Acquiring Background Knowledge for Intelligent Tutoring Systems

Cláudia Antunes [1]
claudia.antunes@ist.utl.pt
[1] Dep of Computer Science and Engineering,
Instituto Superior Técnico, Av. Rovisco Pais 1, 1049-001, Lisboa, Portugal

Abstract. One of the unresolved problems faced in the construction of intelligent tutoring systems is the acquisition of background knowledge, either for the specification of the teaching strategy, or for the construction of the student model, identifying the deviations of students' behavior. In this paper, we argue that the use of sequential pattern mining and constraint relaxations can be used to automatically acquire that knowledge. We show that the methodology of constrained pattern mining used can solve this problem in a way that is difficult to achieve with other approaches.

1 Introduction

In the last years, the construction of intelligent tutoring systems has changed, shifting from a story boarding paradigm to a construction based on the separation of the content and instructional methods [6]. Along with the use of this new paradigm, the explicit and modular representation of the knowledge appeared as a fundamental task, with its acquisition being a central problem.

In order to solve this problem, in the last years, several tools were developed to automatically extract the background knowledge from data. Among the machine learning techniques used in those tools, the unsupervised ones have appeared as the most promising ones [9].

In this paper, we argue that sequential pattern mining is one of the unsupervised techniques of interest to this task, and, in conjunction with constraints and constraint relaxations, it can be used to discover the usual misconceptions or the bug library. A methodology for performing this discovery based on the use of sequential pattern mining and constraint relaxations is proposed and analyzed in this context.

The remaining of this paper is structured as follows: after presenting the sequential pattern mining problem and its solutions, the new methodology is described. This description is followed by a case study, where curricula instantiations are found. The paper concludes by presenting the conclusions and some guidelines for future work.

2 Sequential Pattern Mining

In general, it is possible to distinguish two major classes among unsupervised learning tasks: clustering and pattern mining. While clustering tries to identify groups of elements, that are similar within each group and dissimilar between groups, pattern mining aims at discover the set of "behaviours" that occur in the data a significant number of times. One

of the interesting characteristics of pattern mining is that with a few additional features, pattern mining algorithms are able to discover structured behaviours, and, in particular sequential patterns. These patterns exist when the data to be mined has some sequential nature, i.e., when each piece of data is an ordered set of elements. An example of such data is the sequence of actions performed and results achieved by some student, when interacting with a learning environment.

Sequential Pattern Mining is the machine learning task that addresses the problem of discovering the existing frequent sequences in a given database. The problem was first introduced in 1995 [1], and can be specified as

> *Given a set of sequences and some user-specified minimum support threshold σ, the goal of sequential pattern mining is to discover the set of sequences that are contained in at least σ sequences in the dataset, this is the set of frequent sequences.*

At this point it is important to make some remarks: first, a sequence is an ordered set of itemsets; second, an itemset is a non-empty set of items of interest for the analysis (the itemset composed of two items a and b is represented by (a,b)); finally, a sequence s is contained in another sequence t if all the itemsets of s are contained in some itemset of t, preserving the original order of occurrence.

2.1 Main Drawbacks and the Use of Constraints

In the last years, several sequential pattern mining algorithms were proposed, like *GSP* [9] and *PrefixSpan* [8]. In general, these algorithms act incrementally, discovering a pattern with k items, after the discovery of patterns with (k-1) items. This approach reduces the search space at each step, making use of the anti-monotonicity property. This property is related with the fact that a pattern can only be frequent if all its sub-patterns are also frequent. Despite the reasonable efficiency of pattern mining algorithms, the lack of focus and user control has hampered the generalized use of pattern mining. Indeed, the usually large number of discovered patterns makes the analysis of discovered information a difficult task. In order to solve this problem, several authors have promoted the use of constraints, where a constraint is defined as a predicate on the set of finite sequences. In this manner, given a database of sequences, some user-specified minimum support threshold σ and a constraint, a sequence is frequent if it is contained in at least σ sequences in the database and satisfies the constraint. This approach has been widely accepted by the data mining community, since it allows the user to control the mining process, either by introducing his background knowledge deeply into the process or by narrowing the scope of the discovered patterns. The use of constraints also reduces the search space, which contributes significantly to achieve better performance and scalability levels. When applied to sequential pattern mining, constraints over the content can be just a constraint over the items to consider, or a constraint over the sequence of items. More recently, regular languages have been proposed [4] and used to constrain the mining process, by accepting patterns that may be accepted by a regular language, represented as a deterministic finite automaton (DFA). Subsequent work [2] has shown that regular languages can be substituted by context-free languages, without compromising the performance of the algorithms. This is useful because context-free languages are more expressive than regular ones, and the only adaptation needed is the

substitution of the finite automaton by a pushdown automaton.

3 Acquisition of Background Knowledge for Student Modeling

One of the central components of intelligent tutoring systems is the student model, which is a qualitative representation that accounts for student behaviour in terms of existing background knowledge, and represents the system's belief about the learner's knowledge [11]. It comprises two distinct forms of knowledge: the domain theory and the bug library [9]. The domain theory corresponds to the ideal model of students' behaviour and in some cases it is completely specified. On the other side, the bug library collects the set of misconceptions held and other errors made by a population of students. If it is not impossible to enumerate all these facts, it is certainly very difficult to do it with the existing approaches.

Considering these aspects, we propose a new methodology to acquire background knowledge for student modelling. This methodology assumes that the curriculum knowledge can be represented by a context-free language and the bug library can be found by sequential pattern mining using constraint relaxations. The idea is that the curriculum knowledge can be used to guide the search of the facts in the bug library as a constraint, while the search is performed by a sequential pattern mining algorithm. Since constraints filter all the non-accepted sequences, a softer filter is needed. Constraint relaxations can perform the role of these softer filters, and can enable the discovery of patterns that deviate from the curriculum. Different classes of relaxations express the different levels of deviation of interest to the analyst.

3.1 Constraint Relaxations

The notion of constraint relaxation has been widely used when real-life problems are addressed. In sequential pattern mining they were first introduced in [4], where a regular expression was used to constrain the mining process, and some relaxations were used to improve the performance of the algorithm. A constraint relaxation can be seen as an approximation to the constraint. When used instead of the constraint, it enables the discovery of unknown information that will approximately match user expectations. If these relaxations are used to mine new patterns, instead of simply used to filter the patterns, the discovery of unknown information is possible [3].

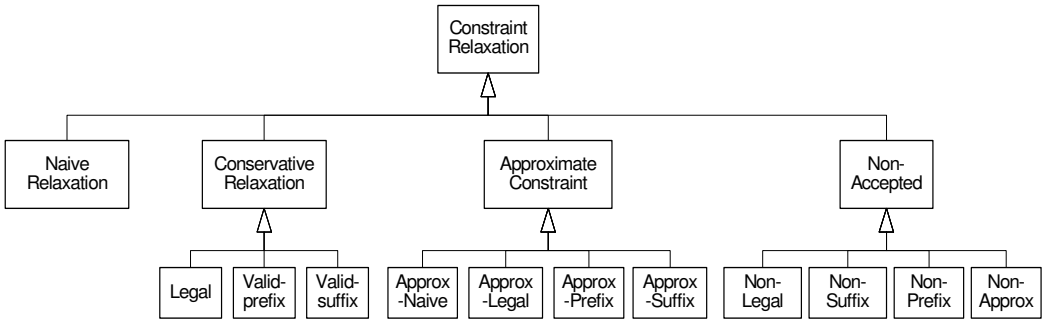

Figure 1. Hierarchy of constraint relaxations

Conservative relaxations group the already known relaxations, used in SPIRIT [4], and a third one – the *Valid-Prefix*. These relaxations impose a weaker condition than the original constraint, accepting patterns that are subsequences of accepted sequences. Although these relaxations have a considerable restrictive power, which improves significantly the focus on user expectations, they do not allow for the existence of errors.

Approx relaxations accept sequences that are approximately accepted by the constraint, which means that the patterns are at an acceptable edit distance from some sequence accepted by the constraint. This edit distance reflects the cost of operations (such as insertion deletion or replacement), that have to be applied to a given sequence, so it would be accepted as a positive example of a given formal language [5]. Approx relaxations can be combined with other relaxations resulting in a new set of relaxations. In general, approximate relaxations can be seen as less restrictive than conservative ones, and can be used to identify the common behaviours of students that made a limited number of errors when comparing to the curriculum knowledge.

The third class of relaxations is the *Naïve* relaxation, which corresponds to a simple item constraint. However, in the context of constraints expressed as formal languages, it can be seen as a relaxation that only accepts patterns containing the items that belong to the alphabet of the language. The naïve relaxation can be used to identify a portion of the frequent behaviours that comprise specific actions, permitting to understand the relations between those actions.

Finally, *Non-Accepted* relaxations accept patterns that are not accepted by the constraint. This type of relaxation is useful when there is a well-known model for the generality of sequences, and the goal is to identify the sequences that are not accepted by that model. In this manner, it is possible to identify low frequency behaviors that are still very significant to the domain. Fraud detection is the paradigm of such task. Note that the difficulties in fraud detection are related with the explosion of discovered information when the minimum support decreases.

It is important to note that the non-accepted relaxation will find all the patterns discovered by the other relaxations, representing a small improvement in the focus on user expectations. An interesting issue is to associate a subset of the alphabet in conjunction with non-accepted relaxation. This conjunction allows for focusing the mining process over a smaller part of the data, reducing the number of discovered sequences, and contributing to reach our goal. Like before, the sub-classes of Non-Accepted relaxations result by combining the non-acceptance philosophy with each one of the others relaxations. While non-accepted relaxation filters only a few patterns, when the constraint is very restrictive, the non-legal relaxation filters all the patterns that are non-legal with respect to the constraint. With this relaxation is possible to discover the behaviours that completely deviate from the accepted ones, helping to discover the fraudulent behaviours. A detailed definition of these relaxations for constraints specified as context-free languages see [3]. In the context of the acquisition of the bug library, the non-accepted relaxation is useful to discover the low-frequent misconceptions. Despite, they are less representative they could be very important as can be seen in the case study.

Table 1. List of common subjects, distributed by scientific area

Common subjects			
Mathematics	AL – Linear Algebra	**Programming Methodologies**	IP – Programming Introduction
	AM1 – Math. Analysis 1		AED – Algorithms and Data Structures
	AM2 – Math. Analysis 2		PLF – Logical and Functional Prog.
	AM3 – Math. Analysis 3		POO – Object Oriented Prog.
	PE – Probability and Statistics	**Architecture and Operating Systems**	SD – Digital Systems
	AN – Numerical Analysis		AC – Computer Architecture
Physics	FEX – Experimental Physics		SO – Operating Systems
	F1 – Physics 1	**Artificial Intelligence**	IA – Artificial Intelligence
	F2 – Physics 2	**Information Systems**	SIBD – IS and Databases
Computer Science	TC – Theory of Computation	**Computer Graphics**	CG – Computer Graphics

4 Curricula Analysis: a case study

In order to demonstrate our claims, consider the curriculum of an undergraduate program on information technology and computer science, with duration of five years (10 semesters) with 20 obligatory subjects (listed in Table 1), 16 subjects from a specific specialty area, an undergraduate thesis and 4 optional subjects in the last year. Also, consider there are four specialty areas: PSI – Programming and Information Systems; SCO – Computer Systems; IAR – Artificial Intelligence and IIN – Information Systems for Factory Automation. This information is usually publicly and previously known, and can be represented as a deterministic finite automaton (as shown on Figure 2). (This DFA shows the curriculum model for each specialty area (from the top to bottom: PSI, SCO, IAR and IIN, respectively; the existence of two different transitions per semester for SCO students, are due to a minor reorganization of the SCO curriculum on 1995/1996).

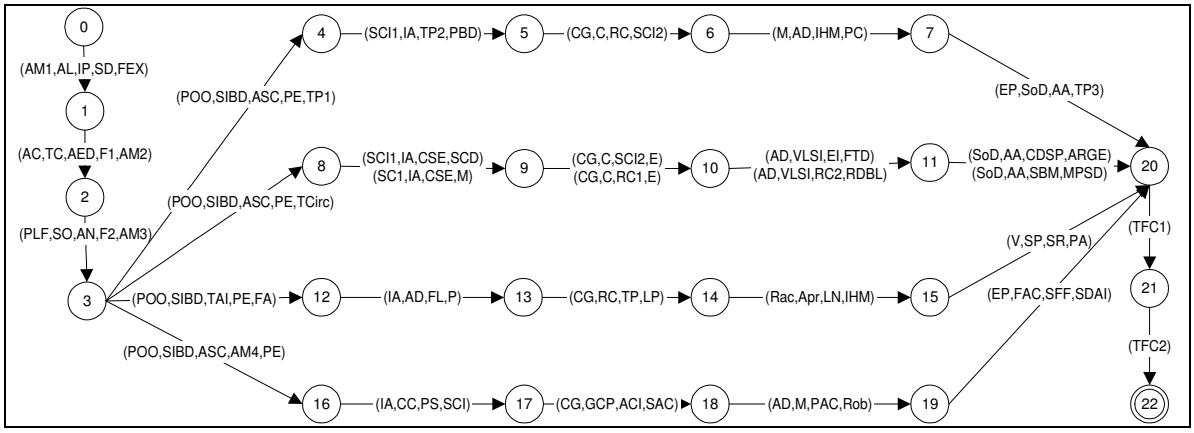

Figure 2. DFA for specifying the model curriculum for LEIC specialty areas

The data used in this study refers to the data of the undergraduate students of the Licenciatura em Engenharia Informática e de Computadores (LEIC) at Instituto Superior Técnico, from the academic years of 1989/90 to 1999/2000. From these students we have only considered the students that have registered for at least eight semesters, and therefore may have concluded the 4th year. In this manner, the dataset is composed of

1440 sequences, corresponding to the curriculum followed by each LEIC student. These sequences have an average length equal to 11.58 semesters. Most of the students (72%) have between 8 and 12 enrolments. In terms of the number of enrolments per semester, its mean is 4.82 enrolments on subjects per semester, with most of students (75%) enrolling on between 4 and 6 units.

Another interesting issue is the distribution of students per specialty area: 55% are PSI students, 19% SCO students, and IAR and IIN have 13% of students each. This distribution conditions the number of enrolments per course. For example, subjects exclusive to Artificial Intelligence and IIN have at most 13% of support. It is interesting to note that only 823 students (57%) have concluded the undergraduate thesis (TFC1 and TFC2). Since it is usual that students only took optional subjects in parallel or after finishing the undergraduate thesis, the support for optional subjects is at most 57%. Since the options are chosen from a large set of choices (130 subjects), their individual support is considerably lower. Indeed the course on Management (G) is the optional course with more students, about 40%.

The goal of this study is to demonstrate that with the use of the program curriculum as background knowledge and the use of constraint relaxations is possible to discover the frequent students' behaviours that approximately follow the curriculum. Note that if the background knowledge is used as a constraint to the sequential pattern mining process, only five patterns can be discovered: one for each specialty area (two for SCO). Moreover, it is probable that each pattern would have very low supports, since just a few students conclude all the subjects on their first enrolment. Next, we will present two experiments that illustrate the utility and effectiveness of our approach, respectively.

4.1 Finding frequent behaviours per specialty area

The first problem is related to the discovery of frequent behaviours per specialty area, and demonstrates the utility of our methodology. It is non trivial due to the difficulty of discovering which optional subjects are frequently chosen by which students. The difficulty of this task resides on the fact that all non-common subjects can be chosen as optional by some student. In this manner, a simple count of each subject support does not give the expected answer, since most of the subjects are required to some percentage of students.

The other usual approach to find out the frequent choices would be to query the database to count the support of each course, knowing that students have followed some given curriculum. However, this approach is also unable to answer the question, since a considerable number of students (more than 50%) have failed one or more subjects, following a slightly different curriculum.

In order to discover frequent behaviours, we have used the new methodology with the constraint based on the DFA in Figure 3, which corresponds to a sub-graph of the previous one. This automaton accepts sequences that represent the curricula on the fourth curricular year for each specialty area. With a constraint defined over this DFA filters all patterns that do not respect the model curriculum for the last two curricular years.

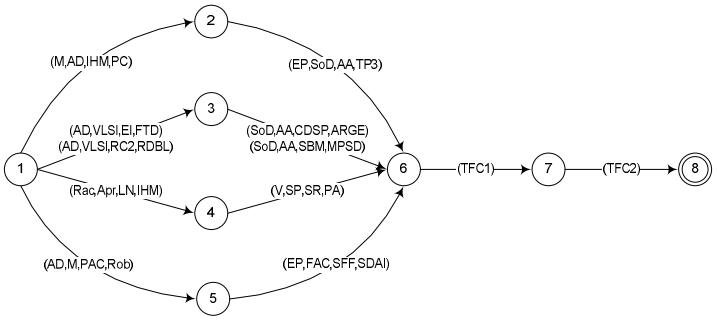

Figure 3. DFA for constraining the search of frequent behaviors per specialty areas

The use of constrained sequential pattern mining (with the specified constraint) would not contribute significantly to answer the initial question, since it would only achieve results similar to the ones achieved by the query described above. However, the use of the *Approx* relaxation that accepts at most two errors (ε=2) chosen from a restricted alphabet composed by every non-common course, allows the discovery of 25 patterns with a support at least equal to 1% (Table 2). These patterns show that, in general, students mostly attend Computer Graphics and Economical subjects.

Table 2. Some of the discovered patterns with optional subjects

Specialty Area	Curricula Instantiations
PSI	(M,AD,IHM,PC)(AA,SoD,EP,TP3)(TFC1,PAC)(TFC2,GF)
	(M,AD,IHM,PC)(AA,SoD,EP,TP3)(TFC1,Econ)(TFC2,GF)
	(M,AD,IHM,PC)(AA,SoD,EP,TP3)(TFC1,Econ)(TFC2,IG)
	(M,AD,IHM,PC)(AA,SoD,EP)(TFC1)(TFC2,TE2)
IIN	(M,AD,PAC,Rob)(EP,SDAI,SFF)(TFC1)(TFC2,IG)
	(M,AD,PAC,Rob)(EP,FAC,SDAI,SFF)(TFC1,IHM)(TFC2,GF)
	(M,PAC,Rob)(EP,FAC,SDAI,SFF)(TFC1,G)(TFC2)
	(M,PAC,Rob)(EP,FAC,SDAI,SFF)(TFC1,TE1)(TFC2)
IAR	(IHM,Rac,LN)(SP,V,PA,SR)(TFC1,TE1)(TFC2)
	(IHM,A,Rac,LN)(SP,V,PA,SR)(TFC1)(TFC2,GF)
SCO	(C)(VLSI,RC2,RDBL,AD)(AA,CDPSD,ARGE,SoD)(TFC1,G)(TFC2)
	(Elect)(VLSI,RC2,RDBL,AD)(AA,CDPSD,ARGE,SoD)(TFC1,G)(TFC2)

It is interesting to note that whenever IIN students have failed on some course on the 4th year, they choose one of two particular courses in Economy; the same is true for PSI and IAR students (shadowed patterns in Table 2). Note that in order to discover these rules, we have to be able to admit some errors on the obligatory curricula per specialty area, which is not easily achieved by executing a query to a database.

4.2 Finding Abandon Reasons

The great difficulty in determining the effectiveness of our mining process is to determine which patterns are the relevant or interesting ones. In order to perform this evaluation, we considered a smaller problem whose results can be easily analyzed. The selected problem is to find the reasons why students abandon LEIC before concluding the

42 subjects required. This problem was chosen because it is easy to enumerate some reasons for abandon in LEIC. By applying common sense, we can suggest two different reasons: the inability to conclude the first computer science specific subjects ('Programming Introduction'-[IP], 'Digital Systems'-[SD], 'Algorithms and Data Structures'-[AED] and 'Computer Architecture'-[AC]), and the inability to conclude the generic engineering subjects ('Linear Algebra'-[AL] and 'Mathematical Analysis 1 and 2'-[AM1, AM2]). This knowledge can be represented by the automaton in Figure 4.

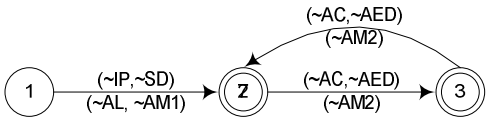

Figure 4. DFA for specifying the anticipated abandon reasons

The dataset used to analyze this question consists on the set of sequences corresponding to the curriculum followed by each LEIC student, with his first enrolment made between 1989 and 1997. The dataset includes all the students that abandoned LEIC (at least temporarily). The dataset (LEICabandons) contains 489 sequences.

In order to choose the relevant patterns, we applied the pattern discovery algorithms on both the LEICabandons and LEIC1989-2001 datasets, and identified as relevant the sequences that are frequent in the first dataset but are not frequent in the second one. With a support of 50%, 91 sequences have been discovered. From these, 79 are relevant by this criterion. A simple analysis of those sequences shows that most students that cancel their registration are not able to conclude more than two subjects in the second semester. Additionally, this analysis shows that the cancellation reasons anticipated and represented in the automaton in Figure 4 are close to the real reasons.

Table 3. Precision and Recall

	Unconst.	Accepted	Legal	Approx (ε=1)	Approx (ε=2)	Approx (ε=3)	Non-Acc
Nr Total Retrieved	91	2	15	20	71	90	89
Nr Retrieved and Relevant	79	2	10	18	63	78	77
Recall	100%	3%	13%	23%	80%	99%	97%
Precision	87%	100%	67%	90%	89%	87%	87%

Assuming these sequences are relevant for this task, it is now possible to evaluate the effectiveness of the new methodology by comparing its results with the results reached with constraints, by counting the number of relevant sequences discovered with each relaxation. Considering the notions of precision and recall usually used for evaluating the effectiveness of information retrieval algorithms, it is possible to compare the use of constraints with the use of relaxations as proposed in this work. When applied to the mining process, precision corresponds to the ratio of relevant patterns retrieved by the process to all patterns retrieved by the process, and recall corresponds to the ratio of relevant patterns retrieved by the process to all relevant patterns in the dataset. Applying those measures it is clear that there are significant differences among the relaxations, as shown in Figure 5.

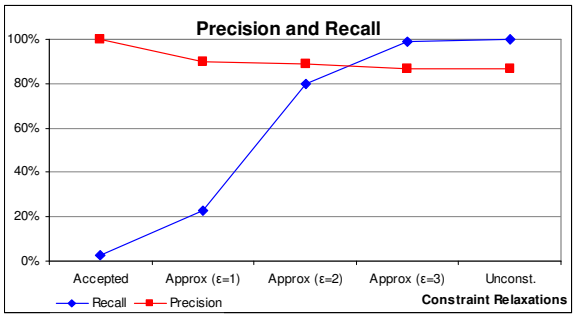

Figure 5. Precision and Recall chart

Since there are only two retrieved sequences when the constraint is used, it is clear that the existing background knowledge is not complete, but is correct. In this manner, the recall of the constrained process is usually very low. The existence of more accurate background knowledge would increase the recall. On the other side, all the accepted sequences are relevant, which makes the precision of the process 100%. The relaxation that shows a better balance between those measures and the efficiency of the process is the approx relaxation, because it is possible to adjust the number of patterns discovered without compromising the precision. By increasing or decreasing the number of errors allowed, it is possible to compensate the excessive selectiveness of the background knowledge. Note that the recall increases considerably when the number of allowed errors increase, but the decrease of the precision is less accentuated. From this analysis and the other experiments presented in this work, it is clear that the great challenge of pattern mining, in general, and of sequential pattern mining, in particular, is to reach the balance between the number of discovered patterns and the user's ability to analyze those patterns with the ability to discover unknown and useful information. The use of constraint relaxations, proposed in this work, represents a first step in this direction.

5 Conclusions

One of the problems in the construction of intelligent tutoring systems is the acquisition of background knowledge, both the teaching strategy and students' frequent behaviour. In this paper, we proposed a methodology to acquire parts of that knowledge, based on the use of sequential pattern mining. Our methodology consists of discovering the frequent sequential patterns among the recorded behaviours, keeping the discovery limited to the sequences that, in some manner, are approximately in accordance to the existing background knowledge. The methodology assumes that the existing background knowledge can be represented by a context-free language, which plays the role of a constraint in the sequential pattern mining process. The use of the constraint relaxations enables the discovery of unknown patterns that correspond to the deviations to the expected behaviours. Different classes of relaxations express different levels of deviation of interest to the analyst. In this paper, we have applied the methodology on identifying the deviations of students' behaviour from a pre-specified curriculum. In order to do that, we have represented the curriculum knowledge as a finite automaton, which establish the order of subjects that a student should attend to finish his graduation. Along with the use of sequential pattern mining, we tried the different relaxations, to answer specific

challenges ranging from the identification of the frequent curricula instantiations, to the discovery of abandon reasons. Although the identified behaviours can be used to classify a student and to predict his next result, alone they do not identify the causes of the failures. An interesting open issue is the correlation of the students' results with a specific teaching strategy. If this strategy and the corresponding expected results are known beforehand, our methodology can be used to identify the steps of the strategy that do not result as expected. In particular, the strategy in conjunction with the expected results can be represented as a context-free language and used as a constraint, to the sequential pattern mining process. In addition, an approx constraint will be able to discover the behaviour patterns that slightly deviate from the expected ones, which identify the failure steps.

References

[1] Agrawal, R and Srikant, R. Mining sequential patterns, *Proc Int'l Conf Data Engineering*, IEEE Computer Society Press, 1995, p. 3-14

[2] Antunes, C. and Oliveira, A.L. Inference of Sequential Association Rules Guided by Context-Free Grammars. *Proc. Int'l Conf. on Grammatical Inference*, 2002, p. 1-13

[3] Antunes, C. and Oliveira, A.L. Constraint Relaxations for Discovering Unknown Sequential Patterns, in B. Goethals and A. Siebes (Eds.): *KDID 2004*, LNCS 3377, Springer-Verlag Berlin Heidelberg, 2005, p.11–32

[4] Garofalakis, M., Rastogi, R. Shim, K. SPIRIT: Sequential Pattern Mining with Regular Expression Constraint. Proc. *Int'l Conf. Very Large Databases*, 1999, 223-234.

[5] Levenshtein, V., Binary Codes capable of correcting spurious insertions and deletions of ones". *Problems of Information Transmission*, 1965, vol.1 1, p. 8-17.

[6] Murray, T Expanding the Knowledge Acquisition Bottleneck for Intelligent Tutoring Systems". *Int'l Journal of Artificial Intelligence in Education*, 1997, vol. 8, p. 222-232.

[7] Murray, T Authoring Intelligence Tutoring Systems: an Analysis of the State of the Art". *Int'l Journal of Artificial Intelligence in Education* 1999, vol. 10, p. 98-129.

[8] Pei J, Han J, Mortazavi-Asl B et al., PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth". *Proc Int'l Conf Data Engineering*. IEEE Computer Society Press, 2001

[9] Sison, R, Shimura, M. Student Modeling and Machine Learning. *Int'l Journal of Artificial Intelligence in Education* 1998, vol. 9, p. 128-158.

[10] Srikant,R, Agrawal,R Mining Sequential Patterns: generalizations and performance improvements. *Proc Int'l Conf Extending Database Technology*. 1996, 3-17

[11] Stauffer, K. Student Modelling and Web-based Learning Systems – Technical Report, Athabasca University. 1996