ISSN 2320 - 2602
Volume 2, No.9, August 2013
International Journal of Advances in Computer Science and Technology
Available Online at http://warse.org/pdfs/2013/ijacst02292013.pdf

SCHEDULING JOB AND WORK FLOW MINING USING NOVEL BASED GROUP MOVEMENT PATTERNS

K.Jeyalakshmi[1], **B.Amuthajanaki**[2]
[1]Assistant Professor, Department of Computer Science,
Hindusthan college of arts & science, Coimbatore, India, jay_lakme@yahoo.com
[2]Research Scholar, Department of Computer Science, Hindusthan College of Arts and Science,
Coimbatore, India, janakikks@gmail.com

ABSTRACT

The main objective of this paper is to reduce the work of the software developers during and in work allocation. Here the work is allotted for each person automatically by splitting the modules into tasks. When developing a big project, more confusion and more discussions may occur, and some clauses may arise between modules while splitting into tasks the person who develops the particular task and performance evolution may not be in accurate, and confuses the actual work allotted by the HR. So the companies struggle to book multiple software projects Simultaneously. So that companies are yielding less works from their employees. This paper is introduced, in order to reduce these problems. Here all the modules will be given as input and it will automatically indicate the number of persons be needed to complete the project and time it needs to complete a particular module. Thus the user can avoid all confusions by using these methods.

Keywords: Tagging, Collaboration, Work items, Subset selection, Filter method, Learning to rank, Workflow management, Graph-based clustering.

1. INTRODUCTION

Our study shows that during last decade, the workflow management concepts and technology have been applied in many enterprise information systems and even in small scale software industries. Workflow management systems such as Staffware, IBM MQSeries, COSA, etc., offer generic modelling and method capabilities for structured business integration processes. By making graphical process definitions, i.e., models describing the entire life cycle of a typical case in isolation, anyone can configure these systems to support business processes. And also pure workflow management systems, many other software systems have adopted workflow technology from various existing methods. Consider, for example, ERP (Enterprise Resource Planning) systems such as SAP, PeopleSoft, Baan and Oracle, CRM (Customer Relationship Management) software, etc. One of the problems is that these systems require a workflow design and a work flow model along with job scheduling, i.e., a developer has to construct a detailed model accurately describing the routing of work. Modelling a workflow is far from trivial: It requires deep knowledge of the workflow language and lengthy discussions with the workers and management involved in the organisation. Instead of starting with a workflow design, we start by gathering information about the workflow processes as they take place in the organisation. We assume that it is possible to record events such that

1. Each modules refers to a task (i.e., a well-defined step in the workflow),

2. Each task refers to a case (i.e., a workflow instance), And

3. Cases are totally ordered (i.e., in the log events are recorded sequentially, even though tasks may be executed in parallel).

Any information system using transactional systems such as ERP, CRM, or workflow management systems will offer this information in some form it may said as the output. Note that they do not assume the presence of a workflow management system. The only assumption the user make is that it is possible to collect workflow logs with event data. These workflow logs are used to construct a process specification which adequately models the behaviour registered. The user use the term process mining for the method of distilling a structured process description from a set of real executions.

2. RELATED WORKS

The idea of process mining is not new [7], [9], [10], [11], [12], [14], [15], Cook and Wolf have investigated similar issues in the context of software engineering processes. In [1], they describe three methods for process discovery: one using neural networks, one using a purely algorithmic approach, and one Markovian approach. The authors consider the latter two the most promising approaches. The purely algorithmic approach builds a finite state machine where states are fused if their futures (in terms of possible behavior in the next k steps) are identical. The Markovian approach uses a mixture of algorithmic and statistical methods and is able to deal with noise. Note that the results presented in [4] are limited to sequential behaviour. Related, but in a different domain, is the work presented in [2], [3], also using a Markovian approach restricted to sequential processes. Cook and Wolf extend their work to concurrent processes in [10]. They propose specific metrics (entropy, event type counts, periodicity, and causality) and use these metrics to discover models out of event streams. However, they do not provide an approach to generate explicit process models. Recall that the final goal of the approach presented in this paper is to find explicit representations for a broad range of process models, i.e., we want to be able to generate a concrete Petri net rather than a set of dependency relations between events. In [5], Cook and Wolf provide a measure to quantify discrepancies between a process model and the actual behaviour as registered using event-based data. The idea of applying process mining in the context of workflow management was first introduced in [6]. This work is based on workflow graphs, which are inspired by workflow products such as IBM MQSeries workflow (formerly known as Flowmark) and In Concert.

3. WORKFLOW MINING

In this section, the rediscovery problem is tackled. Before the user present a mining algorithm, He must be able to rediscover a large class of sound WF-nets, the user investigates the relation between the causal relations detected in the log (i.e., >W) and the presence of places connecting transitions. First, we show that causal relations in >W imply the presence of places. Then, we explore the class of nets for which the reverse also holds. Based on these observations, we present a mining algorithm.

4. MINING ALGORITHM

Based on the results in the previous sections, we now present an algorithm for mining processes. The algorithm uses the fact that for many WF-nets, two tasks are connected iff their causality can be detected by inspecting the log.

Let W be a workflow log over T. is defined as follows:

1. $T_W = \{t \in T \mid \exists_{\sigma \in W} t \in \sigma\}$,
2. $T_I = \{t \in T \mid \exists_{\sigma \in W} t = first(\sigma)\}$,
3. $T_O = \{t \in T \mid \exists_{\sigma \in W} t = last(\sigma)\}$,
4.

$$X_W = \{(A,B) \mid A \subseteq T_W \ \wedge B \subseteq T_W \ \wedge \forall_{a \in A} \forall_{b \in B} a \rightarrow_W b \ \wedge \forall_{a_1,a_2 \in A} a_1 \#_W a_2 \ \wedge \forall_{b_1,b_2 \in B} b_1 \#_W b_2\},$$

5.

$$Y_W = \{(A,B) \in X_W \mid \forall_{(A',B') \in X_W} A \subseteq A' \ \wedge B \subseteq B' \Longrightarrow (A,B) = (A',B')\},$$

6. $P_W = \{p_{(A,B)} \mid \ (A,B) \in Y_W\} \cup \{i_W, o_W\}$,
7.

$$F_W = \{(a, p_{(A,B)}) \mid (A,B) \in Y_W \ \wedge \ a \in A\} \cup \ \{(p_{(A,B)}, b) \mid (A,B) \in Y_W \ \wedge \ b \in B\} \cup \{(i_W, t) \mid t \in T_I\} \ \cup \{(t, o_W) \mid t \in T_O\},$$

and

8. $\alpha(W) = (P_W, T_W, F_W)$.

The mining algorithm constructs a net ð PW ; TW ; FW Þ. Clearly, the set of transitions TW can be derived by inspecting the log. If there are no traces of length one, TW can be derived from >W. Since it is possible to find all initial transitions TI and all final transition TO, it is easy to construct the connections between these transitions and iW and oW. Besides the source place iW and the sink place oW, places of the form p ð A;B Þ are added. For such place, the subscript refers to the set of input and output transitions, i.e.,p ð A;B Þ ¼ A and p ð A;BÞ¼B.A place is added in-between a and b iff a >Wb. However, some of these places need to be merged in case of OR-splits/joins rather than AND-splits/joins. For this purpose, the relations XW and YW are constructed .ðA;B Þ 2XW if there is a causal relation from each member of A to each member of B and the members of A and B never occur next to one another. Note that, if a!Wb, b!W a,or a k Wb, then a and b cannot be both in A(or B). Relation [13] YW is derived from XW by taking only the largest elements with respect to set inclusion.

5. WORKFLOW NETS

Most workflow systems offer standard building blocks such as the AND-split, AND-join, OR-split, and OR-join. These are used to model sequential, conditional, Parallel, and iterative routing. Clearly, a Petri net can be used to specify the routing of cases. Tasks are modelled by transitions and causal dependencies are modelled by places and arcs. In fact, a place corresponds to a condition which can be used as pre and/or post condition for tasks. An AND-split corresponds to a transition with two or more output places, and an AND-join corresponds to a transition with two or more input places. OR-splits/OR-joins correspond to places with multiple outgoing/ingoing arcs. Given the close relation between tasks and transitions, we use the terms interchangeably. A Petri net which models the control-flow dimension of a workflow is called workflow net (WF-net). It should be noted that a WF-net specifies the dynamic behaviour of a single case in isolation

6. CONCLUSION

We have applied our workflow mining techniques to two real applications. The first application is in small scale software industry where the flow of multi disciplinary modules is analyzed. We have analyzed workflow logs of existing modules with the current modules. We have preliminary results showing that process mining is very difficult given the nature of this process. Only by focusing on specific tasks and abstracting from infrequent tasks are we able to successfully mine such processes. The second application concerns the processing of fines by the CJIB (Centraal Justitieel Incasso Bureau), the Dutch Judicial Collection Agency located in Leeuwarden. We have successfully mined the process using information from 130,136 cases. The process comprises 99 tasks and has been validated by the CJIB. This positive result shows that process mining based on thealgorithm and using tools like EMiT and Little Thumb is feasible for at least structured processes. These findings are encouraging and show the potential of the algorithm presented in this paper.

REFERENCES

[1] W.M.P. van der Aalst, "**Verification of Workflow Nets**," Application and Theory of Petri Nets,P. Azema and G. Balbo, eds., pp. 407-426, Berlin: Springer-Verlag, 1997.

[2] W.M.P. van der Aalst, "**The Application of Petri Nets to Workflow Management**, "The J. Circuits, Systems and Computers, vol. 8, no. 1, pp. 21-66, 1998.

[3] "**Business Process Management: Models, Techniques, and Empirical Studies**, "Lecture Notes in Computer Science, W.M.P. van der Aalst, J. Desel, and A. Oberweis, eds., vol. 1806, Springer-Verlag, Berlin, 2010.

[4] W.M.P. van der Aalst and B.F. van Dongen, "**Discovering Workflow Performance Models from Timed Logs**, "Proc. Int'l Conf. Eng. and Deployment of Cooperative Information Systems (EDCIS 2002), Y. Han, S. Tai, and D. Wikarski, eds., vol. 2480, pp. 45-63, 2012.

[5] W.M.P. van der Aalst and K.M. van Hee, **Workflow Management: Models, Methods, and Systems**. Cambridge, Mass.: MIT Press, 2007.

[6] W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster, "**Workflow Mining: Which Processes can be Re discovered?**" BETA Working Paper Series, WP 74, Eindhoven Univ. Of Technology, Eindhoven, 2009.

[7] R. Agrawal, D. Gunopulos, and F. Leymann, "**Mining Process Models from Workflow Logs**, "Proc. Sixth Int'l Conf. Extending Database Technology, pp. 469-483, 1998.

[8] D. Angluin and C.H. Smith, "**Inductive Inference: Theory and Methods**, "Computing Surveys, *vol. 15, no. 3, pp. 237-269*, 1983.

[9] J.E. Cook and A.L. Wolf, "**Discovering Models of Software Processes from Event-Based Data, "ACM Trans. Software Eng. and Methodology**, *vol. 7, no. 3, pp. 215-249*, 1998.

[10] J.E. Cook and A.L. Wolf, "**Event-Based Detection of Concurrency**," Proc. Sixth Int'l Symp. the Foundations of Software Eng. (FSE-6), pp. 35-45, 1998.

[11] J.E. Cook and A.L. Wolf, "**Software Process Validation: Quantitatively Measuring the Correspondence of a Process to a Model**," ACM Trans. Software Eng. and Methodology, *vol. 8, no. 2, pp. 147-176*, 1999.

[12] J. Desel and J. Esparza, "**Free Choice Petri Nets**, "Cambridge Tracts in Theoretical Computer Science, vol. 40, Cambridge, UK: Cambridge Univ. Press, 1995.

[13] J. Eder and G.E. Olivotto, and W. Gruber, "**A Data Warehouse for Workflow Logs**, "Proc. Int'l Conf. Eng. and Deployment of Cooperative Information Systems (EDCIS 2002),Y. Han, S. Tai, and D. Wikarski, eds., pp. 1-15, 2012.

[14] A. Ehrenfeucht, G. Rozenberg, "**Partial (Set) 2-Structures—Part 1 and Part 2**,"Acta Informatica, *vol. 27, no. 4, pp. 315-368*, 1989.

[15] Workflow Handbook 2001, **Workflow Management Coalition**, L. Fischer, ed. Lighthouse Point, Fla.: Future Strategies, 2011.