NRC Publications Archive Archives des publications du CNRC

Abductive workflow mining Buffett, Scott

For the publisher's version, please access the DOI link below./ Pour consulter la version de l'éditeur, utilisez le lien DOI ci-dessous.

<https://doi.org/10.4224/9174610>

NRC Publications Record / Notice d'Archives des publications de CNRC:

<https://nrc-publications.canada.ca/eng/view/object/?id=2fd3cf13-7f40-4b7c-bf43-d07ea1dd0e05>

<https://publications-cnrc.canada.ca/fra/voir/objet/?id=2fd3cf13-7f40-4b7c-bf43-d07ea1dd0e05>

Access and use of this website and the material on it are subject to the Terms and Conditions set forth at

<https://nrc-publications.canada.ca/eng/copyright>

READ THESE TERMS AND CONDITIONS CAREFULLY BEFORE USING THIS WEBSITE.

L'accès à ce site Web et l'utilisation de son contenu sont assujettis aux conditions présentées dans le site

<https://publications-cnrc.canada.ca/fra/droits>

LISEZ CES CONDITIONS ATTENTIVEMENT AVANT D'UTILISER CE SITE WEB.

Questions? Contact the NRC Publications Archive team at

PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca. If you wish to email the authors directly, please see the first page of the publication for their contact information.

Vous avez des questions? Nous pouvons vous aider. Pour communiquer directement avec un auteur, consultez la première page de la revue dans laquelle son article a été publié afin de trouver ses coordonnées. Si vous n'arrivez pas à les repérer, communiquez avec nous à PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca.

National Research
Council Canada

Institute for
Information Technology

Conseil national
de recherches Canada

Institut de technologie
de l'information

NRC - CNRC

Abductive Workflow Mining*

Buffett, S.
September 2008

* published at The 4th Workshop on Business Process Intelligence (BPI 08) to be held in conjunction with Business Process Management (BPM 2008). Milan, Italy. September 1, 2008. NRC 50393.

Copyright 2008 by
National Research Council of Canada

Permission is granted to quote short excerpts and to reproduce figures and tables from this report, provided that the source of such material is fully acknowledged.

Abductive Workflow Mining

Scott Buffett and Bruce Hamilton

National Research Council Canada, Fredericton, NB, E3B 9W4
{Scott.Buffett,Bruce.Hamilton}@nrc.gc.ca

Abstract. We define a new approach to discovering important workflows from event logs, referred to as *abductive workflow mining*, as the process of determining activity that would necessarily imply that certain predetermined critical activity should necessarily take place. Whenever critical activity is observed, one can inspect the abductive workflow to ascertain whether there was sufficient reason for the critical activity to occur. Initial theory surrounding the concept of abductive workflow is defined, and an initial method for discovering abductive workflows is presented. Preliminary experiments show that relatively small and concise abductive workflow models can be constructed, in comparison with constructing a complete model for the entire log.

1 Introduction

The diverse sets of advantages and disadvantages that various existing workflow mining algorithms have to offer means that there is seemingly a solution to fit almost any need. Techniques such as van der Aalst's α -algorithm [4], $\alpha++$ algorithm [7] and heuristic miner [6] offer mining techniques that produce concise, simplified workflows that are easy to visualize, but sacrifice accuracy. Still others such as Petrify [3] and Region Miner [5] achieve high fitness and precision, but as a result can produce workflows that may be too large or complicated to be reasonably understood by a human viewer. In this case, reducing the size and complexity of a workflow model by eliminating components that are not of interest to a particular observer with a specific purpose for viewing the model, while retaining the desirable qualities of fitness and precision, can be quite advantageous. Accomplishing this is the focus of this paper.

Situations where an observer needs to view and understand very specific components of a workflow model for a particular purpose is prevalent in the domain of compliance. In this case, the primary focus commonly seen in workflow mining of understanding how processes work so that they can be improved upon, made more efficient, or simplified, is not as crucial. Instead, the goal here lies in ensuring that individuals' behaviour is deemed appropriate within the context of the overall observed activity. Our approach is based on the notion of abductive reasoning, where hypotheses are found that, if added to a rule base, would necessarily cause an observation to be true. We focus on the instance of workflow mining where there are critical tasks in the underlying process that, if observed, must be scrutinized more diligently to ensure that they are sufficiently

motivated and executed under acceptable circumstances. Given a particular activity for which compliance checking is desirable, it is then the goal to determine exactly which part of the overall workflow model is needed. To accomplish this task, we introduce the concept of *abductive workflow mining*. This concept comes from the particular field of logical reasoning known as *abductive reasoning*. Abductive reasoning refers to the problem of finding *explanations* for a particular set of facts. These explanations are found to be sets of facts and axioms that, if true, will logically imply that the original set of facts are true. We apply this concept to workflow mining by discovering workflows that, if followed in a particular trace, would offer an explanation for why particular tasks were executed. Such a workflow is known as an *abductive workflow*.

The remainder of the paper is presented as follows. In section 2 we discuss required background theory on abductive reasoning that we employ in our techniques. Section 3 then defines abductive workflow, and introduces some necessary theory surrounding the concept. In section 4 we outline a technique for discovering abductive workflows and present a few results demonstrating the performance of our method. Section 5 then discusses conclusions and offers a few directions for future work.

2 Abductive Reasoning

Abduction is an instance of assumption-based reasoning where the focus is to determine hypotheses that, if true, would necessarily explain an observation or evidence in question [1]. This type of reasoning is particularly useful when one observes an interesting or curious event or happening, and may want to know what could possibly cause the event to be true. That is, given evidence E and rule base R , abductive reasoning attempts to find facts that, when added to R , would necessarily imply E . Such facts are then likely to have a causal effect on E in the real-world application being modeled by the rule base.

Abductive reasoning is of particular importance in fault diagnosis in complex systems. Here the rule base represents the operational characteristics of the system, and an observation is typically some fault or error in system function. Abductive reasoning is then used to determine what sort of activities would necessarily cause the fault to occur. These activities are then considered to be the primary causes of the error, and are thus investigated first. So abduction is applied to diagnosis with the idea that actions that would necessarily cause a fault to occur are more likely to be the true cause than actions that are merely consistent with the fault occurring.

We consider the application of abductive reasoning to workflow analysis in a similar way. Here the complex system to be modeled is the business process being analyzed, and the activities in question are the more serious or potentially harmful tasks that are executed within the process. Such critical tasks should not be executed freely; rather there should be an impelling reason for such tasks to be performed. Abductive reasoning can then be applied to determine what sort of activities would be sufficient for validating the critical behaviour in question.

3 Abductive Workflow

Let W be a workflow model constructed based on a set T of tasks and a set C of cases of executions of T . Additionally, let W' be a sub-workflow of W , for which an explanation is desired. Note that W' may often simply consist of a single transition, representing the event to be explained, but in general could be any workflow structure. We consider a workflow W_a to be an *abductive workflow* of W for W' if and only if any sequence of activity observed in W_a necessarily implies that there is associated activity in W' . This means that observing activity in W_a necessarily implies that activity in W' will occur, and thus W_a is an explanation for W' . In this section we introduce the notion of abductive workflow, and describe the associated theory by defining necessary and desirable properties.

Definition 1 (consistent) *Let C be a set of cases and let W be a workflow model (not necessarily built based on C). A case $c \in C$ is said to be consistent with W if there is a valid trace through W that is a (non-empty) substring of c . Moreover, any sequence t of events is consistent with W if there is a valid trace through W that is a (non-empty) substring of t .*

So a case c is consistent with W' if observing c would necessarily cause a trace through W' to be observed, perhaps with extra activity from c occurring before and/or after W' . Let $C^+ \subseteq C$ then be the set of cases in C that are consistent with W' , and accordingly let $C \setminus C^+$ be denoted by C^- . Any case in C^+ being observed will then necessarily cause activity in W' to occur. We use this partition over the set as the basis for the abductive model. First we define an *abductive trace*, which is the key concept in modeling abductive workflow.

Definition 2 (abductive trace) *Let W' be a workflow model and let C be a set of cases partitioned into C^+ and C^- as above. A sequence t_a of events is an abductive trace in C for W' if and only if each of the following hold:*

- t_a is a substring of some $c^+ \in C^+$
- For any $c^- \in C^-$, (1) t_a is not a substring of c^- , and (2) if any possible trace $t_{W'}$ through W' is a substring of t_a , then there is no string S such that the result of replacing $t_{W'}$ with S makes t_a a substring of c^- .

So, not only does an abductive trace t_a cause activity in W' , the part of t_a without the activity from W' necessarily causes activity in W' . This means that the activity in t_a that resides in W' cannot be replaced, and thus the activity in W' *must* occur. To illustrate the necessity for this restriction, consider a simple example with two cases $ABDE$ and $ACDF$, with W' simply being B . Then $ABDE$ is a positive case and $ACDF$ is a negative case. The trace ABD would not be considered an abductive trace, since executing B was not necessary in this situation. Another choice was possible, namely D . Put another way, for any event x , if A is performed immediately before x and D is performed immediately after, it is not necessarily the case that x would be B . However, BDE on the other hand would be an abductive trace (as would $ABDE$, DE , or even simply

E), since it is necessarily the case that an event x followed immediately by DE would have to be B . Using the notion of abductive traces, we define abductive workflow:

Definition 3 (abductive workflow) *Let C be a set of cases yielding workflow W . A workflow W_a is an abductive workflow of W for workflow W' if and only if every valid trace through W_a is an abductive trace in C for W' .*

Thus any activity that is consistent with W_a , minus any activity that forms a valid trace through W' , will necessarily cause a valid trace through W' to be executed.

While any abductive workflow will specify activity that will explain the activity in question, there may be additional explanations that are not captured by the model. That is, it is possible that activity in W' may occur without any explanation from W_a . Ideally, we would prefer an abductive workflow that would capture explanations for every case. That is not to say that we would expect W_a to capture every possible explanation; rather, it would be equally useful to generate abductive workflows that, given any activity in W' , will contain *some* explanation for that activity. To give an indication of how well an abductive workflow performs at offering such an explanation, we define the completeness measure for an abductive workflow.

Definition 4 (completeness measure) *Let $C = C^+ \cup C^-$ be a set of cases yielding workflow W and let W_a be an abductive workflow of W for W' as defined above. The completeness measure for W_a is the probability that a case in C^+ is consistent with W_a .*

The completeness measure is thus a measure of how likely it is that an occurrence of activity in W' will be explained by W_a , which is equal to the probability that a case containing a trace of activity through W' will also contain a trace of activity through W_a .

Definition 5 (complete) *An abductive workflow with completeness measure equal to 1 is said to be complete.*

Another desirable characteristic of abductive workflow is that the model be small and concise, making it easy to check, as well as less likely to introduce errors. There are two ways to measure the size of a workflow model: (1) by considering the length of the traces through the model, and (2) by considering the number of traces through the model. We consider each of these separately.

Definition 6 (task-minimal abductive trace) *An abductive trace t_a is task-minimal if there is no substring of t_a that is an abductive trace.*

Definition 7 (task-minimal abductive workflow) *An abductive workflow is task-minimal if all abductive traces making up the workflow are task-minimal.*

So a task-minimal abductive workflow allows for only small traces, but may allow for more traces than necessary. We tackle this dimension next.

Definition 8 (trace-minimal abductive workflow) *An abductive workflow is trace-minimal if there is no subgraph of the workflow with the same completeness measure.*

Thus an abductive workflow is minimal if it cannot be reduced in any way, thus reducing the number of allowable traces, without giving up some of the explanations it offers. Putting it all together, a best possible abductive workflow is then one that is complete, task-minimal and trace-minimal, and perhaps ideally, has the fewest nodes out of all workflows that meet the above conditions.

4 Testing and Results

While the main focus of the paper is merely to introduce the notion of abductive workflow as well as some of the associated theory behind the ideas, we offer some insight into how abductive workflows can be discovered by presenting a simple technique. To make the computational effort a bit more reasonable, we impose a simplifying assumption. In particular, when searching for substrings of cases in C^+ , we only consider those substrings that contain a trace of activity from W' . This is an attractive assumption for two reasons. First of all, in realistic situations, we believe that events that take place in close proximity to the activity in question are more likely to have a causal relation with that activity than are distant events. Second, it can be relatively easy to identify these substrings.

We start by building a workflow model W for the entire set C . W is then used to discover traces in the model that correspond to a path through W' . An activity a_i that directly precedes this path and an activity a_o that directly follows it are then chosen to form a larger path that contains the path through W' . We then search the rest of the model to determine if there is a sequence that contains a_i and a_o , with activity not consistent with W' in between. If one is found, then a new a_i or a_o is selected, or perhaps a longer sequence of preceding or following activities is selected. If this is not the case, however, then executing a_i and a_o necessarily causes activity in W' to be performed in between, giving an abductive trace. This process continues until a complete model is found.

To get an idea of by how much the size of the workflow graph is reduced by simply considering the abductive model, we ran tests on five example log files that are packaged with the ProM process management software [2]. We used our own miner to build the workflow, and then we chose a single task to be the target and found the corresponding abductive model that explains the target. The average number of transitions and arcs produced for each of the original and abductive model are presented in Table 1. The data shows that, in these tests the abductive model was less than one quarter the size of the original in terms of the number of both transitions and arcs. To show that the abductive model can score a significant reduction on workflows for other miners, we ran the tests on the α miner as well, a particularly tough one to reduce, since very few transitions are used in α -mined workflows. The data shows that significant reduction took place, as the workflows were cut to less than half the size.

Statistic	Our Miner		α Miner	
	Orig. Workflow	Abd. Workflow	Orig. Workflow	Abd. Workflow
Number of transitions	156.2	37.2	12.0	5.6
Number of arcs	318.6	77.0	33.2	15.6

Table 1. Average size of the original and abductive workflow models for both our workflow miner as well as the α miner

5 Conclusions and Future Work

In this paper we discuss a new approach to modeling workflow, referred to as abductive workflow mining. With this approach, small concise workflows can be modeled that are found to necessarily cause some activity in question to occur. This is especially useful in the area of compliance checking where there is a small number of critical tasks that need to be checked. The resulting smaller workflow means that errors are less likely, and can also be more easily understood by a human analyst that might be trying to make sense of what went wrong. Tests on a very naive method for discovering abductive workflows show that the size of workflows can be significantly reduced.

For future work, we plan to further investigate methods for finding small, concise abductive workflows that hold the complete, task-minimal and trace-minimal properties. One idea is to make use of a propositional logic representation, and use logical inference to determine workflows that will necessarily imply the activity in question. Existing reasoning systems such as inference engines and truth maintenance systems could then be utilized to find abductive traces in such a way that certain rules regarding completeness and minimality are satisfied.

References

1. D. Poole, A. Mackworth, and R. Goebel. *Computational Intelligence: A Logical Approach*. Oxford University Press, Inc., New York NY, USA, 1998.
2. ProM. The ProM framework. "http://is.tm.tue.nl/~cgunther/dev/prom/", 2007.
3. Wil M.P. van der Aalst, V. Rubin, B.F. van Dongen¹, E. Kindler, and C.W. Gunther¹. Process mining: A two-step approach using transition systems and regions. Technical report, Eindhoven University of Technology, 2006.
4. W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Maruster. Workflow mining: Discovering process mining models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.
5. B.F. van Dongen, N. Busi, G.M. Pinna, and W.M.P. van der Aalst. An iterative algorithm for applying the theory of regions in process mining. Technical report, Department of Technology Management, Eindhoven University of Technology, 2006.
6. A.J.M.M. Weijters and W.M.P. van der Aalst. Rediscovering workflow models from event-based data using little thumb. *Integrated Computer-Aided Engineering*, 10(2):151–162, 2003.
7. L. Wen, J. Wang, and J.G. Sun. Detecting implicit dependencies between tasks from event logs. In *Frontiers of WWW Research and Development - APWeb 2006*, volume 3841, pages 591–603. Springer Berlin / Heidelberg, 2006.