}_2, \mathbf{x}_3\}$,$\{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3\}$) biclusters, which can either be extended or merged as a single larger bicluster ($\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3\}$,$\{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3\}$).

Fourth, the mining of association rules between sequential patterns, typically referred as sequential rule mining, can be applied with varying confidence thresholds to compose biclusters with located noise. Sequential rule mining was originally proposed by Bettini et al. [70], and since then some methods have been proposed to deal with harder combinatorial nature of this task in comparison to sequential pattern mining. In particular, ERMiner [225] and RuleGrowth [226] are arguably considered state-of-the-art searches for sequential rule mining.

Finally, multi-item assignments can also be used to guarantee robustness to noise. In Figure 6.8c, multiple items are assigned per element based on the distance between the observed value and the centroids of items. Illustrating, let $a_{1,1}$=0.5 and the centroid of items 0 and 1 be respectively 0.2 and 1.1, and the distance threshold be 0.7, then $a_{1,1}$ is assigned to both 0 and 1 items (1.1-0.7< $a_{1,1}$ <0.2+0.7).

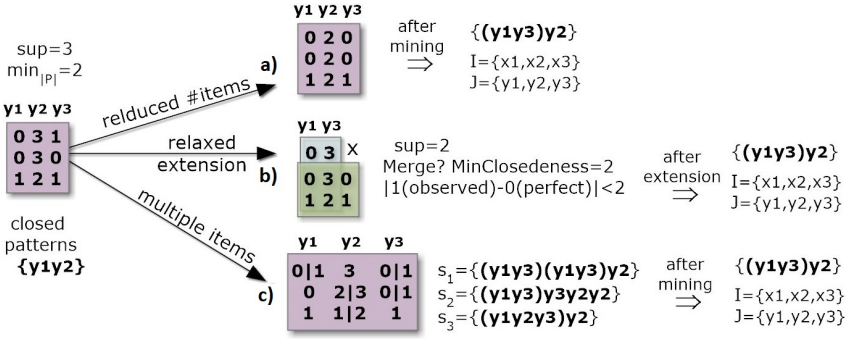

Figure 6.8: Strategies to deal with varying noise relaxations.

Handling of Missings. BicSPAM relies on the strategies proposed in *Chapter III-2* to handle an arbitrary-high number of missing values. As such, it makes available the possibility to: 1) use traditional methods of imputation [631, 183, 301]; 2) remove the element with the associated missing (without the need to remove the associated row and column); 3) handle the missing values separately using a dedicated item; and, more interestingly, 4) rely on multi-item imputations to guarantee robustness to problems associated with single-item imputations. The more relaxed multi-item setting is to replace the missing element by all items. This is a radical alternative to guarantee that

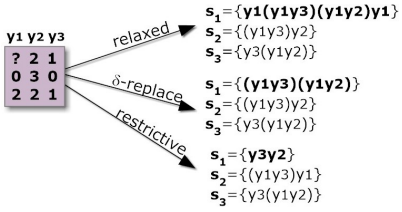

Figure 6.9: Comparison of strategies to handle missing values.

potentially relevant biclusters are not lost due to the presence of missing values. A more conservative alternative (δ-*replace*) considers multiple items around its value-estimation. As illustrated in Figure 6.9, the $\mathbf{s}_1 = \mathbf{y}_3\mathbf{y}_2$ is mapped as $\mathbf{y}_1(\mathbf{y}_1\mathbf{y}_3)(\mathbf{y}_1\mathbf{y}_2)\mathbf{y}_1$ according the relaxed setting. To illustrate the δ-replace alternative, assume that $a_{1,1}$ is estimated to have value 1.5 and δ=0.5, then $a_{1,1} \in \{1, 2\}$ and (since $\mathbf{y}_3$=1 and $\mathbf{y}_2$=2) $\mathbf{s}_1$ is mapped as $(\mathbf{y}_1\mathbf{y}_3)(\mathbf{y}_1\mathbf{y}_2)$.

6.4 Results

In this section, we assess the performance of BicSPAM in three steps. First, we evaluate IndexSpan against competitive SPM methods on synthetic and real data. Second, we parameterize BicSPAM with IndexSpan to compare it against state-of-the-art biclustering methods, assess its efficiency limits and the impact of varying the coherency by controlling the number of co-occurrences and accommodating symmetries. Finally, the biological relevance of BicSPAM's outputs is assessed on real data. Results demonstrate the efficiency, flexibility and robustness. IndexSpan and BicSPAM were implemented in Java (JVM v1.6.0-24). The experiments were run using an Intel Core i5 2.30GHz with 6GB of RAM.

Evaluation Settings. Guidelines from *Chapter II-3* were followed. In particular, to study the performance of BicSPAM, three sets of synthetic data were generated. According to Table II-3.1, a first a set of synthetic matrices was generated by varying structural data properties: size, dimensionality, structure of biclusters and noise factor (up to ±10% of the domain range). We allow for non-coherent overlapping areas among biclusters, which can harden the recovery of the planted biclusters. Results are the average indicators from 30 data instances per setting (15 with Uniform background values and 15 with Gaussian background values).

A second set of datasets was generated to study the efficiency limits of BicSPAM by fixing the number of rows ($|X|$=20000) and varying the number of columns (50≤ $|Y|$ ≤200). Background values were generated as the first set of datasets, and 2 biclusters were planted to occupy 5% of the total area.

Understandably, the mapped (item-indexable) sequences from these tabular datasets are highly dense since each sequence contains all column indexes. In order to study the relevance of IndexSpan for less dense data, we created a third set of datasets where sequential databases were generated with sequential patterns with varying degree of sparsity.

6.4.1 IndexSpan Assessment

Results on Synthetic Data. Figure 6.10 compares the performance of SPM methods for these settings in terms of time and maximum memory usage. IndexSpan is assessed against PrefixSpan and μ-Clustering, as they are competitive baselines to assess efficiency. The impact of mining sequential patterns in the absence and presence of the minimum number of columns per bicluster, δ threshold, is presented for a fair comparison. Two main observations can be derived from this analysis. First, the gains in efficiency from using fast database projections are highly significant, dictating the ability of the SPM task to scale for hard settings. δ-based pruning methods also promote efficiency gains. As such, contrasting with μ-Clustering and PrefixSpan, IndexSpan guarantees acceptable levels of efficiency for matrices up to 2000 columns for a medium-to-large occupation of sequential patterns (~3%-10% of matrix total area). Second, IndexSpan performs searches with minimal memory waste. The memory is only impacted by the lists of sequence identifiers maintained by prefixes during the depth-first search. Memory of PrefixSpan is slightly hampered due to the need to maintain the projected postfixes. μ-Clustering requires the full construction of the pattern-tree before the traversal, which turns this approach only applicable for small-to-medium databases. For an allocated memory space of 2GB, we were not able to construct OPC-Trees for input matrices with more than 40 columns.

To further assess the performance of IndexSpan, we fixed the 1000×75 experimental setting and varied the level of sparsity by removing specific positions on the input matrix, while preserving the planted sequential patterns.

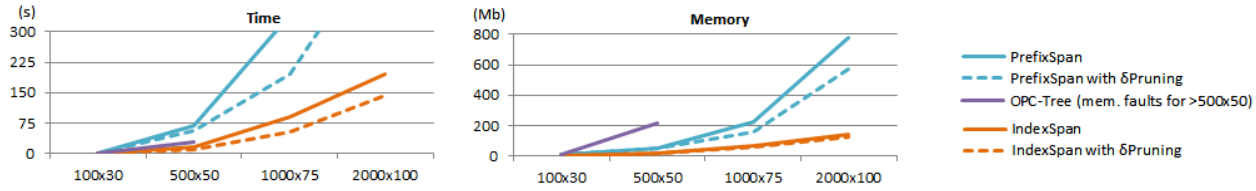

Figure 6.10: Efficiency of IndexSpan against peer SPM methods across data settings with varying size.

We randomly selected these positions to cause a heighten variance of length among the generated sequences. The amount of removals was varied between 0 and 40%. This analysis is illustrated in Figure 6.11. Two main observations can be retrieved. First, to guarantee an optimal memory usage, there is the need to adopt vectors of hash tables in IndexSpan. Second, although the use of these new data structures hampers the efficiency of IndexSpan, the observable computational time is still significantly preferable over the PrefixSpan alternative.

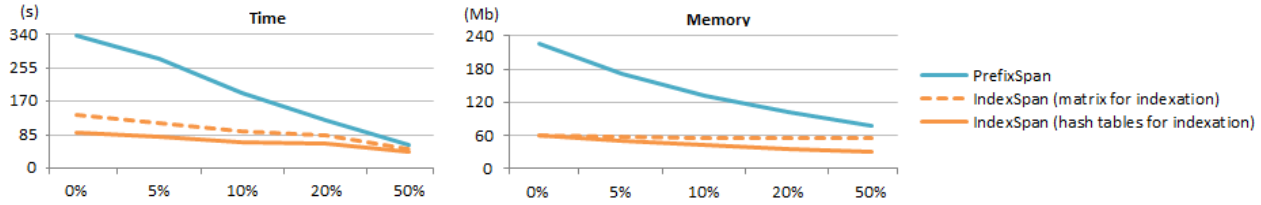

Figure 6.11: Performance for varying levels of sparsity (1000×75 dataset).

Finally, in order to assess the impact of varying the number of co-occurrences vs. precedences, we considered multiple discretizations for the 1000×75 dataset. By decreasing the size of the discretization alphabet, we are increasing the amount of co-occurrences and, consequently, decreasing the number of itemsets per sequence. This analysis is illustrated in Figure 6.12. When the number of precedences per sequence is very small (<10), the efficiency tends to significantly decrease due to the exponential increase of sequential patterns. However, for the remaining discretizations, the efficiency does not strongly differ since the number of frequent patterns is identical and pattern-growth methods are able to deal with co-occurrences and precedences in similar ways (Algorithm 1 *lines 20-27*).

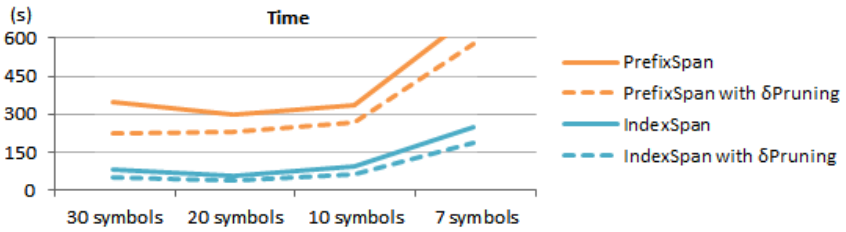

Figure 6.12: Performance for varying weights of precedences vs. co-occurrences.

Results on Real Data. Figure 6.13 compares the time and memory efficiency of IndexSpan for three distinct expression datasets[1]: dlblc (180 items per instance, 660 instances), colon cancer (62 items per instance, 2000 instances), and leukemia (38 items per instance, 7129 instances). The provided results assume the use of a discretization alphabet of 20 items and the θ=8% and δ=5 thresholds. This analysis reinforces the derived observations from synthetic data: μ-Clustering is bounded by the size of the database, and IndexSpan ability to seize item-indexable properly significantly affects the efficiency levels.

[1] http://www.upo.es/eps/bigs/datasets.html
http://www.bioinf.jku.at/software/fabia/gene_expression.html

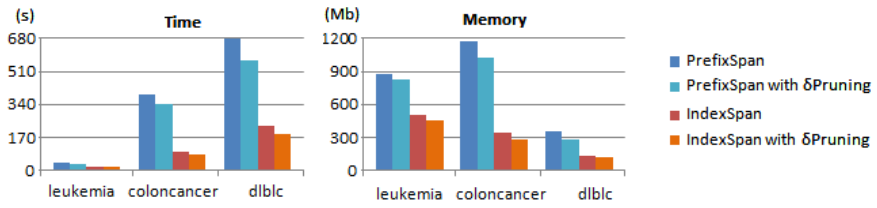

Figure 6.13: Efficiency of BicSPAM over real data.

6.4.2 Assessment of BicSPAM

Comparison of Biclustering Approaches. Four state-of-the-art biclustering approaches were selected: two approaches able to deliver order-preserving biclusters, OPSM [59] and μ-Clustering [415], and two approaches able to discover biclusters under constant, additive and multiplicative models, FABIA with sparse prior Equation [324] and ISA [342]. For this aim, we used the BicAT software [45] to run OPSM and ISA and the R package fabia[2]. The parameterizations of these methods were discussed in *Chapter III-2*. BicSPAM was used with the: *1)* default parameterization, *2)* default parameterization but with sequential patterns gathered from multiple levels of expression ($|\Sigma| \in \{4, 7, 10\}$), and *3)* dynamic data-based parameterization. The support threshold for both BicSPAM and μ-Clustering approaches was incrementally decreased 10% and stopped when the output solution had over 50 biclusters.

The average Jaccard-based scores of these biclustering approaches when 30 data instances per data setting (see Table II-3.1) is illustrated in Figure 6.14. μ-Clustering was excluded due to memory problems for the medium-to-large datasets. For small datasets, its performance is slightly inferior than BicSPAM performance due to: 1) the absence of closing and noise-handling options, 2) delivery of non-maximal biclusters, and 3) absence of converging behavior with expressive stopping criteria. These results confirm the higher performance of BicSPAM in terms of $MS(\mathcal{B}, \mathcal{H})$ (correctness) and $MS(\mathcal{H}, \mathcal{B})$ (completeness). Although OPSM achieves a reasonable performance under the order-preserving assumption, the iterative masking of biclusters degrades the observed match score levels. Additionally, OPSM tends to discover biclusters with varying sizes, which results in a large portion of biclusters with either a very few number of rows or columns. FABIA and ISA approaches are not prepared to discover order-preserving biclusters.

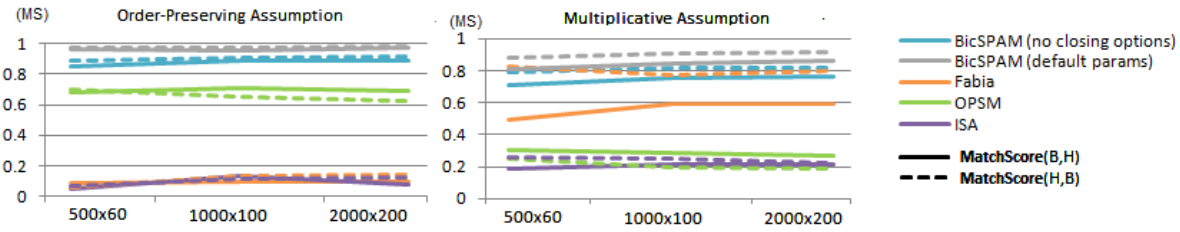

Figure 6.14: Comparing match scores across biclustering approaches using the generated datasets.

Efficiency Limits: To show the boundaries on BicSPAM efficiency when considering 20.000 rows (magnitude of the human genome), we considered the second set of synthetic data with results provided in Figure 6.15. BicSPAM support was decreased until a 5% of coverage is achieved. Two scenarios are depicted: one setting where biclusters are planted and another setting without planted biclusters. In the absence of scalability principles, BicSPAM can handle matrices up to 20.000×100. In the presence of data sampling principles (according to [626]), BicSPAM can scale for the assessed medium-to-large data settings.

Degree of Co-occurrences: Figure 6.16 illustrates the performance of BicSPAM using: the original real-values (leading to sequences with approximately 1 item/index per itemset); a discretization to consider an average of 5%

[2]http://www.bioinf.jku.at/software/fabia/fabia.html

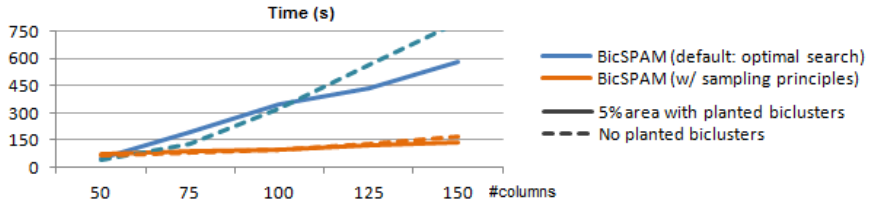

Figure 6.15: Efficiency of BicSPAM for 20000 rows in the absence and presence of sampling options.

of columns per itemset (sequences with 20 itemsets); and a discretization to consider an average of 10% of columns per itemset (sequences with 10 itemsets). These tests were performed using the default parameterizations with no closing options. The retrieved biclusters are shown to match the planted biclusters ($MS(\mathcal{B},\mathcal{H})$ and $MS(\mathcal{H},\mathcal{B})$ above 95% for medium-to-large datasets). These scores are not optimal (100%) due to the exclusion of few rows from the solution as a result of the planted noise or of the allowed overlapping among biclusters. This is also the main reason why the number of discovered biclusters is significantly higher than the number of planted biclusters[3]. As illustrated, this problem is minimized when a merging step (80% overlapping) is considered. Finally, the use of discretization methods decreases the number of precedences, which can lead to a slight decrease in efficiency due to an increase of frequent patterns.

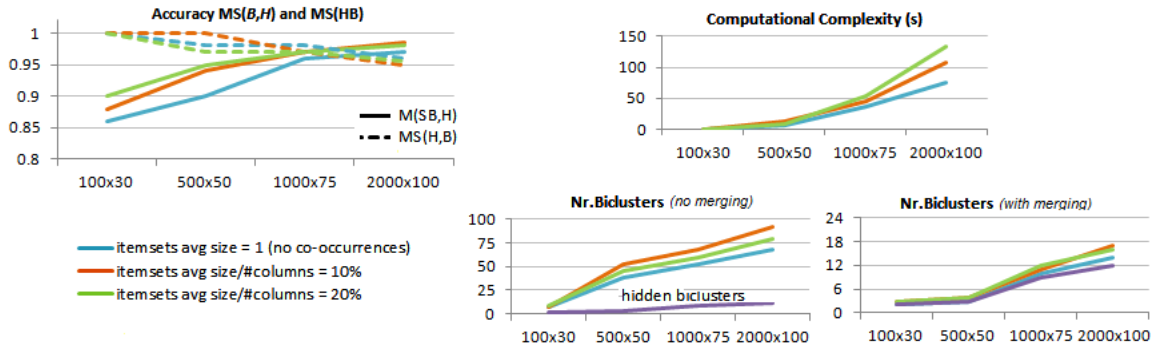

Figure 6.16: Performance of BicSPAM approaches for datasets with varying properties.

Symmetries: Figure 6.17 describes how mining symmetric behavior with BicSPAM compares with the default BicSPAM behavior (dashed lines). For this evaluation, we varied the sign of some rows for each planted bicluster. The default BicSPAM (no symmetries) was tested over the same matrices but using planted biclusters without symmetries. $MS(\mathcal{H},\mathcal{B})$ levels were not included since they are near optimal (~98-99%) across settings. The observed differences in accuracy are related with the higher probability of background values to form a non-planted order-preserving bicluster when considering symmetric behavior (validated by the high number of found biclusters). Finally, the impact of using symmetries in the time complexity is considerably less than the expected $|Y|$ times due to the incorporated heuristics to prune the search space.

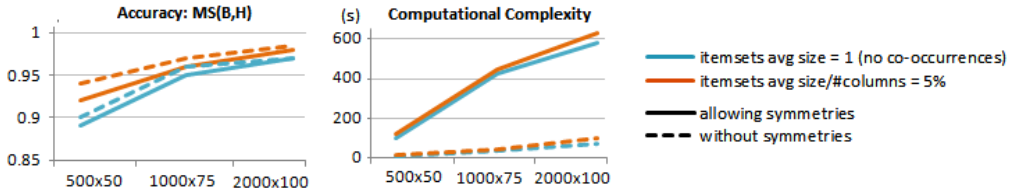

Figure 6.17: Difficulty of mining order-preserving biclusters with and without symmetries.

Other Options: For a more complete view of BicSPAM results on the generated synthetic data, we refer to our published work [311], which further studies impact of parameterizing BicSPAM with different mapping options

[3] $MS(\mathcal{H},\mathcal{B})$ reveals how the hidden biclusters were covered by the nearest found biclusters. Since there is at least one found bicluster with a direct correspondence to each hidden bicluster, BicSPAM has $MS(\mathcal{H},\mathcal{B})$ levels generally higher than $MS(\mathcal{B},\mathcal{H})$.

(including handlers of missing values), mining options (including pattern representations) and closing options (including merging, filtering, extension and reduction procedures).

6.4.3 Biological Relevance of BicSPAM

To assess the relevance of BicSPAM results in biological data contexts, we selected four distinct datasets[4]: dlbcl (180 columns/conditions, 660 rows/genes), yeast (18 columns, 2884 rows), colon cancer (62 columns, 2000 rows), leukemia (38 columns, 7129 rows). These datasets have been previously used by biclustering approaches with flexible coherency criteria [416, 324, 59].

Figure 6.18 illustrates the impact of including symmetries when mining the yeast dataset. We applied BicSPAM with an overall normalization followed by a Gaussian discretization with 20 items, closed pattern representations, and a simple merging procedure with 70% overlapping. Interestingly, we can see that order-preserving solutions that allow for symmetric behavior are able to capture a higher number of biclusters with larger sizes on average. This is an indicator of superior flexibility, which is related with the integrated modeling of (order-preserving) regulatory and co-regulatory behavior.

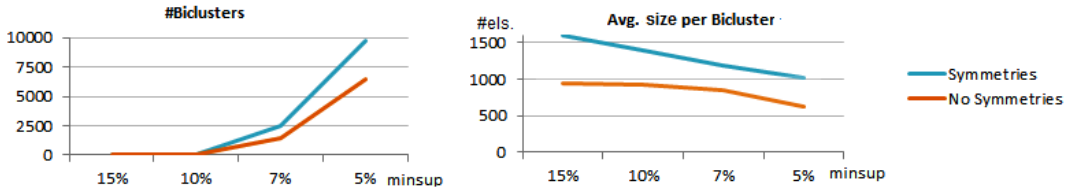

Figure 6.18: Comparing changes in order-preserving models from accommodating symmetries.

To further assess the biological significance of BicSPAM, we performed the analysis for functional enrichment by assessing the over-representation of GO terms (*Chapter II-2*). In this context, we were able to derive an average of 68 significant (and non-similar) biclusters using BicSPAM with default parameterizations per dataset when considering a minimum number of δ=5 conditions and default closing options. We consider a bicluster to be significant if it has at least 5 "biological process" terms with corrected p-values below 0.05. Two illustrative order-preserving biclusters discovered in the yeast dataset are shown in Figure 6.19.

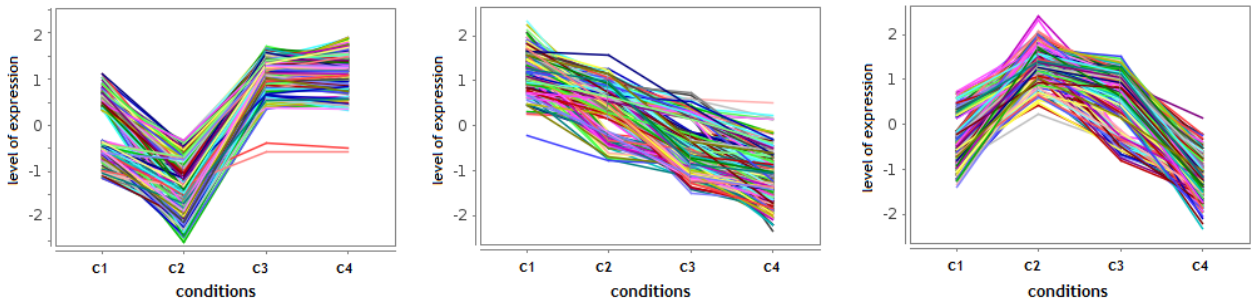

Figure 6.19: Two order-preserving biclusters with small number of conditions for the yeast dataset.

In particular, the average number of significant biclusters increases to over 80 biclusters with a larger number of elements in average when considering symmetries. This is a critical observation since it means that there are groups of genes with biological relevance that can only be discovered through biclustering under a flexible order-preserving setting when symmetries are considered.

Table 6.2 provides an illustrative set of the found order-preserving biclusters with statistical significance. The properties of the biclusters with biological significance are dependent on the type of dataset, number of items (with impact on the number of precedences) and on the allowed closing options.

[4]http://www.molbiolcell.org/content/9/12/3273/suppl/DC1
http://www.upo.es/eps/bigs/datasets.html
http://www.bioinf.jku.at/software/fabia/gene_expression.html

Dataset	♯Genes	♯Conds	♯Precedences	♯Items	Notes	♯p-values <0.01	♯p-values [0.01,0.05]	Best p-value
dlbcl	179	6	4	20	No closing options	5	2	3.12E-4
dlbcl	207	9	5	25	Merging allowed	6	1	2.33E-5
yeast	167	5	3	10	No closing options	11	3	2.12E-4
yeast	240	8	4	15	Extensions allowed	10	1	7.13E-7
colon	769	6	4	25	Merging allowed	12	2	6.08E-8
leukemia	1645	6	3	20	Extensions allowed	9	2	3.47E-9

Table 6.2: Illustrative biclusters passing the GO term-enrichment test at 1% and 5% significance levels after Bonferroni correction.

6.5 Summary of Contributions and Implications

Pattern-based approaches for order-preserving biclustering were proposed to exhaustively discover flexible structures of biclusters with flexible coherency and robustness to noise. As a result, BicSPAM was proposed based on the combined application of orderings with sequential pattern mining (SPM).

To guarantee a heightened efficiency of this learning task, a new SPM method, IndexSpan, was proposed to seize the item-indexable properties associated with dense sequential databases derived from tabular data. BicSPAM uses IndexSpan as the default SPM method due to its superior performance achieved by efficiency gains from fast database projections, minimalist data structures, and early pruning and merging techniques.

To dynamically adapt BicSPAM behavior to the properties of the input data, BicSPAM makes available different SPM methods, pattern representations, preprocessing and postprocessing procedures. Furthermore, we introduced the notion of order-preserving biclusters with symmetries and proposed a method for their effective discovery. BicSPAM contributions are three-fold:

- [*Flexibility*] Discovery of order-preserving biclusters with parameterizable degree of co-occurrences (sensitivity to differences on the observed values) and symmetries; delivery of flexible structures of biclusters to tackle restrictions of greedy approaches;
- [*Robustness*] Strategies for the discovery of solutions with adequate tolerance to noise (extending the set of principles proposed in the context of *Chapter III-2*), thus avoiding the homogeneity restrictions imposed by existing exhaustive approaches and the bias of greedy approaches;
- [*Efficiency*] Scalable searches that surpass efficiency limits of peer approaches, from seizing item-indexable properties, pruning the search space in the presence of size constraints, pushing closing options to the mining step, and making use of data partitioning principles.

Results in both synthetic and real data show that BicSPAM can surpass the drawbacks identified for existing order-preserving approaches, namely more relaxed scalability boundaries and superior robustness to noise and missing values. Results also reveal the importance of adequately tuning the sensitivity to ordering mismatches and of allowing symmetries to simultaneously capture activation and regulatory mechanisms within biological processes.

As future work, we expect to adapt the mining step to search for lengthy sequential patterns by merging smaller sequential patterns discovered under greater support thresholds according to colossal pattern mining principles [714]. This direction also promotes the scalability of BicSPAM. Finally, we expect to integrate contributions from constraint-based pattern mining in BicSPAM to support knowledge-guided biclustering in biological contexts.

7

Biclustering with Efficient Closing Options

Learning descriptive models from regions with adequate size and tolerance to noise is critical in high-dimensional data contexts to minimize their propensity towards under/overfitting. However, in the context of pattern-based biclustering, pattern mining searches are not originally prepared to easily parameterize the patterns' length and tolerance to noise. As such, the effective use of postprocessing procedures is key to adjust the quality and structures of the discovered biclusters, thus promoting their statistical significance and minimizing their under/overfitting risk.

A coherent set of postprocessing procedures – merging, filtering, extension and reduction – were proposed in *Chapter III-1* for this aim. Merging and filtering procedures rely on the observed (dis)similarity among biclusters to either merge or remove sets of biclusters. However, in the presence of large biclustering solutions, the complexity of computing the similarities between all pairs of biclusters is expensive, as well as the combinatorial analysis of the merges and removals to be performed based on the observed similarities. As a result, available procedures are typically sub-optimal and prone to efficiency bottlenecks (thus bounding the complexity of the biclustering task) [308]. Complementarily, extension and reduction procedures allow for the inclusion and exclusion of rows/-columns in the biclusters in order to maximize some homogeneity and significance criteria. However, they also suffer from efficiency bottlenecks, due to the need to apply expensive procedures to identify candidate rows and/or columns per bicluster to perform extensions and reductions [578].

To address these problems, this chapter proposes new algorithms to guarantee the efficiency and robustness of merging/filtering and extension/reduction procedures based on expectations of homogeneity (i.e. coherency and quality) and significance. As a result, this allows the manipulation of pattern-based biclustering solutions according to certain parameterizable properties, including adequate tolerance to noise (affecting quality) and size (affecting significance). In particular, six major contributions are proposed:

- anti-monotonic heuristics and cut-circuit verifications for the efficient computation of similarities;
- new algorithm based on the mapping of the merging task as the discovery of maximal circuits in unweighted undirected graphs;
- new algorithm based on the mapping of the merging task as the frequent itemset mining with variable support (referred as multi-support FIM), where support is indexed by the size of candidate biclusters;
- principles to push postprocessing calculus towards the mining step through adequate data structures for residual computational impact;
- principles for the effective identification of candidate rows and columns for extensions and reductions;
- new algorithm based on constraint-based searches (with succinct properties) to guarantee the efficiency of extension and reduction procedures;

Figure 7.1 visually summarizes the major contributions of this chapter. Accordingly, this chapter is structured as follows. *Section 7.1* overviews the contributions and limitations of related work for the target task. *Sections 7.2* and *7.3* propose efficient merging and extension/reduction procedures. *Section 7.4* assesses the performance of the proposed procedures on both real and synthetic datasets by measuring the costs of their adopt and by comparison with peer procedures from related work. Finally, the implications of this work are synthesized.

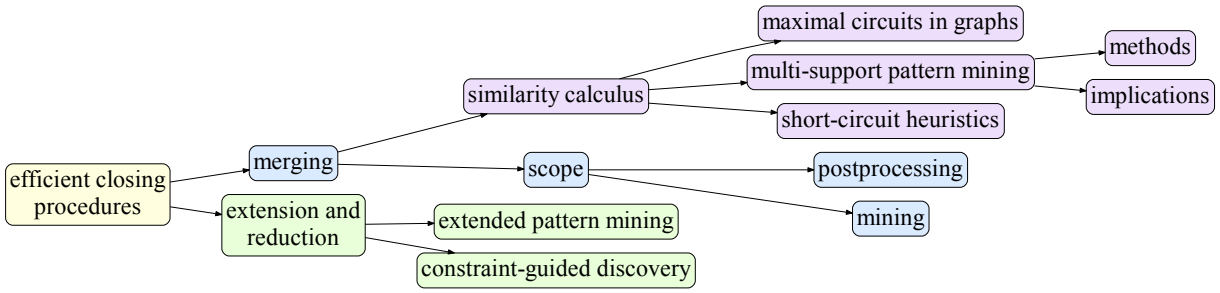

Figure 7.1: Proposed principles to guarantee the efficiency of merging, extension and reduction procedures.

7.1 Related Work: Limitations and Contributions

Throughout *Book III*, multiple contributions were proposed to guarantee the discovery of biclusters with parameterizable properties. In the context of pattern-based biclustering algorithms, these contributions can be categorized depending whether they are applied during the mapping, mining or closing step.

In the context of the mapping step, two major options were proposed with impact on the properties of biclustering solutions. First, multi-item assignments. Despite its residual impact on efficiency and relevance to guarantee robustness to noise, multi-item assignments cannot be use for flexibly controlling the biclusters' size and accepted levels of noise. Second, aggregation of views from discretization alphabets of varying length. Nevertheless, this strategy is dependent on effective and efficient postprocessing options. To check whether these limitations can be surpass, we explore the contributions and challenges associated with possible strategies along the *mining* and *closing* steps.

Mining Options: Discovery of Lengthy Full-Patterns

Existing research on pattern mining with contributions for the discovery of lengthy patterns can be divided according to three major options.

The first option is to efficiently mine patterns under very low support thresholds. There are two main strategies for this end. A first strategy is to incorporate look-ahead heuristics [406], such as the ones used by MaxMiner [49]. A second strategy to start with large candidate patterns and reduce them until the support threshold is satisfied. Contrasting with the traditional generation of incrementally lengthier patterns based on anti-monotonic searches (if an itemset is infrequent, then its supersets are also infrequent), this strategies is associated with monotonic searches (if an itemset is frequent, then its subsets are also frequent). In this context, constraint programming methods can make good use of monotonic constraints to effectively reduce the search space [545]. However, the strategies associated with this first option assume the use of low support thresholds. Understandably, this does not support the final goal of discovering larger regions since an increased pattern length (number of columns/rows) comes at the cost of an heightened decrease in the number of supporting transactions (number of rows/columns). To overcome this drawback, the remaining directions allow for noisy patterns to increase their length without a significant decrease of support levels. These directions assume a tolerance of item mismatches observed for small subsets of the supporting transactions.

The second option is to discover patterns with item-mismatches (frequent itemsets) and gap-based relaxations (sequential patterns) [691, 25, 132, 712]. However, this option is associated with an increased computational complexity of the original methods, possibly compromising the analysis of high-dimensional data [308].

A final option is to rely on searches for approximate patterns under specific principle for composing lengthy patterns [290]. In particular, colossal pattern mining relies on the approximative fusion of smaller patterns [714]. However, these principles have been only proposed in the context of frequent itemset mining and, to our knowledge, have not yet been extended for alternative pattern mining tasks.

Closing Options

Postprocessing options for biclustering were surveyed in *Chapter III-1*. Similarly to the discovery of lengthy full-patterns, their use is critical for (pattern-based) biclustering to guarantee the statistical significance of biclusters and minimize the noise dilemma. Illustrating, small biclusters with high homogeneity (noise-intolerant) can be easily discovered, yet not be significant. In this context, tolerating more noise is important to guarantee that the associated region satisfies the noise dilemma and can be used for further learning tasks without propensity to the underfitting risk. Merging, filtering, extension and reduction procedures were proposed to satisfy this challenge and tackle specific drawbacks asssociated with pattern-based searches, including the presence false positive and false negative rows/columns per bicluster and the partitioning of true biclusters in many smaller biclusters. Below, we discuss the challenges associated with their computational complexity.

The computational complexity of the closing step is essentially determined by merging and extension procedures. The complexity of filtering procedures is inferior to merging procedures as it is only bounded by similarity calculus. The complexity of reduction procedures is typically less than extension procedures due to the lower number of candidate rows/columns to test.

The computational complexity of merging procedures depends on both the calculus of similarity between (possibly) all pairs of biclusters and on the combinatorial analysis of the candidates for merging based on the observed similarities. To illustrate this combinatorial analysis, consider $(\mathbf{B}_1, \mathbf{B}_2)$, $(\mathbf{B}_1, \mathbf{B}_3)$ and $(\mathbf{B}_2, \mathbf{B}_3)$ to be similar pairs of biclusters (candidates for merging). In this scenario, $(\mathbf{B}_1, \mathbf{B}_2, \mathbf{B}_3)$ is also a candidate for merging and, in fact, merging should be jointly applied (if post-merging homogeneity criteria is satisfied) in order to guarantee an exhaustive exploration of the search space.

A naïve solution for the task of merging is to rely on multiple iterations, where for each iteration all pairs of biclusters satisfying certain similarity and/or homogeneity criteria are merged. Nevertheless, this solution is sub-optimal as it prevents the merging of groups of three or more biclusters at a time (leading to the potential loss of candidates for merging between iterations). Furthermore, this solution is associated with efficiency bottlenecks since the computation of similarities is expensive for large solutions, and the number of iterations can easily scale-up. Despite the availability of contributions for the merging of patterns/biclusters [680, 686, 57, 578, 714], they cannot simultaneously satisfy these two drawbacks.

The computational complexity of extension procedures largely depends on the applied strategy. A naïve solution would to perform for each bicluster, statistical tests on the non-included rows and columns. An illustrative statistical test for guaranteeing functionally coherent biclusters is to assess the strength between key columns (rows) of a bicluster and a new row (column) using Fisher's exact test for independence on a contingency table [578]. Understandably, this solution is associated with an impracticable complexity $\Theta(pnm\upsilon)$, where p is the number of biclusters and υ is the cost of each test. As such, heuristics need to be applied in order to perform these tests on a subset of candidate rows and columns respecting already some interestingness criteria. To reduce the computational effort of computing the p-value for every row (column) not belonging to a target bicluster, DeBi [578] proposes the use of the monotonicity property of the hypergeometric distribution: cut-off values yielding $p_i < \alpha$ are pre-computed to identify candidate rows (columns) for extension. An alternative strategy without the need to rely on a high number of statistical tests is to seed a greedy biclustering algorithm with pattern-based biclusters in order to adjust this solution bases. Understandably, this strategy is not able to offer optimality guarantees with regards to a given homogeneity criteria, and is also susceptible to efficiency bottlenecks for a high number of pattern-based biclusters.

7.2 Solution: Efficient Merging Procedures

In this section, we propose the extension of biclustering with efficient procedures to compose biclusters with parameterizable noise tolerance. Equivalently, these procedures can be also applied in the context of full-pattern

mining to compose large full-patterns. Merging procedures have been applied over sets of biclusters by computing the similarities (overlapping degree) between all subsets of biclusters. Merging occurs when a set of biclusters $\cup_k(\mathbf{I}_k, \mathbf{J}_k)$ has an overlapping area above a minimum threshold, meaning that a new bicluster $(\mathbf{I}', \mathbf{J}')$ is composed with $\mathbf{I}' = \cup_k \mathbf{I}_k$ and $\mathbf{J}' = \cup_k \mathbf{J}_k$. This new bicluster can be mapped back as a full-pattern (Def.5.1) when biclustering is not the ultimate goal. The simplest criterion to identify candidates for merging is either to rely on the overlapping area (as a percentage of the smaller bicluster), to compute the overall noisy percentage after the merging, or both. More expressive homogeneity criteria (potentially relying on the real-values provided by the input matrix) can be additionally formulated. Figure 7.2 provides a simplified view of this task, where three larger biclusters are derived from subsets of biclusters satisfying the minimum overlapping constraint. In particular, the placed overlapping criteria in this figure is the shared percentage of the larger bicluster.

When more than two biclusters overlap with each other, two criteria can be consider: to merge all rows and columns if all pairs of biclusters satisfy the considered overlapping criterion (*relaxed setting*) and to merge all rows and columns by comparing the shared area among all with the area of the larger bicluster (*restrictive setting*).

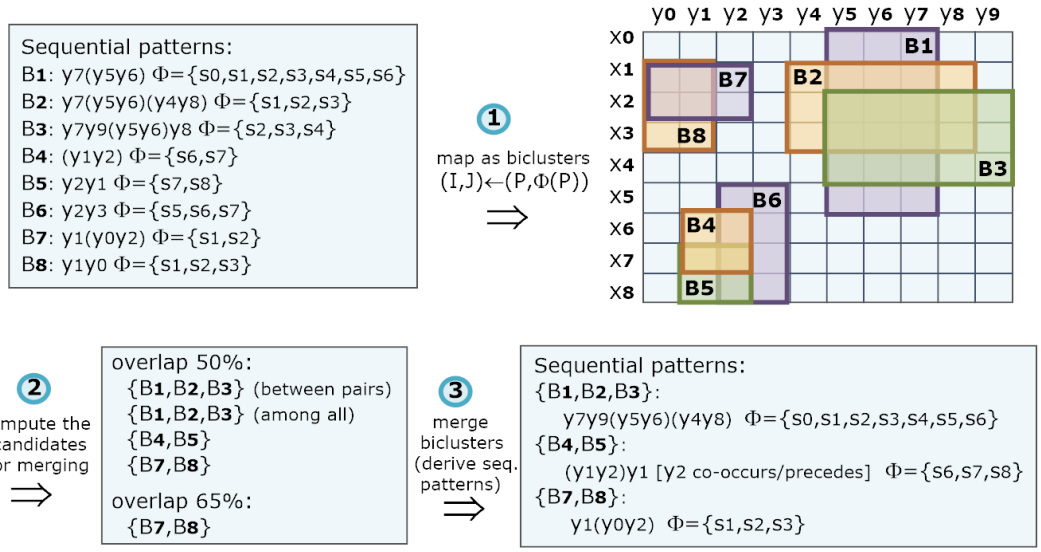

Figure 7.2: Composing lengthier full-patterns by merging biclusters: an illustrative case using sequential pattern mining.

In what follows, *Section 7.2.1* proposes effective heuristics to prune the search space, *Sections 7.2.2* and *7.2.3* combine them with two new procedures targeting respectively relaxed and restrictive settings to merging, and *Section 7.2.4* proposes additional principles to further seize efficiency gains.

7.2.1 Anti-monotonic Heuristics

Consider μ to be the overlapping degree between two biclusters. Since the overlapping degree is typically defined as the number of shared elements by the larger bicluster, heuristics can be defined assuming that biclusters are ordered by size. Consider two biclusters: a smaller bicluster, (I_1, J_1), and a larger bicluster (I_2, J_2). If they do not satisfy $|I_1|\times|J_1| \leq \mu|I_2|\times|J_2|$, we do not need to compute their similarity, neither to compute the similarity for smaller biclusters than (I_1, J_1). Therefore, (anti-)monotonicity can be used as a first heuristic for pruning the search space.

Second heuristic is to further prune the space under (anti-)monotonic searches by computing similarities along one dimension only. Pairs of biclusters not satisfying either $|I_1 \cap I_2| \geq \mu|I_2|$ or $|J_1 \cap J_2| \geq \mu|J_2|$ can be removed without computing the similarities for the remaining dimension. The chosen dimension is the one with average lower size among biclusters.

7.2.2 Merging: Maximal Circuits in Acyclic Graphs

The first proposed procedure, MergeCycle, uses the candidate pairs of biclusters for merging (discovered under previous heuristics) to construct an unweighted undirected graph, where the nodes are the biclusters and their links are given by the candidate pairs, expressing a similarity condition between two nodes/biclusters. Under this formulation, the larger candidates for merging are *edge-disjoint cyclic subgraphs* (or circuits).

Under this mapping, the search for maximal circuits is performed in order to efficiently identify candidate sets of biclusters with an arbitrary number of biclusters. This procedure is illustrated in Figure 7.3, where the cycles in the graph composed of candidate pairs are used to derive the three larger biclusters identified in Figure 7.2.

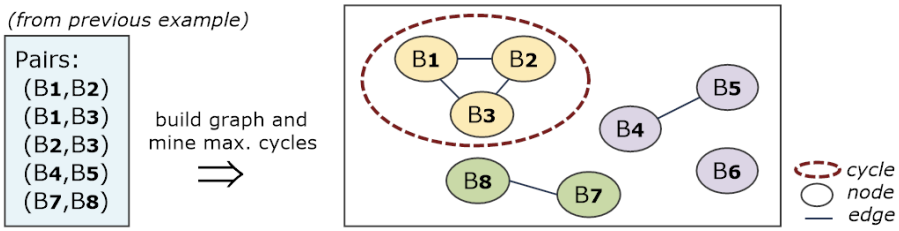

Figure 7.3: Merging candidates from pair candidates as a search by searching for maximal edge-disjoint cycles in unweighted graphs of similar biclusters.

7.2.3 Merging: Multi-support Frequent Itemset Mining

The second proposed merging procedure, MergeFIM, maps the task of merging biclusters as an adapted frequent itemset mining task. Let the elements of the original matrix be associated with a set of transactions (e.g. $\mathbf{x}_{1,1}=a_{1,1}$, $\mathbf{x}_{1,2}=a_{1,2}$ and so forth), and the biclusters be the available items (e.g. if $\mathbf{B}_1$ and $\mathbf{B}_2$ contain element $a_{1,1}$ then $\mathbf{x}_{1,1}=\{B_1, B_2\}$). Recovering the illustrative case presented in Figure 7.2, the new transaction associated with $a_{2,2}$ element would now have three items assigned, $\{B_1, B_2, B_3\}$. In this context, the support represents the number of shared elements for a specific set of biclusters (itemset).

Under this formulation, we cannot rely on a general minimum support threshold, as the minimum number of shared elements to find a candidate for merging depends on the size of the larger bicluster. For this reason, the items (biclusters) are ordered by descending order of their size. When verifying if an itemset is frequent, instead of comparing its support with a minimum support threshold, the support is compared with the minimum support of the larger bicluster ($\mu|I| \times |J|$).

As items are ordered by size, the larger bicluster corresponds to the first item (1-length prefix) of every candidate itemset. In this way, no computational complexity is added, and we guarantee that the outputted itemsets correspond to sets of biclusters that are candidates for merging.

We refer to this variant of the FIM task as multi-support frequent itemset mining. The challenge resides on designing an efficient implementation for multi-support FIM task. For this end, we propose a procedure composed according the three following major steps. First, a minimal transactional database is created. Empty transactions are removed and transactions with one item can be pruned to achieve further efficiency gains. Similarly to MergeCycle procedure, MergeFIM can also rely on the proposed heuristics to reduce the search space in order to produce the pairwise similarities. In this case, transactions with two items can be removed, and an Apriori-based method can be applied with the already 2-length itemsets derived from valid pairs of biclusters.

Second, multi-support FIM is performed using an adapted FIM search with closed itemset representations. For this aim, we replace the general notion of minimum support in FIM searches by an indexable support based on the size of larger bicluster in the context of an itemset. This requires the change of local tests on itemsets, and can be transversally applied on Apriori, pattern-growth and vertical searches. Note that by using closed representations, we avoid subsets of items with the same number of transactions. This means that the multi-support FIM task return the exact number of merges that are required.

Finally, larger biclusters are composed from frequent itemsets (candidate biclusters). This procedure is illustrated in Figure 7.4, and it follows the illustrative case introduced in Figure 7.2.

The efficiency of this procedure is guaranteed due to the observation that the mapped transactional databases tend to be highly sparse since biclusters usually only cover 5% to 10% of the elements in the original matrix.

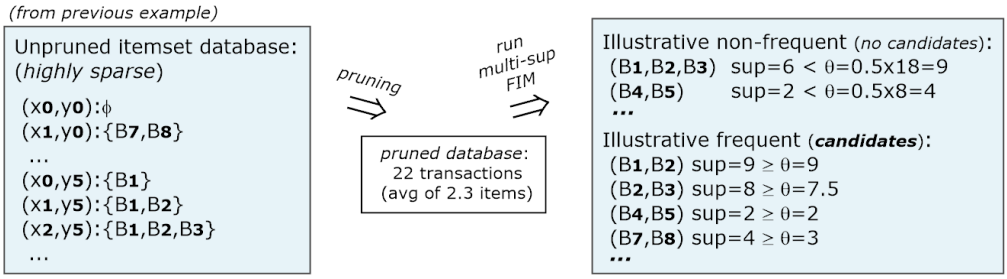

Figure 7.4: Computing candidate biclusters for merging as a multi-support frequent itemset mining task.

Understandably, the utility of solving multi-support FIM tasks goes largely beyond merging procedures, being a means to answer different numeric problems, such as counting and combinatorial calculus. The exploration of the role that MergeFIM may play when answering these problems is, however, out of the scope of this dissertation.

7.2.4 Integrated Mining and Merging of Biclusters

In order to further seize efficiency gains, the computation of similarities associated with the merging procedures can be pushed to the mining step. In this way, overlapping/similarity criteria for merging biclusters can be efficiently checked based on the differences between the ongoing discovered patterns, thus removing the need to check biclusters (even with similar sizes) if they are discovered on different regions of the input matrix. This can be achieved using dedicated distance operations on the underlying data structures associated with the target pattern mining search.

In order to guarantee that this procedure is implemented independently whether we are using Apriori, pattern-growth and vertical searches, we suggest that the discovered patterns are continuously insert in a dedicated data structure and its distances to the closer patterns computed. This dedicated data structure is given by the proposed annotated frequent-pattern trees in *Chapter 5*. Over these annotated FP-trees, simplistic distance operations can be made to test similarities between itemsets and transaction-sets. Illustrating, consider two patterns associated with two branches in the FP-tree. The similarity between the patterns can be directly tested by compared the shared number of nodes against the depth of the larger/smaller branch. Since similarities are computed as soon as a pattern is inserted in the FP-tree (against the previously stored patterns), we guarantee that all pairs of patterns are compared without repeated/unnecessary calculus.

7.3 Solution: Removing Bottlenecks of Extension and Reduction Options

In order to surpass the surveyed bottlenecks associated with extension procedures, we propose a new algorithm based on pattern mining searches applied with revised input thresholds and guided by succinct constraints for the efficient discovery of candidate columns/rows for extensions and reductions.

7.3.1 Parameterized Pattern Mining Searches

The discovery of patterns under more relaxed criteria (such as lower pattern length, support or confidence thresholds) is proposed to guide extension procedures [310]. Illustrating, by reducing the minimum pattern length (number of columns) for the pattern mining search, the resulting biclusters have higher support (number of rows). By comparing the original bicluster with the new bicluster, the additional rows associated with the new bicluster

are candidate rows for extension as they show high similarity with the rows in the original bicluster (shared subset of items).

Similarly, by reducing the minimum support thresholds (and seizing efficiency gains from pruning the search space with a higher minimum pattern length), the identification of candidate columns for extension is effectively handled. When association rule mining is considered, different forms of relaxations can be considered to guide extensions, including less restrictive ways to group the antecedent-consequent.

BicPAM implements these strategies for the identification of candidate rows and columns, and uses the minimum square residue to test whether these candidates can be jointly added or, when this is not the case, how many can be added. For this latter case, the candidates are ordered according to their similarity and the top candidates are used for extensions.

The application of an identical strategy, where pattern mining is applied with more relaxed thresholds, can also be used to guide reductions. Now the focus is on the differences between the larger dimension of the original biclusters against the smaller dimension of biclusters discovered with stricter thresholds. Note, however, that the reduction procedure is only relevant for pattern-based biclustering if there is strong evidence that the merging procedure outputs biclusters with loose homogeneity criteria.

The application of extension/reduction procedures is associated with a computational complexity similar to the mining step since the cost of statistically testing a compact set of candidate rows/columns becomes residual.

7.3.2 Constraint-based Guided Discovery

Although the computational complexity of these procedures are comparable to the complexity of the mining step, the time-and-memory costs associated these procedures can sporadicly exceed the costs of the mining step. This can happen when the impact of reducing the minimum support threshold is not compensated by the gains from pruning the space under a higher minimum pattern length. Understandably, as these costs are associated with postprocessing options, such overhead seems to be unjustifiable. Therefore, in order to foster further efficiency gains, two strategies can be implemented.

First, pattern mining searches can be adapted in order to guarantee that whenever a pattern that satisfies the input thresholds can no longer be extended in one dimension, the minimum support or pattern length is locally decreased. In the context of frequent itemset mining, this can be implemented using the previously introduced multi-support FIM task. Understandably, this strategy assumes that extensions are performed prior to merging and filtering procedures, and therefore it can suffer from an unnecessary computational overhead.

Second, we propose the incorporation of succinct constraints. These constraints show properties of interested that can be explored by pattern mining searches (either Apriori, vertical and pattern-growth [489, 520, 81]) for a guided pruning the search space. Illustrating, consider that the bicluster ($\mathbf{I} = \{\mathbf{x}_2, \mathbf{x}_7, \mathbf{x}_9, \mathbf{x}_{16}, \mathbf{x}_{18}\}, \mathbf{J} = \{\mathbf{y}_3, \mathbf{y}_8, \mathbf{y}_{27}, \mathbf{y}_{32}\}$) is discovered, and a looser support (less rows) is considered to discover candidate columns. First, the succinct constraint $\{\mathbf{x}_2, \mathbf{x}_7, \mathbf{x}_9, \mathbf{x}_{16}, \mathbf{x}_{18}\} \subseteq \mathbf{I}'$ implies the search for extensions over the new bicluster $(\mathbf{I}', \mathbf{J}')$ can be applied using the $\{\mathbf{x}_2, \mathbf{x}_7, \mathbf{x}_9, \mathbf{x}_{16}, \mathbf{x}_{18}\}$-conditional database. Second, the complementary succinct constraint $\mathbf{J}' \subseteq \{\mathbf{y}_3, \mathbf{y}_8, \mathbf{y}_{27}, \mathbf{y}_{32}\}$ can be used to further seizes efficiency gains by guiding the exploration of the search.

7.4 Results and Discussion

This section evaluates the performance of the proposed algorithms against competitive alternatives. The postprocessing procedures were implemented in Java (JVM version 1.6.0-24), and experiments computed using an Intel Core i5 2.30GHz with 6GB of RAM.

Results on Synthetic Data. To assess the impact and efficiency of postprocessing procedures, we relies on the 1000×100 setting from the data settings described in Table II-3.1 assuming an order-preserving regularity, the

absence of plaid structures and parameterizable levels of noise.

Figure 7.5 provides the gains of accuracy from correctly adjusting biclustering solutions in the presence of merging procedures (for varying overlapping degrees) and extensions. As observed in Figure 7.5a, by decreasing the minimum overlapping threshold from 100%, match scores increase since the merging step allows for the recovery of violations in the orders associated with the target order-preserving biclusters. However, this improvement in behavior is only verified up to a certain overlapping threshold. The correct identification of this threshold can lead to significant gains (near 15 percentage points for this experiment), and it is thus important to optimally address the noise dilemma. Figure 7.5b shows that the inclusion of rows and columns that still preserve a desirable homogeneity (given by 1-MSR>90%) is always beneficial against the baseline option. In sum, the overlapping threshold and homogeneity criteria for merges and extensions are for these illustrative cases the means to fix an adequate tolerance to noise.

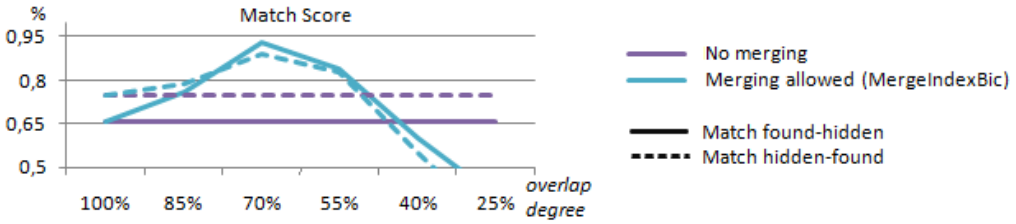

Figure 7.5: Impact of postprocessing procedures for the 1000×100 setting with planted noise: a) merging with varying overlapping thresholds for data with 5% of planted noise, and b) extensions when varying the amount of planted noise.

To assess the efficiency of the proposed principles for merging biclusters, we maintained the previous experimental settings and compared combinatorial procedures (where similarities are computed for all pairs of biclusters, and the composition of larger candidates is recursively accomplished) in the absence and presence of anti-monotonic heuristics against the proposed mappings: discovery of maximal cycles and multi-support frequent itemset mining. Multi-support procedure relies on Charm[1] method for the efficient delivery of closed frequent itemsets. Figure 7.6 illustrates this analysis for varying overlapping degrees. As observed, the proposed mappings are required as traditional procedures do not scale. Additionally, when these mappings are combined with the proposed heuristics, further efficiency gains are observed. Finally, multi-support FIM outperforms the discovery of maximal cycles for hard settings, include the cases where biclustering solutions have a high number of (possibly large) biclusters, and there are large candidates for merging, or for loose overlapping thresholds.

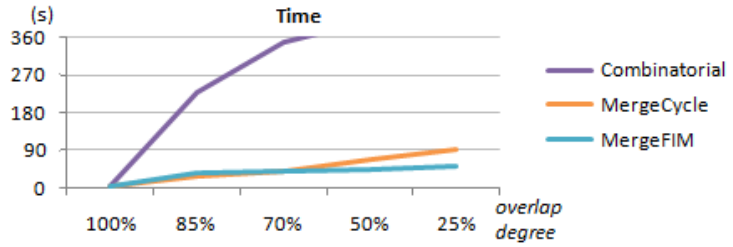

Figure 7.6: Comparing the efficiency of merging biclusters when using: combinatorial searches, maximal cycles in graphs, patterns from multi-support FIM. Options assessed in the presence and absence of heuristics to compute pairwise similarities.

Similarly, Figure 7.7 assesses the efficiency of using the proposed principles to guide the extension of biclusters (mapped from closed frequent itemsets) for datasets with varying size. The discovery of candidate rows/columns from pattern mining searches with relaxed inputted thresholds is able to surpasses the bottlenecks of combinatorial procedures and DeBi-inspired alternatives. Further significant efficiency gains are observed in the presence of succinct constraints. Figure III-2.12 complements this analysis with a view on the effectiveness of the proposed principles in the presence of varying levels of noise.

[1] Implementation from SPMF: http://www.philippe-fournier-viger.com/spmf/

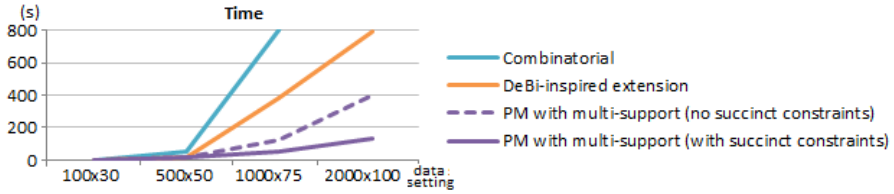

Figure 7.7: Comparing the efficiency of extending biclusters when using: combinatorial searches, DeBi-inspired procedures and pattern-based searches with relaxed thresholds in the absence and presence of succinct constraints.

Results on Real Data. The relevance of using adequate postprocessing procedures to affect the homogeneity and significance of (pattern-based) biclustering solutions was already largely demonstrated throughout *Chapters III-2-III-6*, as shown by the biological significance of BicPAM, BiP and BicSPAM outputs. In particular, Tables III-2.1, III-2.2, III-2.3, III-3.2, III-4.1, III-4.3 and III-6.2 pinpoint the relevance of BicPAM, BiP or BicSPAM when parameterized with the proposed procedures in this chapter. To briefly complement these results, we use three gene expression datasets – gasch, dlblc and hughes (described in *Section III-2*) – to illustrate the gains from merging and extending (pattern-based) biclusters. Figure 7.8 shows the differences on the total and percentage of functionally enriched biclusters from merging and extending the biclusters found by BicPAM without closing options. For this analysis, a functionally enriched bicluster has at least 10 overrepresented terms with Bonferroni corrected p-values below 0.01. We observe an increase on the relative percentage of highly enriched terms, motivating the relevance of the introduced closing options.

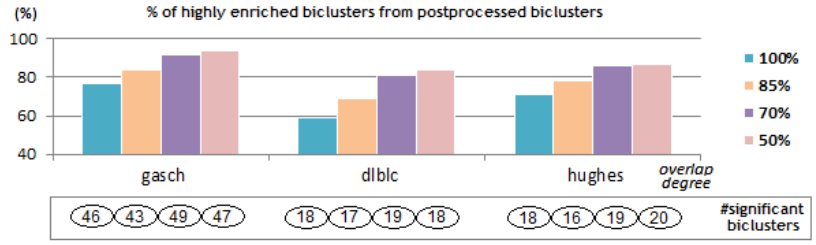

Figure 7.8: Biological relevance of using merging and extension procedures for leukemia data.

7.5 Summary of Contributions and Implications

In this chapter, closing procedures were proposed to efficiently learn (pattern-based) biclustering models with adequate size and quality. These procedures aim to surpass the surveyed computational limits associated with the available mining and postprocessing procedures.

We first proposed efficient merging algorithms based on principles that guarantee a scalable computation of: 1) similarities among all pairs of biclusters by pushing checks to the mining step and making use of anti-monotonic heuristics; and 2) the effective computation of candidate subsets of biclusters by mapping the merging task as maximal cycle discovery in undirected graphs and as multi-support frequent itemset mining.

Second, we proposed the extension of pattern mining searches (with adaptable support, pattern length and confidence thresholds) to efficiently compute sets of candidate rows/columns for extensions and reductions.

The implementation of these procedures in BicPAM is achieved by testing residue-based homogeneity thresholds and (possibly) minimum size expectations. As such, these procedures enable the parameterizable manipulation of biclusters in order to guarantee that the discovered biclusters satisfy both certain homogeneity and significance criteria (according to Def.I-1.10). Results on both synthetic and real datasets show the superior performance of these procedures against naïve and peer options.

8

Effective Biclustering of Large-Scale Network Data

The increasing precision and completeness of biological and social networks provide an unprecedented opportunity to understand the organization and dynamics of the cell functions [43] and social behavior [571]. In particular, the discovery of functional network modules has been largely used to characterize, discriminate and predict biological functions [582, 473, 43, 576] and social activity [487, 258]. For simplicity sake, this chapter places primarily the attention on the analysis of biological and social network data. Yet, the proposed contributions are applicable to other types of networks, such as coautorship/citation networks or networks inferred from financial transactions.

The task of discovering modules can be mapped into the discovery of coherent regions in weighted graphs, where nodes represent the molecular units (typically genes, proteins or metabolites) or individuals, and the edges' weights represent the strength of the interactions between the biological or social entities. In this context, a large focus has been placed on the identification of dense regions [243, 145, 174, 30], where each region is given by a statistically significant set of highly interconnected nodes. In recent years, a high number of biclustering algorithms has been proposed to discover dense regions from (bipartite) graphs by mapping them as adjacency matrices and searching for dense submatrices [30, 145, 56, 473, 426]. A bicluster is then given by two subsets of strongly connected nodes.

Despite the relevance of biclustering to model local interactions, the focus on dense regions comes with key drawbacks. First, since dense biclusters can only disclose strongly connected modules, they are usually associated with either trivial, already known putative modules. Second, the weights of the interactions may not reflect the true association strength. In biological networks, inaccurate weights are often associated with less studied genes, proteins and metabolites (with error penalizations highly dependent on the organism under study) and therefore the weights in the network may not reflect the true role of these molecular interactions in certain cellular processes [610]. In particular, the presence of (well-studied) regular/background cellular processes may mask the discovery of sporadic or less-trivial processes.

Furthermore, although many biclustering algorithms are able to find flexible coherencies in (adjacency) matrices [429], two major challenges have been preventing their application to biological and social networks. First, the relevance and biological/social meaning of network modules with flexible coherency is still unclear [313]. Second, the hard combinatorial nature of biclustering regions with flexible coherency, together with the high dimensionality of matrices derived from biological/social networks are often associated with memory and time bottlenecks, and/or undesirable restrictions on the structure and quality of biclusters.

This chapter aims to answer these problems by: 1) analyzing the biological/social relevance of modeling non-dense regions in a network, and 2) enabling the efficient learning of flexible biclustering models from large-scale networks. For this end, we propose the algorithm BicNET (Biclustering NETworks). BicNET integrates principles from pattern-based biclustering algorithms [305, 310] and adapts their data structures and searches to explore efficiency gains from the inherent sparsity of biological and social networks. Furthermore, we motivate the relevance of finding non-dense yet coherent modules and provide a meaningful analysis of BicNET's outputs. In this context, this chapter provides six major contributions:

- Extension of pattern-based biclustering to learn from data with arbitrary-high sparsity;
- Principles to guarantee the efficiency of the algorithm, including adequate data structures and searches;
- Principles for biclustering modules in weighted graphs given by flexible coherency assumptions, including constant, order-preserving, symmetric modules with non-dense yet meaningful interactions, and plaid structures to accommodate weight variations associated with network topology;
- Principles for biclustering modules robust to missing and noisy interactions;
- Principles for biclustering different types of networks, including homogeneous and heterogeneous networks, and networks with either quantitative/weighted or qualitative/labeled interactions;
- Both theoretical and empirical evidence of the (biological) relevance of modules discovered under flexible coherency and robustness criteria over (biological) network data.

Results gathered from synthetic and real data show: the relevance of the proposed efficiency principles for biclustering large-scale networks, and the effectiveness of BicNET to discover a complete set of non-trivial yet coherent and (biologically) significant modules.

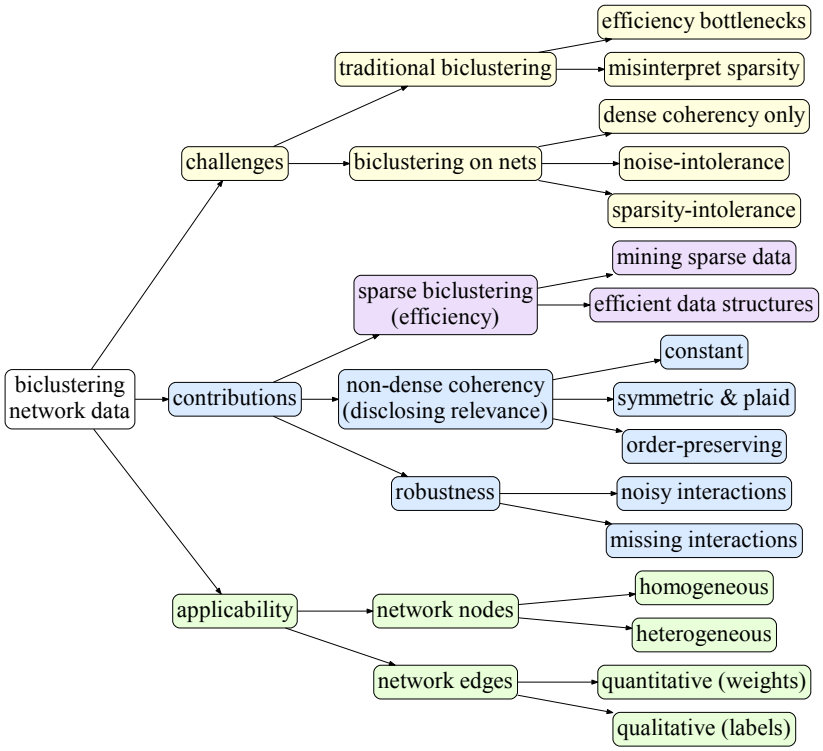

Figure 8.1: Structured view on the existing challenges, proposed contributions (and their applicability) for an effective and efficient (pattern-based) biclustering over network data.

Figure 8.1 provides a structured view on the challenges and proposed contributions of (pattern-based) biclustering network data. Accordingly, this chapter is organized as follows. *Section 8.1* provides background on the target task and surveys major contributions from related work. *Section 8.2* surveys the contributions and limitation of related work. *Section 8.3* proposes the BicNET algorithm. *Section 8.4* provides empirical evidence for the relevance of BicNET to unravel non-trivial yet relevant modules in synthetic and real networks. Finally, we draw conclusions and highlight directions for future work.

8.1 Background

Biological and Social Network Data. Network data can be described as a linked collection of interrelated objects/nodes. Network data is typically classified according to the homogeneity of nodes and interactions: homogeneous

networks have single node type and interaction type; heterogeneous networks have multiple object and/or interaction types. Networks can also have semantic information annotated to nodes and/or interactions.

In biological contexts, homogeneous networks are given for instance by protein-protein interactions (PPI) and gene interactions (GI). Heteregeneous networks are either associated with different types of interactions (binding, activation and repression, etc.) or with distinct types of nodes: 1) different molecular entities (proteins, protein complexes, metabolites, genes, etc.), 2) host and viral molecules, or 3) biological entities and terms/functions.

In social contexts, homogeneous networks model a single type of interactions between individuals, typically inferred from friendship links, email communication, instant messaging, mobile calls, logs from enterprise information systems, or linked web pages. Heterogeneous network can accommodate different types of social interaction/activity and of nodes by possibly distinguishing subjects according to their role or profile or by capturing interactions between individuals and communities.

Biclustering Flexible Network Modules. The surveyed networks can be mapped into (bipartite) graphs for the subsequent discovery of network modules associated with interconnected regions. Defs.8.1 and 8.2 extend previous formal view on biclustering of real-valued matrices and tabular data from last chapters towards the task of biclustering network data with (possibly) flexible coherency assumptions.

Def. 8.1 Given a weighted bipartite graph with two sets of nodes $\mathbf{X}=\{\mathbf{x}_1,..,\mathbf{x}_n\}$ and $\mathbf{Y}=\{\mathbf{y}_1,..,\mathbf{y}_m\}$, and interactions $a_{ij} \in \mathbb{R}$ relating nodes $\mathbf{x}_i$ and $\mathbf{y}_j$, the task of **biclustering weighted graphs** aims to find a set of biclusters $\mathcal{B}=\{\mathbf{B}_1,..,\mathbf{B}_p\}$, where each bicluster $B_k=(I_k,J_k)$ is a subgraph (module) given by two subsets of nodes, $\mathbf{I}_k \subseteq \mathbf{X} \wedge \mathbf{J}_k \subseteq \mathbf{Y}$, satisfying specific criteria of *homogeneity* and (possibly) *significance*.

This task can be solved with traditional biclustering on real-valued matrices by mapping the bipartite graph into an adjacency matrix, where rows and columns are given by the nodes and the values by the weighted interactions. In this case, subsets of rows and columns define a bicluster associated with a network module with coherent interactions.

Def. 8.2 Let the elements in a bicluster $a_{ij} \in (\mathbf{I},\mathbf{J})$ have specific coherency. A bicluster is **dense** when the average strength of its interactions, $\frac{1}{|\mathbf{I}||\mathbf{J}|}\Sigma_{\mathbf{x}_i\in\mathbf{I}}\Sigma_{\mathbf{y}_j\in\mathbf{J}}|a_{ij}|$, is significantly high.
A **constant** coherency is observed when $a_{ij}=k_j$ where k_j is the expected strength of interactions between nodes in $\mathbf{I}$ and $\mathbf{y}_j$ node from $\mathbf{J}$. In the presence of **symmetries**, $a_{ij}=k_jc_i$ where $c_i\in\{-1,1\}$.
An **Order**-preserving assumption is verified when the values for each node in one subset of nodes induce the same linear ordering across the other subset of nodes.
A **plaid** coherency considers cumulative contributions on the elements where biclusters/subgraphs overlap.

8.2 Related Work: Limitations and Contributions

Biclustering Biological Networks

A large number of algorithms has been proposed to find modules in unweighted and/or weighted graphs mapped from homogeneous and/or heterogeneous biological networks [473, 125, 582]. In unweighted graphs, shortest-path algorithms [255], clique detection with Monte Carlo optimization [596], probabilistic motif discovery [65] and clustering on graphs with network distance functions [125] have been, respectively, applied to discover modules in PPIs (yeast), GIs (E. coli) and metabolic networks. In unweighted bipartite graphs, the densest regions correspond to bicliques. Bicliques can be efficiently mined using density-constrained biclustering [144], Motzkin-Straus optimization [174], formal concepts [381] and pattern-based biclustering [56, 473, 683, 426]. In weighted graphs, the density of a module is given by the average strength of interactions. Strength is either determined by a measure of confidence (when it is predicted from literature or diverse data sources) or by the functional correlation between nodes (when it is derived from experimental data). Densely weighted modules have been discovered with betweenness-based partitioning [125], graph flow-based clustering [528] and several biclustering

approaches, including SAMBA [616], multi-objective searches [444] and pattern-based biclustering [154, 145, 30]. The application of these methods over homogeneous and viral-host PPIs show that protein complexes largely match the found modules [444, 125, 528].

The discovery of dense network modules has been largely accomplished with pattern-based biclustering algorithms [145, 154, 56, 473, 683, 426, 30] due to their intrinsic ability to exhaustively discover flexible structures of biclusters. This task should not be mingled with the task of inferring networks from biclustering alternative data sources (Pointer 8.1). Frequent patterns in discrete networks can be mapped as biclusters with specific coherency strength determined by the number of symbols (ranges of weights) assigned to the interactions. In unweighted graphs, closed frequent itemset mining and association rule mining were applied to study interactions between proteins and protein complexes in yeast proteome network [683] and between HIV-1 and human proteins to predict and characterize host-cellular functions and their perturbations [473, 426]. More recently, association rules were also used to obtain a modular decomposition of positive and negative GIs ($a_{ij} \in \{-1,0,1\}$) for understanding between-pathaway and within-pathway models of GIs [56]. In weighted graphs, Dao et. al [154] and Atluri et. al [30] relied on the loose antimonotone property of density to propose weight-sensitive pattern mining searches. Finally, DECOB [145], originally applied to PPIs and GIs from human and yeast, uses an additional post-filtering step to output of non-similar modules only.

Pointers 8.1 Biclustering for network inference

A rather different use of biclustering models for network data analysis is given when biclustering is perform on different datasets to identify candidate interactions between nodes from biclusters for network reconstruction/inference [529]. In biological domains, the goal is to score or infer previously unknown relationships among biological entities using link prediction scores from biologically significant biclusters discovered from expression data [77], genome-wide data [554] proteomic and metabolomic data [462, 155]. For this purpose, evolutionary, least squares fitting-based and Bayesian approaches to biclustering have been used [156, 157, 17, 611].

Despite the relevance of the surveyed work and of additional contributions proposed in the context of labeled data (Pointer 8.2), they are centered on dense modules' discovery, and thus insufficient to answer the target task.

Pointers 8.2 Supervised learning from discriminative network modules

Some of the surveyed works have been extended to discover discriminative modules, often referred as multigenic markers, for classification tasks such as function prediction [154, 582, 426]. These modules are critical to surpass the limitations of single gene markers and topological markers, since they have been shown to be more robust and stable markers [142, 141]. Network-based (bi)clustering methods for function prediction have been comprehensively reviewed by Sharan et. al [582].

Biclustering Social Networks

The discovery of modules in social networks (also referred as community detection, subgraph discovery) has been solved recurring to both dedicated methods and the outputs of alternative well-known tasks such as network modeling, topology/architecture extraction, information diffusion/propagation, and link prediction [618, 80]. Illustrating, the identification of dense modules has been combined with topology extraction to interpret community structures using genetic algorithms [530]. In the context of link mining, the discovery of modules with qualitative interactions satisfying certain coherency criteria has received some attention to surpass the inherent limitation of (statistical) models only able to analyze numeric and binary networks [245, 246]. Graph mining approaches for community detection have been surveyed by Tang and Liu [618]. In particular, pattern mining algorithms for graph analysis have been show to be effective towards this end [713, 685].

Similarly to related work on biological network, by mapping networks as (bipartite) graphs, techniques for searching (maximal) bicliques have been proposed using formal concept analysis [488] and biclustering [258]. Despite the relevance of biclustering for the discovery of coherent communities, it has been scarcely applied. BSN [294] combines min–max procedures with and pattern searches to construct hierarchical biclusters and discover dense interactions among nodes in a social network in order to easily interpret its structure. TeamFinder [210] is

an alternative biclustering method allows the analysis of heterogeneous nodes by aggregating subjects/nodes in a network based both on their skills (annotated labels) and strength of interactions.

The problem with the surveyed research is, again, the lack of focus on the discovery of network modules with flexible coherency and parameterizable quality.

Biclustering Modules with Flexible Coherency

Although the state-of-the-art is primarily focused on the discovery of dense network modules, slight variants of this coherency have been proposed [616, 30], and discussed by Dittrich et al. [177] and Ideker et al. [340]. Despite the large availability of traditional biclustering algorithms able to find biclusters with flexible coherency [429], empirical evidence shows that they are not prepared to deal with the sparsity and/or high-dimensionality of adjacency matrices mapped from networks (see *Section 8.4*). A first attempt of applying biclustering algorithms with flexible coherency over (biological) networks was presented by Tomaino et al. [628]. Despite its relevance, this work only considered very small networks (less than 50 nodes and 200 interactions), and did not explore the (biological) meaning and relevance of the target coherencies associated with the discovered biclusters.

8.3 Solution

In what follows, we first introduce principles to enable the sound application of biclustering over network data. Second, we motivate the relevance of discovering coherent modules following constant, symmetric and plaid models. Third, we show how to discover modules robust noisy and missing interactions. Finally, we extend pattern-based searches to optimally handle the inherent structural sparsity of biological and social networks.

Biclustering Weighted Graphs

For an effective application of state-of-the-art biclustering algorithms to (weighted) graphs derived from network data, two principles should be satisfied. First, the weighted graph should be mapped into a minimal bipartite graph. In heterogeneous networks, multiple bipartite graphs are created (each with two disjoint sets of nodes with heterogeneous interactions). The minimality requirement can be satisfied by identifying subsets of nodes with cross-set interactions but without intra-set interactions to avoid unnecessary duplicated nodes in the disjoint sets of nodes (see Figure 8.2). This is essential to avoid the generation of large bipartite graphs and subsequent very large matrices.

Second, when targeting non-dense coherencies from homogeneous networks, two real-valued adjacency matrices need to be derived from the bipartite graph (a matrix with rows and columns mapped from the disjoint sets of nodes and its transpose). A third matrix, and the one used for the subsequent learning, should be created as the sum of the previous matrices. Despite the relevance of this second principle, some of the few attempts to find non-dense biclusters in biological/social networks fail to satisfy it [628], thus delivering incomplete and often inconsistent solutions.

Under the satisfaction of the previous two principles, a wide-range of biclustering algorithms can be applied to discover modules with flexible coherencies [429]. Yet, only pattern-based biclustering [305, 310, 314] is able to guarantee an exhaustive yet efficient discovery of flexible structures of biclusters with parameterizable coherency and quality criteria. This provides the necessary context to measure the relevance and impact of discovering modules with non-dense coherency and noise-tolerance. In particular, we rely on BicPAM, BiP and BicSPAM algorithms [310, 303, 311], which, respectively, use frequent itemset mining and association rule mining to find biclusters with constant, plaid and order-preserving coherencies in the absence and presence of symmetries. Furthermore, they integrate the dispersed contributions from previous pattern-based algorithms and address some of their limitations, providing key principles to surpass discretization problems and robustly handle noise and missing values. Figure

8.2 provides a view on how transactions can be derived from (heterogeneous) network data for the discovery of constant modules (see previous chapters for details on the itemization, mining and postprocessing steps).

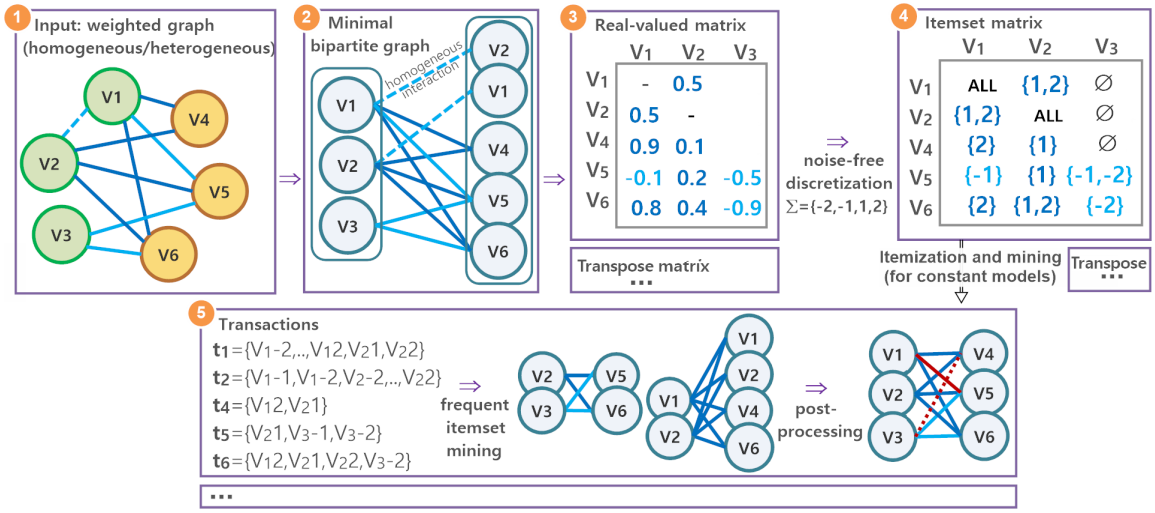

Figure 8.2: Pattern-based biclustering of (heterogeneous) networks: mining real-valued matrices derived from minimal weighted bipartite graphs.

8.3.1 Network Modules with Flexible Coherencies

Constant Model. Given a bicluster defining a module with coherent interactions between two sets of nodes, the constant coherency (Def.8.2) implies that the nodes in one set show a single type of interaction with the nodes in the remaining set. Consider an illustrative biological network with a set of interactions between genes and proteins, where their absolute weight defines the strength of the association and their sign determines whether the association corresponds to activation or repression mechanisms. The constant model guarantees that when a gene is associated with a group of proteins, it establishes the same type of interaction with all these proteins (such as heightened activation of the transcription of a complex of proteins). When analyzing the transposed matrix (by switching the disjoint sets of the bipartite graph), similar relations can be observed: a protein coherently affects a set of genes (softly repressing their expression, for example). In biological domains, the constant model can also disclose relevant interactions between homogeneous groups of genes, proteins and metabolites. Figure 8.3 provides an illustrative constant module.

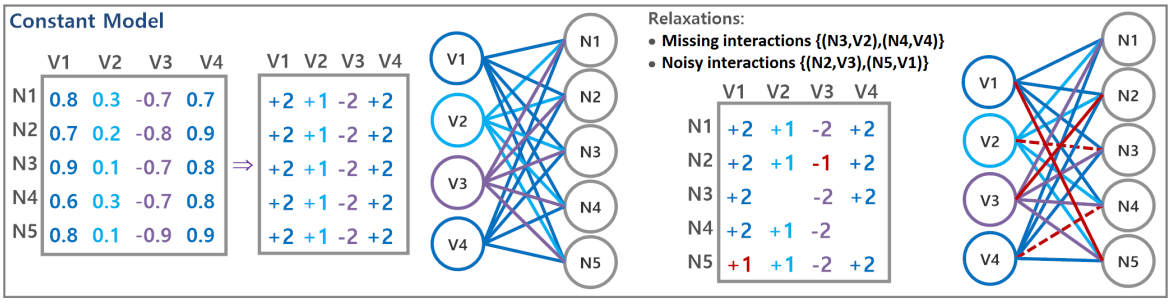

Figure 8.3: Biclustering (noise-tolerant) modules with the constant model.

The proposed constant model can be directly applied to networks with qualitative interactions capturing distinct types of regulatory relations. In biological domains, interactions associated with *binding*, *activation* or *enhancement* associations are commonly observed for a wide-variety of PPIs [426, 473]. Alternatively, in social domains, distinct types of interactions between subjects (based on the type of friendship, nature of interaction and user's profile) are also increasingly available [210, 246].

The constant model is essential to guarantee that social/biological entities with non-necessarily high (yet coherent) influence on another set of entities are not excluded. The constant coherency is in general more flexible

than the dense coherency, leading to the discovery of larger modules. The exception is when the dense coherency is not given by highly weighted interactions, but instead by all interactions independently of their weight (extent of interconnected nodes).

Symmetric Model. The presence of symmetries is key to simultaneously capture positive and negative interactions associated with the interactions of a single node [310]. In biological contexts, this is critical to model activation and repression mechanisms. In this contexts, the symmetric model introduces a new degree of flexibility by enabling the discovery of more complex regulatory modules, where a specific gene/protein may show symmetric regulatory behavior according to the expected pattern, yet still respect the observed coherency. In social contexts, symmetries are useful to coherently model activity distinguishing interest (such as endorsements and subscription) from oblivion. Figure 8.4 illustrates the symmetric model, where rows with symmetries are identified with dashed lines.

Plaid Model. The plaid assumption [303] is essential to describe overlapping regulatory/social behavior associated with cumulative effects in the strength of interactions between nodes that appear in multiple functional modules. Illustrating, consider that two genes interact in the context of multiple biological processes, a plaid model can consider their cumulative effect on their interaction's weights (based on the expected weight associated with each active process). This is also valid for the regulatory influence between proteins and for heterogeneous networks. The plaid assumption of GIs and PPIs also provides insights on the network topology and molecular functions, revealing hubs and core interactions (based on the amount of overlapping interactions), and between- and within-pathway interactions (based on the interactions inside and outside of the overlapping areas).

In social networks, the plaid assumption is critical to describe activity from overlapping communities and to adequately model centrality individuals that act as hubs between communities. Figure 8.4 illustrates a plaid model associated with two overlapping modules. These modules could not be discovered without a plaid assumption.

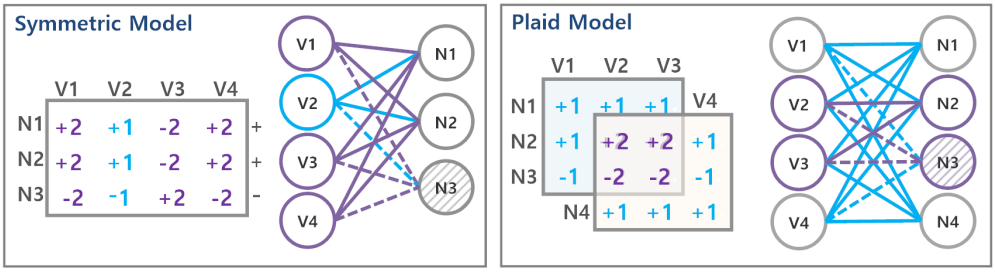

Figure 8.4: Biclustering modules with the symmetric and plaid models.

Order-Preserving Model. An order-preserving module/bicluster is defined by a set of nodes with a preserved relative degree of influence on another set of nodes [311, 415]. Illustrating, given an order-preserving module with two proteins acting as a transcription factors of a set of genes/proteins/metabolites, this implies that these proteins show the same ordering of regulatory influence on the biological molecules. Order-preserving regularities are also observed across subjects within a social community based on their degree of involvement and activity. Order-preserving modules may contain interactions according to the constant model (as well as modules with shifting and scaling factors [311]), leading to more inclusive solutions associated with larger and less noise-susceptible modules. The order-preserving model is thus critical to accommodate non-fixed yet coherent influence of a node on another set of nodes, tackling the problem of noisy scores from less-studied regions in the network.

An order-preserving coherence with symmetries is often used to model biological settings where the degree of both the activation and repression of groups of genes, proteins and metabolites is preserved. Figure 8.5 provides illustrative order-preserving modules in the absence and presence of symmetries.

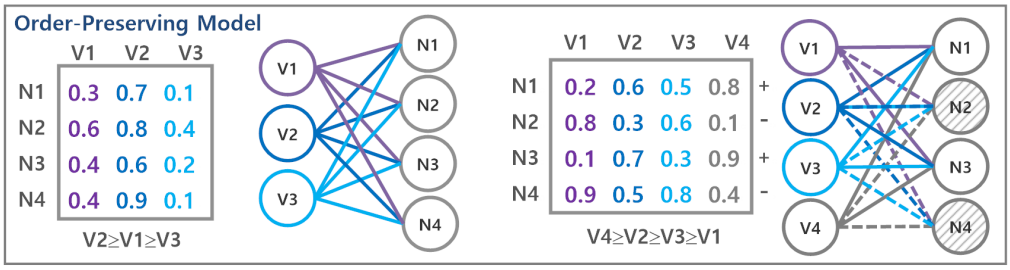

Figure 8.5: Biclustering modules with flexible coherencies: the order-preserving model.

8.3.2 Modules with Noisy and Missing Interactions

An undesirable restriction of exhaustive searches for (possibly dense) modules is that they may exclude relevant nodes associated with a bicluster if those nodes do not interact with all of the nodes in one subset of nodes from the bicluster. Understandably, meaningful modules with missing interactions are common since existing biological networks are still largely incomplete. Pattern-based biclustering is able to recover missing interactions recurring to well-established and efficient postprocessing procedures (based on the merging and extension of the discovered modules) [310].

Furthermore, the scoring scheme of interactions might be prone to experimental noise, preprocessing biases and structural noise, not always reflecting the true interactions. In biological networks, this is particularly common for less-studied and unstable genes or proteins. In social networks, this is associated with the highly stochastic nature of user behavior.

Pattern-based biclustering also allows the assignment of multiple symbols to specific interactions [310], thus avoiding the exclusion of noisy interactions (see Figure 8.2). Although default parameterizations are provided to guarantee an adequate tolerance to noise, the level of sparsity and noise of the modules can be parametrically controlled using thresholds based on quality expectations. Figure 8.3 illustrates a coherent module with corrections associated with missing interactions (red dashed lines) and noisy interactions (red continuous lines).

8.3.3 BicNET: Efficiently Biclustering Network Data

Understandably, the task of discovering modules with the introduced coherencies is more complex than finding dense modules (complexity discussed in [310]). Empirical evidence shows that state-of-the-art biclustering algorithms are only scalable for biological/social networks up to a few hundreds of nodes (see *Section 8.4*). Nevertheless, a key property distinguishing network data from tabular data is their underlying sparsity. Illustrating, some of the densest PPI and GI networks from well-studied organisms still have a density below 5% (ratio of interconnected nodes after excluding nodes without interactions). Similar density ratios are observed for social networks derived form web communities [606]. While traditional biclustering depends on operations over matrices, pattern-based biclustering algorithms are prepared to mine transactions of varying length. This property makes pattern-based biclustering able to exclude missing interactions from searches and thus surpass memory and efficiency bottlenecks. Based on this observation, we propose BicNET (**Bic**lustering Biological **NET**works), a pattern-based biclustering algorithm for the discovery of network modules with non-trivial coherencies and robustness to noise. Additionally, BicNET relies on the following principles to explore further efficiency gains.

We propose a new data structure to efficiently preprocess data: an array, where each position (node from a disjoint set in the bipartite graph) has a list of pairs, each pair representing an interaction (corresponding node and the interaction weight). Discretization and itemization procedures are performed by linearly scanning this structure three times. Thus, their time and memory complexity is linear on the number of interactions.

Pattern-based searches commonly rely on bitset vectors due to the need to retrieve not only the frequent patterns but also their supporting transactions in order to compose biclusters. However, bitset vectors are costly in terms of memory, and the associated intersection operations are computationally expensive for large-scale networks. For

this reason, we rely on the recently proposed F2G miner [315] and on revised implementations of Eclat and Charm miners where diffsets are used to address the bottlenecks of bitsets. These pattern-based searches guarantee an efficient discovery of constant, symmetric and plaid models.

Furthermore, the underlying pattern mining searches of BicNET are dynamically selected based on the properties of the network to optimize their efficiency. Horizontal versus vertical data formats [310] are selected based on the ratio of rows and columns from the mapped matrix. Apriori (candidate generation) versus pattern-growth (tree projection) searches [310] are selected based on network density (pattern-growth searches are preferable for dense networks). We also push the computation of similarities between all pairs of biclusters (the most expensive postprocessing procedure) into the mining step by checking similarities with distance operators on a compact data structure to store the frequent patterns.

8.4 Results and Discussion

Results are organized as follows. *Section 8.4.1* compares the performance of BicNET against state-of-the-art biclustering algorithms using artificial networks. *Section 8.4.2* applies BicNET over large-scale PPI and GI networks to show the relevance of discovering modules with flexible coherencies and parameterizable levels of noise and sparsity. BicNET is implemented in Java (JVM v1.6.0-24). Experiments were computed using an Intel Core i5 2.30GHz with 6GB of RAM.

Synthetic Data. Accordingly to the generation procedures proposed in *Section II-3.1.5*, networks with planted coherent modules were generated respecting the commonly observed topological properties of biological and social networks. Table II-3.2 describes the used data settings. The major variables used to affect the structural properties of network data include:

- number of nodes (from 200 to 10000 nodes), density (from 1% to 25%) and distributions of the weight (positive and negative ranges revealing the interaction strength);
- degree of noisy and missing interactions (from 0% to 20%).
- number, size (Uniform distribution on the number of nodes), shape (imbalance on the size of the disjoint sets of each subgraph), overlapping, and coherency (dense, constant, symmetric, plaid and order-preserving) of the planted biclusters.

Real Data. We used four biological networks: GIs in yeast from DryGIN [372] and STRING v10 [610] databases, and two licensed PPIs in human and E. coli from STRING v10 [610] database. The scores in these networks reveal the expected strength of influence/physical interaction between genes/proteins. DryGIN networks are inferred from experimental data, while STRING networks are primarily inferred from literature and knowledge bases. Table 8.1 shows some basic statistics of the selected networks.

Type	Organism	♯Nodes	♯Interactions	Density	Notes on the weight of interactions
GI	Yeast	4455	191309	1.0%	Weights (65% negative) from double-mutant arrays [372].
GI	Yeast	6314	423335	1.1%	Known and predicted associations benchmarked
PPI	E. Coli	8428	3293416	4.6%	from multiple data sources and text mining,
PPI	Human	19247	8548002	2.3%	and combined through an integrative score [610].

Table 8.1: Biological networks used to assess the relevance and efficiency of BicNET.

Performance Metrics. Given the set of planted modules $\mathcal{H}$ in a synthetic network, the accuracy of the retrieved modules $\mathcal{B}$ can be given by biclustering match scores surveyed in *Chapter 2*. In particular, to address the problem of Jaccard-based and FC matches (only focused on one of the two subsets of nodes at a time [310]) and RNIA (loose matching criteria [310]), we rely on the integrative metric proposed in Eq.2.9 (*Section 2.2*). Efficiency and domain significance are used to complement this analysis.

8.4.1 Results on Synthetic Data

Figure 8.6 compares the efficiency of BicNET with state-of-the-art biclustering algorithms with flexible coherence criteria using networks with varying size and density and planted modules with constant coherency. We selected FABIA (with sparse prior equation and decreasing sparsity until able to retrieve a non-empty set of biclusters) [324], ISA [342], xMotifs [477], CC [134] and OPSM [59] to discover modules with flexible coherency. We preserved the parameterizations used in previous chapters. Three major observations can be retrieved. First, BicNET shows heightened efficiency levels, constrasting with peer biclustering algorithms. Understandably, as most of the remaining algorithms are only prepared to analyze (non-sparse) matrices, they show efficiency bottlenecks for even small networks. Second, the majority is not able to accurately recover the planted modules as they cannot interpret missing interactions. Third, although SAMBA [616] and some pattern-based biclustering algorithms, such as BiMax and DECOB [473, 145], are able to discover dense models efficiently, they are not prepared to discover modules with alternative coherence criteria.

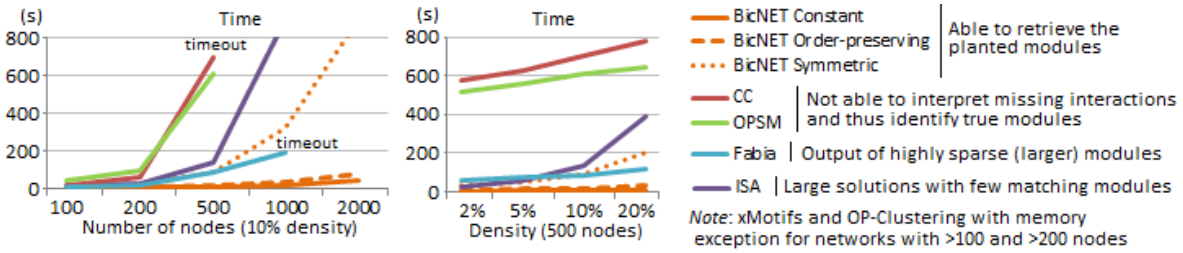

Figure 8.6: Efficiency of flexible biclustering algorithms to discover constant modules in synthetic networks with varying size and density.

Figure 8.7 zooms-in on the performance of BicNET by quantifying the efficiency gains in memory and time from using adequate data structures (replacing the need to use matrices) and searches (replacing the need to rely on bitset vectors). It also shows that the cost of assigning multiple symbols per interaction are moderate, despite resulting in an increased network density.

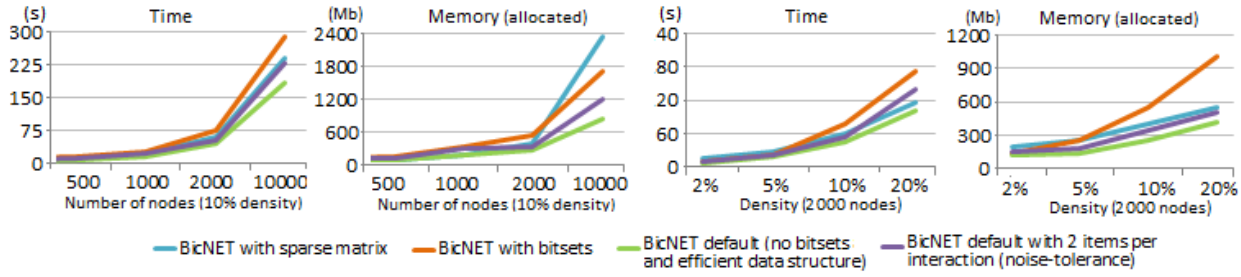

Figure 8.7: Efficiency gains of BicNET when using sparse data structures, pattern mining searches providing robust alternatives to bitset vectors, and noise handlers.

Figure 8.8 compares the performance of BicNET with peer algorithms for discovering dense network modules (hypercliques) in the presence of noisy and missing interactions. This analysis clearly shows that existing pattern-

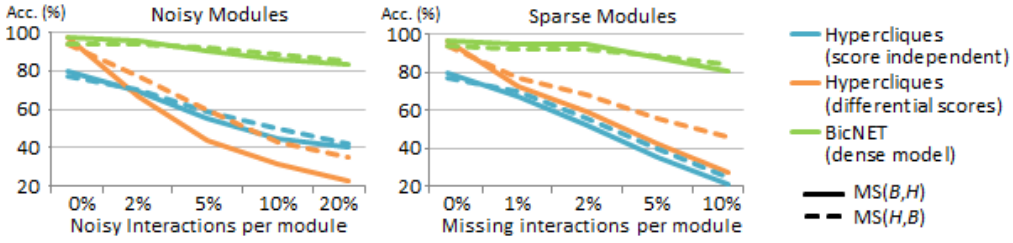

Figure 8.8: Accuracy of BicNET against peer pattern-based searches to discover dense modules on networks (2000 nodes, 10% density) with varying degree of noise and missings.

based searches for hypercliques have no tolerance to errors since their accuracy rapidly degrades for an increased number of planted noisy/missing interactions. Thus, they are not able to deal with the natural incompleteness and scoring uncertainty associated with biological networks. On the other hand, the observed accuracy levels of BicNET demonstrate its robustness to noise (validating the importance of assigning multiple ranges of weights for some interactions) and to missing interactions (showing the effectiveness of BicNET's postprocessing procedures).

Finally, Figure 8.9 shows that, even in the presence of medium-to-high levels of noise, BicNET can be effectively applied for the discovery of modules with distinct coherencies. All of the target coherencies are associated with searches showing high levels of accuracy, with the plaid model being slightly worse than its peers due to the inherent harder nature of this task when multiple modules overlap according to a complex schema. Additionally, order-preserving models have higher propensity to define modules with false positive nodes for dense networks due to the higher probability of background values to respect this coherency.

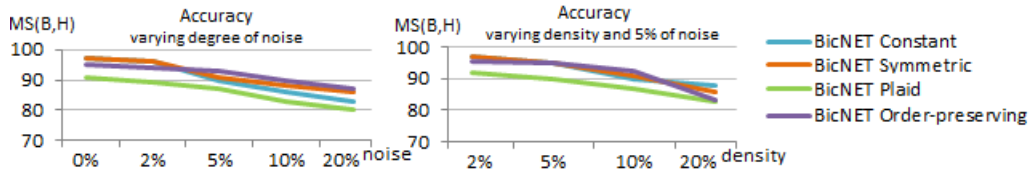

Figure 8.9: Correct recovery of modules from noisy networks with planted constant, symmetric and plaid coherencies (networks with 2000 nodes).

8.4.2 Results on Real Data

Results gathered from the application of BicNET over real biological networks are provided in three parts. First, we show basic statistics that motivate the relevance of using BicNET against peer algorithms. Second, we explore the biological relevance of the retrieved modules when considering varying levels of tolerance to noise and different forms of coherency. Finally, we analyze less-trivial modules (such as modules characterized by the presence of plaid effects, flexible constant patterns or symmetries), and provide a brief analysis of their enriched terms and transcription factors.

The biological significance of the retrieved modules from real data is here computed by assessing the over-representation of Gene Ontology (GO) terms with an hypergeometric test using GOrilla [194]. A module is significant when its genes or proteins show enrichment for one or more of the "biological process" terms by having a (Bonferroni corrected) p-value below 0.01.

Figure 8.10 shows some of the properties of BicNET solutions for the four biological networks described in Table 8.1. In particular, 97% of the BicNET's modules discovered in DRYGIN's yeast GIs were significantly enriched, while all the BicNET's modules discovered in STRING's yeast GIs were significantly enriched. BicNET is able to discover the largest number of (non-similar and statistically significant) biclusters. The analysis of the enriched terms for these modules (see Tables 8.2 and 8.4) against the significant terms found in other biclustering solutions supports the completeness of BicNET's solutions, as well as their exclusivity and relevance since the majority of the enriched modules were not discovered by peer algorithms (see Table 7.3). The biological significance of peer biclustering algorithms focused on dense regions is further hampered by noise and discretization errors (in accordance with Figure 8.10). Alternative biclustering algorithms able to discover non-dense regions were not able to scale. The subsequent analyzes (Tables 8.2–) provide further empirical evidence for the relevance, completeness and exclusivity of BicNET solutions.

Modules with Flexible Coherency. A subset of the overall modules collected from the application of BicNET over the selected biological networks is provided in Table 8.2. This table gathers modules with varying: tolerance to noise (overlapping threshold for merging procedures varied between 60% and 90%), coherency assumption (dense, constant and order-preserving models) and coherency strength (D_1-D_4 with $\mathcal{L}$={-2,-1,1,2}, Y_1-Y_5 and H_1-H_3 with

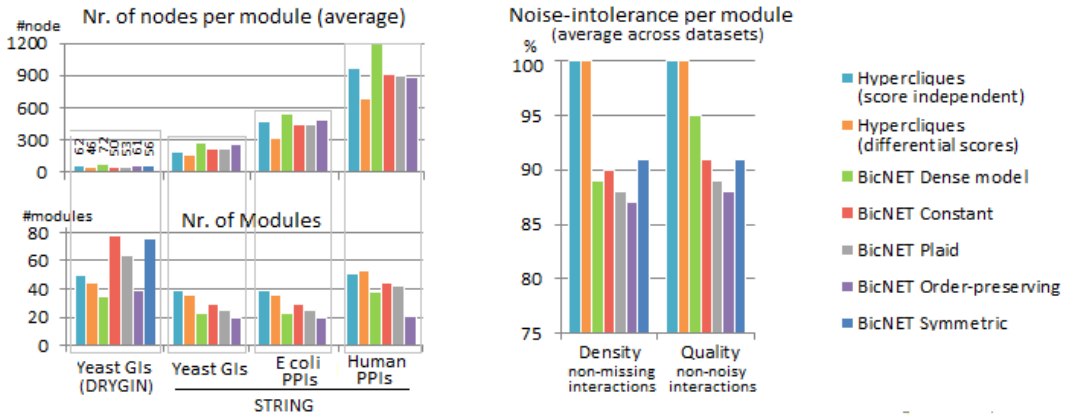

Figure 8.10: Properties of BicNET solutions against hypercliques discovered in GI and PPI networks (described in Table III-8.1) when considering varying coherency criteria.

$\mathcal{L}$={1,2,3}, Y_6 and H_4 with $\mathcal{L}$={1,2,3,4}). All of the modules were discovered using multi-item assignments whenever values were found to be near a discretization boundary. The collected results show that all of BicNET's modules had not only highly enriched terms, but also the enriched terms were found to be functionally related (taxonomically closed biological processes [440]). This observation suggests that the discovered modules are characterized by a cohesive set of putative biological functions. To support this observation, Figures 8.11 and 8.12 provide an hierarchical visualization of some of the enriched terms (recurring to GOrilla tool [194]) for a subset of the discovered modules.

	ID	Homogeneity	♯Nodes $\lvert I\rvert\times\lvert J\rvert$	Putative functionality: group of enriched terms (p<1E-10)
STRING(yeast)	Y1	dense (high noise-tolerance)	231×14	Metabolic processes with incidence on protein, peptide and amide metabolism and biosynthesis.
	Y2	dense (medium noise-tolerance)	217×9	Metabolism of nitrogen compounds and some organic substances.
	Y3	constant (few high a_{ij})	103×8	Amino acid activation and tRNA metabolism for tRNA aminoacylation.
	Y4	constant (few high a_{ij})	206×6	Organic acid metabolic process and its subterms.
	Y5	constant (few high or low a_{ij})	55×7	Signal transduction and its subterms.
	Y6	constant (few high or low a_{ij})	43×6	Phosphorylation related terms (with incidence on protein phosphorylation).
	Y7	order-preserving	176×12	Transport of organic acids (with incidence on aminoacid transmembrane transport).
	Y8	order-preserving	235×9	Oxidation-reduction process and metabolism of aminoacids. Assembly of ribonucleoprotein.
	Y9	order-pres. (few high a_{ij})	146×8	Transport of molecules (highest enrichment found for drug transmembrane).
STR.(human)	H1	dense (high noise-tolerance)	811×28	Multiple metabolic processes with incidence on transcription activity.
	H2	dense (high noise-tolerance)	787×25	Regulation of metabolic processes (both positive and negative regulation).
	H3	constant (few high a_{ij})	693×14	Regulation of intracellular signal transduction (over 20 highly enriched terms).
	H4	constant (few high a_{ij})	645×10	Regulation of molecular functions (incidence on catalytic activity).
	H5	order-preserving	720×24	Establishment of protein localization (protein targeting to ER and membrane).
	H6	order-preserving	733×29	Protein phosphorylation and its subterms.
DryGIN	D1	dense (high noise-tolerance)	28×17	Organelle localization (establishment of spindle and nuclear localization).
	D2	constant (with pos and neg a_{ij})	22×10	Chromatin remodeling and nucleosome organization.
	D3	constant (with pos and neg a_{ij})	21×7	Transport processes for the establishment of protein localization.
	D4	constant (with pos and neg a_{ij})	19×9	Regulation of growth (incidence on filamentous growth).
	D5	order-preserving	39×7	Organelle and nucleous organization.
	D6	order-preserving	54×6	Regulation of cellular metabolic processes (both positive and negative regulation).

Table 8.2: Description of the biological role of an illustrative set of BicNET's modules with varying properties.

Three major observations are retrieved from the conducted analyzes. First, the combination of the dense model with the provided procedures to foster robustness leads to higher enrichment factors as key genes/proteins with subtler yet functional relevance were not excluded from the modules. Nevertheless, this form of coherency is mainly associated with broader biological processes, such as general metabolic and regulatory processes (see Y_1, Y_2, H_1 and H_2 modules). Second, the constant model is indicated to guarantee a focus on less trivial modules associated with a compact set of more specific biological processes. Modules Y_3-Y_6, H_3-H_4 and D_2-D_4 are example of the relevance of considering non-dense interactions since these interactions are often related with latent or secondary (yet critical) cellular functions. Third, the order-preserving coherency is associated with modules as

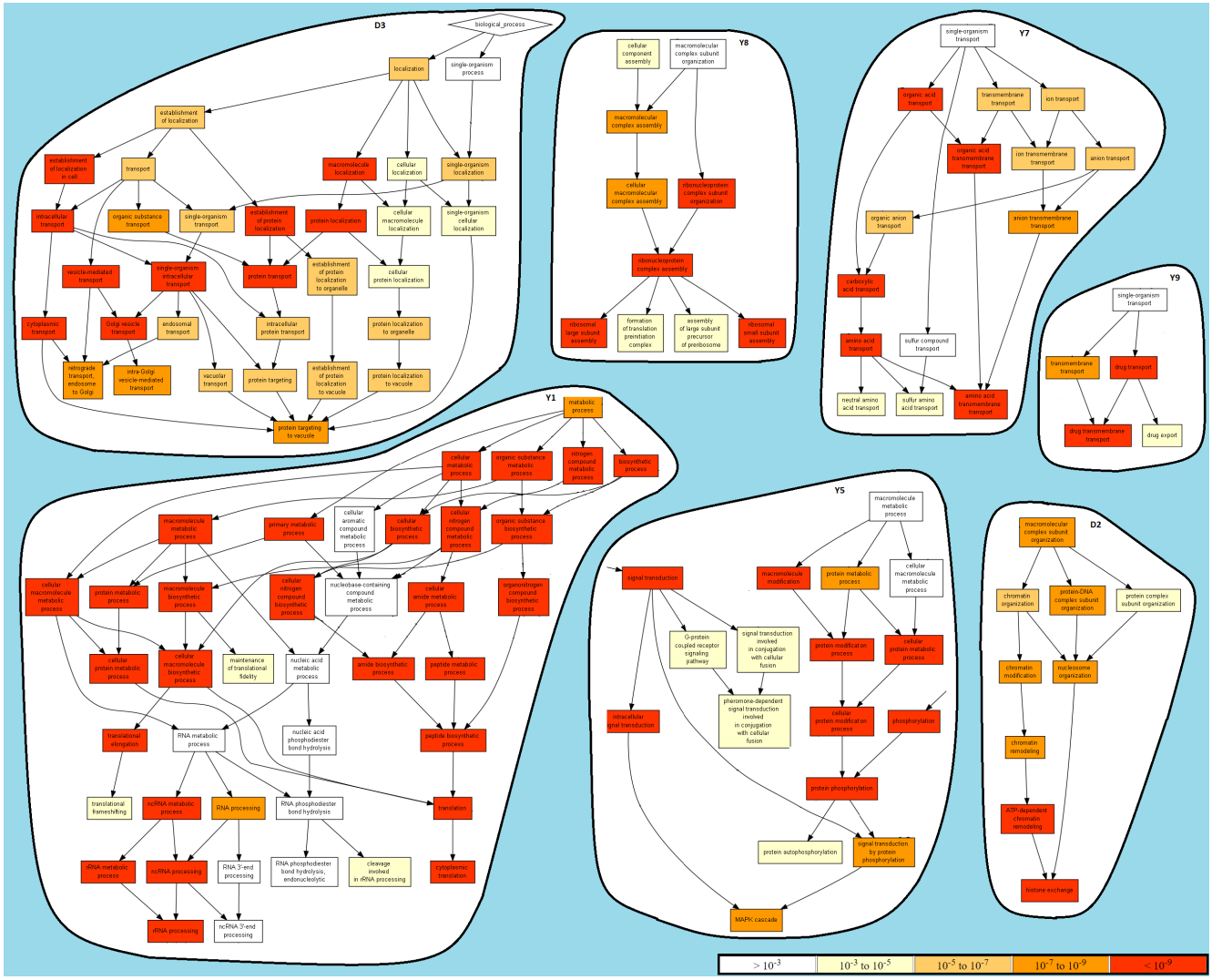

Figure 8.11: Taxonomy of enriched terms for BicNET's modules from yeast GIs (on STRING and DryGIN networks).

large as the ones provided under the noise-tolerant dense coherency, yet with the additional benefit of enabling the presence of weaker interactions as long as their coherency among the nodes is respected.

Non-trivial Modules. The provided modules in Table 8.2 already show flexible properties to surpass the limitations of the existing methods for network module discovery. Even so, BicNET can be used to further disclose less trivial modules, such as modules characterized by the presence of constant patterns with multiple symbols, symmetries and plaid effects. For this purpose, we parameterized BicNET with simple constraints (Section) to guarantee that such modules appear in the output. Table 8.3 shows an illustrative set of such modules with significantly enriched terms. All of the illustrated modules show coherent patterns of interaction between nodes and have an average amount of 5 to 10% of missing interactions. This analysis reinforces that BicNET is well positioned to find modules with varying size, coherency and quality. Illustrating, the constant modules G_6 and G_7 differ with regards to their number of nodes and quality (G_7 is larger and more more tolerant to noisy interactions). Understandably, the number of nodes per module is naturally affected by the size and sparsity of the target network. The discovered modules clearly show non-trivial yet meaningful correlations (as they include interactions with coherent yet non-differential scores), whose relevance is pinpointed by the number of highly enriched terms after correction.

Table 8.4 lists some of the enriched termsfor the modules in Table 8.3, showing their functional coherence and role to unravel putative biological processes. Interestingly, as illustrated in Table III-8.5, some of the identified modules are part of an additive plaid model (with in-between condition [303]). Illustrating, modules G_6 and S_4 share, respectively, 21% and 42% of their interactions with modules G_7 and S_2 under a plaid assumption. Some

	ID	Type	♯Nodes |I|×|J|	Items	♯Terms p<1E-15	Notes
DryGIN	G1	constant	18×9	{-4,..,-1}	27	Module with coherent strong (-4) and soft (-1) negative interactions.
	G2	symmetric	4×9	{-3,..,3}	13	Varying levels of strong (mainly positive) interactions ({±3,±2}).
	G3	symmetric	5×6	{-2,-1,1,2}	12	Module with either all positive or negative interactions per "row"-node ({±1,±2}).
	G4	constant	7×5	{1,2}	12	Module with coherent strong (2) and soft (1) positive interactions.
	G5	symmetric	7×5	{-2,-1,1,2}	11	Module with either all positive or negative interactions per "row"-node ({±1,±2}).
	G6	order	14×11	{-3,..,3}	25	Preserved precedences and co-occurrences per "row"-node before postprocessing.
	G7	order	42×8	{-2,-1,1,2}	50	Noise-tolerant module with mostly preserved orderings per "row"-node.
STRING	S1	order	155×14	{1,2,3,4}	169	Preserved precedences and co-occurrences per "row"-node before postprocessing.
	S2	constant	80×18	{1,2,3}	98	Module with mostly of non-dense interactions ({1,2}).
	S3	constant	83×10	{1,2}	93	Module with non-dense positive interactions before postprocessing ({1}).
	S4	constant	50×20	{1,2,3}	70	Module with non-dense positive interactions ({1,2}) before postprocessing.
	S5	constant	45×31	{1,2,3}	76	Module with mostly dense interactions (scores in {2,3}).
	S6	constant	55×85	{1,2}	143	Module with mostly dense interactions ({2}).

Table 8.3: Exclusivity and relevance of BicNET solutions: properties of found modules.

properties of the two illustrative sets of overlapping modules are provided in Table 8.5. Without this assumption, only smaller modules (excluding key nodes) could be obtained, resulting in a lower enrichment of their terms.

The analysis of the enriched transcription factors (TFs) for each putative biological process in Table 8.4 further supports the previous functional enrichment analyzes. For this end, we retrieved the TFs that are more *representative* (high coverage of the genes in the module) and *significant* (high functional enrichment: p-value<1E-3). Illustrating, G_1 has diverse TFs regulating different families of histones, such as Jhd1p [620]; in G_4 we found regulators of meiosis, including Sin3p [620]; the TFs of G_7 activate genes required for cytokinesis (exit from mitosis); in S_1 we found TFs associated with responses to oxygen-related stress, such as the activation of beta-oxidation genes

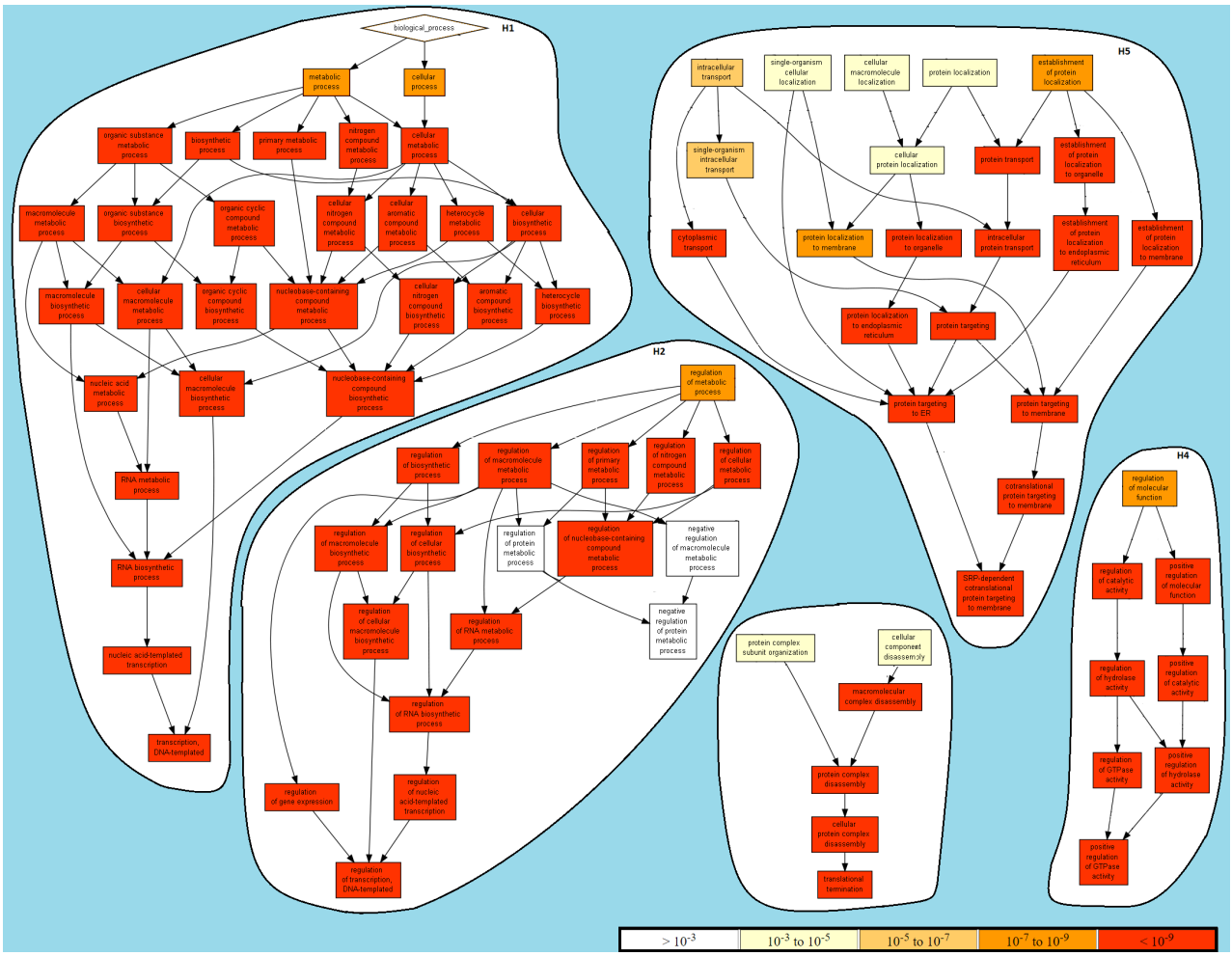

Figure 8.12: Taxonomy of enriched terms of BicNET's modules discovered from human PPIs.

	ID	Terms description (♯)	♯Terms p<1E-15	♯Nodes
DryGIN	G1	Histone modification; regulation of histone H3-K79 methylation, histone H2B ubiquitination, H2B conserved C-terminal lysine ubiquitination, H3-K4 methylation (4);	6	27
	G2	Regulation of gluconeogenesis; glutamate metabolic and catabolic processes (2); nicotinamide riboside metabolic process; nicotinamide nucleotide biosynthetic process;	6	13
	G3	Positive and negative regulation of transcription from RNA polymerase II; Invasive growth response to glucose limitation and hyperosmotic salinity response by regulating RNA polymerase II (5);	5	12
	G4	Meiotic anaphase I; activation of anaphase-promoting complex activity involved in meiotic cell cycle;	4	12
	G5	Negative reg. of phospholipid biosynthesis; lipid homeostasis; isopropylmalate and oxaloacetate transport;	4	11
	G6	Cotranslational protein targeting to membrane; protein insertion into mitochondrial membrane; protein import into peroxisome membrane; reg. sporulation; actin filament bundle assembly involved in cytokinesis;	5	25
	G7	Acetate fermentation, acetyl-CoA biosynthesis (from acetate), reg. transcription on exit from mitosis;	7	50
STRING	S1	Response to hypoxia; oxidation-dependent protein catabolic process; anaerobic respiration; age-dependent response to reactive oxygen species; cellular response to oxidative stress;	36	169
	S2	Positive & negative reg. of mitotic and nuclear cell cycle, DNA replication, budding cell apical bud growth;	16	98
	S3	Transport of aerobic electron, acetyl-CoA, vacuolar transmembrane, amine, transport (5); ribose phosphate metabolic process; D-ribose metabolic and catabolic processes (2);	22	93
	S4	Heterochromatin maintenance involved in chromatin silencing; sister chromatid segregation;	6	70
	S5	Cytoplasmic and mitochondrial translation (4); regulation of translational fidelity; ADP biosynthesis;	6	76
	S6	rRNA processing; separation, cleavage & maturation of SSU-rRNA (5); ribosomal (large subunit) biogenesis;	14	143

Table 8.4: Illustrative set of biologically significant BicNET's modules: description of the highly enriched terms in the modules presented in the previous table [620, 135].

ID	Modules with meaningful overlapping regions	Pattern	♯Nodes $\|I\|\times\|J\|$	%Overlapping interactions
G6	G7 from Table III-8.4 (orders preserved in overlapping regions before cumulative effect);	order	42×8	21%
	G8: tRNA re-export from nucleus; nuclear mRNA surveillance of mRNP export;	constant	12×10	62%
	G9: More general module (background) including cellular responses to pH;	constant	41×6	16%
S4	S2 from Table III-8.4 (satisfying the relaxed additive model proposed in [303]);	constant	80×18	42%
	S7: Telomere maintenance; translocation; protein import into nucleous;	constant	104×20	37%
	S8: Response to ionizing radiation; ribose phosphate metabolic process;	constant	59×31	45%
	S9: Positive regulation of mitochondrial translation in response to stress;	constant	50×20	89%

Table 8.5: Sets of modules with meaningful overlapping areas (satisfying the in-between plaid assumption [303]).

by Pip2p [620]; proteins regulating S_2 respond to DNA damaging, such as Plm2p and Abf1p [135]; membrane sensors, such as Ure2p, are active in the regulation of genes in S_3; S_4 has proteins promoting the organization and remodeling of chromatin, including Abf1p, Plm2p and Rsc1p [135]; regulators of ribosomal biogenesis, such as Sfp1p (100% representativity), and of its subunits, such as Cse2p [620], are core TFs for S_6.

Concluding Note. When analyzing networks derived from knowledge-based repositories and literature (such as the networks from STRING [610]), the flexibility of coherence and noise-robustness is critical to deal with uncertainty and with the regions of the network where scores may be affected due to the unbalanced focus of research studies. When analyzing networks derived from data experiments (such as the GIs from DRYGIN [372]), the discovery of modules with non-necessarily strong interactions (e.g. given by the constant model) is critical to model less-predominant (yet key) biological processes, such as the ones associated with early stages of stimulation or disease.

8.5 Summary of Contributions and Implications

This work motivates and answers the task of biclustering large-scale network data to discover modules with flexible yet meaningful coherency and robustness to noise. In particular, we explored the relevance of mining non-trivial modules in networks with homogeneous and heterogeneous nodes and with quantitative and qualitative interactions. We proposed BicNET algorithm to extend state-of-the-art contributions on pattern-based biclustering with efficient searches on networks, thus enabling the exhaustive discovery of constant, symmetric and plaid models in biological and social networks. Additional strategies are further incorporated to retrieve modules with noisy and missing interactions, thus addressing the limitations of the existing exhaustive searches on networks.

Empirical evidence confirm the limits in efficiency of existing biclustering algorithms for the discovery of non-dense modules from network data. Contrasting, BicNET enables the analysis of dense networks with up to one billion interactions. Results on synthetic and biological networks confirm its efficiency and relevance to discover non-trivial (yet coherent and significant) modules.

Future Work. Seven possible directions are identified for future work. First, we aim to enlarge the conducted biological analysis to further establish relationships between modules and biological functions. In this context, non-dense models might give new clues about the organization of genes, proteins and metabolites,and support the characterization of molecular entities with yet unclear roles. Complementarily, we expect to apply BicNET for the analysis of non-trivial communities from social networks.

Second, we expect to use the knowledge regarding missing and noisy interactions within the discovered modules to predict unknown interactions and to test the confidence of the existing interactions.

Third, the plaid model can be used to further explore the network structure and identify hubs based on the overlapping interactions between modules. Moreover, understanding the plaid effects is critical to further assess the type of connectivity between modules.

Fourth, we aim to motivate the relevance and study the experimental impact of using alternative coherency assumptions, such as given by biclustering algorithms with distinct homogeneity criteria or by pattern-based biclustering under additive and multiplicative models.

Fifth, we expect to extend the proposed contributions for predictive tasks in annotated networks based on the discovery class-conditional modules with discriminative power.

Sixth, additional principles from research on pattern-based biclustering can be used to guarantee the scalability of BicNET on large-scale network data given, for instance, by large web communities or by complete representations of molecular interactions in the human organism. For this end, we aim to enable the use of data partitioning strategies, parallelization in multi-processor or distributed settings, and the search for approximate patterns.

Finally, we expect to extend BicNET for the integrative analysis of network and tabular data. In biological domains, the analysis of GI and PPI networks has been combined with expression data analysis in order to guide the learning task and further validate results. Illustrating, Ideker et al. [341] found connected subnetworks which yield a high score measured in p-values obtained from gene expression; Hanisch et al. [297] defined distance functions, based on both expression and network information, which were subsequently integrated into clustering procedures; Segal et al. [576] provided integrative probabilistic models; and Ulitsky and Shamir [641] tested dense modules against co-expression homogeneity.

9

Flexible Pattern-based Biclustering: Putting All Together

To enable the effective use of pattern-based biclustering by the scientific community, this chapter introduces BicPAMS (Biclustering based on PAttern Mining Software) to consistently integrate the previously proposed contributions, further explore accuracy and efficiency gains, and provide an user-friendly environment to parameterize the properties of the target biclustering models.

As such, this chapter provides five major contributions:

- integration of contributions proposed in the context of BicPAM [310], BicNET [313], BicSPAM [311], BiP [303], DeBi [578] and BiModule [500]. In particular:
  - exploration of efficiency gains from synergies associated with their integrated application;
  - proposal of a consistent algorithmic basis and the analysis of its computational complexity;
- extension of BicPAMS to effectively learn from datasets with uninformative elements;
- proposal of supportive environment to run and adjust the behavior of pattern-based biclustering:
  - default and dynamically parameterized behavior based on the input data and desirable outputs;
  - declarative, graphical and programmatic interfaces.

As a result, BicPAMS is proposed to efficiently deliver exhaustive biclustering solutions with parameterizable structures, coherency and quality, yet powerful default behavior. Accordingly, this section is organized as follows. *Section 9.1* explores relevant synergies and describes the algorithmic behavior of BicPAMS. *Section 9.2* extends BicPAMS with simplistic constraints to guarantee an adequate learning from sparse data contexts when only a part of the overall data elements is relevant. *Section 9.3* discusses its default parameterizations. *Section 9.4* describes the different interfaces made available by BicPAMS for the customization of the target descriptive models. Finally, we draw implications from initial empirical evidence.

9.1 Integrative View of Contributions

The inherent simplicity, efficiency, flexibility and robustness of pattern-based biclustering explains why they are receiving an increasing attention in recent years [310, 311, 500, 578, 442]. In particular, some of the major contributions of pattern-based biclustering proposed throughout *Book III* include: *1)* efficient analysis of large matrices based on monotone searches, their extension for data contexts with large informative regions and item-indexable properties, and the support for partitioned data settings and approximate patterns [290]; *2)* biclusters with parameterizable coherency strength and assumption; *3)* flexible structures of biclusters (arbitrary size and positioning of biclusters) and searches (no need to fix the number of biclusters apriori) [500, 578]; *4)* robustness to noise, missings and discretization problems [310]; and *5)* applicability for different data structures, including tabular data with non-identically distributed features and sparse data. The relevance of these contributions was largely motivated and empirically demonstrated [310, 303, 311, 578]. BicPAM [310], BicSPAM [311], BiP [303] and BicNET [313] go beyond precedent efforts towards pattern-based biclustering [578, 500], originally prepared to model constant coherencies in real-valued matrices, in order to efficiently accommodate additive, multiplicative,

Coherence and *method*	*Behavior*	*Tackled Limitations*
Constant BicPAM [310]	BicPAM integrates previous contributions from pattern-based biclustering according to three steps. The core step is the *mining* step, corresponding to the application of a target pattern miner. This step is driven by the chosen paradigm, pattern representations and search properties. *Preprocessing* is responsible for the itemization of a real-valued data, to fix the coherence strength and to handle outliers and missings. *Postprocessing* includes the merging, filtering and extension options to affect the quality and structure of the target biclustering solutions.	Exhaustive (yet scalable) searches; Flexible structures; Native way of handling noise; Parameterizable coherence strength; Extensible for multi-class settings; Easy incorporation of constraints.
Additive and Multiplicative BicPAM [310]	BicPAM makes use of the observed differences (additive) and of the least common divisor (multiplicative case) between the maximum and observed values to identify and remove shifting and scaling factors (iterative corrections). Pruning strategies are provided to avoid redundant calculus.	Flexible structure and parameterizable quality; Precise modeling of shifts and scales across rows.
Order Preserving BicSPAM [311]	Biclustering is parameterized with sequential pattern mining, for an exhaustive and flexible discovery of biclusters respecting noise-tolerant orderings. For this aim, column indexes are reordered per row according to their values, and biclusters are mapped from the frequent subsequences. BicSPAM also allows a parameterizable variation of the degree of co-occurrences (elements with similar values) versus precedences to affect coherency.	Surpasses efficiency and robustness issues of exhaustive peers; Flexible structures with guarantees of optimality, addressing the problems of greedy peers.
Symmetric Bic(S)PAM [310, 311]	BicPAM and BicSPAM rely on combinatorial sign-adjustments across rows to model symmetries; integrate them with scales, shifts and orderings; and make use of pruning principles to guarantee the efficiency of the search.	Discovery of non-constant biclusters with symmetries; Parameterizable properties.
Plaid BiP [303]	BiP extends existing searches to recover excluded areas (due to cumulative contributions on overlapping areas between biclusters) and to remove noisy areas that are not described by a plaid assumption. For learning complex plaid models (non-trivial overlapping aspects), BiP relies on converging behavior by incrementally removing contributions. Finally, without degrading efficiency, it provides new composition functions (weighted additive and multiplicative functions) and relaxations to deal with noise and non-linear cumulative effects.	Addresses the exact additive plaid assumption by using relaxations and weighted compositions; No need for all the elements in the dataset to follow a plaid assumption; Models non-constant coherencies.

Table 9.1: Recent breakthroughs on pattern-based biclustering: algorithms and tackled limitations.

symmetric and plaid coherency assumptions robust to noise in tabular and network data. Table 9.1 synthesizes the properties of these recent algorithms and the tackled problems from related research.

Below, we structure these contributions according to the coherency, structure and quality of pattern-based solutions and the efficiency of the underlying searches. Following this analysis, major synergies resulting from the integration of these contributions are explored (*Section 9.1.1*), and the algorithmic basis and computational complexity of BicPAMS analyzed (*Sections 9.1.2* and *9.1.3*).

Packing Contributions

Coherency. As highlighted in Table 9.1, BicPAMS enables exhaustive searches of biclusters following a parameterizable *coherency assumption* (constant, additive, multiplicative, order-preserving and plaid). BicPAMS also offers the possibility to either select coherency on rows or on columns, and accommodates principles to discover biclusters with mixtures of coherency assumptions from tabular data with categoric and non-identically distributed features. Finally, BicPAMS provides the possibility to select a desirable *coherency strength* of biclusters and to include or neglect symmetric effects in order to effectively deal with both discrete and real-valued data with either negative or strictly positive ranges of values.

Structure. BicPAMS revises pattern mining searches to guarantee that biclustering can be performed in the presence of a meaningful stopping criteria, such as the minimum number of (dissimilar) biclusters or minimum percentage of the elements in the original dataset covered by the found biclusters. The minimum number of rows or columns of biclusters can be optionally inputted to guide the search. Different pattern representations can be used to affect the structure. Also, BicPAMS makes available post-processing procedures with parameterizable criteria to merge, reduce and extend biclusters. These procedures can be used to further customize the positioning of biclusters. Finally, BicPAMS gives the possibility to model plaid structures by assuming a parameterizable composition of contributions from biclusters on areas/interactions where they overlap.

Quality. BicPAMS solutions make use of post-processing procedures and, among other strategies, the possibility to assign multiple items to a single element in the matrix to guarantee robust solutions. Different relaxation levels are supported, given the user the possibility to to calibrate the tolerance to noise by, for instance, assigning a

parameterizable number of symbols to an element based on the observed value. Similarly, BicPAMS guarantees robustness to missings by offering the possibility to either remove them (yet allowing biclusters to have a bounded amount of missing values) or by imputing a parameterizable number of items based on their value expectations.

Efficiency. BicPAMS also relies on enhanced pattern mining searches (including the introduced F2G and IndexSpan searches) to explore efficiency opportunities associated with: specificities of the biclustering task, inputted constraints and desirable outputs. BicPAMS is able to rely on frequent itemsets (using Apriori-based, vertical and dedicated pattern-growth searches reliant on efficiently annotated tree structures), association rules, sequential patterns (using dedicated searches able to seize efficiency gains from item-indexable properties), sequential rules, and actionable patterns. BicPAMS also implements searches able to deliver a fixed number of patterns (top-K patterns) if the user wants to focus on specific number of dissimilar regions. All searches are integrated with heuristics that guarantee an effective pruning of the search space in the presence of constraints such as minimum number of columns. Additionally, principles are incorporated to seize efficiency gains from the condensed pattern representations, including principles from Charm and Bide+.

BicPAMS makes available adequate data structures and operations to guarantee heightened time-and-memory effeciency in sparse data contexts (e.g. network data analysis) and dense data contexts (e.g. low coherency strength with multi-item assignments). It also guarantees the efficiency of postprocessing procedures based on similarity-calculus during the mining step and constraint-guided extensions and reductions. Finally, BicPAMS is easily extensible to seize efficiency gains from mining approximate patterns, and from its application over distributed and partitioned data settings.

9.1.1 Synergies

Three major synergies can be explored when the previously summarized contributions are integrated. *First*, the diverse options provided in the context of BicPAM (*Chapters III-1-III-3* and *III-5*), BiP (*Chapter III-4*), BicSPAM (*Chapters III-6* and *III-7*), and BicNET (*Chapter III-8*) can be transversally used to compose pattern-based biclustering algorithms using the structured view provided in *Chapter III-1* (Figure III-1.5). In particular, the selection of specific options can be used to mimic the behavior of previous pattern-based biclustering algorithms, such as BiModule [500] and DeBi [578], as well as to compose new algorithms. This possibility turns BicPAMS an attractive means for the systematic study and assessment of pattern-based biclustering.

Second, BicPAMS guarantees the efficient discovery of biclustering models with multiple coherency assumptions at a time. This option is relevant as it enables the discovery of a complete composition of regions with varying regularities. In order to promote the efficiency of the searches in this context, two strategies can be considered. These strategies are based on the principle that biclusters with stricter coherency assumptions are contained in biclusters with more flexible assumptions. As such, a first strategy is to first discover biclusters with stricter coherency assumptions are then use them to guide the discovery of other biclusters. Illustrating, constant biclusters can be first discovered, and their columns and rows be used under succinct constraints to guide the discovery of larger biclusters given, for instance, by additive, symmetric or order-preserving assumptions.

A second strategy, and default option in BicPAMS, is to first discover biclusters with more flexible coherency assumption. Since biclusters with stricter coherency are contained in these biclusters, all the elements in the input matrix not covered by flexible biclusters can be removed and then the search for biclusters with stricter coherency applied on the resulting sparse dataset. Illustrating, order-preserving biclusters can be first discovered and all the remaining elements removed for the discovery of additive or constant biclusters. Since biclustering solutions commonly cover less than 20% of overall elements in the input data matrix, the subsequent searches for biclusters with stricter coherency have a residual complexity. As a result, the computational complexity is uniquely bounded by the search for biclusters with the most flexible coherency assumption.

Third, when multiple coherency assumptions are considered: mining and closing steps are applied one time only.

Closing procedures are also jointly applied when inputting multiple coherency strength thresholds. Depending on the chosen strategies to guarantee robustness, either mapping or closing steps can be jointly applied when considering varying levels of tolerance to noise.

9.1.2 Algorithmic Solution

The algorithmic basis of BicPAMS is described in Algorithm 7. Although BicPAMS follows a plug-and-play style, default and data-driven parameterizations are made available. In particular, *lines 45-49* and *42-43* describe BicPAM behavior in the absence of user-driven parameterizations.

9.1.3 Computational Complexity Analysis

The computational complexity of BicPAMS is bounded by the pattern mining task and computation of similarities among biclusters. For this analysis, we discuss the major computational bottlenecks associated with each one of the three major steps of BicPAMS.

Within the *mapping* step: outlier detection, normalization, discretization, and noise correction procedures (such as the assignment of multiple items) are linear on the size of the matrix, $\Theta(nm)$. The optional distribution fitting tests and parameter estimations to dynamically select an adequate discretization procedure are also $\Theta(nm)$. These tests and estimations rely on the calculation of approximated statistical ratios[1]. Handling missings by removing the respective element or by replacing them by a special dedicated item is also $\Theta(nm)$. However, when an imputation method is selected, the complexity is upper bounded by $\Theta(hnm)$, where h is the number of missing values. In BicPAMS, the nearest neighbor rows and columns are computed for the estimation of each missing value.

The cost of the *mining step* depends on two factors: the complexity of the pattern miner and the need for iterations for the discovery of non-constant coherency assumptions. The cost of the pattern mining task depends essentially on: the number and size of transactions (γnm, where $\gamma \geq 1$ captures the increase in size related with noise and missings handlers), selected mining procedures (FIM, SPM or association/sequential rules), the frequency distribution of items for non-order-preserving assumption ($\{\mathcal{L} \times Y\} \rightarrow \mathbb{N}$), the minimum support θ, the pattern representation and the type of search. Empirical evidence show that the complexity of the mining step when iteratively applied with a decreasing support threshold is bounded by the search with lowest support. A detailed analysis of the complexity of the pattern mining task been attempted in literature [548] and it is out of the scope of this paper. The reader should also keep in mind that there has been proposals to guarantee the scalability of pattern miners recurring to partitioning and approximate methods [290]. Let $\Theta(\wp(\gamma, n, m, |\mathcal{L}|, \theta))$, or simply $\Theta(\wp)$, be the complexity of a pattern mining task. When there is the need for the iterative application of the core mining procedure for the discovery of symmetric, additive and multiplicative, the overall search is bounded by $\Theta(d \times \wp)$, where $d = min(\binom{n}{2}, m)$ when allowing symmetries, $d = min(\binom{n}{|\mathcal{L}|}, m)$ when allowing shifts, and $d = min(\binom{n}{\sharp lcm}, m)$ when allowing scaling factors.

The cost of the *closing* step depends essentially on two factors: the complexity of computing similarities among biclusters (required for merging and filtering biclusters) and the complexity of extending and reducing biclusters. To compute similarities, a directed graph is dynamically created. Only biclusters sharing interactions over a threshold based on the input overlapping degree are candidates for merging and filtering. Although these similarities are computed under cut-circuiting heuristics, their computational complexity is significantly larger than the posterior the application of multi-support FIM or maximal circuit discovery to merger sets of similar biclusters. As such, the average complexity of these tasks is bounded by $\Theta(\binom{k}{k/2}\bar{r}\bar{s})$, where k is the number of biclusters and $\bar{r}\bar{s}$ their average size. Extending biclusters relies on quick tests based on the coherency of candidate columns or rows and therefore the complexity of this task is respectively $\Theta(k'\bar{r}m)$ or $\Theta(k'n\bar{s})$, where k' is the number of biclusters after merging and filtering. Removing rows or columns from biclusters is $\Theta(k'\bar{r}\bar{s})$.

[1] http://cran.r-project.org/doc/contrib/Ricci-distributions-en.pdf (accessed 13 Oct 2014)

Algorithm 7: BicPAMS Core Steps (simplified code presentation)

Input: matrix, coherency, orientation, nrItems, patternMiner, patternRep, normalizer, discretizer, missingsHandler, noiseHandler, extender, merger, reductor, filter, stopCriteria */*default when absent*/*

```
main begin
    itemizedData ← runMappingStep(matrix, nrItems, normalizer, discretizer, missingsHandler, noiseHandler, orientation);
    biclusters ← runMiningStep(coherency, patternMiner, itemizedData, patternRep, stopCriteria, nrItems, orientation);
    return runClosingStep(biclusters, extender, merger, filter);

runMappingStep begin
    mask ← getMissingsAndOutliersMask(matrix);
    /*to preprocess sparse and noisy data*/ discData ← discretize(matrix, nrItems, normalizer, discretizer, mask);
    if isColumn(orientation) then discData ← transpose(discData);
    treatedData ← appHandlers(discData, missingsHandler, noiseHandler); /*inc. multi-item assignments and imputations*/
    // mapping data into transactional or sequential databases (observations with possibly varying size):
    if isOrderPreserving(coherency) then return createSequences(treatedData);
    else return createTransactions(treatedData);

runMiningStep begin
    if isConstant(coherency) || isOrderPreserving(coherency) then freqItemsets ← runPM(patternMiner, itemizedData, patternRep, stopCriteria);
      if isPlaid(coherency) then freqItemsets ← runPlaid(patternMiner, itemizedData, stopCriteria); /*according to Alg.III-3*/;
    else freqItemsets ← runAddMultSymmPM(coherency, patternMiner, itemizedData, patternRep, stopCriteria);
    // recover columns and rows from frequent itemsets (according to Def.III-1.7 or Def.III-6.4)
    return getBiclusters(freqItemsets, nrItems, orientation);

runClosingStep begin
    biclusters ← merge(biclusters, merger);
    biclusters ← filterBiclusters(biclusters, filter);
    biclusters ← extend(biclusters, extender);
    return increaseConsistency(biclusters, filter);

runAddMultSymmPM begin
    factors ← ∅;
    freqItemsets ← ∅;
    foreach column j in itemizedData do
        // variations for additive (differences), multiplicative (least common multiples) and symmetric models
        colAdjusts ← computeFactor(itemizedData[·][j],coherency);
        if colAdjusts ∈ factors then continue;
        else factors ← factors ∪ colAdjusts;
        alignedData ← alignDataByRows(colAdjusts,itemizedData);
        freqItemsets ← freqItemsets ∪ runPM(patternMiner, alignedData, patternRep, stopCriteria);
        if allCombinations(factors) then break; /*simple combinatorial calculus to prune the search*/;
    return freqItemsets;

runPM begin
    if estimateLimits(stopCriteria) then
        // simple statistical calculus based on the frequency of items
        (minRows,minColumns) ← findLowerLimitsExpectations(data);
        freqItemsets ← PM(patternMiner, minRows, minColumns, data, patternRep); /* F2G or IndexSpan by default */
    else
        minSupport ← 0.5;
        freqItemsets ← ∅;
        while minAreaPercentageAchieved(freqItemsets,stopCriteria) || minNrBiclusters(freqItemsets,stopCriteria) do
            freqItemsets ← PM(patternMiner, minSupport, data, patternRep);
            minSupport ← minSupport×0.8;
    return freqItemsets;
```

In this context, the complexity of BicPAMS is bounded by $\Theta(hnm + d\wp + \binom{k}{k/2}\bar{r}\bar{s} + k'(\bar{r}m + n\bar{s}))$, which for settings resulting in a large number of biclusters after the mining step ($k \gg k'$) is approximately $\Theta(d\wp + \binom{k}{k/2}\bar{r}\bar{s})$.

9.2 Learning from Sparse Data: Discarding Uninformative Elements

Input tabular data can be accompanied with a-priori knowledge regarding uninformative regions. This is the case of tabular datasets mapped from network data, where missing interactions correspond to uninformative elements that should be removed for the prompt application of pattern-based searches. It is also the case of tabular datasets where only a (possibly small) subset of overall features per observation are filled, as often observed in medical data domains. Furthermore, specific tabular data contexts are associated with non-interesting elements, such as elements with non-differential/baseline expression in gene expression data. In these settings, the ranges of uninformative values (or items) should be removed from the input matrix. This option can be easily specified through a succinct constraint: $remove(S)$ where $S \subseteq \mathbb{R}^+$ (or $S \subseteq \mathcal{L}$).

Understandably, as pattern mining searches rely on transactions with arbitrary length, this solution is soundly supported. In fact, since BicPAMS integrates BicNET's principles, it is able to effectively explore time-and-memory efficiency gains when specific elements from the input matrix are removed. Due to the properties associated with learning from sparse data, empirical evidence shows that the removal of a given α% of the elements is commonly associated with an exponential magnitude of efficiency gains ($\gg \alpha$% of seconds or bytes). Despite the structural simplicity of this behavior, this possibility is not supported by state-of-the-art biclustering algorithms [314].

In the presence of very large datasets, the ranges of less-interesting values to remove can be parametrically increased in order to guarantee narrower searches and thus guarantee the scalability of the biclustering task. Illustrating, more relaxed thresholds can be defined to identify: 1) undifferentiated elements in expression data, 2) entries with low-counts in tabular datasets extracted from text, 3) inconclusive ratings in collaborative filtering data, 3) unprofitable decisions from trading data, 4) healthy evaluations from medical data, among others.

Three major factors impact the sparsity of the preprocessed data for the mining step: 1) structural sparsity either associated with non-imputed missing values (tabular data) or disconnected nodes (network data), 2) increased sparsity when uninformative regions are removed (according to background knowledge or user expectations), and 3) decreased sparsity (increased density) due to application of multi-item assignments on the informative elements and imputation strategies. These two later options can be adequately controlled to guarantee an adequate balance between robustness and efficiency.

9.3 Default and Dynamic Parameterizations of Pattern-based Biclustering Algorithms

BicPAMS makes available parameterizations proposed in the context of previous pattern-based biclustering approaches (such as the number of items proposed in BiModule [500], α-significance levels for closing procedures from DeBi [578], confidence thresholds from GenMiner [442], or coherency strength from RAP [509]), and further extends them to allow the ability to customize the coherency, structure and quality of solutions. In this context, there is the need to guarantee that BicPAMS provides a robust and friendly environment to be used by users without expertise in (pattern-based) biclustering.

For this aim, BicPAMS makes available: 1) default parameterizations (data-independent setting) and 2) dynamic parameterizations based on the properties of the input dataset (data-dependent setting). Default parameterizations include: 1) zero-mean row-oriented normalization; 2) overall Gaussian discretization $m/4$ items for order-preserving coherencies (for an adequate trade-off of precedences vs. co-occurrences) and a set of $\{3,5,7\}$ items for the remaining coherencies; 3) integrated discovery of multiple coherencies (*Section 9.1.1*); 4) F2G search for closed FIM and association rule mining, and IndexSpan search for SPM; 5) multi-items assignment (allocation of 2 items for values in range $c \in [a,b]$ when $\frac{min(b-c,c-a)}{b-a} < 25\%$); 6) imputation of two $\lfloor|\mathcal{L}|/2\rfloor$ items for missing values; 7) merging procedure (multi-support FIM with similarities pushed into the mining step) with 80% overlapping; 8) filtering procedure for biclusters with 80% Jaccard-based similarity against a larger bicluster; 9) no extension or reduction procedures. For the default setting, BicSPAM iteratively decreases the support threshold 10% (starting with θ=50%) until the output solution discovers 50 dissimilar biclusters or a minimum coverage of 10% of the elements in the input matrix.

The dynamic parameterizations differ with regards to the following aspects. First, the fit of different distributions are tested to select adequate normalization and discretization procedures. Second, the data size and dimensionality is used to parameterize the mining step (according to *Sections III-1.4.1* and *III-6.3.2*). Third, moderate and relaxed missing handlers are selected if the input dataset has, respectively, over 2% and 5% of missing elements. Fourth, in the presence of data with very high size and/or dimensionality, data partitioning procedures are used. Assuming coherency on columns, empirical evidence shows that scalability principles are required when the condition $(m < 20.000 \wedge n < \lambda \times 1000) \vee (m < 4.000 \wedge n < \lambda \times 2000)$ is not satisfied for the: constant assumption when λ=1, order-preserving assumption when λ=0.1, and remaining coherency assumptions when λ=0.5.

9.4 Declarative, Graphical and Programmatic Interfaces

Declarative Interface: Parameterizations as Native Constraints. BicPAMS introduces the option of inputting parameterizations in the form of simplistic constraints, referred as native constraints, where parameters are assigned to values. When parameters are not assigned to values, the default values discussed in previous section are assumed. Below we enumerate some of the most relevant native constraints that can be incorporated to affect the behavior of BicPAMS along its major steps: 1) mapping step, including coherency strength δ or number of items $|\mathcal{L}|$ and noise tolerance (determining the minimum distances considered for multi-item assignments and imputations); 2) mining step, including the coherency assumption and orientation, stopping criteria, expectations (such as minimum number of rows or columns), pattern representation, and algorithmic choice; and 3) closing step, including the maximum percentage of noisy and/or missing elements per bicluster (based on merging procedures [310]); and minimum homogeneity thresholds for the target biclusters (using extension and reduction procedures with a parameterizable merit function [310]).

Graphical Interface. BicPAMS is provided within online and desktop graphical interfaces to soundly parameterize pattern-based biclustering algorithms and to visualize their outputs. The soundness of pattern-based biclustering is guaranteed by either disabling options when certain parameters are selected (e.g. the order-preserving assumption is removed when closed frequent itemset mining searches are selected) or by displaying messages of error (e.g. an error message is displayed when additive models with symmetries are used with an even number of items since the absence of an item with '0'-value correspondence can invalidate a correct analysis of shifts). Upon running BicPAMS, when stopping criteria is met, a message of success is displayed, enabling a view with the graphical representation of each bicluster (see Figures 9.2 and 9.1). Additionally, BicPAMS supports the loading of input data and the exportation of results according to a wide-variety of formats.

- **Online Interface**. A web application is provided to run BicPAMS without installation requirements through a set of asynchronous webservices, concurrently computed within a remote machine (Intel CPU with 2.4GHz and a shared 16GB RAM). Soundness is further guaranteed by providing dynamic form checks. The privacy and security of the requests is assured. Finally, the online interface further supports the possibility to display results in a refreshing page or via e-mail (useful when running time consuming requests);
- **Desktop Interface**. In addition to the online interface, BicPAMS is also provided as a desktop applet to avoid waiting times associated with: the uploading of large datasets or the processing of online requests due to the presence of a high number of concurrent requests. The implemented behavior, provided parameters and soundness checks are essentially identical to the online interface, with slight differences on the way files are uploaded and results are displayed. Figure 9.2 provides an illustrative snapshot.

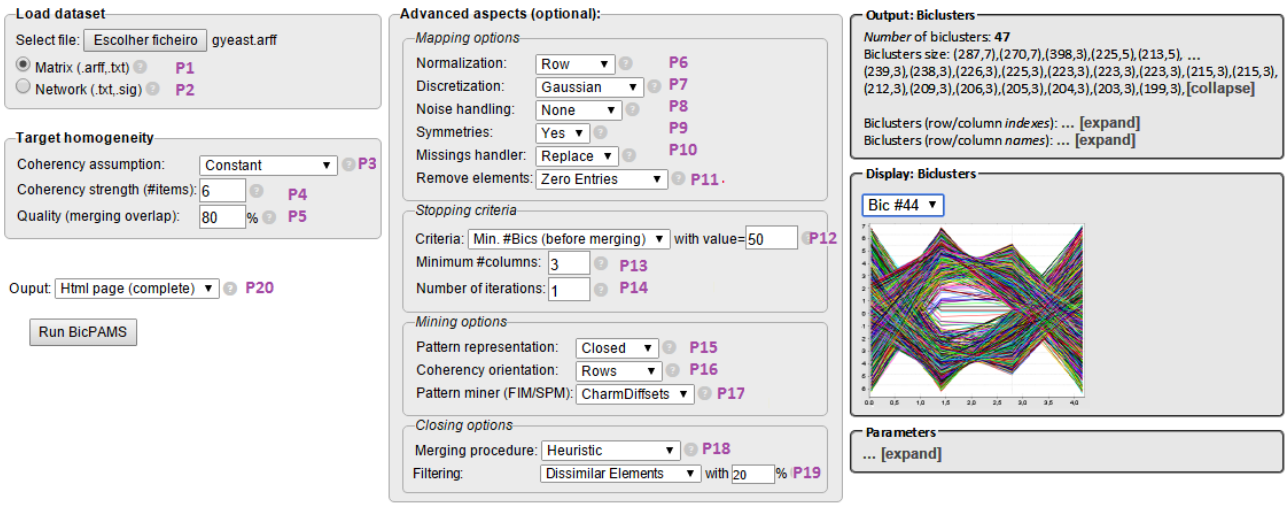

Figure 9.1: BicPAMS online interface: sound parameterization of pattern-based biclustering and visualization of outputs.

As an in-depth description of the graphical interfaces of BicPAMS is out of the scope of this dissertation, we synthesized their major options in Table 9.2 (numeration according Figure 9.1).

Programmatic Interface. Alternatively to the previous interfaces, BicPAMS makes available a programmatic interface based on a Java API with the respective source code and accompanying documentation. This interface is essential to: extend pattern-based biclustering algorithms for alternative tasks, such as classification and indexation, and easily adapt its behavior in the presence of data with very specific regularities. Detailed scenarios where the use of additional parameters is further explored are provided in the webpage of the author.

Input and Output	[P1] Matrix	The accepted file formats include attribute-relation (.ARFF) and standard matrices (such as .TXT). The first line of standard matrices should specify the column identifiers, while the first entry of each line should specify the row identifier. Tabular data can be either delimited by tabs, spaces or commas.
	[P2] Network	BicPAMS accepts any input file format (such as .TXT or .SIG) assuming that: the first line specifies the column identifiers, and each other line specifies an interaction/entry within the network. An entry specifies the nodes and the association strength. Entries can be either delimited by tabs, spaces or commas. In addition to the file, the column index identifying the first node, second node and association strength needs to be inputted. Illustrating, for a network with header "idProteinA,nameProteinA,idProteinB,nameProteinB,weight", the user should fix (node1,node2,score) indexes as (0,2,4) or (1,3,4). Finally, the user can specify whether each entry is directional from the first node towards the second node or bidirectional. Bidirectional entries increase the density of the network.
	Output	Upon successfully running BicPAMS, a textual and graphical display of the outputs is provided. The user can select whether the output is displayed in the same HTML page (complete solution) or sent through e-mail (containing the textual display only up to a maximum size of 10Mb).
Biclustering Models	[P3] Coherency Assumption	The coherency assumption defines the correlation of values within a bicluster. In constant models, an observed pattern (possibly containing different items) is preserved across rows (or columns). In additive or multiplicative models, shifting or scaling factors are allowed per row (or column) in order to allow meaningful variations of the original pattern. In order-preserving models, the values per row induce the same ordering across columns. A plaid model considers the cumulative effect of the contributions from multiple biclusters on areas where their rows and columns overlap. Previous models can further allow symmetric factors.
	[P4] Coherency Strength	The number of items determines the allowed deviations from expected values. Illustrating, a gene expression matrix parameterized with 5 items will have 2 levels of activation ({1,2}), 2 levels of repression ({-1,-2}) and 1 level of unchanged expression ({0}). By going beyond the differential values, BicPAMS enables the discovery of non-trivial yet coherent and meaningful correlations. To maintain consistency, additive (multiplicative) models should be used with an uneven (even) number of items. When considering order-preserving models, the number of items should be increased to balance the degree of co-occurrences versus precedences.
	[P5] Quality	This field specifies the maximum number of allowed noisy/missing elements (determining the minimum overlapping threshold for merging procedures). The tolerance of biclusters to noise can be additionally addressed using noise handlers (see mapping options) and alternative postprocessing procedures.
	[P15] Pattern Representation	Closed patterns (default option) enable the discovery of maximal biclusters (biclusters that cannot be extended without the need of removing rows and columns). The use of maximal patterns leads to biclusters with the columns' size maximized. This gives a preference towards flattened biclusters, possibly neglecting both vertical and smaller biclusters. Finally, the use of simple/all frequent patterns leads to biclustering solutions with a high number of biclusters (possibly contained by another bicluster), which can be useful to guide postprocessing steps. As the user specifies one of these three options, the available pattern miners are dynamically updated.
	[P16] Orientation	Coherency can be either observed across rows (default) or columns (searches are applied on the transposed matrix). When the number of columns highly exceeds the number of rows (or vice-versa when searches are applied on the transposed matrix), pattern miners with vertical data formats such as Eclat should be preferred.
Mapping Options	[P6] Normalization	Depending on the properties of the input data, the user can either normalize data per Row, Column or for the Overall data elements or ignore normalization by selecting the None option. Both outliers and missing values are handled separately.
	[P7] Discretization	Real-valued data needs to be discretized to apply pattern-based biclustering (see noise handling to understand how BicPAMs guarantees robustness to discretization drawbacks). The user can select the cut-off points of a Gaussian distribution (default) or fixed ranges of values (equal sized intervals after excluding outliers). Note that fixed ranges can lead to an imbalanced distribution of items. The user can bypass this option for symbolic data by selecting the None option.
	[P8] Noise Handler	Multi-item assignments can be considered to handle deviations on the expected values within a bicluster caused by noise or discretization issues. By selecting this option, 2 items are assigned to elements with a value near a boundary of discretization (value in range $c \in [a, b]$ when $\min(b\text{-}c, c\text{-}a)/(b\text{-}a)<25\%$). In this context, the data elements become associated with a varying number of items, thus increasing the size of data for analysis.
	[P9] Symmetries	This option should be selected if the input data is composed by positive and negative values (as it naturally affects the properties of the outputted biclusters). When using symmetric ranges, additive (multiplicative) models should be parameterized with an odd (even) number of items to guarantee consistent shifts (scales).
	[P10] Missings Handling	The user can specify what happens in the presence of missing values. Since BicPAMS is natively prepared to analyze sparse data, the Remove option (default) simply signals the algorithms to exclude missings from the searches. Alternatively, the Replace option uses WEKA's imputation methods to fill missings (the error of imputations can be minimized by simultaneously recurring to noise handlers). We suggest the use of Remove option for network data and other meaningfully sparse datasets since BicPAMS is able to discover biclusters with missing interactions (see Quality parameter).
	[P11] Remove Uninformative Elements	This option supports the possibility to remove uninformative elements within the inputted matrix (or uninformative interactions within the inputted network). Zero Entries can be selected to remove the {0}-items, while the Differential option is used to focus on items with high absolute value (e.g. {-3,-2,2,3} when $\|\mathcal{L}\|$=6). Uninformative elements may correspond to: 1) weak interactions in networks, 2) unchanged expression, 3) healthy evaluations from clinical data, among others.
Mining Options	[P12] Stopping Criteria	The search algorithm runs until any of the available stopping criteria is met. The available options are: 1) minimum number of biclusters before merging (default), 2) minimum covered area by the discovered biclusters (as a percentage of the elements of the input data matrix or network), and 3) minimum support threshold (minimum number of rows per bicluster specified as a fraction of overall rows). The value associated with the selected option should be additionally specified. We suggest the definition of a high number of biclusters (>50) as the default option, in order to guarantee an adequate exploration of the input dataset.
	[P13] Minimum ♯Columns	The minimum number of columns per bicluster can be optionally inputted to promote efficiency and align the outputs according to user expectations. A good principle to fix this value is to use the square root of the number of columns (interactions per nodes) of the input matrix (network).
	[P14] ♯Iterations	BicPAMS default behavior relies on a single iteration. For data with large coherent regions that may prevent the discovery of smaller (yet relevant) regions, the number of iterations can be increased to guarantee their discovery. On every new iteration, 25% of the most selected data elements (from the biclusters discovered from the previous iteration) are removed to guarantee a focus on new regions. 3 iterations already guarantee an adequate space exploration for hard data settings.
	[P17] Pattern Miner	The available pattern mining algorithms are dynamically provided based on the selected coherency assumption and pattern representation. Sequential pattern miners (SPM) are provided for order-preserving models: PrefixSpan and IndexSpan (an optimized algorithm able to explore efficiency gains from item-indexable properties) are made available for simple pattern representations, while BIDE+ is provided for closed pattern representations. Frequent itemset miners (FIM) are selected for the remaining coherency assumptions. AprioriTID, F2G (pattern-growth method for data with a large extent of coherent areas) and Eclat (vertical method for data with a high number of columns) are made available for simple pattern representations. CharmDiffsets, AprioriTID and CharmTID are made available for closed pattern representations, while CharmMFI with diffsets is provided for maximal pattern representations.
	Scalability	In addition, the user can specify whether data partitioning principles are applied or not to guarantee the scalability of the searches (suggested for data with >100Mb).
Closing	[P18] Merging	Different merging procedures are made available (according to [308]): heuristic (default option) for an efficient and effective yet non-exhaustive merging; and combinatorial and multi-support FIM alternatives for an exhaustive yet more costly postprocessing step.
	[P19] Filtering	Filtering is essential to guarantee compact solutions (applied after merging). A biclustered is filtered if it has not enough Dissimilar Elements, Dissimilar Rows or Dissimilar Columns against a larger bicluster. Considering a filtering option with 20% of dissimilar elements. In this context, biclusters sharing more than 80% of their elements against a larger bicluster are removed.

Table 9.2: Parameters of BicPAMS (according to Figure 9.1): taking the most of unsupervised analysis of tabular data.

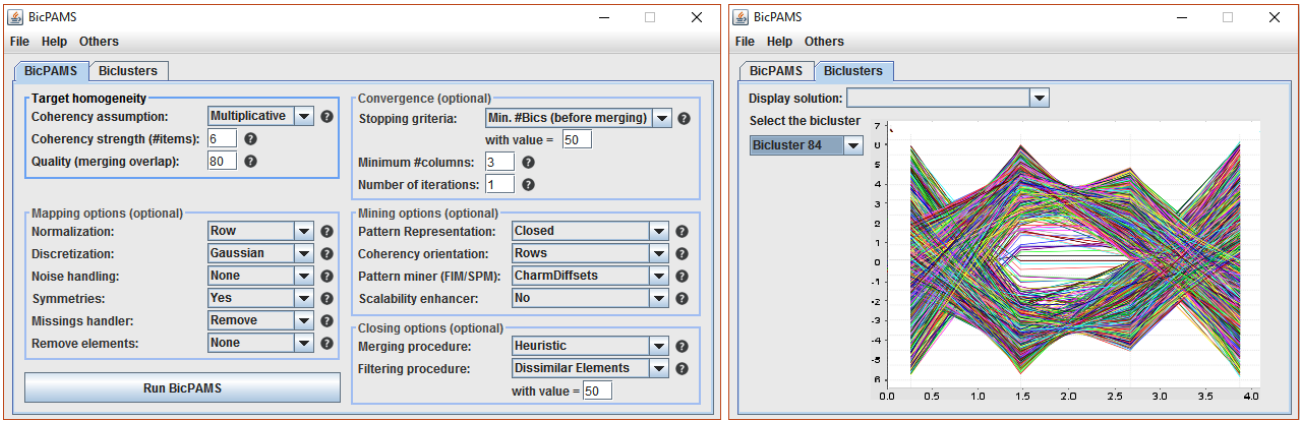

Figure 9.2: BicPAMS desktop interface to soundly run pattern-based algorithms and visualize outputs.

9.5 Summary of Contributions and Implications

BicPAMS integrates and extends the state-of-the-art contributions on pattern-based biclustering, tackling the commonly placed restrictions on the allowed structure, coherency and quality of biclusters. As such, new possibilities become available, as well as opportunities to further explore efficiency gains when distinct pattern-based searches are applied in an integrated fashion. This is the case when multiple coherency assumptions are targeted since their degree of flexibility can be used to narrow the search space.

BicPAMS extends the principles for learning from sparse data, originally proposed in the context of BicNET, to guarantee their applicability over tabular data with uninformative elements.

In order to guarantee the usability and robustness of the parametrically rich BicPAMS, both default parameterizations and parameterizations dynamically driven from the properties of input data are proposed.

Complementarily, BicPAMS makes available declarative, graphical and programmatic interfaces to: 1) easily control the coherency (including assumption, orientation and strength), quality (including tolerance to noisy elements and small-to-medium deviations on the observed values) and significance (including minimum size expectations) of biclustering models, 2) check the soundness of the (integrative) searches, 3) visualize results, and 4) extend and adapt pattern-based biclustering.

Empirical Evidence. The collected results throughout *Book III*, associated with pattern-based biclustering algorithms made available by BicPAMS, provide substantial empirical evidence of its effectiveness and relevance. Results over synthetic data confirm the superiority of BicPAM, BiP, BicSPAM and BicNET against peers, and their inherent ability to provide exhaustive yet efficient searches for biclusters with parameterizable coherency and quality. In particular, we demonstrate their ability to deal with varying coherence strength and structures of biclusters, as well as their robustness to discretization problems (assessed against planted deviations on the values) and to varying amounts of noisy and missing elements. Biological analyzes from the application of the algorithms in BicPAMS over gene expression data and biological networks reveal their unique ability to find complete, meaningful and non-trivial biclusters with heightened biological relevance that could not be discovered by other biclustering algorithms. This analysis was further corroborated by their transcriptional regulation.

Future Work. In the mid-term, we expect to embed BicPAMS within the biclust package for R; integrate BicPAMS with BiCAT [45]; extend the declarative, graphical and programmatic interfaces with the assessments provided in *Book V* to guarantee the statistical significance of biclustering models; and support discriminative analyzes in the presence of labeled data by incorporating the contributions provided in *Book VI*. We also expect to provide an in-depth analysis of its relevance for the analysis of structurally sparse data.

10

Constraint-based Biclustering in the Presence of Background Knowledge

One of the current key problems in the field of biclustering is how to incorporate the available domain knowledge to guide the learning process. Initial attempts to use background knowledge for biclustering based on user expectations [207, 271, 381] and knowledge-based repositories [650, 442, 486] show its importance to explore efficiency gains and guarantee relevant solutions.

In this context, two valuable synergies can be identified. First, the optimality and flexibility of pattern-based biclustering solutions provide an adequate basis upon which knowledge-driven constraints can be incorporated. Contrasting with pattern-based biclustering, alternative biclustering algorithms place restrictions on the structure, coherency and quality of biclusters, which may prevent the incorporation of certain constraints [429, 310]. Second, the effective use of background knowledge to guide pattern mining searches has been largely researched in the context of domain-driven pattern mining [520, 83].

Despite these synergies, there is a lack of research on the feasibility and impact of integrating domain-driven pattern mining and biclustering. In particular, there is not yet a solid ground on how to map the commonly available background knowledge in the form of constraints to guide the biclustering task. Additionally, the majority of existing pattern-based biclustering algorithms rely on searches dependent on bitset vectors [442, 578, 500], which may turn their performance impracticable for large and dense biological datasets. Although new searches became recently available for biclustering large and dense data [315], there are not yet contributions on how these searches can be adapted to seize the benefits from the available domain knowledge.

In this chapter, we address these problems. We extend pattern-based biclustering algorithms recurring to principles from domain-driven pattern mining to seize large efficiency gains in the presence of background knowledge. Furthermore, we show how functional annotations and constraints with succinct, (anti-)monotone and convertible properties can be used to guide the biclustering task. As such, the major contributions are five-fold:

- integrative view of constraint-based pattern mining, full-pattern mining and pattern-based biclustering. The applicability of this view over frequent itemsets, association rules and sequential patterns is shown;
- principles for biclustering tabular data in the presence of an arbitrary number of annotations per observation and/or derived from knowledge repositories and literature;
- list of meaningful constraints succinct, (anti-)monotone and convertible properties for biological and social data contexts;
- principles to specify, process and incorporate native, nice and tougher constraints;
- extension of pattern-growth searches to optimally explore efficiency gains from constraints with succinct, (anti-)monotone and convertible properties. In particular we show:
  - F2G [315] compliance with state-of-the-art pruning principles on (conditional) FP-Trees;
  - IndexSpan [308] compliance with prefix-monotone verifications during database projections.

Figure 10.1 provides a structured view on the proposed contributions and of their applicability.

To achieve these goals, we propose BiC2PAM (**BiC**lustering with **C**onstraints using **PA**ttern **M**ining), an algo-

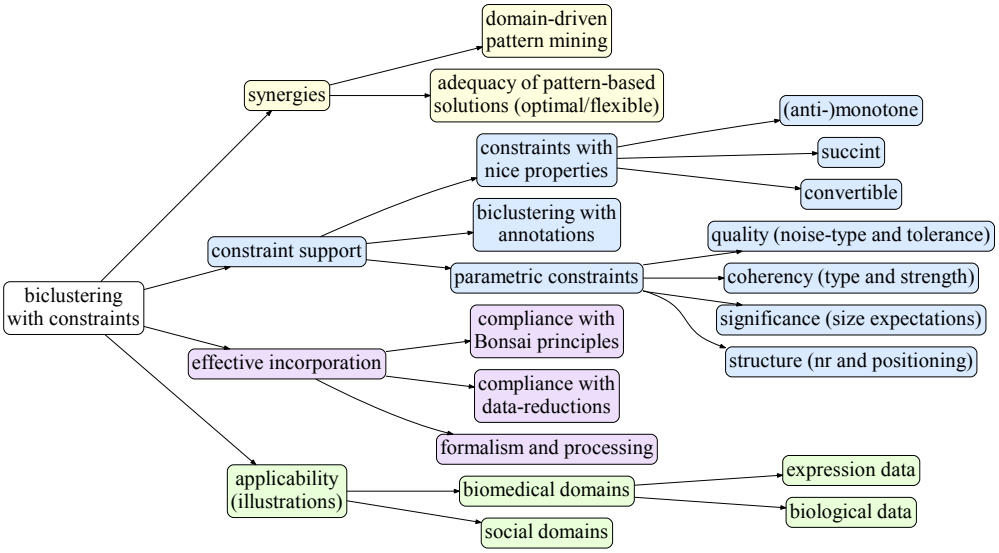

Figure 10.1: Proposed contributions for an effective incorporation of constraints with distinct properties into the (pattern-based) biclustering tasks.

rithm that integrates recent breakthroughs on pattern-based biclustering (covered in *Chapter III-9* [310, 311, 303]) and extends them to effectively incorporate constraints.

Experimental results show the importance of incorporating background knowledge within pattern-based biclustering to seize large efficiency gains by adequately pruning the search space and to guarantee non-trivial and (biologically) relevant solutions.

This chapter is structured as follows. *Section 10.1* provides the background on domain-driven pattern mining for pattern-based biclustering. *Section 10.2* surveys key contributions and limitations from related work. *Section 10.3* lists meaningful constraints in gene expression data and social/biological networks, and provides an algorithmic basis (BiC2PAM) for their incorporation. *Section 10.4* extends BiC2PAM to explore further efficiency gains. *Section 10.5* provides initial empirical evidence of BiC2PAM's efficiency and ability to unravel non-trivial yet significant biclusters. Finally, concluding remarks are synthesized.

10.1 Background

As motivated, the discovery of exhaustive and flexible structures of biclusters with parameterizable homogeneity is a desirable condition to effectively incorporate knowledge-driven constraints. Despite its relevance, most of the existing biclustering algorithms are either based on greedy or stochastic approaches, producing sub-optimal solutions and placing restrictions (e.g. fixed number of biclusters, non-overlapping structures, and simplistic coherencies) that prevent the flexibility of the biclustering task [429]. Pattern-based biclustering appeared in recent years as an effective attempt to address these limitations. As such, we redirect the reader to the previous chapters (*Book III*), in order to understand the structural properties of this class of biclustering algorithms and their role for guaranteeing the discovery of flexible structures of biclusters with parameterizable coherency and quality. At its core, pattern-based biclustering relies on the (iterative application of the) full-pattern mining task. In fact, when assuming the principles proposed in *Chapter III-5* to guarantee the efficiency of closing options, the computational complexity of pattern-based biclustering is essentially bounded by the complexity of the full-pattern mining task.

A full-pattern is essentially a frequent itemset, association rule, sequential pattern, sequential rule or subgraph with frequency and length above certain thresholds, from which the set of supporting transactions in the respective transactional/sequential/graph database are known. Frequent itemsets can be discovered to learn constant, additive and multiplicative models, sequential patterns are used to learn order-preserving models, and rules can be composed to learn plaid models or tolerate parameterizable levels of localized noise.

Let $\mathcal{L}$ be a finite set of items, and a *pattern* P to be a composition of items, either an itemset ($P \subseteq \mathcal{L}$), association rule ($P : P_1 \rightarrow P_2$ where $P_1 \subseteq \mathcal{L} \wedge P_2 \subseteq \mathcal{L}$) or sequence ($P{=}P_1..P_n$ where $P_i \subseteq \mathcal{L}$).

Let a database D be a finite set of observations, each defining a composition of items (either an itemset, rule or sequence). Given a (real-valued) matrix A, let D be a transactional database derived from A by concatenating the items with their column indexes (each observation is an itemset) or be a sequential database derived from A by ordering column indexes according to the values per observation (each observation is a sequence).

A *full-pattern* is a tuple $(P, \Phi_P, \psi_P, \Upsilon_P)$, where P is the pattern in D, $\Phi_P \in X$ is its coverage (observations satisfying P), $\Psi_P \in Y$ is the set of indexes (columns), and Υ_P be the original pattern in A. Given a matrix A, the mapped database D, and a minimum support θ_1 and pattern length θ_2 thresholds, *full-pattern mining* consists of computing: $\{(P, \Phi_P, \psi_P, \Upsilon_P) \mid sup_P \geq \theta_1 \wedge |P| \geq \theta_2\}$. In this context, a set of maximal biclusters can be mapped from closed full-patterns. *Basics III-5.1* provides illustrative instantiations of these concepts.

Constraint-based Biclustering

To formalize the task targeted by this chapter, we introduce below the concept of constraint in the context of biclustering, and further describe different types of constraints according to the selected full-pattern mining task. A constraint is traditionally seen as a conjunction of relations (predicate) over a set of variables describing a given dataset. Definitions 10.1 and 10.2 revise this notion to guarantee its proper applicability within (pattern-based) biclustering tasks.

Def. 10.1 In the context of pattern mining, a *constraint* is a predicate on the powerset of items $C : 2^{\mathcal{L}} \rightarrow \{true,false\}$. In the context of full-pattern mining, a ***full-constraint*** is a predicate on the powerset of original items, transactions, indexes and/or concatenations, $C : \{2^{\mathbf{Y}} \times 2^{\mathcal{L}}, 2^{\mathbf{X}}, 2^{\mathbf{Y}}, 2^{\mathcal{L}}\} \rightarrow \{true,false\}$. A full-pattern $(P, \Phi_P, \psi_P, \Upsilon_P)$ satisfies a full-constraint C if $C(P, \Phi_P, \psi_P, \Upsilon_P)$ is true.

Def. 10.2 A ***biclustering constraint*** is a predicate on a bicluster's values per column, rows I, columns J and pattern $\varphi_{\mathbf{B}}$, $C : \{2^{\mathbf{Y}} \times 2^{\mathcal{L}}, 2^{\mathbf{X}}, 2^{\mathbf{Y}}, 2^{\mathcal{L}}\} \rightarrow \{true,false\}$. A bicluster $\mathbf{B}$ satisfies a constraint C if $C(\varphi_{\mathbf{B}} \cdot \mathbf{J}, \mathbf{I}, \mathbf{J}, \varphi_{\mathbf{B}})$ is true (or, alternatively, when the associated full-pattern satisfies a full-constraint).

Consider a tabular dataset with $n{=}3$ transactions and $m{=}4$ columns mapped into a transactional database with $\mathcal{L}{=}\{a,b,c\}$. An illustrative full-constraint is $y_1 a \in P \wedge \{\mathbf{x}_2, \mathbf{x}_3\} \subseteq \Phi_P \wedge \mathbf{y}_4 \in \Psi_P \wedge \{a\} \subseteq \Upsilon_P$. Minimum support and pattern length are the default full-constraints for full-pattern mining: $C_{support} = |\Phi_P| \geq \theta$ and $C_{length} = |P| \geq \theta$.

More interesting constraints with properties of interest include regular expressions or aggregate functions. In the presence of matrices with numeric or ordinal values, further constraints can be specified. In this context, a *cost table* is specified in addition to the alphabet of items (e.g. {a:0,b:1,c:2}). Depending on the type of full-pattern, multiple constraints can be applied against a cost table, including the paradigmatic cases of aggregate functions such as length, maximum, minimum, range, sum, mean and variance [521].

Some of these constraints are said to exhibit *nice properties* when their input can be effectively pushed deep into the pattern mining task [520] to prune the search space and therefore achieve efficiency gains. Below, we explore different types of constraints according to the selected full-pattern mining task for biclustering: itemset, rule-based and sequential-pattern constraints.

Itemset Constraints. Regular expressions and aggregate functions are the most common form of constraints to guide frequent itemset mining. In this context, efficiency gains can be seized in the presence of constraints with succinct, (anti-)monotone and convertible properties.

Def. 10.3 Let $\mathcal{L}$ be a set of items and P be an itemset, $P \subseteq \mathcal{L}$. Let each item $\sigma \in \mathcal{L}$ have a correspondence with a real value, $c : \mathcal{L} \rightarrow \mathbb{R}$, according to a well-defined *cost table*. Let v be a real-valued constant and $range(P)=max(P)-min(P)$, $max(P)=max \bigcup_{\sigma \in P} c(\sigma)$, $min(P)=min \bigcup_{\sigma \in P} c(\sigma)$ and $avg(P)= \sum_{\sigma \in P} \frac{c(\sigma)}{|P|}$ be well-defined predicates. In this context:
A constraint C is **monotone** if for any P satisfying C, P supersets satisfy C (e.g. $range(P) \geq v$).
A constraint C is **anti-monotone** if for any P not satisfying C, P supersets do not satisfy C (e.g. $max(P) \leq v$).
Given a pattern P' satisfying a constraint C, C is **succint** over P if P contains P' (e.g. $min(P) \leq v$).
A constraint C is **convertible** with regards to an ordering of items R_Σ if for any itemset P satisfying C, the P suffixes satisfy C or/and itemsets with P as suffix satisfy C (e.g. $avg(P) \geq v$).

To instantiate the formalized constraints, consider three observations ($\mathbf{x}_1 = \{a, b, c\}$, $\mathbf{x}_2 = \{a, b, c, d\}$, $\mathbf{x}_3 = \{a, d\}$), a minimum support θ_1=1 and length θ_2=2, and the cost table {a:0,b:1,c:2,d:3}. The set of closed full-patterns satisfying: the monotone constraint $range(P) \geq 2$ is $\{(\{a, b, c\}, \{t_1, t_2\}), (\{a, d\}, \{t_1, t_3\}), (\{b, d\}, \{t_2\})\}$; the anti-monotone constraint $sum(P) \leq 1$ is $\{(\{a, b\}, \{t_1, t_2\})\}$; the succint $P \supseteq \{c, d\}$ is $\{(\{a, b, c, d\}, \{t_2\})\}$; and the convertible constraint $avg(P) \geq 2$ is $\{(\{b, c, d\}, \{t_2\})\}$.

Association Rule Constraints. Constraints satisfying these properties can be also effectively applied in the context of association rule mining (for the discovery of noise-tolerant biclusters [305, 303]). In this context, constraints need to be satisfied by the antecedent, consequent, or can be alternatively applied during the generation of frequent itemsets, prior to the composition of rules.

Additional constraints to guarantee specific correlation/interestingness criteria [614] or the dissimilarity and minimality of rules [20] can be specified.

In the context of rule-based biclustering, a full-constraint is evaluated against the union of items on the antecedent and consequent and the union of supporting transactions of the antecedent and consequent. Given $P : P_1 \rightarrow P_2$ and a constraint C, P satisfies C if the full-pattern given by $(\Upsilon_{P_1 \cup P_2}, \Phi_{P_1} \cup \Phi_{P_2}, \psi_{P_1 \cup P_2}, P_1 \cup P_2)$ satisfies C.

Sequential Pattern Constraints. The introduced concepts can be further extended for the incorporation of constraints in the context of sequential pattern mining (for the discovery of order-preserving biclusters [311]). A sequence P is an ordered set of itemsets, each itemset being a set of indexes in $\mathbf{Y}$. Given a matrix $(\mathbf{X}, \mathbf{Y})$ with n=5 rows and m=3 columns and a minimum support θ_1=3, $(y_2 \leq y_1 \wedge y_2 \leq y_3, \{\mathbf{x}_2, \mathbf{x}_4, \mathbf{x}_5\}, \{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3\}, <y_2(y_1 y_3)>)$ is an illustrative full-pattern. Interestingly, the sequential pattern Υ_P does not explicitly disclose the value expectations φ_B. Instead, Υ_P is associated with an ordering relation (such as $y_2 \leq y_1 \wedge y_2 \leq y_3$). In this context, the following constraints can be specified: item constraints (e.g. $\{\mathbf{y}_1, \mathbf{y}_3\} \subseteq P$); length constraints (minimum/maximum number of precedences and/or co-occurrences); super-pattern constraints (patterns that contain a particular set of patterns as sub-patterns – $y_2 \leq y_1 \subseteq P$); and, more interestingly, regular expressions (e.g. $P \equiv y_\bullet \leq \{y_\bullet, y_\bullet\}$). Constraints concerning value expectations can be also specified using the values from a given ordering based on the median of values from the supporting rows and columns. As a result, aggregate functions can be additionally specified within sequential pattern constraints.

With regards to properties of the aforementioned constraints: length constraints are anti-monotonic, while super-pattern constraints are monotonic. Item constraints, length constraints and super-pattern constraints are all succinct. Some aggregate constraints and regular expressions can also show nice properties [526].

10.2 Related Work

Related work is surveyed according to: 1) contributions and limitations of existing attempts to perform biclustering with domain knowledge; 2) state-of-the-art on domain-driven pattern mining; and 3) contributions for full-pattern mining and their propensity to accommodate domain knowledge.

Knowledge-driven Biclustering. The use of domain knowledge to guide biclustering has been increasingly motivated since solutions with good homogeneity and statistical significance may not necessarily be biologically rel-

evant. However, only few biclustering algorithms are able to incorporate domain knowledge. AI-ISA [650], GenMiner [442] and scatter biclustering [486] are able to annotate data with functional terms retrieved from repositories with ontologies, and use these annotations to guide the search.

COBIC [468] is able to adjust its behavior (maximum-flow/minimum-cut parameters) in the presence of background knowledge. Similarly, the priors and architectures of generative biclustering algorithms can also incorporate background knowledge [309]. However, COBIC and its generative peers only accommodate search-specific constraints and are not able to deliver flexible biclustering solutions.

Fang et al. [207] proposed a constraint-based algorithm enabling the discovery of dense biclusters associated with high-order combinations of single-nucleotide polymorphisms (SNPs). Data-Peeler [271], as well as algorithms from formal concept analysis [381] and bi-sets mining [69], are able to efficiently discover dense biclusters in binary matrices in the presence of (anti-)monotone constraints. However, this set of algorithms impose a very restrictive form of homogeneity in the delivered biclusters.

Domain-driven Pattern Mining. A large number of studies explored how constraints can be used to guide pattern mining tasks. Two major paradigms are available: constraint-programming (CP) [83] and dedicated searches [489, 520]. CP allows pattern mining to be declaratively defined according to sets of constraints [83, 364]. These declarative models are expressive as they can allow for complex mathematical expressions on the set of full-patterns. Nevertheless, due to the poor scalability of CP methods, they have been only used in highly constrained settings, small-to-medium data, or to mine approximative patterns [364, 83].

Pattern mining methods have been adapted to seize efficiency gains from different types of constraints [489, 520, 82]. These efforts aim to replace naïve solutions based on post-filtering to guarantee the satisfaction of constraints. Instead, the constraints are pushed as deeply as possible within the mining step for an optimal pruning of the search space. The nice properties exhibited by constraints, such as anti-monotone and succinct properties, have been initially seized in the context of frequent itemset mining by Apriori methods [489] to affect the generation of candidates. Convertible constraints, can hardly be pushed in Apriori methods but can be adequately handled by pattern growth methods such as FP-Growth [520]. FICA, FICM, and more recently MCFPTree, are FP-Growth extensions to further explore opportunities from diverse constraints [520]. The inclusion of monotone constraints is more complex. Filtering methods, such as ExAnte [81], are able to combine anti-monotone and monotone pruning based on reduction procedures. Empirical evidence shows that these reductions are optimally handled within pattern growth methods by adequately growing and pruning small FP-Trees (referred as FP-Bonsais) [82].

These contributions have been extended for association rule mining [599, 82]. In particular, nice properties have been studied for item constraints [599], support constraints [657], bounds interestingness criteria [51], and constraints on the structure and dissimilarity of rules (respectively referred as schema and opportunistic) [44].

Similarly, some studies proposed ways to effectively incorporate constraints within Apriori and pattern-growth searches for sequential pattern mining (SPM). Apriori searches were first extended to incorporate temporal constraints and user-defined taxonomies [598]. Mining frequent episodes in a sequence of events [435] can also be viewed as a constrained SPM task by seeing episodes as constraints in the form of acyclic graphs. SPIRIT [237] revises the Apriori search to incorporate a broader range of constraints with nice properties and regular expressions. Pattern growth searches based on data projections, such as PrefixSpan, were only later extended by Pei et al. [526, 525] to support a wide-set of constraints with nice properties. Although multiple studies have been proposed on the use of temporal constraints for SPM, including length and gap constraints [24, 526], these constraints are not relevant for the aim of learning order-preserving models.

Full-Pattern Mining with Constraints. There are three major classes of full-pattern mining searches [305, 290, 424]: 1) AprioriTID-based searches, generally suffering from costs of candidate generation for dense datasets and low support thresholds; 2) searches with vertical projections, which show efficiency bottlenecks for data with a high number of transactions since the bitset cardinality becomes large and associated intersection procedures

expensive; and 3) recently proposed pattern-growth searches based on the annotation of original pattern-growth structures with transactions' identifiers. In particular, F2G [315] and IndexSpan [308] (default options in BicPAM, BiP, BicNET and BicSPAM biclustering algorithms [310, 303, 313, 311]) were the first pattern-growth searches for full-pattern mining with the aim of surpassing memory and time bottlenecks associated with bitset and diffset structures preserved by AprioriTID and vertical-based searches.

Despite the large number of contributions from domain-driven pattern mining, the ability of pattern-growth searches to effectively incorporate *full*-constraints with nice properties (Def.10.1) was not yet demonstrated.

10.3 Pattern-based Biclustering with Background Knowledge

In this section, we extend BicPAMS to accommodate constraints. As follows, *Section 10.3.1* provides principles for biclustering annotated data; *Section 10.3.2* lists meaningful constraints with nice properties for gene expression analysis and network data analysis; and *Section 10.3.3* proposes principles for their specification, processing and incorporation.

10.3.1 Biclustering with Annotations extracted from Knowledge Repositories and Literature

Domain knowledge comes often in the form of annotations associated with specific rows and columns in a matrix (or nodes in a network). These annotations are often retrieved from knowledge repositories, semantic sources and/or literature. Annotations can be either directly derived from the properties associated with each row/column/node (e.g. properties of a gene or a sample in gene expression data) or can be implicitly predicted based on the observed values by using feature extraction procedures. For instance, consider the set of functional annotations associated with Gene Ontology (GO) terms [440]. A GO term is associated with an interrelated group of genes associated with a specific biological process. Since a gene can participate in multiple biological processes, genes can have an arbitrary number of functional annotations. As such, rows in an expression matrix (or nodes in a biological network) can be annotated with a non-fixed number of labels.

Pattern-based biclustering supports the integrated analysis of matrices and annotations recurring to one of two strategies. First, association rules or sequential rules can be used to guide the biclustering task in the presence of annotations according to the principles introduced by Martinez et al. [442]. In this context, annotations can either appear in the consequent, antecedent or on both sides of an association rule. Biclusters can then be inferred from these rules using the principles introduced by Henriques et al. [305]. Illustrating, a rule $\{y_1 2, y_4 2\} \rightarrow \{T_1, T_2\}$ supported by $\{\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_5\}$ rows can be used to compose a bicluster $(\{\mathbf{y}_1, \mathbf{y}_4\}, \{\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_5\})$ with elements consistently associated with annotations T_1 and T_2. Learning association rules with levels of confidence (or alternative interestingness scores) below 100% [303] is relevant to discover biclusters with consistent annotations without imposing a subset of annotations to appear on all rows/columns of each bicluster.

Second, the annotations can be included directly within data since pattern mining is able to rely on rows with an arbitrary length. To this aim, annotations are associated with a new dedicated symbol and appended to the respective rows, possibly leading to a set of observations with varying length. Consider the annotations T_1 and T_2 to be respectively associated with genes $\{\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_4\}$ and $\{\mathbf{x}_3, \mathbf{x}_5\}$, an illustrative transactional database of itemsets for this scenario would be $\{\mathbf{x}_1 = \{a_{11}, .., a_{1m}, T_1\}, \mathbf{x}_2 = \{a_{21}, .., a_{2m}\}, \mathbf{x}_3 = \{a_{31}, .., a_{3m}, T_1, T_2\}, ...\}$. Databases of sequences (for order-preserving biclustering) can be composed by appending terms either at the end or the beginning of each sequence.

Given these enriched databases, pattern mining can then be applied on top of these annotated transactions with succinct, (anti-)monotone and convertible constraints. Succinct constraints can be incorporated to guarantee the inclusion of certain terms (such as $P \cap \{T_1, T_2\} \neq 0$). This is useful to discover, for instance, biclusters with genes participating in specific functions of interest. (Anti-)monotone convertible constraints can be, alternatively incorporated to guarantee, for instance, that a bicluster associated with a discovered pattern is functionally consistent,

meaning that it can be mapped to a single annotation. The $|P \cap \{T_1, T_2\}| \geq 1$ constraint is anti-monotone and satisfies the convertible condition: if P satisfies C, the P suffixes also satisfy C.

Interestingly, the two previous strategies can be seen as equivalent when assuming that the discovery of the introduced class of association rules is guided by rule-based constraints and the discovery of patterns from annotated data is guided by itemset/sequence constraints.

10.3.2 Biological Constraints with Properties of Interest

Different types of constraints were introduced in Definition 10.3. In order to show how these constraints can be specified and instantiated, this section provides examples of meaningful constraints for gene expression and network data analysis.

Note that similar constraints can be formulated for the analysis of alternative biological data, including: structural genome variations to enable the discovery of high-order single-nucleotide polymorphisms; genome-wide data to find promoters where mutations or appearing binding sites show properties of interest; or medical data to force the inclusion of certain clinical features or to focus on less-trivial disease markers.

Gene Expression Data Analysis. For illustrative purposes, consider Figure 10.2 to be associated with a symbolic expression matrix (and associated "price table"), where the rows in the matrix correspond to different genes and their values correspond to the observed expression levels for a specific condition (column). The {-3,-2}, {-1,0,1} and {2,3} sets of symbols are respectively associated with repressed (down-regulated), default (preserved) and activated (up-regulated) expression levels.

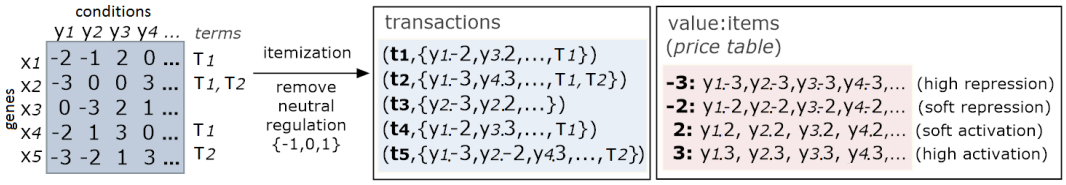

Figure 10.2: Illustrative symbolic dataset and "price table".

First, *succinct* constraints in gene expression analysis allow the discovery of genes with specific constrained levels of expression across a subset of conditions. Illustrating, $min(\varphi_B)$=-3 implies an interest in biclusters (putative biological processes) where genes are at least highly repressed in one condition. Alternatively, succinct constraints can be used to discover non-trivial biclusters by focusing on non-highly differential expression (e.g. patterns with symbols {-2,2}). Such option contrasts with the large focus on dense biclusters [429], thus enabling the discovery of less-trivial yet coherent modules.

Second, *(anti-)monotone* constraints are key to capture background knowledge and guide biclustering. For instance, the non-succinct monotonic constraint $countVal(\varphi_B) \geq 2$ implies that at least two different levels of expression must be present within a bicluster (putative biological process). In gene expression analysis, biclusters should be able to accommodate genes with different ranges of up-regulation and/or down-regulation. Yet, the majority of existing biclustering approaches can only model a single value across conditions [429, 310]. When constraints, such as the value-counting inequality, are available, efficiency bottlenecks can be tackled by adequately pruning the search space.

Finally, *convertible* constraints also play an important role in biological settings to guarantee, for instance, that the observed patterns have an average of values within a specific range. Illustrating, the anti-monotonic convertible constraint $avg(\varphi_B) \leq 0$ indicates a preference for patterns with repression mechanisms without a strict exclusion of activation mechanisms. These constraints are useful to focus the discovery on specific expression levels, while still allowing for noise deviations. Understandably, they are a robust alternative to the use of strict bounds from succinct constraints with maximum-minimum inequalities.

Biological Network Data Analysis. To motivate the relevance of inputting similar constraints for the analysis of biological networks, we use again the tabular dataset provided in Figure 10.2. In this context, rows and columns correspond to nodes associated with biological entities (such as genes, proteins, protein complexes or other molecular compounds), and the values in the matrix correspond to the strength of the interactions between the nodes. As such, the strength of the interactions is either negative {-3,-2} (e.g. inhibition), weak {-1,0,1} or positive {2,3} (e.g. activation).

First, succinct constraints can be specified for the discovery of sets of nodes with specific interaction patterns of interest. Illustrating, $\{-2, 2\} \subseteq \varphi_B$ implies an interest on non-dense network modules (coherent interactions with soft inhibition and activation) to disclose non-trivial regulatory activity, and $min(\varphi_B) = -3 \wedge max(\varphi_B) = 3$ implies a focus on modules with the simultaneous presence of highly positive and negative interactions.

Second, (anti-)monotone constraints are key to discover network modules with distinct yet coherent regulatory interactions. For instance, the non-succinct monotonic constraint $countVal(\varphi_B) \geq 3$ implies that at least three different types of interactions must be present within a module.

Finally, convertible constraints are useful to place non-strict expectations on the desirable patterns, yet still accommodating deviations from expectations. Illustrating, $avg(\varphi_B) \leq 0$ indicates a preference for network modules with negative interactions without a strict exclusion of positive interactions.

Constraints with nice properties can be alternatively applied for networks with qualitative interactions. Regulatory interactions, such as "binds", "activates" or "enhances", are increasingly observed for a wide-variety of protein-protein and gene interaction networks [426, 473]. In this context, assuming the presence of {a,b,c} types of biological interactions, an illustrative anti-monotone constraint is $|\varphi_B \cap \{a, b\}| \geq 0$.

Biological Data Analysis with Full-Constraints. Although less motivated, constraints can be also defined on the powerset of rows, columns and/or values per columns. In fact, the minimum support and minimum pattern length can be seen as constraints over I and J indexes, respectively. An alternative constraint over I and J is to require that biclusters include a minimum number rows/columns from a particular subset of rows/columns of interest. An illustrative succinct constraint in $Y \times \mathcal{L}$ is $P \cap \{y_2\text{-}3, y_23\} \neq \emptyset$, which implies an interest in biclusters with differential expression (or interactions) associated with the $\mathbf{y}_2$ sample/gene/node.

Other Data Contexts. Similarly, meaningful constraints with nice properties can be formulated for the analysis of:

- medical data to force the inclusion of certain clinical features or to focus on less-trivial markers of disease;
- collaborative filtering data to focus on specific rating behavior or for contrast analysis;
- text data to center the learning on specific words or expectations regarding the coherency of counts;
- structural genome variations to enable the discovery of high-order single-nucleotide polymorphisms;
- trading data to focus on buying/holding/selling profiles of interest (e.g. risk) for specific markets.

The constraints illustrated throughout this section represent a small subset of possible constraints of interest for the sake of motivating their relevance. A careful specification of constraints should always be made as function of the learning scope and the input dataset. Understandably, an exhaustive listing and discussion of relevant constraints for biomedical and social data contexts is considered to be out the scope of this work.

10.3.3 Incorporation of Full-Constraints

We propose BiC2PAM (**BiC**lustering with **C**onstraints using **PA**ttern **M**ining) to effectively incorporate full-constraints (including the set of constraints motivated in previous section). BiC2PAM's extensions to the existing contributions on pattern-based biclustering [310, 313, 311, 303, 578] are two-fold. First, a precise formalism was defined to represent full-constraints (with identical notation to the one introduced along this work) and new processing procedures were implemented for their parsing and interpretation. Under these principles, the desirable properties

of biclustering solutions can be defined with sharp usability. BiC2PAM supports not only the specification of full-constraints (Definition 10.2), but further makes available the possibility to specify native constraints to customize the structure, coherency and quality of biclustering solutions (as described in *Appendix*). Second, BiC2PAM implements different strategies to incorporate distinct types of constraints:

- if native constraints are inputted, BiC2PAM maps them into parameterizations along the mapping, mining and closing steps of BicPAMS (*Appendix*);
- if constraints without nice properties are inputted, BiC2PAM satisfies them using post-filtering verifications;
- if constraints with nice properties are inputted, BiC2PAM implements pruning heuristics from previous research on constraint-based Apriori-based methods [657, 237].

In the context of the formal view on constraint-based full-pattern mining introduced in *Section* , when constraints over Υ_P (constraints in $2^{\mathcal{L}}$) are inputted, they are mapped as constraints over $P \in 2^{Y \times \mathcal{L}}$. For instance, the $a \in \Upsilon_P$ succinct constraint is mapped as $P \cap \{y_1 a, ..y_m a\} \neq \emptyset$.

Similarly, constraints from $\psi_P \in 2^Y$ are mapped to constraints over $P \in 2^{Y \times \mathcal{L}}$. Illustrating, $y_2 \in Y$ is mapped as $P \cap \{y_2 a, y_2 b, ..\} \neq \emptyset$.

Finally, constraints from $\Phi_P \in 2^X$ are incorporated by adjusting the Apriori searches to effectively prune the search space. Consider a succinct constraint that specifies a set of transactions to be included in the resulting biclusters. In this case, as soon as a generated candidate is no longer supported by any transaction of interest, there is no need to further generate new candidates and, thus, the search space can be pruned at this point.

Understandably, despite the inherent simplicity of incorporating constraints with nice properties in Apriori-based searches, there is a critical drawback: the inability to rely on key pattern-growth searches, such as F2G (for the discovery of constant/additive/symmetric/plaid biclusters) and IndexSpan (for the discovery of order-preserving biclusters). These pattern-growth searches were previously shown to be able to mine large data with superior efficiency [315, 308]. Adding to this observation, there is a considerable agreement that the underlying structures of pattern-growth searches, such as frequent-pattern trees and prefix-growth trees, provide a more adequate representation of the search space for an improved pruning.

Figure 10.3 provides a highly simplified illustration of BiC2PAM behavior.

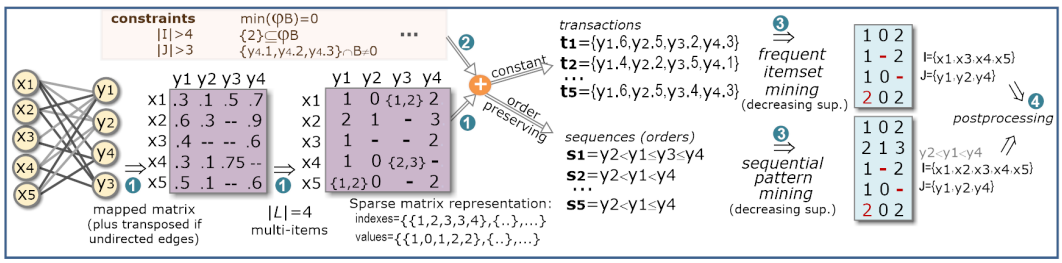

Figure 10.3: Simplified illustration of BiC2PAM behavior: 1) transactional and sequential databases derived from a multi-item matrix; 2) constraints are processed; 3) pattern mining searches are applied with a decreasing support; and 4) the discovered pattern-based biclusters that satisfy the inputted constraints are postprocessed.

10.4 Exploring Optimum Efficiency Gains from Constraints with Nice Properties

Although native constraints are effectively incorporated in BiC2PAM through adequate parameterizations of pattern-based biclustering algorithms, the incorporation of constraints with nice properties can only be easily supported under Apriori-based searches. Furthermore, despite the large consensus that pattern-growth searches are better positioned to seize efficiency gains from constraints than peer methods based on bitset vectors, there is not yet proof whether this observation remains valid in the context of full-pattern mining. As such, *Sections 10.4.1* and *10.4.2* extend the recently proposed F2G and IndexSpan algorithms to guarantee an optimal pruning of the search space in the presence of constraints. These extensions are integrated in BiC2PAM.

10.4.1 F2G-Bonsai: F2G with Constraints

F2G implements a pattern-growth search that does not suffer from efficiency bottlenecks of peer searches since it relies on FP-tree structures where transaction-IDs are stored without duplicates (see *Chapter 5*). The FP-tree is recursively mined to enumerate all full-patterns. Unlike other pattern-growth searches, transaction-IDs are not lost at the very first scan. Full-patterns are generated by concatenating the pattern suffixes with the full-patterns discovered from conditional FP-trees where suffixes are removed. F2G behavior is illustrated in Figure 5.2. In this section, we first show the compliance of F2G with principles to handle succinct and convertible constraints [520]. Second, we show compliance of F2G with principles to handle difficult combinations of monotone and anti-monotone constraints [82].

Compliance with Different Types of Constraints. Unlike candidate generation methods, pattern growth methods (such as FP-Growth) provide further pruning opportunities. Pruning principles can be standardly applied on both the original database (full FP-Tree) and on each projected database (conditional FP-Tree).

The CFG method extends FP-Growth [520] to seize the properties of nice constraints under three simple principles. First, supersets of itemsets violating anti-monotone constraints are removed from each (conditional) FP-Tree. Illustrating, in the presence of $sum(\Upsilon_P) \leq 3$, when analyzing the $y_1 2$ conditional database, the following items $\cup_{i=1}^{m}\{y_i 2, y_i 3\}$ can be removed to avoid conflicts as their sum violates the illustrated constraint. For an effective pruning, it is recommended to order the symbols in the header table according to their value and support [520, 521]. F2G is compliant with these pruning heuristics, since it allows the rising of transaction-IDs in the FP-Tree according to the order of candidate items for removal in the header table (see Algorithms 4 and 5).

For the particular case of an anti-monotone convertible constraint, itemsets that satisfy the constraint are efficiently generated under a pattern-growth search [521]. This is done by assuming that original/conditional FP-trees are built according to a price table and by pruning patterns that no longer satisfy an anti-monotone convertible constraint since the inclusion of new items will no longer satisfy the constraint. Illustrating, since $\{y_1\text{-}3, y_4 2, y_2 3\}$ does not satisfies $avg(\Upsilon_P) \leq 0$, there is no need to further build $\{y_1\text{-}3, y_4 2, y_2 3\}$-conditional trees. Therefore, This principle provides an important criterion to stop FP-tree projections and/or prune items in a (conditional) FP-tree.

Finally, the transactions and items within a (conditional) FP-tree that conflict with a given constraint can be directly removed without causing any changes on the resulting set of valid patterns. Illustrating, given $min(\Upsilon_P) = 0$ constraint, the transactions $\mathbf{x}_1 = \{y_1\text{-}1, y_2 3, y_3 1\}$ and $\mathbf{x}_4 = \{y_1 1, y_2\text{-}1, y_3 2\}$ can be directly removed as they do not satisfy the given succinct constraint. Similarly, given the same constraint, $min(\Upsilon_P) = 0$, the items with values below 0 can be removed. For the illustrative $\mathbf{x}_1$ and $\mathbf{x}_4$ transactions, this means removing $a_{1,1} = y_1\text{-}1$ and $a_{4,2} = y_2\text{-}1$ items.

Also, constraint checks can be avoided for subsets of itemsets satisfying a monotone constraint. Illustrating, no further checks are needed in the presence of $countVal(\Upsilon_P) \geq 2$ constraint when the range of values in the suffix of a pattern is ≥2 under the $\{y_1 0, y_1 1\}$-conditional FP-Tree.

Combination of Constraints with Nice Properties. The previous extensions to pattern-growth searches are not able to effectively comply with monotone constraints when anti-monotone constraints (such as minimum support) are also considered. In FP-Bonsai [82], principles to further explore the monotone properties for pruning the search space are considered without reducing anti-monotone pruning opportunities. This method is based on data-reduction operations originally implemented in ExAnte to seize efficiency gains from the properties of monotone constraints. There are two data-reductions: μ-reduction, which deletes transactions not satisfying C; and α-reduction, which deletes from transactions single items not satisfying C. Thanks to the recursive projections of FP-growth, the ExAnte data-reduction methods can be applied on each conditional FP-tree to obtain a compact number of smaller FP-Trees (FP-Bonsais). The FP-Bonsai method can be combined with the previously introduced principles, which are particularly prone to handle succinct and convertible anti-monotone constraints. F2G can be extended to support these reductions on the (conditional) FP-Trees by guaranteeing that transactions consistently

rise up. The only requirement is to preserve the order of items in the header table (see Algorithm 6). As such, it complies with the FP-Bonsai extension.

10.4.2 IndexSpanPG: IndexSpan with Constraints

The work of Pei et al. [526] provides principles to extend pattern-growth searches with prefix-based database projections and no candidate generation to effectively incorporate regular expressions and constraints with nice properties. For this aim, the prefix-monotone property is defined. A constraint is called *prefix-monotone* if it is prefix anti-monotonic or prefix monotonic. With a prefix-monotone constraint, there is only the need to search in the projected databases for prefixes that satisfy the constraint. When a constraint C is: 1) prefix anti-monotonic, if $C(P)$=false, then there exists no sequential patterns containing P has a prefix and also satisfies C; 2) prefix monotonic, if $C(P)$=true, then every sequential pattern having P as a prefix satisfies C; and 3) a regular expression, if the prefix of a given sequential pattern is conflicting with the regular expression C, then there is no need to further expand (i.e. there are no sequential patterns with the same prefix that also satisfy C). As such, since monotonic, anti-monotonic and regular expression constraints are prefix-monotone they can be pushed deep into the search. Understandably, the efficiency gains associated with such constraints cannot be attained under Apriori-based searches [237]. Although succinct constraints are not necessarily prefix anti-monotonic or prefix monotonic, they can also be easily pushed deep into the mining process (independently of the applied SPM method).

According to these principles, we extended IndexSpan [308], an extension of PrefixSpan proposed in *Chapter III-6*, to explore efficiency gains from the intrinsic properties of the order-preserving biclustering task. IndexSpan is compliant with the enumerated principles. The minimalist data structures, fast database projections and early pruning techniques [308] do not interfere with the underlying prefix-growth behavior, the essential requirement to incorporate prefix-monotone constraints. Furthermore, given the fact that IndexSpan explores item-indexable properties associated with the order-preserving biclustering task, testing constraints is done in an efficient and elegant way. This is true with regards to: 1) the validation of whether an anti-monotonic constraint (or regular expression) cannot be satisfied by a given prefix (in order to stop its growth), and 2) the validation of whether a a monotonic constraint cannot be satisfied by a given (projected) sequence (in order to prune the search).

10.5 Results and Discussion

This section provides empirical evidence of the soundness of the proposed contributions and of the relevance of using constraints within (pattern-based) biclustering to prune the search space and guarantee biologically significant solutions. For this end, we assess the performance of BiC2PAM on synthetic data, expression data and biological networks in the presence of domain knowledge. BiC2PAM is implemented in Java (JVM v1.6.0-24). The experiments were computed using an Intel Core i5 2.30GHz with 6GB of RAM.

BiC2PAM was extended for biclustering data with annotations and further parameterized with the proposed F2G-Bonsai search for the discovery of constant biclusters with itemset constraints and with the proposed IndexSpanPG for the discovery of order-preserving biclusters with sequential pattern constraints. BiC2PAM was run with default behavior for pattern-based biclustering (according to [311, 310]). Biclusters with >70% of overlapping area were merged in a postprocessing step [308]. We specified the stopping criteria of BiC2PAM to be defined by a minimum number of 20 dissimilar biclusters for synthetic data and 50 dissimilar biclusters for real data.

10.5.1 Results on Synthetic Data

The generated data settings are described in Table II-3.1. Constant and order-preserving biclusters with different area, shape and coherency strength ($|\mathcal{L}|\in\{4,7,10\}$) were planted. Reported results are the average of 30 instances per setting. BiC2PAM was applied with F2G and a default merging option (70% of overlapping).

Uninformative Elements. A simplistic yet relevant form of domain knowledge is the knowledge regarding the uninformative elements of a given dataset. To this end, the ranges of values (or symbols) to remove can be specified under a succinct constraint $S \notin P$ where $S \subseteq \mathbb{R}^+$ (or $S \subseteq \mathcal{L}$). The application of this constraint within BiC2PAM leads to the removal of these elements prior to the mining step, resulting in significantly large efficiency gains as shown by Figure 10.4. This figure describes the impact of removing a varying extent of uninformative elements from synthetic data on the biclustering task. Despite the simplicity of this constraint, existing biclustering algorithms are not able to support this behavior, which undesirably impacts their efficiency and the adequacy of the outputted biclustering solutions.

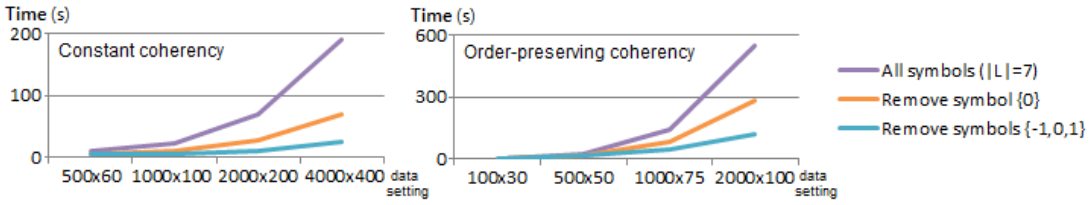

Figure 10.4: Efficiency gains of BiC2PAM from succinct constraints specifying uninformative elements for varying data settings with constant and order-preserving biclusters and coherency strength defined by $|\mathcal{L}|$=7.

Incorporating Annotations. Figure 10.5 assesses the ability of BiC2PAM to discover biclusters with functional consistency when an input matrix (2000×200) is annotated with an arbitrary number of labels per row[1]. For this end, rows were generated with a varying number of annotations, {10±4, 4±2}, where each annotation is observed on a varying number of rows, {200±10, 100±10}. For this analysis, we guaranteed that the hidden biclusters have a high degree of functional consistency by imposing that the majority (85%±10pp) of their rows share a common annotation. As such, BiC2PAM was parameterized with succinct constraints guaranteeing that at least one annotation is consistently observed for all the rows of each bicluster before postprocessing (before the application of extension, merging and reduction procedures). Despite the higher complexity from mining heterogeneous data (input data plus a large amount of annotations), results show that BiC2PAM is in fact more efficient than the baseline option. Furthermore, the observed match scores suggests that the presence of annotations may play an important role in guiding the recovery of true biclusters.

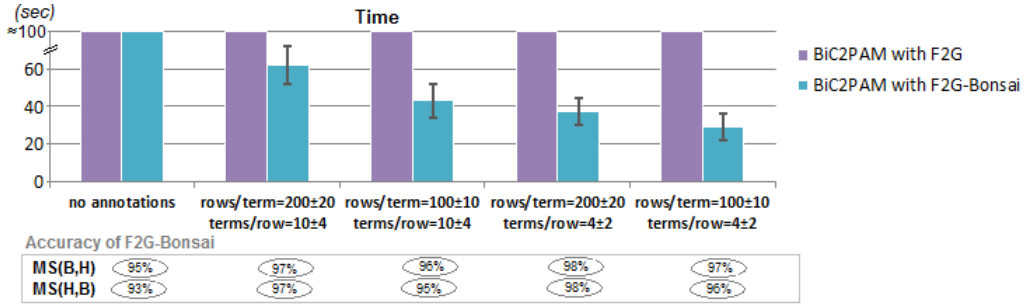

Figure 10.5: BiC2PAM ability to biclustering data with varying distributions of annotations (efficiency and Jaccard-based match scores [310] collected for the 2000×200 setting).

Itemset Constraints. In order to test the ability of BiC2PAM to seize efficiency gains in the presence of itemset constraints with nice properties, we applied BiC2PAM over the 2000×200 data setting (generated with 5 background symbols $\mathcal{L}$={-2,-1,0,1,2} and hidden biclusters with constant assumption) in the presence of succinct, monotone and convertible constraints. For the baseline performance, constraints were satisfied using post-filtering procedures. Figure 10.6 shows the impact of inputting disjunctions of succinct constraints in the performance of BiC2PAM. As observed, the ability of BiC2PAM to effectively prune the search space in the presence of these constraints is associated with significant efficiency gains. Moreover, they enable a focus on less-trivial regions from the input data space (e.g. $-1 \in \varphi_B \vee 1 \in \varphi_B$).

[1] Datasets available in http://web.ist.utl.pt/rmch/software/bic2pam/

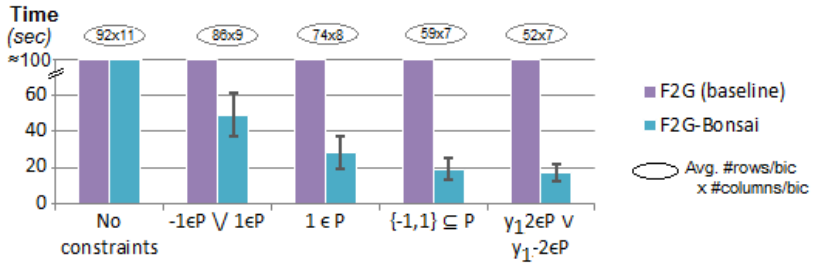

Figure 10.6: BiC2PAM's efficiency in the presence of succinct constraints (2000×200 setting with constant assumption).

Figure 10.7 measures the performance of BiC2PAM when constraints with monotone, anti-monotone and convertible properties are inputted. To this end, we show the efficiency gains from parameterizing the underlying F2G miner with diverse principles, and further test F2G's ability to deal not only with constraints satisfying a single property but multiple properties of interests (e.g. $\gamma_1 < sum(\varphi_B) < \gamma_2$). Results confirm that the proposed enhancements can lead to a substantial pruning of the search space. In particular, CFG principles [520] are used to seize efficiency gains from convertible constraints and FP-Bonsai [82] to seize efficiency gains from monotonic constraints.

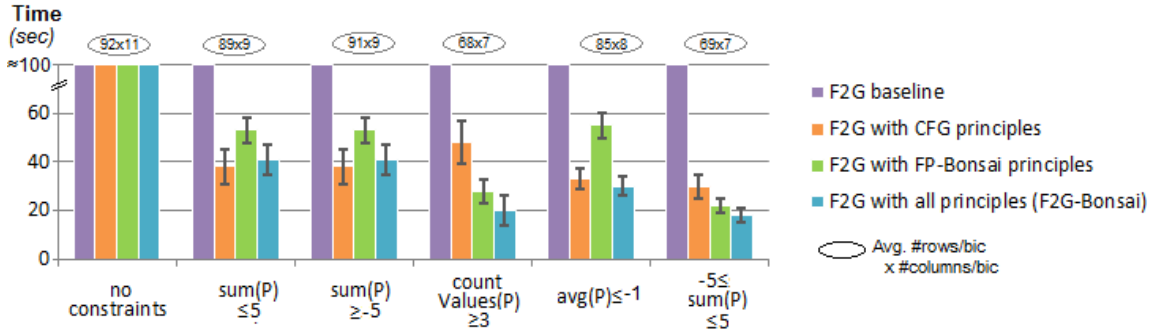

Figure 10.7: BiC2PAM's efficiency with (combined) anti-monotone, monotone and convertible constraints (2000×200 setting with constant coherency). Impact of enhancing BiC2PAM with CFG [520] and FP-Bonsai [82] principles.

Sequential Pattern Constraints. Figure 10.8 extends the previous analyses towards the constraint-guided discovery of order-preserving biclusters with regular expressions. For this analysis, BiC2PAM was parameterized with IndexSpan and IndexSpanPG and applied over the 1000×100 setting with a varying set of constraints (minimum number of precedences and ordering constraints). Results show that increased efficiency gains can be attained from pruning data regions that do not satisfy these constraints.

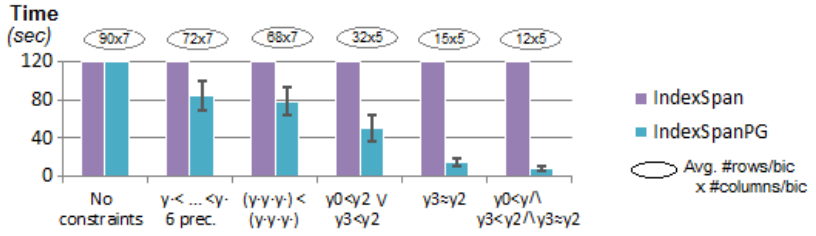

Figure 10.8: BiC2PAM performance with sequence constraints when learning order-preserving solutions (1000×100 setting).

10.5.2 Results on Biological Data

Real Data. We selected expression and network datasets with varying properties. Four gene expression datasets were considered: *dlblc* (660 genes, 180 conditions) with human responses to chemotherapy [560], *hughes* (6300 genes, 300 conditions) to study nucleosome occupancy [395], and *yeast-cycle* (6221 genes, 80 conditions) and *gasch* (6152 genes, 176 conditions) measuring yeast responses to environmental stimuli [239]. Three biological networks from STRING v10 database [610] were additionally considered. These networks capture the gene interactions

within human (6314 nodes, 423335 interactions), Escherichia coli (8428 nodes, 3293416 interactions) and yeast (19247 nodes, 8548002 interactions) organisms. The scores in these networks are inferred from literature and multiple data sources, revealing the expected strength of correlation between genes.

Uninformative Elements. In gene expression data analysis, elements from the input matrix with default/non-differential expression are generally less relevant. Similarly, in the context of network data analysis, interactions with low weights are generally of reduced interest for module discovery. In these contexts, these data elements can be removed from the learning under a succinct constraint. Figures 10.9-10.10 measures the impact of inputting such succinct constraints on the efficiency of BiC2PAM and on the properties of the outputted biclusters (assuming constant coherency). For this analysis, we analyze performance of BiC2PAM on both expression data (Figure 10.9) and network data (Figure 10.10) from different organisms. Results show that by inputting such simplistic constraints, very high efficiency gains can be obtained. Additionally, the removal of uninformative elements allows the focus on more relevant regions of the input data space and is associated with slightly smaller biclusters due to the greater ability to exclude such elements from the solution space.

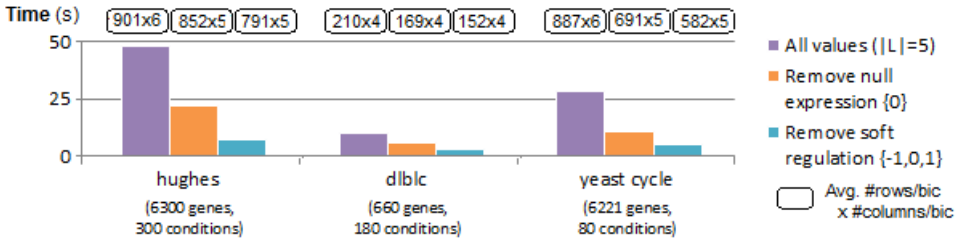

Figure 10.9: Efficiency of BiC2PAM with knowledge regarding the uninformative elements for the analysis of expression data (hughes, dlblc, yeast-cycle) when assuming a constant coherency with $|\mathcal{L}|$=5.

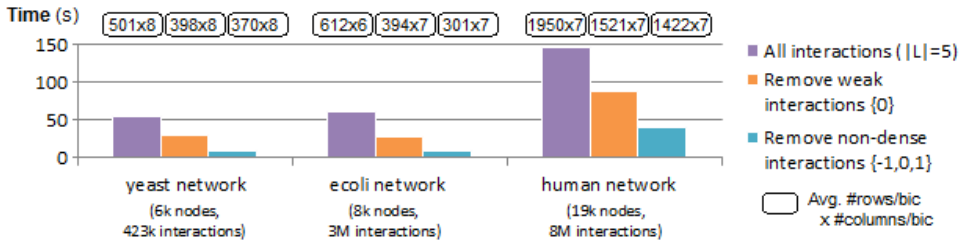

Figure 10.10: Efficiency of BiC2PAM with knowledge regarding the uninformative elements for the analysis of network data (human, E. coli, yeast from STRING [610]) when assuming constant coherency with $|\mathcal{L}|$=5.

Annotations. Figure 10.11 measures the impact of incorporating functional terms from ontologies for the analysis of biological data (assuming an underlying constant coherency). To this end, we collected for each gene from human and yeast organisms the set of functional terms associated with the biological processes represented in gene ontology from GOToolBox [440]. BiC2PAM was then applied over expression and network data in the presence of these annotations. Results confirm that BiC2PAM is able to integratively learn from data and annotations without further costs in efficiency, and to guarantee the functional consistency of the outputted biclusters (as expectedly demonstrated by the analysis of the enriched terms).

Succinct, Monotone and Convertible Constraints. Figures 10.12 and 10.13 show the impact of inputting biologically meaningful constraints in the efficiency and effectiveness of BiC2PAM. For this purpose, we used the complete gasch dataset (6152×176) [240] with five levels of expression ($|\mathcal{L}|$=6). The impact of considering a diverse set of constraints in the efficiency levels of BiC2PAM is provided in Figure 10.12. The observed results demonstrate the relevance of using meaningful constraints with succinct, (anti-)monotone and convertible properties not only to guarantee a user-guided focus on specific regions of interest, but also to promote the tractability to perform biclustering to solve computationally complex biological problems and analyzes.

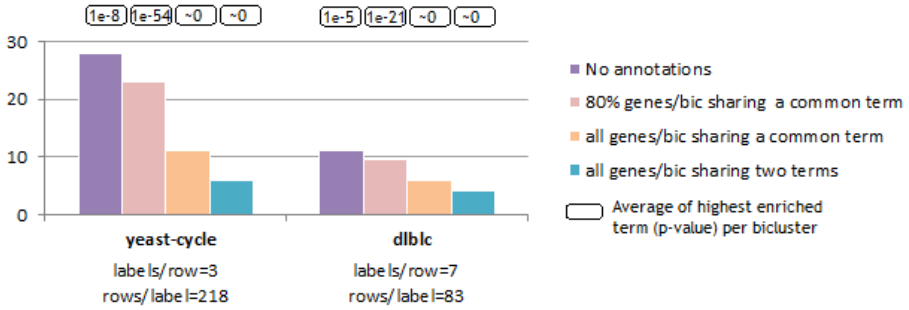

Figure 10.11: Performance of BiC2PAM for biclustering biological datasets (yeast-cycle and dlblc) annotated with representative human and yeast GO terms (terms associated with biological processes with more than 50 genes).

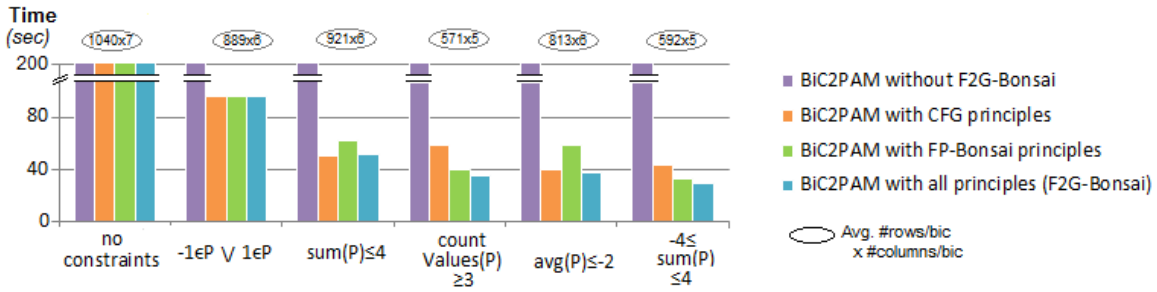

Figure 10.12: Efficiency gains from using biologically meaningful constraints with succinct/monotone/convertible properties within BiC2PAM for the analysis of the gasch dataset (6152×176).

The impact of these constraints in the relevance of pattern-based biclustering solutions is presented in Figure 10.13. The biological relevance of each bicluster was derived from the analysis of functionally enriched GO terms based on the application of hypergeometric tests [440]. A bicluster is considered significantly enriched if it has a set of correlated over-represented terms with Bonferroni corrected p-values below 10^{-3}. Two major observations can be retrieved. First, when focusing on properties of interest (e.g. differential expression), the average significance of biclusters increases as their genes have higher propensity to be functionally co-regulated. This trend is observed despite the smaller size of the constrained biclusters. Second, when focusing on rare expression profiles (≥3 distinct levels of expression), the average relevance of biclusters slightly decreases as their co-regulation is less obvious. Yet, such non-trivial biclusters hold unique properties with potential interest that can be further investigated. To our knowledge, BiC2PAM is the only available biclustering algorithm able to rely on user expectations and other forms of knowledge to focus the search on these non-trivial yet coherent and potentially interesting regions from the input data space.

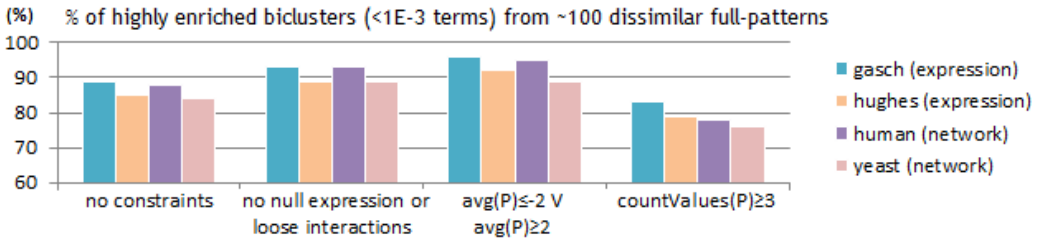

Figure 10.13: Biological relevance of F2G for multiple constraint-based profiles of expression.

10.6 Summary of Contributions and Implications

This chapter motivates the task of biclustering data in the presence of background knowledge. To answer this task, we explored the synergies between pattern-based biclustering and domain-driven pattern mining. As a result, BiC2PAM algorithm was proposed with two major goals: 1) to learn biclustering models in the presence of an arbitrary number of annotations from knowledge repositories and literature, and 2) to effectively incorporate

constraints with nice properties derived from user expectations and available knowledge. Under BiC2PAM, not only the relevance of the output solutions can be promoted, but also heightened efficiency gains can be explored.

In this chapter, we also showed the consistency of integrating the fields of constraint-based pattern mining and pattern-based biclustering based on the notion of full-patterns; surveyed the major impediments of existing research towards this end; and extended pattern-growth searches with state-of-the-art principles to prune the search space by pushing constraints with nice properties deep into the mining process. In particular, we show the compliance of F2G with principles to effectively prune (conditional) FP-Trees, and the compliance of IndexSpan with principles to effectively prune prefix-growth structures. These searches were accordingly extended to support pattern-based biclustering with constant and order-preserving assumptions.

Meaningful constraints with succinct, monotone, anti-monotone and convertible properties were presented for different data domains, including gene expression analysis and network data analysis, to focus the search space on non-trivial yet relevant regions. Furthermore, constraints on the rows and columns associated with a given pattern were shown to be consistently specified and effectively supported. Finally, we discussed the relevance of associating price tables to sequential patterns mapped from tabular data and provided a simplistic mapping to support this possibility.

Results from synthetic and real data quantify the efficiency gains from using distinct constraints, demonstrating the relevance of their effective incorporation. We further provide empirical evidence of the relevance of incorporating annotations derived from knowledge repositories to increase the functional consistency of biclustering solutions, as well as of accommodating constraints to guarantee the discovery of non-trivial yet meaningful biclusters in biological data.

Future Work. Three major directions are identified for upcoming work on this field. First, the extension of these contributions towards classification tasks with a focus on the discriminative properties of biclusters. Second, an in-depth systematization of constraints with nice properties across the data domains from Table I-1.1. Finally, a broader quantification of the impact of incorporating constraints across biomedical and social real-world problems.