POQL: A New Query Language for Process-Oriented Case-Based Reasoning

Gilbert Müller and Ralph Bergmann

Business Information Systems II
University of Trier
54286 Trier, Germany
[muellerg][bergmann]@uni-trier.de,
http://www.wi2.uni-trier.de

Abstract. Sharing and reuse of best-practice process models is an important knowledge management approach for business process modelling. Process-oriented Case-Based Reasoning (PO-CBR) supports this by retrieving and adapting processes or workflows based on models stored in the repository, which requires an expressive query language. Hence, we present a novel query language for workflows that enables to express generalized query terms and negation. Further, it allows a ranking of the repository workflows.

Keywords: Process-oriented Case-based Reasoning, Business Process Querying, Workflows

1 Introduction

Nowadays, business processes have to be organized highly flexible due to increasing globalization and competitive pressure. Thus, business processes and workflows implementing them have to be promptly defined or adapted to new circumstances. For this purpose, sharing and reuse of existing best-practice process models is an important approach that introduces knowledge management concepts into business process modelling [11]. Thus, important knowledge management tools are searchable repositories of process models that enable the retrieval of reusable processes and may in addition propose ways of reusing them. Process-oriented Case-based Reasoning (POCBR) [18] is a research area that deals with applying case-based reasoning (CBR) to experiential knowledge represented in process models and workflows and thus provides the foundations for building knowledge management tools supporting process modelling. Current research in POCBR addresses the similarity-based retrieval and adaptation of process and workflow models. However, the formulation of queries for the purpose of modelling and adaptation support has not been discussed extensively.

 In: R. Bergmann, S. Görg, G. Müller (Eds.): Proceedings of the LWA 2015 Workshops: KDML, FGWM, IR, and FGDB. Trier, Germany, 7.-9. October 2015, published at http://ceur-ws.org

Current POCBR research usually assumes that a query just consists of a partial workflow/process currently being designed. Then retrieval searches for similar workflows that mostly match the current query. Thus, the found workflows can be considered a source of the auto-completion of the current partial workflow, which can be supported by case-based adaptation methods. However, for appropriate retrieval and adaptation, a query language is needed that is able to capture as best as possible all current requirements on the workflow/process to be created. In this paper, we address this issue by proposing a new, more expressive approach for the formulation of queries in POCBR and we sketch a way of tweaking existing retrieval methods to deal with this language.

2 Foundations

We build this research upon our previous work in POCBR which addresses the similarity-based retrieval and adaptation of semantic workflows [7,19,20]. The methods developed so far are implemented in the prototypical software system CAKE [6] and analyzed in various application domains. In this paper, we illustrate our approach in the domain of cooking recipes. A cooking recipe is represented as a workflow describing the instructions for cooking a particular dish. We now briefly outline previous relevant work as the main foundation of this paper.

Broadly speaking, workflows consist of a set of activities (also called tasks) combined with control-flow structures like sequences, parallel (AND) or alternative (XOR) branches, as well as repeated execution (LOOPs). Tasks and control-flow structures form the control-flow. In addition, tasks exchange certain data items, which can also be of physical matter, depending on the workflow domain. Tasks, data items, and relationships between them form the data flow. In our work we extend this traditional view of workflows by adding semantic annotations to (potentially) all workflow items as a means to support case-based reasoning. In formal terms, a semantic workflow is defined as a directed graph $W = (N, E, S, T)$ where N is a set of nodes and $E \subseteq N \times N$ is a set of edges. Nodes and edges have types assigned by the function T, which partitions the nodes N of a workflow into a single workflow node N^W and several data nodes N^D, task nodes N^T, and control-flow nodes N^C. Likewise we distinguish data-flow edges E^D, control-flow edges E^C and part of edges N^P. Further, nodes have a semantic description from a semantic meta data language Σ, which is assigned by the function $S : N \to \Sigma$.

2.1 Semantic Workflows

Figure 1 shows a simple fragment of a workflow graph from the cooking domain. Here, the tasks represent the cooking steps and the data items refer to the ingredients being processed by the cooking steps. The main source of knowledge in POCBR is a repository of semantic workflows (in CBR terminology called the case base) available for reuse. In order to obtain a reusable workflow similarity

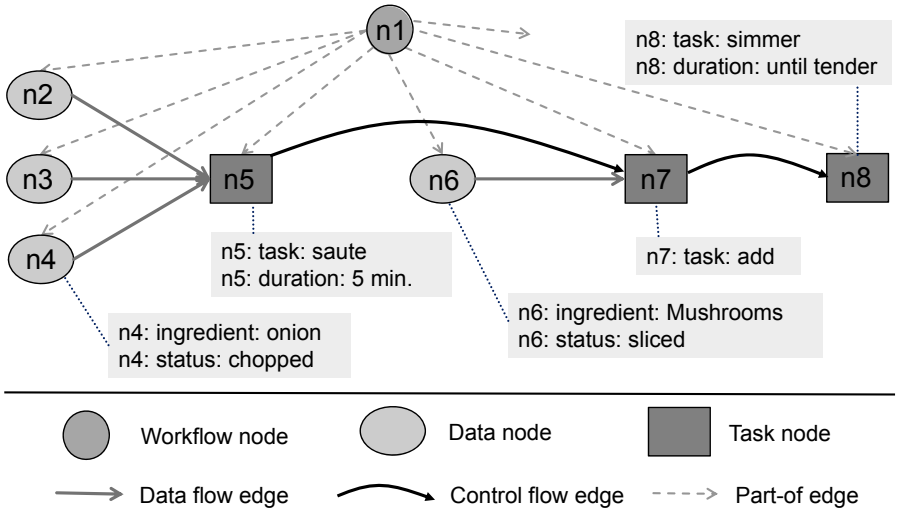


Fig. 1. An example workflow graph

search or process model querying [11,21] can be applied. According to Dijkman et al. [11] "the main difference between querying and similarity search is that querying searches for exact matches of a query to a part of a process model, while similarity searches for inexact matches of the query to a complete process model". We focus on similarity search as it is able to provide results even if exact matches are not available, which is very likely in many application scenarios. Various approaches for similarity search of workflows have been proposed in the literature, such as graph edit metrics, graph/subgraph isomorphism, most common subgraph approaches [3,7,10,14,15]. In our research [7], we developed an approach that follows the tradition of CBR and uses explicitly modelled local similarity measures for task and data items, based on task and data ontologies, which are also used for the semantic annotation of the workflows. The overall similarity $sim(QW, CW) \in [0, 1]$ between a query workflow QW and a case workflow CW from the repository is defined as an optimization problem aiming at finding the best possible type-preserving, partial, injective mapping of the nodes and edges of QW to those of CW. The optimization target is the average similarity of the mapped nodes and edges. This similarity measure assesses how well the query workflow is covered by the case workflow. In particular, the similarity is 1 if the query workflow is exactly included in the case workflow as a subgraph.

3 Query Language Requirements

Previous work on POCBR (including our own) is limited by the type of queries that can be considered. As described in the previous section, a query is a single

(partial) workflow that describes task and data items and structural relationships of the desired workflow. This can be roughly considered a conjunctive query, as the ideal workflow contains the whole query workflow, i.e., all components it contains. Thus, disjunction and negation cannot be expressed. However, the user may also want to express undesired workflow elements and structures. For example, some tasks or data elements must not occur in the workflow or a certain sequence of activities is undesired. Also, disjunction/generalization is sometimes required, e.g. by providing more general conditions, such as specifying a class of tasks (from the ontology) or by specifying that a certain task must occur some time (but not necessarily directly) before another one. More expressive queries are not only desirable for retrieving more suitable workflows but are also essential to guide automatic adaptation methods from POCBR as they can provide hints concerning which workflow elements need to be added, deleted, or moved to a different position. Besides these usage scenarios, literature also lists a wide range of additional purposes for queries [17,1], e.g., dependency analysis between workflow elements [9] and decision making support [12]. However, in the following we focus our investigations on retrieval and adaptation support. Based on this, we derive the following main requirements for a new query language, which have also been partially mentioned in the literature [16,2]:

- **Expressiveness:** The query language must be expressive enough to be able to represent the relevant requirements of the user. Thus, the query language should not only be able to represent what the user desires, it should be also able to handle undesired workflow elements, which requires a kind of negation. Additionally, generalization of structure and items is required.
- **Intuitiveness:** The query language should be easy to understand. Thus, new notations should be only introduced if required and it should be based on the already known concepts. Additionally, a visual query language is preferable as workflows can become very complex and thus also its queries.
- **Ranking:** For the specified query language it must be able to identify all matching workflows. Moreover, as fully matching workflows are very unlikely in many application scenarios, it must be possible to rank the workflows w.r.t. suitability for the query, if they don't match perfectly.

Thus, the similarity-based retrieval must be extended towards a retrieval considering suitability w.r.t. a more expressive query.

4 POQL: A New Query Language for Workflows

POQL is a workflow query language developed according to the previously mentioned requirements. It extends the purely conjunctive query approach by introducing negation and generalized query workflows. Formally, a POQL query Q is defined as follows: $Q = DW \wedge \neg RW_1 \dots RW_n$. Here, DW is the desired workflow representing properties that the searched workflow should fulfill, RW_i are restriction workflows, each of which represents one undesired situation that

should be avoided. The desired workflow and the restriction workflows are so-called generalized query workflows. They are in principle workflows in the previously introduced sense, but they are extended to represent generalizations by two means. As a consequence, generalized query workflows are not executable workflows anymore, but they are just a means to express a query in a way similar to a workflow. Thereby, the intuitiveness requirement can be fulfilled as building a query is very similar to building a workflow.

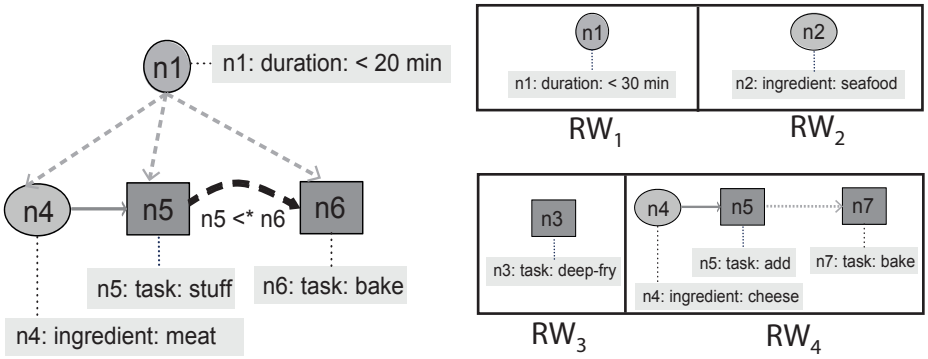
Generalized Task and Data Labels: While workflows from the repository usually contain ontology instances for tasks and data items specifying the flow of activities in executable terms, a generalized workflow [20] is allowed to contain concept labels of the data- and task ontology, thus specifying classes of them. For example, a recipe workflow may specify that meat is desirable without specifying in detail which meat should be used. For the representation of generalized task and data labels, the meta-data language used for semantic annotation must also include the ontology concepts.

Transitive Control-Flow and Dataflow Connectedness: While workflows exactly specify the control- and dataflow of a workflow, a generalized query workflow may just specify that a certain tasks must be executed some time before another task or that one task produces data that at some later point is used by another tasks. In formal terms these relations are defined as follows.

1. Let $t_1 < t_2$ define that there exists a control-flow edge between $t_1 \in N^T$ and $t_2 \in N^T$ defining that t_1 is executed before t_2. We define that two tasks $t_1, t_2 \in N^T$ are transitively control-flow connected $t_1 <^* t_2$ iff $t_1 < t_2 \vee \exists t \in N^T : t_1 < t < *t_2$. We represent $t_1 < *t_2$ in a generalized query graph by introducing a new type of edge leading from t_1 to t_2.
2. Let $t_1 \ltimes t_2$ denote that the tasks $t_1, t_2 \in N^T$ are data-flow connected, i.e., it holds $t_1 \ltimes t_2$ iff $t_1 < t_2 \vee \exists d \in N^D : ((t_1, d) \in E^D \wedge (d, t_2) \in E^D)$. Based on this we define that two tasks $t_1, t_2 \in N^T$ are transitively data-flow connected $t_1 \ltimes^* t_2, if t_1 \ltimes t_2 \vee \exists t \in N^T : t_1 \ltimes t \ltimes^* t_2$. To represent transitive data-flow connectedness, a new type of edge is introduced. For example, this edge can be used to express in the query that after a specific preparation step using an ingredient another preparation step follows that uses the same ingredient.

Figure 2(a) shows an example of a generalized query workflow. This query specifies that a workflow is desired that requires less than 20 minutes of preparation time, which contains some kind of meat (generalized data label) , and the preparation steps stuff and bake. Furthermore, it is specified that the tasks for stuffing the meat must occur before the baking tasks (transitive control-flow connected).

The restriction workflows can be constructed in a similar manner as illustrated in figure 2(b). The four restrictions shown specify that the searched workflow should require less than 30 minutes of preparation time (RW_1), that it should not contain any seafood (RW_2) or deep-fry preparation steps (RW_3), as a frying machine is maybe not available. Furthermore, it is required that no



(a) Query Workflow Example (b) Restriction Workflow Examples

Fig. 2. POQL Query Example

cheese is added to a dish component which is later baked (RW_4), thus casserole recipes are undesired.

Please note that in principle it would be possible to allow more than one desired workflow in a query. However, this is not necessary as it would not extend the expressiveness of the language, as several desired workflows could be easily merged into a single desired workflow representing the same query semantics.

The processing of a POQL query requires ranking all workflow from the repository and presenting the best ranked results. For a query which does not include any restriction workflows, our approach for workflow similarity can be extended in a straight forward manner. Generalized task and data labels are addressed by applying taxonomic similarity measures, which are well-established in CBR [5].

To consider transitive control-flow and dataflow connectedness, the workflow graph of all workflows in the repository is extended to represent all relations $t_1 < *t_2$ and $t_1 \ltimes^* t_2$ which must be pre-computed during the initialization of the repository. Given this, these relations can be matched in the same manner as all other edges of the graph. However, this approach obviously increases the complexity of the representation and thus the complexity of the similarity assessment, which is an issue of our future research.

To handle restriction workflows in POQL requires dealing with negation and conjunction. We propose an approach adopted from fuzzy logic [8] by treating the similarity value computed by comparing a case workflow with a generalized query workflow as a fuzzy membership value. Then we can apply fuzzy negations and a t-norm to compute the conjunction. In particular, we compute the rank of a workflow as follows:

$$rank(Q, CW) = min\{sim(DW, CW), 1-sim(RW_1, CW), \ldots, 1-sim(RW_n, CW)\} \quad (1)$$

The resulting $rank(Q, CW) \in [0, 1]$ reflects how well the workflow matches the query, while the following conditions hold:

1. $Rank = 1$ if the case workflow exactly matches the desired workflow and does not contain any subworkflow which is somehow similar to any of the restriction workflows.
2. $Rank = 0$ if the case workflow contains a subworkflow which exactly matches one of the restriction workflows.
3. $Rank \in]0, 1[$ if the case workflow partially matches the desired workflow and if all restriction workflows are at most matching partially (not exactly).

5 Conclusions

We presented the novel query language POQL for POCBR which is highly intuitive and enables not just the retrieval but also the adaptation of workflows. A POQL query can be easily constructed using a graphical workflow editor by introducing additional link types among tasks as well as ontological concepts for semantic annotation. We presented an approach that allows an ordering of the best workflows from the repository by a ranking approach.

In process model querying, BPMN-Q [1,21] is a related approach applied to BPMN business processes. The approach is able to identify processes that match the partial modelled workflow. However, the approach is so far neither able to consider the dataflow nor undesired data or tasks. Furthermore, there is no ranking between the processes found. Awad et al. [2] extend BPMN-Q by regarding semantics between workflow elements. The related approaches presented by Beeri et al. [4] or by Markovic et al. [16,17] are not able to support negations or to rank the results by similarity which both is required for the modelling and adaptation support of workflows. Recently, PQL [13] has been presented, which also addresses the querying and changing of process models. However, in contrast to our work where required changes are implicitly derived from the query, PQL required to define those changes explicitly in a SQL-like statement.

Future work will extend the query language to support other usage scenarios (see section 3), i.e., scenarios in which only fragments of workflows are searched rather than complete workflows. Additionally, an evaluation of the presented POQL will be undertaken.

Acknowledgements. This work was funded by the German Research Foundation (DFG), project number BE 1373/3-1.

References

1. Awad, A.: BPMN-Q: A language to query business processes. In: EMISA. vol. 119, pp. 115–128 (2007)

2. Awad, A., Polyvyanyy, A., Weske, M.: Semantic querying of business process models. In: Enterprise Distributed Object Computing Conference, 2008. EDOC'08. 12th International IEEE. pp. 85–94. IEEE (2008)
3. Bae, J., Liu, L., Caverlee, J., Rouse, W.B.: Process mining, discovery, and integration using distance measures. In: Web Services, 2006. ICWS'06. International Conference on. pp. 479–488. IEEE (2006)
4. Beeri, C., Eyal, A., Kamenkovich, S., Milo, T.: Querying business processes. In: Proceedings of the 32nd international conference on Very large data bases. pp. 343–354. VLDB Endowment (2006)
5. Bergmann, R.: Experience management: foundations, development methodology, and internet-based applications. Springer-Verlag (2002)
6. Bergmann, R., Gessinger, S., Görg, S., Müller, G.: The collaborative agile knowledge engine cake. In: Proceedings of the 18th International Conference on Supporting Group Work. pp. 281–284. GROUP '14, ACM, New York, NY, USA (2014)
7. Bergmann, R., Gil, Y.: Similarity assessment and efficient retrieval of semantic workflows. Inf. Syst. 40, 115–127 (Mar 2014)
8. Burkhard, H.D., Richter, M.M.: On the notion of similarity in case based reasoning and fuzzy theory. In: Soft computing in case based reasoning, pp. 29–45. Springer (2001)
9. Dai, W., of Waterloo. School of Computer Science, U.: A Query-based Approach to Workflow Process Dependency Analysis. University of Waterloo (2005)
10. Dijkman, R., Dumas, M., García-Bañuelos, L.: Graph matching algorithms for business process model similarity search. In: Business process management, pp. 48–63. Springer (2009)
11. Dijkman, R.M., La Rosa, M., Reijers, H.A.: Managing large collections of business process models-current techniques and challenges. Computers in Industry 63(2), 91–97 (2012)
12. Hepp, M., Leymann, F., Domingue, J., Wahler, A., Fensel, D.: Semantic business process management: A vision towards using semantic web services for business process management. In: e-Business Engineering, 2005. ICEBE 2005. IEEE International Conference on. pp. 535–540. IEEE (2005)
13. Kammerer, K., Kolb, J., Reichert, M.: PQL - A Descriptive Language for Querying, Abstracting and Changing Process Models. In: BPMDS'15. pp. 135–150. No. 214 in LNBIP, Springer (June 2015)
14. Kapetanakis, S., Petridis, M., Knight, B., Ma, J., Bacon, L.: A case based reasoning approach for the monitoring of business workflows. In: Case-Based Reasoning. Research and Development, pp. 390–405. Springer (2010)
15. Ma, Y., Zhang, X., Lu, K.: A graph distance based metric for data oriented workflow retrieval with variable time constraints. Expert Systems with Applications 41(4, Part 1), 1377 – 1388 (2014)
16. Markovic, I.: Advanced querying and reasoning on business process models. In: Abramowicz, W., Fensel, D. (eds.) Business Information Systems, Lecture Notes in Business Information Processing, vol. 7, pp. 189–200. Springer Berlin Heidelberg (2008)
17. Markovic, I., Costa Pereira, A., de Francisco, D., Muoz, H.: Querying in business process modeling. In: Di Nitto, E., Ripeanu, M. (eds.) Service-Oriented Computing - ICSOC 2007 Workshops, Lecture Notes in Computer Science, vol. 4907, pp. 234–245. Springer Berlin Heidelberg (2009)
18. Minor, M., Montani, S., Recio-Garca, J.A.: Process-oriented case-based reasoning. Information Systems 40(0), 103 – 105 (2014)

19. Müller, G., Bergmann, R.: Workflow streams: A means for compositional adaptation in process-oriented cbr. In: Case-Based Reasoning Research and Development, pp. 315–329. Springer (2014)
20. Müller, G., Bergmann, R.: Generalization of Workflows in Process-Oriented Case-Based Reasoning. In: 28th International FLAIRS Conference. AAAI, Hollywood (Florida), USA (2015)
21. Sakr, S., Awad, A., Kunze, M.: Querying process models repositories by aggregated graph search. In: Business Process Management Workshops. pp. 573–585. Springer (2013)