Application of Tree-structured Data Mining for Analysis of Process Logs in XML format

Dang Bach Bui **Fedja Hadzic** **Michael Hecker**

Department of Computing
School of Electrical Engineering and Computer Science
Curtin University of Technology,
Email: dang.buibach@postgrad.curtin.edu.au, f.hadzic@curtin.edu.au,
Michael.Hecker@cbs.curtin.edu.au

Abstract

Process logs are increasingly being represented using XML based templates such as *MXML* and *XES*. Popular XML data mining techniques have had limited application to directly mine such data. The majority of work in the process mining field focuses on process discovery and conformance checking tasks often utilizing visualization and simulation based techniques. In this paper, an approach is proposed within which a wider range of data mining methods can be directly applied on tree-structured process log data. Clustering, classification and frequent pattern mining are used as a case in point and experiments are performed on publicly available real-world and synthetic data. The results indicate the great potential of the proposed approach in adding to the available set of methods for process log analysis. It presents an alternative where process model discovery is not the pre-requisite and a variety of methods can be directly applied.

Keywords: process/event log analysis, clustering, frequent subtree mining, classification, XML/MXML/XES mining

1 Introduction and Related Works

A business process is a set of related activities following some logical order whose objective is to create a complete product or service for a customer or a market (Aguilar-Savén 2004). In process-aware information systems, when a business process is executed it leaves traces in an event log (also called process log) of the system (van der Aalst 2011). Cook et al. (Cook & Wolf 1995) was the first to introduce the concept of process mining in the software engineering domain, while Agrawal (Agrawal et al. 1998) generalized the concept to workflow management systems. The purpose is to give valuable information about the processes captured by business information systems. It significantly increases productivity and saves on cost by providing an insight view on business process by different simulation, modeling, analysis and data mining techniques (van der Aalst 2011). The process mining techniques have been applied in many domains e.g. software development, health care, public administration, etc. (van der Aalst 2011)

The aim of process discovery task is to find a model that best describes the workflow of a business process. Many types of models are graphical-based and describe a variety of control flow constructs e.g. sequence, loop, choice, parallel, synchronization. Conformance checking aims to find any difference between the log and the model. Non-conformance may indicate either that the model does not reflect the reality, or deviating process instances which require corrective actions (van der Aalst 2011). Process enhancement analyzes other dimensions of the process log, e.g. actors and activities (work distribution), actors and actors (social network analysis), actor behaviors and decision mining (Rozinat & van der Aalst 2006), to optimize the business process. The majority of research has focused on the process model discovery task or on methods utilizing the process model as the base for analysis. For example, sequential event logs are often analyzed to discover the underlying process model (Günther & Van Der Aalst 2007, Weijters & van der Aalst 2003, van der Aalst et al. 2004, Maruster et al. 2002). Clustering of sequential event logs was used for the pre-processing step in process model discovery in (Bose & van der Aalst 2009, De Medeiros et al. 2008, Greco et al. 2006). Classification learning to predict the next possible event was studied in (Goedertier et al. 2008). The problem of mining frequent patterns of workflow schema executions was introduced in (Greco et al. 2005). The authors presented specialized graph mining algorithms to deal with structural constraints imposed by the workflow schemas and their instances. Please refer to (van der Aalst 2011, Tiwari et al. 2008) for a complete overview of process mining algorithms and techniques.

There is a recent momentum in representing event logs in XML format. The first XML standard created for event log is *MXML* (Günther & van der Aalst 2006) and more recently, *XES* standard was proposed in (Verbeek 2011). Other attempts in using XML to store event logs are presented in (Kim 2006) and (Gonçalves et al. 2002). Semi-structured documents such as XML are known for their ability to represent the contextual information among different data items in a domain specific way. Due to their hierarchical nature XML documents are commonly represented as rooted ordered labeled trees (Hadzic et al. 2011*b*, Zaki 2005). While sequence mining techniques have been applied in the process mining field, to our best knowledge, no tree-structured data mining techniques have been specifically explored for mining of MXML/XES event logs. For example, the frequent subtree mining methods are the basis for discovering interesting associations among tree-structured data objects in XML data, but their utilization in the process mining field is still to be explored. Same holds for other methods that take structural aspects into account during tasks such as XML clustering (Kutty et

al. 2011, Hadzic et al. 2011*a*) and classification (Kim et al. 2010). Note that a synthetic process log dataset was used in (Hadzic et al. 2011*a*), but the main purpose was to compare the time performance and quality of clustering solution between algorithms. The work in (Greco et al. 2005) demonstrated the benefit of applying frequent pattern mining that incorporates workflow schema structure during the analysis of workflow execution. Similarly, the process mining field would benefit by the application of tree-structured data mining techniques, as they not only preserve the order of events but also their context and structural organization within the workflow. In the case of web logs it has already been shown in (Hadzic et al. 2011*b*, Zaki 2005) that a subtree-based pattern is more informative than an itemset or a sequential pattern as it captures the structural properties as well as the navigational behavior over the web site structure. Hence in the case of event/process logs similar reasoning would apply, and a subtree pattern would capture the workflow execution pattern over the overall structure of the business process at hand. The subtree patterns also preserve the context of the events and event attributes within a trace.

The process log is often characterized by repetition of events within a trace and the underlying tree-structures of an XES document can grow quite large and complex which can pose a problem to the performance of tree-structured data analysis. To alleviate the complexity associated with mining complex structures and to enable a wider range of data analysis/mining techniques to be directly applied on tree-structured data, a structure-preserving flat data format of tree-structured data has been recently proposed in (Hadzic et al. 2011*a*, Hadzic 2011). An interesting implication of the method is that the exact positions of nodes/attributes are taken into account during the knowledge discovery process. This property can be useful in process mining as events/actions are distinguished based on their context or exact occurrence within a trace of events. Using this technique as a basis, a general approach is proposed capable of encompassing a broad range of business process aspects during the analysis phase. Decision tree learning, frequent pattern mining and clustering methods are used as case in point and applied to publicly available synthetic and real world data. The results indicate the capability of the approach in discovering interesting descriptive and discriminating characteristics of workflows. A variety of data mining techniques can be utilized within the approach, and applied directly to MXML/XES data to satisfy different application needs. This work extends the available pool of process mining techniques and does not require business process models as the basis for discovery.

The paper is organized as follow. An illustrative scenario that motivates the direction of this work is presented in Section 2. Section 3 describes the proposed approach components of which are tested on real world and synthetic data in Section 4. Section 5 concludes the paper and discusses our future work.

2 Motivation Scenario

In the data mining field an XML document is modeled as a rooted ordered labeled tree, which can be denoted as $T = (v_0, V, L, E)$, where (1) $v_0 \in V$ is the root vertex; (2) V is the set of vertices or nodes; (3) L is a labelling function that assigns a label $L(v)$ to every vertex $v \in V$; (4) $E = (v_1, v_2) \mid v_1, v_2 \in V \; AND \; v_1 \neq v_2$ is the set of edges in the tree, and (4) for each internal node the children are ordered form left to right. Frequent subtree mining (Hadzic et al. 2011*b*, Zaki 2005) is important to enable the discovery of association among tree-structured data object, and several algorithms have been developed that mine different subtree types (Hadzic et al. 2011*b*, Zaki 2005, Tan et al. 2008, Hadzic et al. 2008, Chi et al. 2005, 2004). A closed subtree is a subtree for which none of its proper supertrees has the same support, while for a maximal subtrees, no supertrees exist that are frequent. For a detailed overview of the frequent subtree mining field, please refer to (Hadzic et al. 2011*b*, Chi et al. 2004).

Fig. 1 shows a tree representation of a simplified example extracted from Dutch hospital process log (Bose & Aalst 2012) originally stored in a XES file. No matter what representation of the data is, process mining algorithms such as alpha (van der Aalst et al. 2004), heuristic miners (Weijters & van der Aalst 2003), fuzzy miner (Günther & Van Der Aalst 2007), etc. understand them as a set of sequences of events, and from that discover the generative process model. An example of process model discovered by heuristic algorithm is shown in Fig. 2a. It is noticeable that the process model discovered does not include any information about the attribute values of each event e.g. department information of where the activities were administered. In order to incorporate the contextual information of each event into the mining process, we consider the set of traces as a set of trees rather than as a set of sequences as is commonly done. Using the frequent closed subtree mining (Chi et al. 2005) on the tree database gives us different subtree patterns, each of which occur at different traces. An example of closed subtrees at support = 2 is given Fig. 2b. The subtree at the top occurs in trace (1) and (3) of Fig. 1. These subtree patterns preserve the order of events (according to pre-order traversal of the subtree) and thus represent the frequent path of executions. The frequent paths of execution (*consult* => *administration*) shown in Fig. 2b do not contain the event *phone consult*, *blood test* and *cytologic* because the minimum support value is set to 2; if we lower that parameter longer paths of executions can be detected. The benefit of using subtree mining is that the activities and their contextual information are preserved in the pattern e.g. the path of execution information is enriched with their administering departments e.g. *Radio therapy*, *Obstetrics Lab.*

The order of events within a frequent path of execution is consistent among the matched instances in the database. However, the positional information of those individual events is not indicated in Fig. 2b and in reality there could be other events that occur in between the events belonging to the frequent path execution. The latter case can be observed from our example that the subtree pattern in the lower part of Fig. 2b does not match with trace (4) of Fig. 1 i.e. there is an extra *phone consult* event between the *consult* and *administration* events in the process log. In reality, this result can make us believe that under no circumstances an additional *phone consult* happens after the first *consult* which could cause misjudgments of the process. In general, the position of the captured patterns could be an important lead to many discoveries. Furthermore, process logs in XML format can be quite complex with many nodes in the underlying tree repeating throughout the traces. This can cause combinatorial problems and hinder the analysis due to the large volume of irrelevant patterns captured through frequent subtree. These characteristics of the data and the desired analysis at lower level of detail motivate us to explore the technique recently proposed in (Hadzic 2011) and utilized in (Hadzic et al. 2011*a*) for clustering, which forms the basis of the

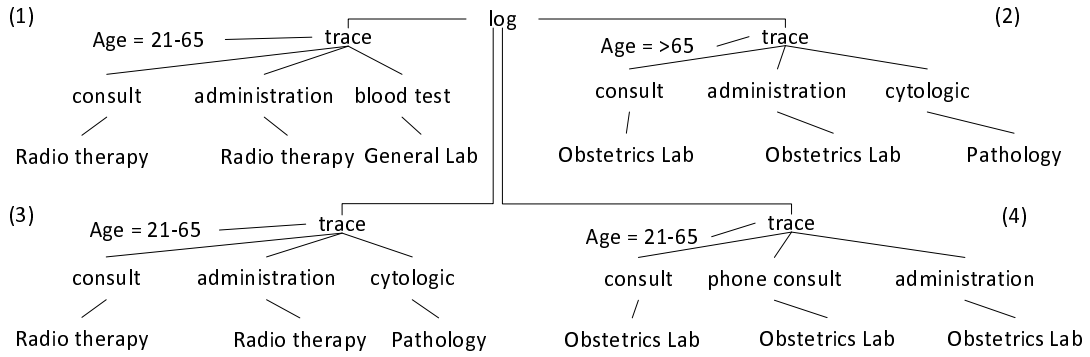

Figure 1: Traces and their elements presented as hierarchical structure

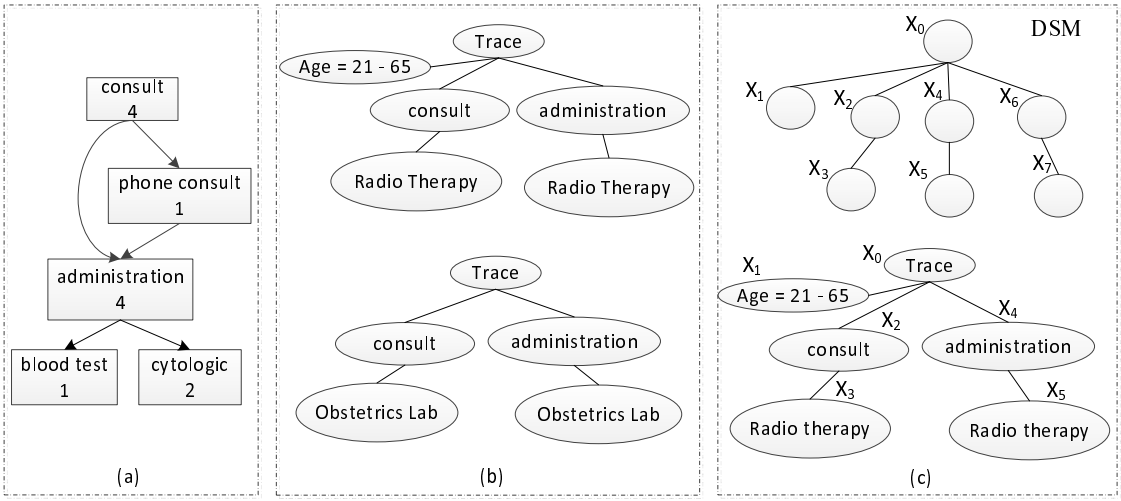

Figure 2: Patterns learned from (a) Heuristic Miner (b) Frequent subtree mining (c) DSM based frequent subtree mining

proposed approach described in the next section.

3 Proposed approach

The proposed approach shown in Fig. 3 consists of four main phases, pre-processing, database structure model (*DSM*) extraction and tree to flat conversion (Hadzic 2011), knowledge discovery and interpretation. Depending on the specific purpose of the process mining task, different pre-processing techniques could be used e.g. grouping/removal, discretization, filtering etc. If the MXML/XES data is not ready, extract transform and load methods can be used (van der Aalst 2011). This semi-structured file is modeled as a set of rooted ordered labeled trees and represented in a pre-order string encoding (Zaki 2005). A pre-order string encoding lists the node labels in the sequence of the pre-order traversal of a tree, and uses a special symbol (e.g. *-1*), when backtracking up the tree. For example, the pre-order string encoding of the trace (1) in Fig. 1 is *'trace', 'Age = 21-65', '-1', 'consult', 'Radio therapy', '-1', '-1', 'administration', 'Radio therapy', '-1', ' -1', 'blood test', 'General Lab', '-1', '-1'*.

The technique (Hadzic 2011) utilized in this approach converts tree-structured data into a flat data structure format (henceforth referred as table) while preserving both structural and attribute-value information. The approach starts by first extracting the *DSM* (Hadzic et al. 2011*a*, Hadzic 2011) of which each tree instance is a valid subtree. This DSM contains the most general structure where every instance from the tree database can be matched to. The DSM tree is shown on top of Fig 2c. The pre-order string encoding of DSM will become the first row of the table with nodes X_i (i corresponds to the pre-order position of the node in the tree) and backtracks b_j (j corresponds to the backtrack number) are used as the attribute names. For each record, when a label is encountered, it is placed to the matching column under the matching node X_i in the DSM structure. When a backtrack (*-1*) is encountered, a valued *1* is placed to the matching backtrack b_j. Remaining entries are assigned a value of *0* (non-existence). The resulting table is called *DSM-Flat*. Table 1 shows the flat representation of the traces in Fig. 1. This conversion process enables the application of frequent pattern mining, clustering, classification and prediction techniques originally developed for vectorial data directly to tree structured process data (knowledge discovery phase in Fig. 3). The discovered knowledge patterns can be re-mapped to tree structure by using the DSM. For example, a frequent subtree mining task can be done by first converting Table 1 into itemset format as shown in Table 2 and then performing the frequent itemset mining. One of the frequent itemsets found at support=2 is *(X_0)'trace', (X_1)'Age=21-65', (X_2)'consult', (X_3)'Radio therapy', (X_4)'administration', (X_5)'Radio therapy'*, which can be mapped into a tree as shown at the bottom of Fig. 2c using the method described in (Hadzic et al. 2011*a*). In comparison to the respective traditional subtree displayed at the top of Fig. 2b, one can see that using the DSM approach, the positional

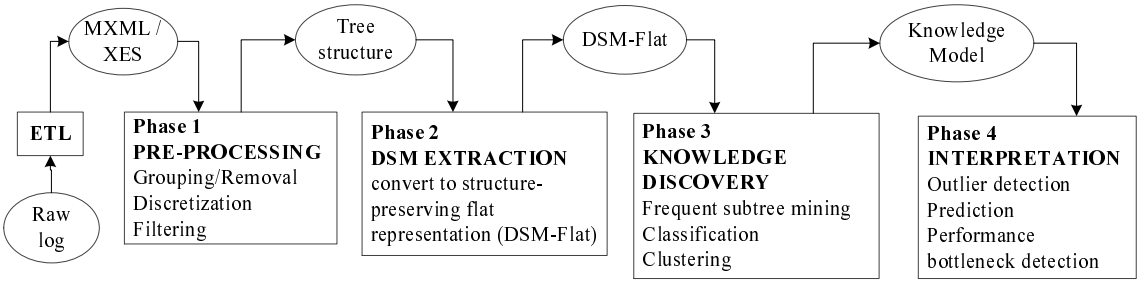

Figure 3: The proposed approach

X_0	X_1	b_0	X_2	X_3	b_1	b_2	X_4	X_5	b_3	b_4	X_6	X_7	b_5	b_6
trace	Age21-65	1	consult	Radio therapy	1	1	administration	Radio therapy	1	1	blood test	General Lab	1	1
trace	Age>65	1	consult	Obstetrics Lab	1	1	administration	Obstetrics Lab	1	1	cytologic	Pathology	1	1
trace	Age21-65	1	consult	Radio therapy	1	1	administration	Radio therapy	1	1	cytologic	Pathology	1	1
trace	Age21-65	1	consult	Obstetrics Lab	1	1	phone consult	Obstetrics Lab	1	1	administration	Obstetrics Lab	1	1

Table 1: Flat representation of tree database

information of each node is indicated. In process mining this interprets to an exact occurrence of an event (or any other aspect of the process) within a trace. This characteristic of distinguishing subtree based upon their exact occurrences would cause the DSM method not to detect the subtree displayed at the bottom of Fig. 2b at support = 2. This is because the right hand side of the subtree (i.e. node *administration* and *Obstetrics Lab*), occurs at different position within trace (2) and (4) of Fig. 1. Hence, the DSM approach would not consider groups of events similar if additional/different events occur within the group (e.g. *phone consult* in trace (4) of Fig. 1). This is important as the analyst can be certain that the DSM based subtree pattern reflects exact similarity of groups of events across traces, where no additional or different events occur in between. Moreover, using the DSM approach, structural complexity associated with mining of complex process logs is avoided by flat representation while structural characteristics of the data are still preserved. For a detailed description of the DSM approach please refer to (Hadzic et al. 2011*a*, Hadzic 2011).

Besides indicating the structural characteristics of the knowledge patterns discovered, in the interpretation phase of the approach, the patterns will be evaluated for their specific use in a given application. For example in outlier/exception detection, the low occurring frequent subtree patterns can indicate characteristics of outlying or exceptional cases. The difference to the more frequently occurring patterns reflecting the norm will be investigated. The instances characterized by such rare patterns may correspond to outliers. In context of clustering, clusters covering only small percentage of instances are suspect of being outliers. The cluster characteristics will be compared to see what the difference is between the outliers and clusters with many instances. The classification methods can be useful for outlier prediction purposes. Hence, once the outlying instances are detected, the instances themselves will be labeled as outlying and others as norm and classification techniques are run to discover a classification model. This knowledge model can then be used to predict the outlying behavior, when a set of preconditions during business process execution path become true. Generally speaking, the classification methods are useful when the process log can be labeled with respect to a particular business need to learn more about a particular business aspect (e.g. duration, performance bottleneck, known cases of exceptions/fraud). This will be demonstrated in the next section when applied on data where labels can be logically assigned.

In the cases where no label can be logically assigned to the process log, we adopt a different approach to enable one to discover different variations in process executions, and learn about descriptive characteristic of each variation, as well as discriminating characteristics with respect to others. This approach is illustrated as a whole in Fig. 4. It starts by performing clustering on the process logs to group process instances which have similar process execution paths. To discover the descriptive characteristics of each cluster we apply frequent pattern mining to detect the common subtree pattern(s) (paths of execution) among instances (step 2a). To detect discriminating characteristics among the clusters, process instances are assigned a virtual cluster label so that classification models can be discovered (step 2b). The results of this process when applied on process log from a Dutch Hospital (Bose & Aalst 2012) are discussed in the next section.

4 Approach Discussion

The approach proposed in the previous section can be better justified when it is compared with other approaches or applied in real environment. However, there are no comparable methods at this moment and we are waiting to apply this approach in practice in order to get feedbacks from domain experts. At this phase, the approach design is motivated as follows.

- Phase 1 (Pre-processing): the process log is originally in semi-structured format. This data should be converted into a tree-structured form in order to enable the direct mining of the contextual and chronological information of the events. This maximizes the potential of the knowledge discovered.
- Phase (DSM extraction): recent tree mining methods are not able to mine the tree database without loosing all or part of the position information of the subtree patterns. Furthermore, the complexity of the frequent subtree pattern searching is high. The DSM extraction phase transform the tree database into a flat representation that enables the direct application of traditional data mining techniques.
- Phase 3 (Knowledge discovery): depending the goals one wants to achieve, different data mining techniques can be utilized at this phase. For

	Itemsets
1	(X_0)trace, (X_1)Age21-65, (X_2)consult, (X_3)Radio therapy, (X_4)administration, (X_5)Radio therapy, (X_6)blood test, (X_7)General Lab
2	(X_0)trace, (X_1)Age>65, (X_2)consult, (X_3)Obstetrics Lab, (X_4)administration, (X_5)Obstetrics Lab, (X_6)cytologic, (X_7)Pathology
3	(X_0)trace, (X_1)Age21-65, (X_2)consult, (X_3)Radio therapy, (X_4)administration, (X_5)Radio therapy, (X_6)cytologic, (X_7)Pathology
4	(X_0)trace, (X_1)Age21-65, (X_2)consult, (X_3)Obstetrics Lab, (X_4)phone consult, (X_5)Obstetrics Lab, (X_6)administration, (X_7)Obstetrics Lab

Table 2: Itemset format

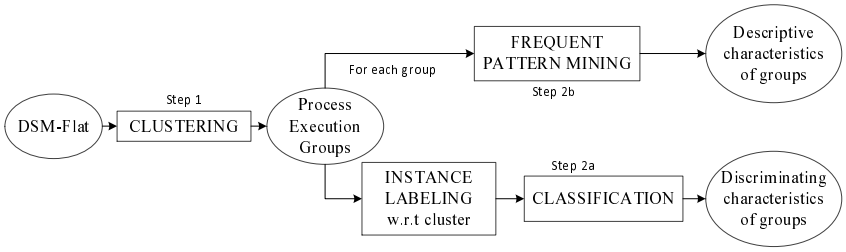

Figure 4: Proposed method for unsupervised process log analysis

example, the process instances can be labeled according to different business requirements and then be used to train classifiers for prediction purposes. One common application is that in case the process analyst is not familiar with the domain of the process log, clustering methods may help identify groups of similar process instances. Frequent pattern mining can then be applied to identify the descriptive characteristics of each group. Classification techniques are able to identify the discriminative characteristics among the groups.

- Phase 4 (Interpretation): in this phase, the results obtained from previous phases should be analyzed and interpreted in a way is understandable and actionable to the domain experts.

5 Experiments and Discussion

The proposed approach is tested on a real life hospital dataset and two synthetic datasets regarding insurance claim and telephone repair. For all experiments we follow the generally proposed four-phase approach Fig. 3). Note that in Phase 2, we do not separate the XML elements, attributes and their values into separate nodes. Therefore in our tree representation of process logs a node label can be a representative of both the element and the element value, while the hierarchical properties of the document are adhered to. The tree mining and frequent item set mining algorithms were run on a Linux machine, Intel Xeon E5345 at 2.33 GHz, 8 GB RAM and 4MB Cache Open SUSE 10.2 64bit. The classification, clustering and DSM extraction/DSM-flat conversion tasks were executed on Windows Server 2008 64-bit machine with 128GB of RAM, quad socket quad core Xeon E7330 (2.4 GHz).

5.1 Hospital dataset

The process log used in this experiment is taken from a Dutch Hospital. Each trace in the input file has a set of attributes e.g. *diagnosis, diagnosis code, treatment, treatment code, start time, end time, age* etc. Each event in a trace includes a set of nine attributes such as: *org:group* (the department in which the activity occurred), *concept:name* (the name of the activity), *time:timestamp* etc. The dataset describes a large variety of processes characterizing different treatments/diagnosis, which themselves are overlapping. These properties of the dataset were described in (Bose & Aalst 2012), with a clear indication that

k	Correlation	Euclidean	Jaccard
3	0.517(0.153)	0.388(0.001)	0.306(0)
4	0.574(0.197)	0.552(0.001)	0.287(0.001)
5	0.586(0.209)	0.467(0.012)	0.452(0.001)
6	0.594(0.213)	0.573(0.01)	0.41(0.002)
7	0.61(0.223)	0.607(0.028)	0.471(0.006)
9	0.653(0.25)	0.637(0.04)	0.5(0.002)

Table 3: Average internal similarity and external similarity of different clustering solutions

one needs to focus on only one aspect/segment of the data, for example a single diagnosis or treatment code. Hence, in this experiment we focus on the set of processes where the describing attribute is a single diagnosis code or treatment code. A subset of data that contains traces of patients who are diagnosed with the code *M13* (most frequent diagnosis) is selected for analysis. For traces that have multiple treatment codes, the codes are concatenated into a single treatment code, while preserving the order of the concatenation across records with same treatment code combination. Other trace attributes such as *specialism code, treatment code combination ID*, etc. are removed as they are either unique for each trace or contained elsewhere in the trace. Event attributes *lifecycle:transition* and *number of executions* are removed as their values are the same across all logs. The structural properties of the subset of the dataset reflecting diagnosis code *M13* are as follows: |transactions|= 252; avg. encoding length = 934.5; max. tree size = 2796; avg. tree height = 2; avg. tree fan-out = 7.4; avg. tree size = 467.7; max tree height = 2; max. tree fan-out = 352. Unlike the approach described in (Bose & Aalst 2012) where the original dataset is preprocessed and examined according to one perspective e.g. time perspective, organizational perspective, urgent and non-urgent case perspective, our approach as described in Fig. 3 and Fig. 4 gears toward a more holistic solution which is more beneficial for users that are not familiar to the domain. In what follows we describe how our method is applied to the hospital dataset.

Discovering process execution groups: it is easier to perform the pattern analysis if the hospital dataset can be split into different homogeneous groups of instances. This is best done in unsupervised way as we do not have specific knowledge about the domain. The hospital data is clustered into different process execution groups. We utilized the DSM based clustering technique proposed in (Hadzic et al. 2011*a*). It converts the data to a structure-preserving flat representation and uses the CLUTO clustering

Cluster	Size	ISim	ESim
0	13	0.999	0.003
1	28	0.88	0.002
2	58	0.552	0.001
3	5	0.488	0
4	37	0.2	0.001
5	74	0.193	0

Table 4: Clustering result at k=4 and Euclidean distance measure is used

toolkit (Karypis 2003) to form clusters. The only difference is in the representation as in (Hadzic et al. 2011*a*) each XML entity (element, attribute and their values) is represented as an individual node, while in this work the attributes and/or values of an XML element are mapped to the same node as the element itself. Please note that in these experiments we do not provide comparisons of the method with other clustering methods for tree-structured data, as extensive comparison on data of varied complexity including a complex synthetic process log, is already provided in (Hadzic 2011). The number of clusters (k) is trial with different values and Euclidean, Jaccard and correlation distance measures are used. Table 3 shows the average internal similarity and external similarity values of all clustering solutions with the latter shown in parentheses. The clustering solution that has the smallest number of k and the largest gap between the average of internal and external similarity is selected for further analysis. The best result is achieved with parameter k=4 using Euclidean distance measure, which produced a clustering solution having six clusters. The detailed internal and external similarity measures (abbreviated as ISim and ESim, respectively) of each cluster are presented in Table 4. It took 9.25s to run the whole process including DSM extraction, DSM-flat conversion and clustering. It can be seen from the table that the top two clusters outperform the remaining clusters in term of the difference between the internal similarity and external similarity.

Discovering discriminating characteristics of groups: groups of similar process instances are identified in the first step in the proposed method. It is useful for a process analyst if he/she can identify the key differences in characteristics of instances among different groups. This can be done by applying classification algorithm on each group; due to their high number of instances and cluster quality, the first three clusters are selected for the classification task. For each cluster, 70% of instances are reserved for the training set and the remainder for the test set. Because the number of instances for each class is different, we apply the oversampling method to balance the examples of each class. The *C4.5* tree induction algorithm (Quinlan 1993) and the *Rapidminer* software(Mierswa et al. 2006) with default parameter settings are used in all classification tasks of this paper. The resulting decision tree is shown in Fig. 5 and it has accuracy of 83.3%. The whole tree induction, applying model and performance evaluation was accomplished in under 1 second. The left most branch of the decision tree shows a rule that if X_2= *Age=retired*, X_{28}=*NO* and X_{19} =*duration=0* then the instance is classified as belonging to cluster 0. Due to the property of the DSM approach of representing tree-structured data, the position of an attribute X_n in a DSM tree can always be inferred. In the hospital dataset, each event has seven attributes, thus if X_4 stores the node *event* of the 1st event of a trace then the node *event* of the 3rd event (if it exists) is located at X_{28} in the DSM tree. With this knowledge in mind, we can interpret the above rule as if the patient is > 65 years old and there are no more than 3 events (e.g. consultations) in the process (the whole treatment) and the duration of the 2nd event is less than 1 day, this patient treatment process belongs to cluster 0.

Discovering descriptive characteristics of groups: each group identified by the clustering algorithm in the first step of the proposed method contains process instances that shares similar characteristics. A frequent pattern mining algorithm can reveal the characteristics that are prevalent among its instances. This can be done either by a frequent closed subtree mining algorithm *CMTreeMiner* or a frequent closed itemset mining algorithm *LCM* (Uno et al. 2004) applied on the flat representation of the tree acquired using DSM method (Hadzic 2011) (abbreviated as LCM-DSM). Both frequent pattern mining minimum supports are set at 90% and the results are shown in Fig. 6 and Fig. 7, respectively. Note that since in our scenario the aim is to discover the patterns that reflect the characteristics of majority of process execution instances within a group, a rather large support threshold is used. For different application aims one could choose lower support thresholds, and this would typically result in more specific patterns characterising smaller subsets of process execution instances within a group.

The subtrees of Fig. 6 indicate some distinguishing characteristics of cluster 0 and 1. For example, cluster 0 is characterized by 2 events, each having producer code of *SGNA* where the 2nd event has attribute *name = administratief* (administration). On the other hand, cluster 1 is characterized by 3 events, the first one having attributes *name = vevolgconsult* (follow up consultation) and *activity code = 411100*. Further, cluster 0 is characterized by *AgeGroup = retired*, and cluster 1 by *AgeGroup = working*. Note however, that from these subtrees one cannot be certain whether any other events occurred between detected common events among the instances and/or whether the additional events differed. On the other hand the subtrees detected using LCM-DSM in Fig. 7 confirms that no additional events occurred between the common events of cluster 0, and that they were the first executed events within the traces. In fact, the DSM based subtrees also indicate how many events in total occurred in the traces of the instances (90% in this case) of the cluster, i.e. 3 events for cluster 0 (X_4, X_{12} and X_{20}) and 4 events for cluster 1 (X_4, X_{12}, X_{20} and X_{28}). Furthermore the attribute *name = administratief* in the 2nd event of traditional subtree of cluster 0 in Fig. 6 did not occur in DSM-based subtree of cluster 0 in Fig. 7, which also indicates that this attribute was not frequent in the 2nd event but was frequent when its occurrence in the 2nd and 3rd event was counted together. We have confirmed this with the instances of cluster 0 and have found that the attribute *name = administratief* occurs 7 times in the 2nd event and 6 times in the 3rd event. Another difference is that the attributes *name = vevolgconsult* and *activity code = 411100* did not occur in the DSM based subtree of cluster 1. The DSM subtree of cluster 1 indicates that there was another event (at X_4) before the event at node X_{12} (i.e. the 2nd event in the traces). We have inspected the instances of cluster 1 and have found that the association of *name = vevolgconsult* and *activity code = 411100* occurs 18 times in 1st event and 7 times in 2nd event.

These differences indicate the benefit of DSM in cases when we would like to find the exact location of each repeated or outlying values, or know the exact

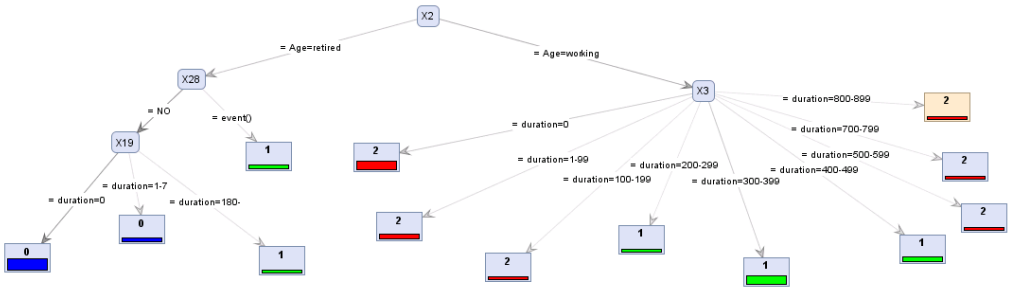

Figure 5: Decision tree learned from three clusters of process instances

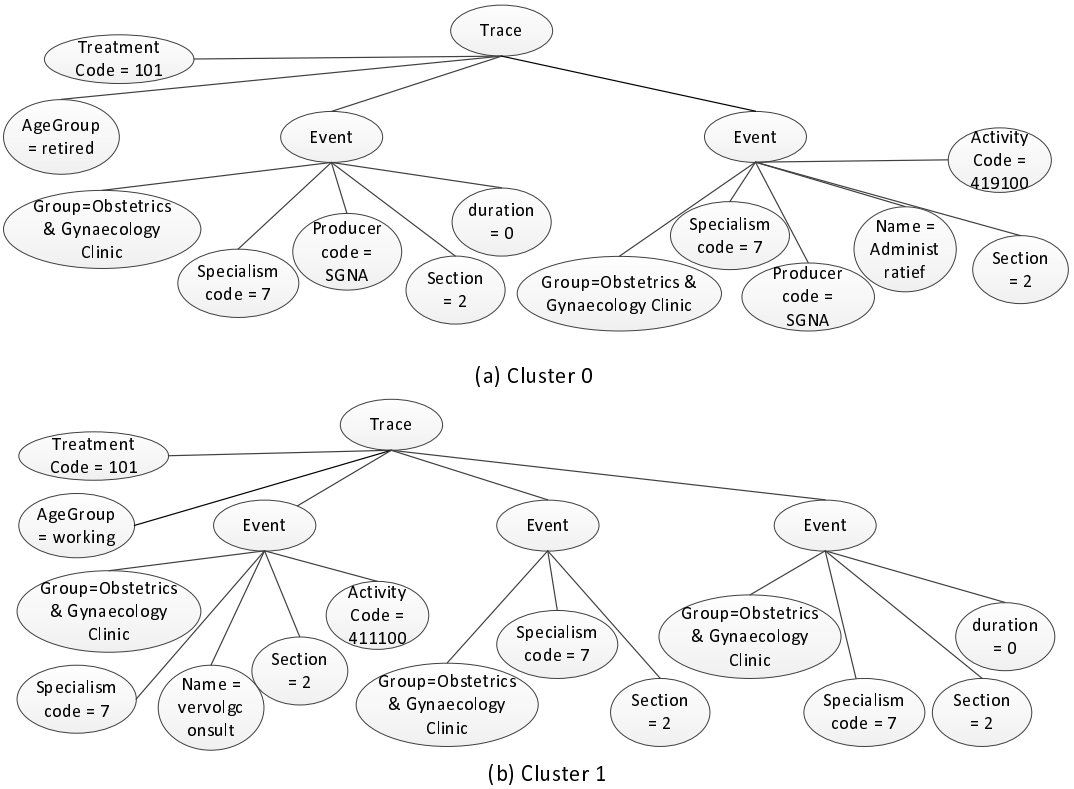

Figure 6: Frequent subtrees identified by CMTreeMiner

occurrence of an event and its characteristics within the trace as a whole. However, we do not claim that the traditional subtree mining would not be useful, but that the different approaches each have its own useful characteristics. For example the traditional subtrees reflect the occurrences of events and their characteristics no matter in which part of the trace they occurred, while the DSM based subtrees provide further detail of their exact location within a trace, and both could be used in a complementary way to obtain a more comprehensive analysis.

5.2 Insurance dataset

The synthetic insurance process log describes the handling of claims in an insurance company. The log contains 46138 events related to 3512 cases (claims). One typical process is that the customers file the claims, the center checks information, registers to the system, the claim is then quickly checked by a claim handler, after that it is fully examined; the officer advises the claimant and starts payment; finally the claim is closed. The purpose of this experiment is to build a classification model to identify four possible outcomes of a claim such as *processed*, *rejected*, *insufficient information* or *not liable*. Each trace is first labeled from one of the four values as described above. The structural properties of the tree database are as follows |transactions|= 3512; avg. length of encoding = 114.1; max. tree size = 73; avg. height of trees = 2; avg. fan-out of trees= 4; avg. size of trees = 57.5; max height of trees = 2; max. fan-out of trees = 19. The decision tree model discovered from the balanced dataset is displayed in Fig. 8. The running time is 2 seconds and has accuracy of 100% evaluated using ten-fold cross-validation.

Fig. 8 shows that if activity *end* happens at the 7th event (at position X_{35}), the claim is identified as *not liable*; otherwise the claim would be *insufficient information*. Furthermore, at position X_{53}, we know that if an eleventh event is available the claim is *processed*, otherwise it is *rejected*. This indicates another useful property of the DSM-Flat representation, as specific points of difference between events of traces of different class can be directly detected rather than manually searching for the differences within the often

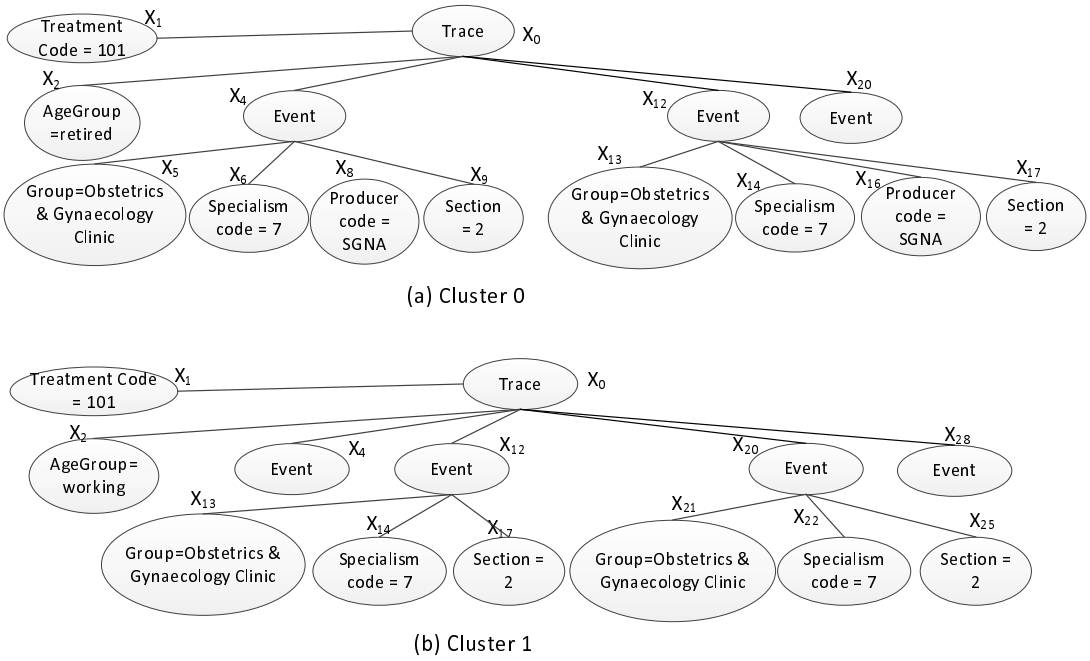

Figure 7: Frequent subtrees identified by DSM-LCM method

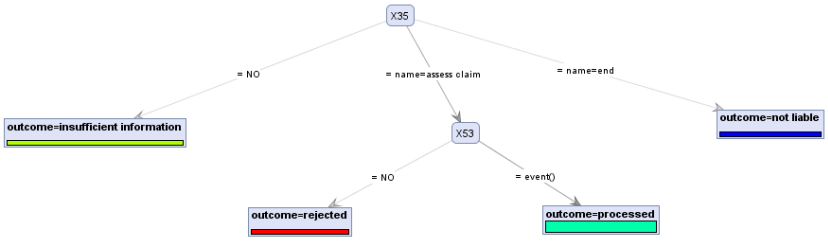

Figure 8: Decision tree for outcome of a process instance

large frequent pattern set in case of frequent subtree mining application.

5.3 Telephone repair dataset

The synthetic telephone dataset describes artificial logs of telephone repair processes. One example of a process instance starts by a customer registering a telephone for repair; the telephone is analysed; then transferred to either simple repair team or complex repair team; at the same time the customer is informed of the condition of their device; once the telephone is repaired it is tested; if not fixed it is then sent back for repair; the case is archived after the telephone is fixed. In this dataset we try to predict (1) the time needed to repair each telephone (class values were *0*, *1* and *2* hours) and (2) the complexity of the repair which can be *simple*, *complex* or *both*. Note that the number of attributes in each event of this dataset was the same but some of them could differ in order. Hence, in this case they were first sorted in the same order to ensure that columns in the flat representation contain values of the same attribute. The structural properties of the tree database are as follows: |transactions|= 1104; avg. encoding length = 106; max. tree size = 112; avg. tree height = 2; avg. tree fan-out = 4.5; avg. tree size = 53.5; max tree height = 2; max. tree fan-out = 27.

Prediction of duration: The resulting C4.5 decision tree model (took 6s to build and evaluate) has the size of 69 with 58 leaf nodes and accuracy of 82.7% evaluated using ten-fold cross-validation. Due to its large size, the decision tree is not shown here. Inspecting the rules of the decision tree shows that the duration of a process is classified as 0 if $X_{90} = X_{67} = X_{41}$ =NO. Note that X_{90} contains the 4th attribute (*number of repair*) of the 8th event, X_{67} contains the the 5th attribute (*defect fixed*) of the 13th event and X_{41} contains the 5th attribute (*defect fixed*) of the 8th event node. X_{27}, which corresponds to the 1st attribute (*resource*) of the 6th event, names the officer responsible for the repair at that stage. From the decision model, we observe that officer *SolverC3* performed poorly as most of his/her tasks are completed in one hour while others finish in less than one hour. While this is a synthetic dataset, this kind of analysis is also useful for detecting performance bottlenecks and investigating into resource optimization.

Prediction of complex process: a trace is labeled *simple* (or *complex*) if it contains any *simple* (or *complex*) repair, as part of the descriptive attributes of actions within an event. If both *simple* and *complex* repairs exist then the trace is labeled *both*. The resulting decision tree for the three-class classification evaluated using ten-fold cross-validation is displayed in Fig. 9. It took 2 seconds to build and evaluate the model and the classification accuracy is at 92.94

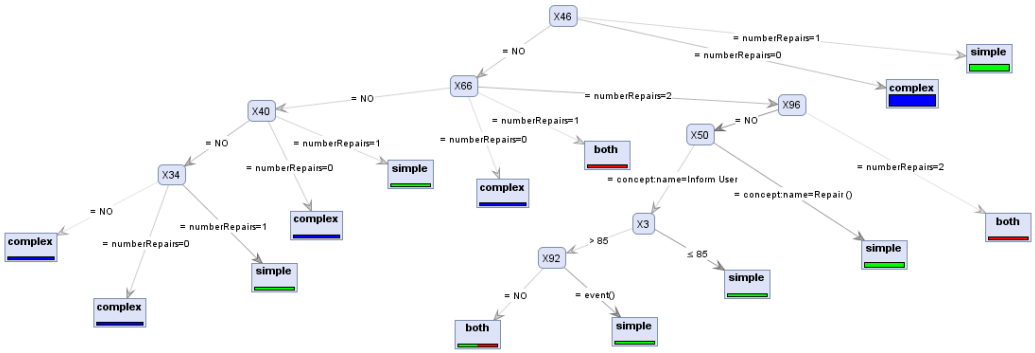

Figure 9: Decision tree for complexity prediction of a process instance

6 Conclusion and Future Work

Providing that process logs are increasingly available in XML format, this paper introduces an approach for direct application of a wide range of data mining/analysis methods to tree structured process logs. In the experiments using two synthetic XES datasets, decision tree learning is applied to directly detect all discriminating criteria. For the experiment on real world hospital process log, a combination of clustering, frequent pattern mining and classification techniques is used. We demonstrate how using the proposed approach one can detect and characterize different groups representing different process executions within the business, as well as detect the discriminating characteristics between the different groups. We have also illustrated some important differences and implications for process log analysis between subtrees extracted using the traditional subtree mining approach and the DSM approach which takes the node position into account. In our future work we will explore the use of the proposed approach for exception detection/analysis/prediction, model conformance checking, and in general discovery of patterns encompassing broad aspects of processes.

References

Agrawal, R., Gunopulos, D. & Leymann, F. (1998), Mining process models from workflow logs, *in* 'Proceedings of the 6th International Conference on Extending Database Technology: Advances in Database Technology', EDBT '98, Springer-Verlag, London, UK, UK, pp. 469–483.

Aguilar-Savén, R. S. (2004), 'Business process modelling: Review and framework', *International Journal of Production Economics* **90**(2), 129 – 149.

Bose, R. P. J. C. & Aalst, W. M. P. (2012), Analysis of patient treatment procedures, *in* F. Daniel, K. Barkaoui, S. Dustdar, W. Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw & C. Szyperski, eds, 'Business Process Management Workshops', Vol. 99 of *Lecture Notes in Business Information Processing*, Springer Berlin Heidelberg, pp. 165–166.

Bose, R. P. J. C. & van der Aalst, W. M. P. (2009), Context aware trace clustering: Towards improving process mining results., *in* 'SDM', SIAM, pp. 401–412.

Chi, Y., Muntz, R. R., Nijssen, S. & Kok, J. N. (2004), 'Frequent subtree mining - an overview', *Fundam. Inf.* **66**(1-2), 161–198.

Chi, Y., Xia, Y., Yang, Y. & R. Muntz, R. (2005), 'Mining closed and maximal frequent subtrees from databases of labeled rooted trees', *IEEE Trans. on Knowl. and Data Eng.* **17**(2), 190–202.

Cook, J. E. & Wolf, A. L. (1995), Automating process discovery through event-data analysis, *in* 'Proceedings of the 17th international conference on Software engineering', ICSE '95, ACM, New York, NY, USA, pp. 73–82.

De Medeiros, A. K. A., Guzzo, A., Greco, G., Van Der Aalst, W. M. P., Weijters, A. J. M. M., Van Dongen, B. F. & Saccà, D. (2008), Process mining based on clustering: a quest for precision, *in* 'Proceedings of the 2007 international conference on Business process management', BPM'07, Springer-Verlag, Berlin, Heidelberg, pp. 17–29.

Goedertier, S., Martens, D., Baesens, B., Haesen, R. & Vanthienen, J. (2008), Process mining as first-order classification learning on logs with negative events, *in* 'Proceedings of the 2007 international conference on Business process management', BPM'07, Springer-Verlag, Berlin, Heidelberg, pp. 42–53.

Gonçalves, M. A., Luo, M., Shen, R., Ali, M. F. & Fox, E. A. (2002), An xml log standard and tool for digital library logging analysis, *in* 'Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries', ECDL '02, Springer-Verlag, London, UK, UK, pp. 129–143.

Greco, G., Guzzo, A., Manco, G. & Sacca, D. (2005), 'Mining and reasoning on workflows', *IEEE Trans. on Knowl. and Data Eng.* **17**(4), 519–534.

Greco, G., Guzzo, A., Pontieri, L. & Sacca, D. (2006), 'Discovering expressive process models by clustering log traces', *IEEE Trans. on Knowl. and Data Eng.* **18**(8), 1010–1027.

Günther, C. W. & van der Aalst, W. M. P. (2006), A generic import framework for process event logs, *in* 'Proceedings of the 2006 international conference on Business Process Management Workshops', BPM'06, Springer-Verlag, Berlin, Heidelberg, pp. 81–92.

Günther, C. W. & Van Der Aalst, W. M. P. (2007), Fuzzy mining: adaptive process simplification based on multi-perspective metrics, *in* 'Proceedings of the 5th international conference on Business process management', BPM'07, Springer-Verlag, Berlin, Heidelberg, pp. 328–343.

Hadzic, F. (2011), A structure preserving flat data format representation for tree-structured data, *in* 'PAKDD QIMIE Workshop', Shenzhen, China, pp. 221–233.

Hadzic, F., Hecker, M. & Tagarelli, A. (2011*a*), Xml document clustering using structure-preserving flat representation of xml content and structure, *in* 'Proceedings of the 7th international conference on Advanced Data Mining and Applications - Volume Part II', ADMA'11, Springer-Verlag, Berlin, Heidelberg, pp. 403–416.

Hadzic, F., Tan, H. & Dillon, S. T. (2011*b*), *Mining of Data with Complex Structures*, Springer.

Hadzic, F., Tan, H. & Dillon, T. (2008), Mining unordered distance-constrained embedded subtrees, *in* 'Proceedings of the 11th International Conference on Discovery Science', DS '08, Springer-Verlag, Berlin, Heidelberg, pp. 272–283.

Karypis, G. (2003), Cluto: A clustering toolkit, Technical report, University of Minnesota.

Kim, H., Kim, S., Weninger, T., Han, J. & Abdelzaher, T. (2010), Ndpmine: efficiently mining discriminative numerical features for pattern-based classification, *in* 'Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part II', ECML PKDD'10, Springer-Verlag, Berlin, Heidelberg, pp. 35–50.

Kim, K. (2006), A xml-based workflow event logging mechanism for workflow mining, *in* 'Proceedings of the 2006 international conference on Advanced Web and Network Technologies, and Applications', APWeb'06, Springer-Verlag, Berlin, Heidelberg, pp. 132–136.

Kutty, S., Nayak, R. & Li, Y. (2011), Xml documents clustering using a tensor space model, *in* 'Proceedings of the 15th Pacific-Asia conference on Advances in knowledge discovery and data mining - Volume Part I', PAKDD'11, Springer-Verlag, Berlin, Heidelberg, pp. 488–499.

Maruster, L., Weijters, A. J. M. M., Aalst, W. M. P. v. d. & Bosch, A. v. d. (2002), Process mining: Discovering direct successors in process logs, *in* 'Proceedings of the 5th International Conference on Discovery Science', DS '02, Springer-Verlag, London, UK, UK, pp. 364–373.

Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M. & Euler, T. (2006), Yale: rapid prototyping for complex data mining tasks, *in* 'Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining', KDD '06, ACM, New York, NY, USA, pp. 935–940.

Quinlan, J. (1993), *Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA.

Rozinat, A. & van der Aalst, W. M. P. (2006), Decision mining in prom, *in* 'Proceedings of the 4th international conference on Business Process Management', BPM'06, Springer-Verlag, Berlin, Heidelberg, pp. 420–425.

Tan, H., Hadzic, F., Dillon, T. S., Chang, E. & Feng, L. (2008), 'Tree model guided candidate generation for mining frequent subtrees from xml documents', *ACM Trans. Knowl. Discov. Data* **2**(2), 9:1–9:43.

Tiwari, A., Turner, C. J. & Majeed, B. (2008), 'A review of business process mining: state-of-the-art and future trends', *Business Process Management Journal* **14**, 5–22.

Uno, T., Kiyomi, M. & Arimura, H. (2004), LCM ver.2: Efficient mining algorithms for Frequent/Closed/maximal itemsets, *in* 'Proc. 1st Int'l workshop on open source data mining: frequent pattern mining implementations'.

van der Aalst, W. M. P. (2011), *Process Mining - Discovery, Conformance and Enhancement of Business Processes*, Springer.

van der Aalst, W., Weijters, T. & Maruster, L. (2004), 'Workflow mining: Discovering process models from event logs', *IEEE Trans. on Knowl. and Data Eng.* **16**(9), 1128–1142.

Verbeek, H.M.W., B. J. D. B. v. . A. W. v. d. (2011), Xes, xesame, and prom 6, *in* E. E. Soffer, P. & Proper, ed., 'Information Systems Evolution (CAiSE Forum 2010)', Vol. 72, Springer Berlin, pp. 60–75.

Weijters, A. J. M. M. & van der Aalst, W. M. P. (2003), 'Rediscovering workflow models from event-based data using little thumb', *Integr. Comput.-Aided Eng.* **10**(2), 151–162.

Zaki, M. J. (2005), 'Efficiently mining frequent trees in a forest: Algorithms and applications', *in IEEE Transaction on Knowledge and Data Engineering* **17**, 1021–1035.