Discovering User Communities in Large Event Logs

Diogo R. Ferreira, Cláudia Alves

IST – Technical University of Lisbon, Portugal
{diogo.ferreira,claudia.alves}@ist.utl.pt

Abstract. The organizational perspective of process mining supports the discovery of social networks within organizations by analyzing event logs recorded during process execution. However, applying these social network mining techniques to real data generates very complex models that are hard to analyze and understand. In this work we present an approach to overcome these difficulties by focusing on the discovery of communities from such event logs. The clustering of users into communities allows the analysis and visualization of the social network at different levels of abstraction. The proposed approach also makes use of the concept of modularity, which provides an indication of the best division of the social network into community clusters. The approach was implemented in the ProM framework and it was successfully applied in the analysis of the emergency service of a medium-sized hospital.

Key words: Process Mining, Social Network Analysis, Hierarchical Clustering, Community Structure, Modularity

1 Introduction

The goal of process mining [1, 2] is to discover, analyze and understand business processes based on the run-time behavior recorded in event logs. Such analysis can be performed on three different perspectives [3]: the *process perspective* focuses on extracting models for the control-flow of the business process; the *case perspective* focuses on the behavior, properties and data elements associated with individual process instances; and the *organizational perspective* focuses on understanding the roles and groups of people participating in the process. Other issues such as performance [4] and conformance [5] can be studied as well.

While much attention has been devoted to the process perspective through several techniques – such as the α-algorithm [6], the Heuristic Miner [7], the Genetic Miner [8], and the Fuzzy Miner [9] –, the organizational perspective is based mostly on techniques developed by [10] and [11]. Also, there has been considerable concern about making control-flow models more understandable [12], but no comparable effort has been done to facilitate the understanding of very large and complex social networks arising from the analysis of real-world event logs.

In this work we describe an approach to deal with such large models by employing a hierarchical clustering technique to discover community structure [13] in social networks. Such approach facilitates the analysis and visualization of the social network at different levels of abstraction. It also provides an effective means to discover user groups based on the actual user interactions as in [10], rather than on task similarity as in [11]. The distinction is relevant because although hierarchical clustering has already been used in [11], it has been applied to group users according to the similarity of the tasks they perform. Here we will be interested in applying hierarchical clustering to the social network obtained by considering the *working together* metric [14].

2 Extracting social networks from event logs

Figure 1 illustrates a purchase process comprising several steps. First, it is necessary to fill out a requisition form and send it for approval. If not approved, the requisition is archived. If approved, the product is ordered from a supplier and two branches will run in parallel: at the warehouse an employee receives the product and updates the stock; at the accounting department someone else will take care of payment. When these two branches complete, the requisition is closed.

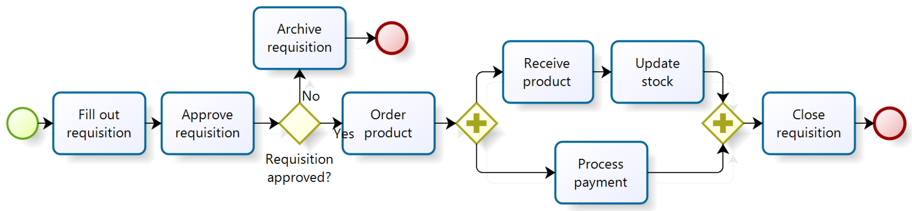


Fig. 1. Purchase process example

Table 1 shows an excerpt of the event log that could be generated by executing this requisition process. There may be several instances of this process, and several participants performing different tasks. Case 1 represents a requisition that was successfully completed, while case 2 is a requisition that was not approved and was afterward resubmitted as case 3.

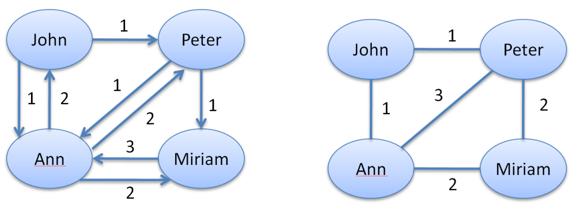
There are basically two ways to study the interaction between participants recorded in such an event log [14]:

– *Handover of work:* captures the number of times each user performs a task just before another user. This results in a directed graph where nodes represent users and arcs are labeled with the number of times a user hands over work to another user. Figure 2(a) shows the resulting graph for the three cases recorded in the requisition process.
– *Working together:* for each pair of users, it captures the number of cases where these users have worked together. This results in an undirected graph where

Table 1. Example of an event log

case id	task id	user id	timestamp
1	Fill out requisition	John	2010-03-29 10:15
1	Approve requisition	Ann	2010-03-30 09:05
1	Order product	John	2010-03-30 14:20
2	Fill out requisition	Miriam	2010-04-02 11:40
1	Receive product	Peter	2010-04-05 08:00
1	Update stock	Peter	2010-04-05 08:10
2	Approve requisition	Ann	2010-04-05 09:30
2	Archive requisition	Peter	2010-04-06 12:20
1	Process payment	Ann	2010-04-07 08:10
3	Fill out requisition	Miriam	2010-04-09 15:40
...	...	...	...

nodes represent users and arcs are labeled with the number of cases. Figure 2(b) shows the resulting graph for the three complete cases.



(a) handover of work (b) working together

Fig. 2. Example networks for three cases of the requisition process

The application of either of these approaches results in a social network (directed or undirected) that depicts the interactions between the participants in a process. For structured processes, the *handover of work* approach may be interesting as it captures some of the control-flow together with the social network. However, for unstructured processes which produce event logs with a lot of ad-hoc behavior, or in the presence of a significant amount of noise, the *working together* metric becomes more useful as it focuses on the social network alone, while the control-flow can be studied with other specialized techniques. For the purpose of understanding large social networks extracted from real-world event logs, we will be using mainly the *working together* metric.

3 Clustering the social network

Clustering, and in particular hierarchical clustering [15], is an essential technique to analyze social networks by aggregating nodes that are close together. The distance (actually, similarity) between nodes can be measured according to a number of different metrics, such as the two approaches described above, which represent different ways of capturing the interaction between each pair of users. Once individual nodes have been grouped together to form clusters, it is possible to measure the similarity between clusters to decide whether any pair of clusters should be merged together. Repeating this merging iteratively leads to a hierarchical clustering approach, which provides a range of cluster configurations from having a cluster for each individual node to having a single cluster that contains all nodes.

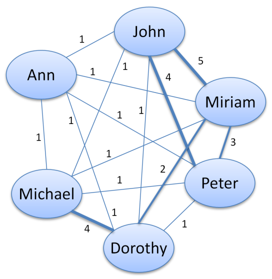
The similarity between clusters can be computed in different ways based on the similarity between individual nodes. Let c_r and c_s be two clusters with n_r and n_s nodes respectively, and let $d(i,j)$ denote the similarity between node i in c_r and node j in c_s. Then the similarity between the two clusters is given by:

$$D(c_r, c_s) = \frac{1}{n_r n_s} \sum_{i \in c_r} \sum_{j \in c_s} d(i,j) \tag{1}$$

This equation specifies the well-known *average linkage* function [16] which provides a measure of the average similarity between pairs of nodes in two different clusters. When computing $D(c_r, c_s)$ for every pair of clusters c_r and c_s, one can find the pair of clusters for which the similarity is maximal, and these become the best candidates for being merged in the current iteration. If there are several candidate pairs of clusters with the same maximal similarity, then an untying procedure is required, as will be explained in the next section. For the moment we will assume there are no ties.

Figure 3 shows an example of a small network created using the working together metric. On the right is the adjacency matrix, where A_{ij} denotes is the weight of the arc between nodes i and j. Clustering begins by considering that each individual node is a cluster, and then finding the pair of clusters with maximum similarity in the network.

In the first iteration, Equation (1) simplifies to $D(c_i, c_j) = d(i,j) = A_{ij}$ and the best candidate is the pair {John, Miriam} with $D = 5$. These nodes become a single cluster, as shown in Figure 4(a). In the second iteration, the best candidate for merging is {Michael, Dorothy} with $D = 4$. Note that another link with $d = 4$ exists between Peter and John, but since John is part of the cluster {John, Miriam}, the average similarity between Peter and {John, Miriam} is $D = 3.5$. Therefore, {Michael, Dorothy} with $D = 4$ becomes the second cluster. It is only in the third iteration that Peter joins {John, Miriam} and the cluster becomes {John, Miriam, Peter}, as in Figure 4(c). In the fourth iteration the clusters {Michael, Dorothy} and {John, Miriam, Peter} have $D = 7/6$ and are merged together; at this stage the only other option would be to include Ann in one of these clusters, but $D = 1$ in either case. In the fifth and final iteration, Ann is



A_{ij}	John	Miriam	Peter	Dorothy	Michael	Ann
John	–	5	4	1	1	1
Miriam	5	–	3	2	1	1
Peter	4	3	–	1	1	1
Dorothy	1	2	1	–	4	1
Michael	1	1	1	4	–	1
Ann	1	1	1	1	1	–

Fig. 3. Diagram and adjacency matrix for a small network

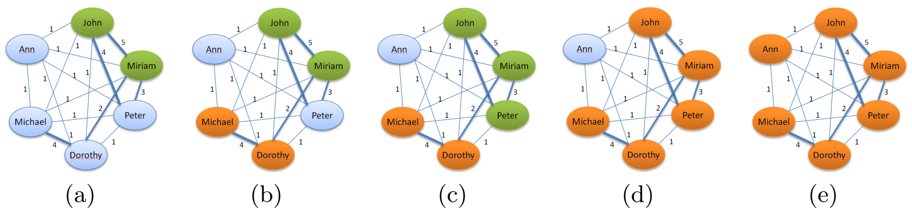


Fig. 4. Clustering of a small network in five iterations

included in {Michael, Dorothy, John, Miriam, Peter} and the network becomes a single cluster, as shown in Figure 4(e).

4 Using modularity to find communities

Clearly, in going from N clusters to a single cluster one should know where to stop. Between the two extremes there are several possible explanations for the social network. In the example above, it appears that {John, Miriam} form one group, while {Michael, Dorothy} form another, and the question is whether Peter should be included in the group of {John, Miriam} as well. This decision requires a criterion either to stop the clustering algorithm at some point [17, 18] or to determine the optimal number of clusters [19, 20].

In particular, we are looking for the cluster configuration which best describes the user groups within the social network. A group should contain people who, between themselves, work together more often than they work with people from outside the group or from other groups. Groups should be densely connected on the inside, while on the outside there should be sparse connections between them. A natural way to measure this property is through the concept of *modularity* [21], which can be computed as:

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \cdot \delta(c_i, c_j) \qquad (2)$$

where the sum is over each arc ij in the network. In the above equation, m is the sum of the weight of all arcs, A_{ij} is the adjacency matrix, k_i is the *degree* of node i (i.e. the sum of all arcs emanating from node i), and $\delta(c_i, c_j) = 1$ if nodes i and j belong to the same cluster, zero otherwise. Modularity has been successfully used to discover communities in different kinds of networks [13, 22].

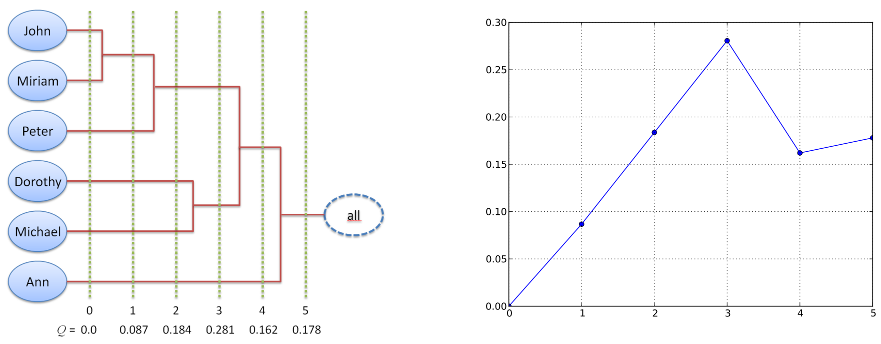


Fig. 5. Dendogram and modularity plot for the small network

In the example from the previous section, the modularity of the initial network is zero since no two nodes belong to the same cluster. After the first iteration, it becomes $Q \simeq 0.087$ and increases to $Q \simeq 0.184$ in the second iteration. It reaches a maximum of $Q \simeq 0.281$ in the third iteration, and in the fourth and fifth it drops to $Q \simeq 0.162$ and $Q \simeq 0.178$, respectively. Figure 5 shows a dendogram and a plot of modularity, where the peak at iteration 3 is clearly visible. The same trend will be observed in practice: modularity keeps increasing monotonically up to a certain point when it reaches a clear maximum, and proceeding further will decrease it. In the small network being used as an example, this means that the best cluster configuration is obtained in iteration 3.

5 Implementation in ProM

We have implemented hierarchical clustering with modularity as a plug-in for the ProM framework [23]. Figure 6 shows a screenshot of the resulting plug-in which, among other features, automatically arranges and colors nodes according to the cluster they belong to. The plug-in is able to build a social network from an event log using either the working together or the similar tasks metric; it is able to perform hierarchical clustering based on single linkage, complete linkage, and average linkage; it is able to decide between several possible merges based on the modularity of the resulting network; and it is able to plot the modularity as well as show the cluster configuration obtained in each iteration.

Internally, the plug-in implements the following algorithm:

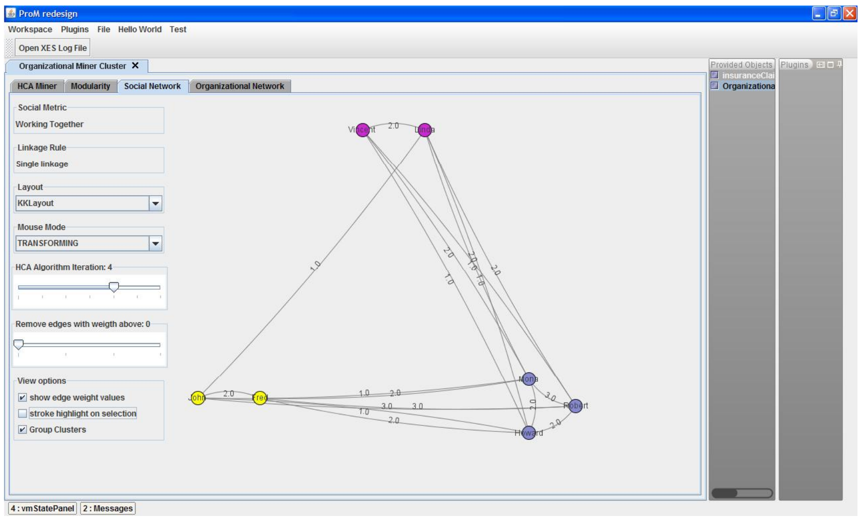

Social Metric: Working Together
Linkage Rule: Single linkage
Layout: KKLayout
Mouse Mode: TRANSFORMING
HCA Algorithm Iteration: 4
Remove edges with weigth above: 0
View options:
- [✓] show edge weight values
- [ ] stroke highlight on selection
- [✓] Group Clusters

Fig. 6. The hierarchical clustering plug-in implemented in ProM version 6

1. Let N be the number of nodes in the initial network. Then N clusters are created, each with one individual node.
2. Calculate the similarity $D(c_r, c_s)$ between every pair of clusters c_r and c_s according to the selected linkage function. For average linkage, the similarity is given by Equation (1) with $d(i,j) = A_{ij}$. For single linkage, $D(c_r, c_s) = \min\{d(i,j)\}$, and for complete linkage, $D(c_r, c_s) = \max\{d(i,j)\}$, where i is a node from c_r and j a node from c_s.
3. Choose the pair of clusters having maximum similarity D. If several pairs have the same maximum similarity, calculate the modularity of the resulting networks in case each of these pairs is merged; then choose the pair that leads to the highest modularity. In the (rare) event that both similarity and modularity are the same, then choose any of those pairs indifferently.
4. Repeat steps 2 and 3 until the network becomes a single cluster with all nodes. After each iteration, store the resulting cluster configuration and the corresponding value of modularity. Modularity can be computed according to Equation (2).

Besides plotting modularity values, the plug-in also allows an inspection of the cluster configuration obtained after each iteration. This way the user can navigate through all the results produced during clustering and analyze them in terms of structure and modularity.

6 Case study

The Hospital of São Sebastião (HSS) is a public hospital with approximately 300 beds, located in *Santa Maria da Feira*, Portugal. The hospital provides several medical specialties, namely Anesthesia, Cardiology, Gastroenterology, Gynecology, Immunology, Internal Medicine, Neurology, Obstetrics, Oncology, Ophthalmology, Orthopedics, Otolaryngology, Pediatrics, Pneumology, Psychiatry, Surgery, and Urology. The hospital also has the facilities to carry out medical exams in many of these specialties, and it has an emergency service running 24x7.

In its daily activity, the hospital makes use of an ERP (Electronic Patient Record) system called Medtrix, which was developed in-house. The system provides an integrated view of all clinical information about each patient, and its database is therefore a valuable source of data to perform process mining and analysis. In this case study, we focused on the organizational perspective and our goal was to capture the structure of work teams that collaborate in the clinical cases that are handled in the emergency service. For this purpose, we had access to three different event logs, as shown in Table 2. In these event logs, each process instance corresponds to a new patient that arrived at the emergency.

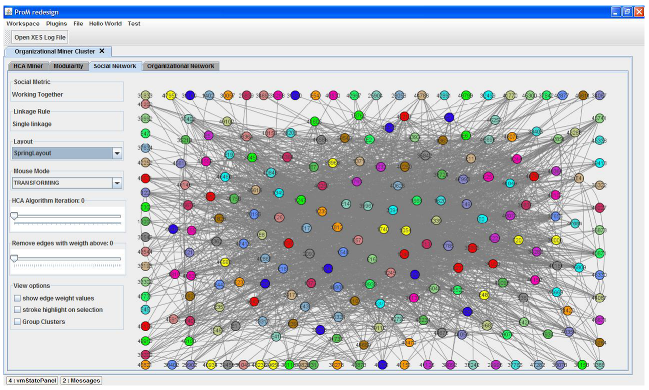
Table 2. Main characteristics of the event logs used in the case study

Time span	No. participants	No. process instances	No. events	No. activities
12 days	131	1868	11506	18
14 days	231	4851	22803	18
6 months	507	78623	536735	21

We conducted several experiments with these event logs, but here we will focus only on the collaboration between medical doctors. Our goal was to discover which specialists work with other specialists, and for that purpose we did an analysis of the social network based on the working together metric. For this analysis, some preprocessing was applied to the event logs, namely: the activities performed by other members of the staff, such as nurses and medical imaging personnel, were excluded; and all process instances (cases) having a single doctor (i.e. doctors working alone) were excluded as well. Even then, the resulting social network was large and difficult to understand, as shown in Figure 7.

We therefore turned to clustering analysis, and the cluster configurations obtained using different linkage functions were all similar to the one presented in Figure 8. Here, each node corresponds to a doctor or group of doctors, and different colors correspond to different medical specialties. From this and with further analysis we were able to draw conclusions about the following specialties:

– *Emergency*: Emergency doctors are the ones who collaborate the most. Although there are several communities for this specialty, they are all interlinked. The size of these communities may reach as much as 30 elements, which represent the largest communities across all specialties. These specialists also work together with doctors from almost all other specialties. Regardless of how the network is clustered, emergency doctors have always one or two communities that play a central role in the network.


ProM redesign
Workspace Plugins File Hello World Test
Open XES Log File
Organizational Miner Cluster
HCA Miner Modularity Social Network Organizational Network
4 : vmStatePanel 2 : Messages

Social Metric: Working Together
Linkage Rule: Single linkage
Layout: SpringLayout
Mouse Mode: TRANSFORMING
HCA Algorithm Iteration: 0
Remove edges with weigth above: 0
View options
- [ ] show edge weight values
- [ ] stroke highlight on selection
- [ ] Group Clusters

Fig. 7. The initial social network before clustering analysis

– *Pediatrics*: This is the specialty showing the second highest tendency to work in group, i.e. pediatrics working with other pediatrics. There are several communities comprising only Pediatrics, and they communicate between them. There is a certain tendency of this specialty to create islands. We may find a single community of pediatrics isolated from the rest of the social network as in Figure 9(a), or we may find a small group of pediatric communities that communicate between them but are isolated from the rest of the network as in Figure 9(b). The size of these communities goes up to 4 elements.
– *Obstetrics/Gynecology*: This specialty is often isolated from other communities, and these specialists also tend to work in isolation between themselves. The size of these communities is typically 1 or 2 elements. Occasionally, these specialists collaborate with emergency doctors.
– *Orthopedics*: Communities in Orthopedics are very rare. The communities that exist contain a single element, and they always appear at the periphery of the network, as shown in Figure 8. The same applies to the remaining specialties.

In this study we were also able to discover that some specialties never work together, such as: Obstetrics/Gynecology with Orthopedics; Obstetrics/Gynecology with Pediatrics; Orthopedics with Pediatrics; General surgery with Pediatrics; General surgery with Orthopedics.

The results above were all obtained from the iteration (of the clustering algorithm) having the highest value of modularity. It is comforting to realize that this concept works as well in large networks as it did in the small network used as example. In effect, in all experiments of this case study, modularity evolved in a similar way to that depicted in Figure 10. Basically, it keeps increasing monotonically up to a certain point when it reaches a maximum, and proceeding further will decrease it noticeably. The iteration with highest modularity then provides the best cluster configuration. It should be noted that modularity can also be used to compare the results obtained using different linkage functions.

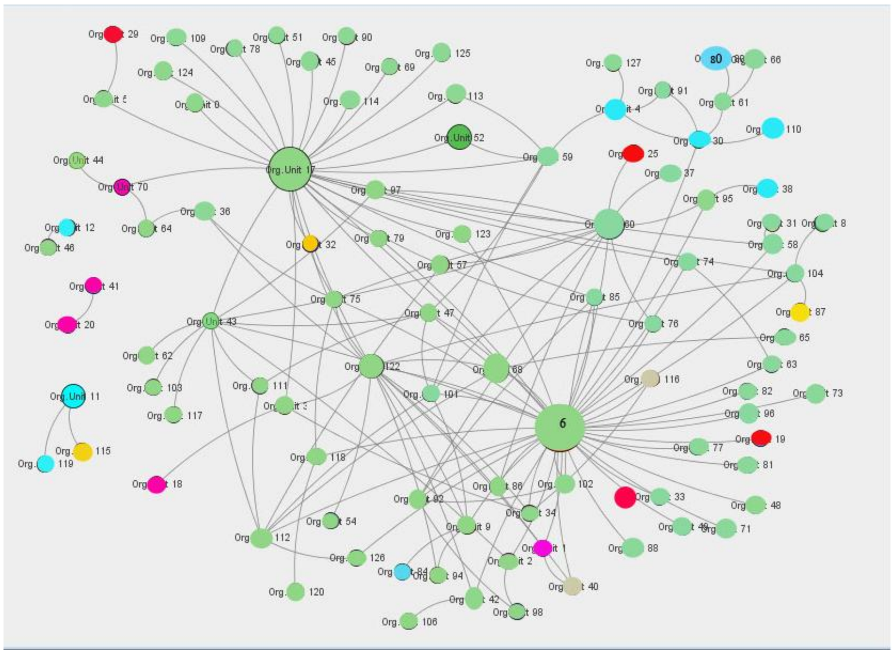
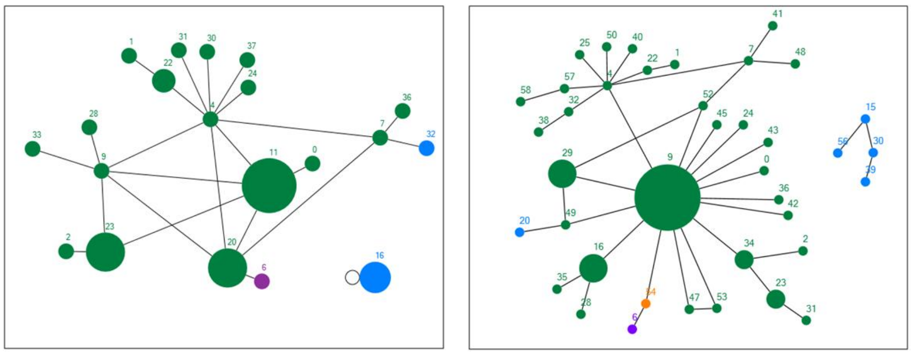
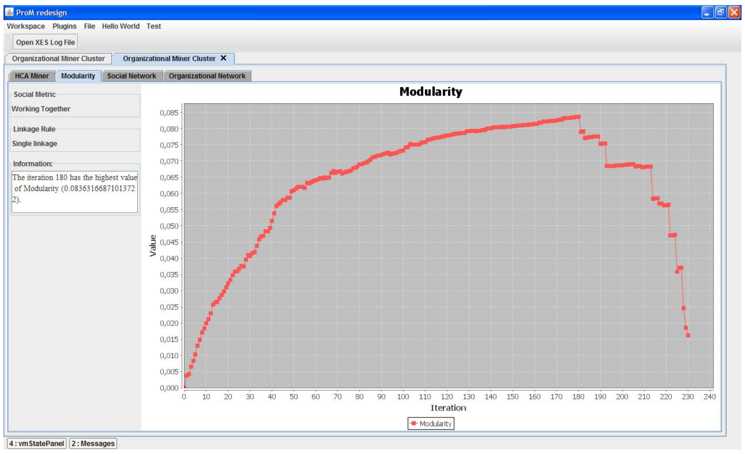


Fig. 8. Results obtained from the event log of 12 days, considering only medical doctors, and using complete linkage (GREEN = Emergency doctors; BLUE = Pediatrics; PINK = Obstetrics/Gynecology, RED = Orthopedics, YELLOW = Emergency relay; DARK PURPLE = General surgery, LIGHT PURPLE = Neurology; GRAY = Internal Medicine)



(a) Single linkage (b) Complete linkage

Fig. 9. Results obtained from the event log of 14 days, considering only medical doctors



Social Metric

Working Together

Linkage Rule

Single linkage

Information:

The iteration 180 has the highest value of Modularity (0.08363166871013722).

Fig. 10. Modularity per iteration of clustering in an experiment from the case study

7 Conclusion

In this paper we described how to use hierarchical clustering together with the concept of modularity to analyze social networks obtained from large event logs. The clustering iteration with the highest value of modularity determines the best division of the social network into a set of clusters, which correspond to the discovered communities. This approach facilitates the analysis of social networks such as the one that we dealt with in the case study.

There are several metrics than can be used to build social networks from event logs, such as similar tasks and working together. In this case study, it was the working together metric that proved more challenging – and useful – to understand the collaboration between a large set of actors. As future work, we plan to support the handover of work metric as well. The approach has been implemented in ProM version 6 and we hope it will be of interest for other researchers and practitioners in the field of process mining.

References

1. van der Aalst, W.M.P., van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G., Weijters, A.J.M.M.: Workflow mining: A survey of issues and approaches. Data and Knowledge Engineering **47**(2) (2003) 237–267
2. Tiwari, A., Turner, C., Majeed, B.: A review of business process mining: state-of-the-art and future trends. Business Process Management Journal **14**(1) (2008) 5–22
3. van der Aalst, W.M.P.: Business alignment: using process mining as a tool for delta analysis and conformance testing. Requirements Engineering **10**(3) (2005) 198–211

4. Hornix, P.: Performance analysis of business processes through process mining. Master's thesis, Eindhoven University of Technology (2007)
5. Rozinat, A., van der Aalst, W.M.P.: Conformance checking of processes based on monitoring real behavior. Information Systems **33**(1) (2008) 64–95
6. van der Aalst, W.M.P., Weijters, A.J.M.M., Maruster, L.: Workflow mining: Discovering process models from event logs. IEEE Transactions on Knowledge and Data Engineering **16** (2004) 1128–1142
7. Weijters, A., van der Aalst, W., de Medeiros, A.A.: Process mining with the heuristics miner algorithm. BETA Working Paper Series WP 166, Eindhoven University of Technology (2006)
8. de Medeiros, A.K.A., Weijters, A.J.M.M.: Genetic process mining. Lecture Notes in Computer Science **3536** (2005) 48–69
9. Günther, C., van der Aalst, W.: Fuzzy mining – adaptive process simplification based on multi-perspective metrics. Lecture Notes in Computer Science **4714** (2007) 328–343
10. van der Aalst, W.M.P., Reijers, H.A., Song, M.: Discovering social networks from event logs. Computer Supported Cooperative Work **14**(6) (2005) 549–593
11. Song, M., van der Aalst, W.M.P.: Towards comprehensive support for organizational mining. Decision Support Systems **46**(1) (2008) 300–317
12. Veiga, G.M., Ferreira, D.R.: Understanding spaghetti models with sequence clustering for ProM. Lecture Notes in Business Information Processing **43** (2010) 92–103
13. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. Proceedings of the National Academy of Sciences of the United States of America **99**(12) (June 2002) 7821–7826
14. van der Aalst, W.M.P., Song, M.: Mining social networks: Uncovering interaction patterns in business processes. Lecture Notes in Computer Science **3080** (2004) 244–260
15. Johnson, S.C.: Hierarchical clustering schemes. Psychometrika **32**(3) (1967) 241–254
16. Murtagh, F.: A survey of recent advances in hierarchical clustering algorithms. Computer Journal **26**(4) (1983) 354–359
17. Lv, T.Y., Su, T.X., Wang, Z.X., Zuo, W.L.: An auto-stopped hierarchical clustering algorithm integrating outlier detection algorithm. Lecture Notes in Computer Science **3739** (2004) 464–474
18. Han, K.J., Narayanan, S.S.: A robust stopping criterion for agglomerative hierarchical clustering in a speaker diarization system. In: Proceedings of InterSpeech, Antwerp, Belgium (2007) 1853–1856
19. Milligan, G.W., Cooper, M.C.: An examination of procedures for determining the number of clusters in a data set. Psychometrika **50**(2) (1985) 159–179
20. Jung, Y., Park, H., Du, D.Z., Drake, B.L.: A decision criterion for the optimal number of clusters in hierarchical clustering. Journal of Global Optimization **25**(1) (2003) 91–111
21. Newman, M.E.J.: Modularity and community structure in networks. Proceedings of the National Academy of Sciences **103**(23) (2006) 8577–8582
22. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Physical Review E **70**(6) (December 2004) 066111
23. van Dongen, B.F., de Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P.: The ProM framework: A new era in process mining tool support. Lecture Notes in Computer Science **3536** (2005) 444–454