SIMILARITY MATCHING OF BUSINESS PROCESS VARIANTS

Noor Mazlina Mahmod, Shazia Sadiq and Ruopeng Lu
School of Information Technology and Electrical Engineering, The University of Queensland
St Lucia, QLD 4072, Brisbane, Australia

Keywords: Similarity Matching, Business Process Variants, Flexible Work Practice.

Abstract: Evidence from business work practice indicates that variance from prescribed business process models is not only inevitable and frequent, but is in fact a valuable source of organizational intellectual capital that needs to be captured and capitalized, since variance is typically representative of preferred and successful work practice. In this paper, we present a framework for harnessing the value of business process variants. An essential aspect of this framework is the ability to search and retrieve variants. This functionality requires variants to be matched against a given criteria. The focus of this paper is on the structural criteria which is rather challenging as query process structures may have different levels of similarity with variant process structures. The paper provides methods for undertaking the similarity matching and subsequently providing ranked results in a systematic way, as well as a reference architecture within which the methods may be deployed.

1 INTRODUCTION

Instance adaptation of business processes is an ongoing issue due to various reasons such as the frequent change in underlying business objectives and operational constraints, and the emergence of unexpected events that cannot be handled by predefined exception handling policies, collaborative and/or knowledge intensive work, and gap of process models from preferred work practices. Consequently, the execution of process instances needs to be changed at runtime causing different instances of the same business process to be handled differently according to instance specific conditions.

The typical consequence of instance adaptation is the production of a large number of *process variants*. An executed process instance reflects a variant of realization of process constraints, and provides valuable knowledge of organization at the operational level. There is evidence that work practices at the operational level are often diverse, incorporating the creativity and individualism of knowledge workers and potentially contributing to the organization's competitive advantage. Such resources can provide valuable insight into work practice, help externalize previously tacit knowledge, and provide valuable feedback on subsequent process design, improvement, and evolution.

In this paper, we propose building a repository to systematically capture, structure and subsequently deliberate on the decisions that led to a particular design. The focus is on providing a means to search and retrieve process variants on the basis of their structural similarity to a user defined query. In the subsequent sections, we will first present the related work for this topic. We then discuss the overall variant management framework. Then, in section 4 we will introduce the notion of structural similarity. This notion is used to conduct the matching analysis as well as the ranking computation process, which are respectively presented. Finally in section 5, we conclude with a summary of contributions of this work and its interesting extensions.

2 RELATED WORK

The goal of the work presented in this paper is to find an effective means to facilitate the search and retrieval of process variants that have total or partial structural match with a given process query i.e. we want to produce an effective method to find the degree of structural similarity and to compute the structural similarity rank between the process variants.

The notion of similarity matching analysis is in general a hard problem. It has been addressed from

Mazlina Mahmod N., Sadiq S. and Lu R. (2008).
SIMILARITY MATCHING OF BUSINESS PROCESS VARIANTS.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - ISAS*, pages 234-239
DOI: 10.5220/0001691002340239

various aspects e.g. string matching (Koudas et al, 2004) image and video matching (Shen et al, 2007), and graph matching (Chen et al, 2005). For business processes, notable work has been reported on process equivalence (van der Aalst et al, 2006) that takes into account execution sequences to conduct the similarity analysis. Another approach is presented in (Chen et al, 2005) to detect semantic business process variants using ontology approach.

In (Lu & Sadiq, 2006), a selective reduce technique has been introduced to reduce process variants that can be visually compared to conduct the structural matching between the process variant and the process query. This process graph reduction technique introduced in (Sadiq & Orlowska, 2000) will be used and applied in the structural matching analysis carried out by this paper as well. More over, the flows counting algorithm introduced in (Lu & Sadiq, 2006) is enhanced and modified to produce an improved algorithm to compute the total structural match and the different types of partial structural match as presented in section 4.

3 VARIANT MANAGEMENT

A prerequisite to utilizing process variance for the benefit of instance adaptation and process improvements is the creation of the variants, such that they can described, captured and eventually utilized. We rely on an instance adaptation framework (Sadiq et al, 2005) based on the principle of late modelling (Weber et al, 2007) to achieve a systematic creation of process variants. The framework allows variants to be created under well defined but minimal constraints, thus ensuring that variant representations do not have drastic differences that makes querying and eventually learning from them practically infeasible.

Process Variant Repository (PVR) provides a well-formed structure to store past process designs, as well as an instrument to utilize process variants as an information resource (Lu & Sadiq, 2006). The capture of executed process variants in the repository and the subsequent retrieval of preferred process variants are the two major functions of PVR.

Fig. 1 presents an overview of PVR reference architecture, details in (Lu & Sadiq, 2007).

We observe that a process variant at least contains information from the following dimensions: ***Structural*** dimension contains the process model based on which the process instance is executed. ***Behavioral*** dimension contains execution information. ***Contextual*** dimension contains descriptive information (annotations) from the process modeller.

In this paper, we primarily focus on the search and retrieval of variants based on its structural dimension. However, there is evidence (Lu & Sadiq, 2007) that multi-criteria search and retrieval can allow further refinement.

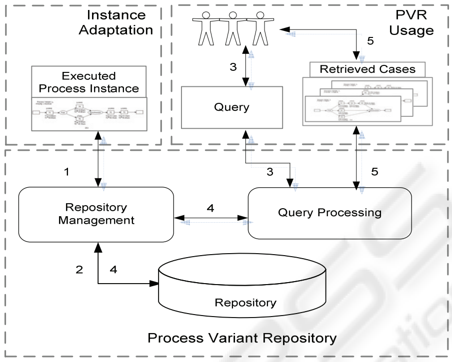

Figure 1: Reference architecture of PVR.

Definition 1 (Process Model). A process model W is a pair (N, E), which is defined through a directed graph consisting a finite set of nodes N, and a finite set of flow relations (edges) $E \subseteq N \times N$. Nodes are classified into tasks T and coordinators C, where $N=C \cup T$, and $C \cap T = \emptyset$. *Task* is the set of tasks in W, and C contains coordinators of the type {*Begin*, *End*, *Fork*, *Synchronizer, Choice, Merge*}, which have typical workflow semantics.

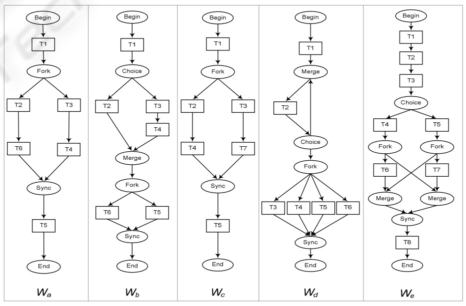

Figure 2: Example process models of process variants.

Figure 2 presents example process models of different process variants. Suppose these process variants belong to a network diagnosis process in a Telco company as described above, and tasks *T1,…,T8* correspond to a list of network testing activities. For example, T1 represents "*Send Test Message*", T2 represents "*Test Hub201*", and T3 "*Test ExchangeA30*" *etc*. In the rest of this paper, we omit the full task names for clarity.

Definition 2 (Process Variant). A process variant V is defined by (id, W, B, T, C, M), where

– *id* is the identifier for the process variant;

– W is the process model (N, E) for V defined on the task set $T \subseteq N$;
– B is a set of behavior properties defined for the process which may include execution sequences, resources utilized, time durations etc (see (Lu & Sadiq, 2007)0 for more details on behavior properties for variants)
– $T=\{T_1, \ldots, T_n\}$ is the set of tasks in V. Each task may also contain task level behavior properties.
– C is an annotation that textually describes the design of the variant;
– M is the set of modeler(s) who participated in the instance adaptation for V.

The schema for process variants contains *instance level* (*id, W, B, T, C, M*) and *task level* features (T). The *id* can be combined with the variant symbol V, *i.e.*, V_{10} denotes variant V with the feature (*id*, 10). Occasionally we omit the subscript i for V when there is no ambiguity. Each element in V is referred to as a *feature* of V. In this way, the schema of process variant is defined by a list of features from structural, behavioral and contextual dimensions. The process variant repository is the set of all collected process variants, that is $PVR=\{V_1, \ldots, V_n\}$.

4 STRUCTURAL MATCHING

The notion of structural similarity for business processes is rather involved. There have been some notable attempts to define structural relations, e.g. see equivalence, subsume and transform relations in (Sadiq & Orlowska, 2000), as well as similarity based on execution sequences in (van der Aalst et al, 2006). Due to the labelling of nodes in process graphs, as well as the specialized semantics of modelling constructs, the equivalence notion in process graphs is somehow computationally simplified.

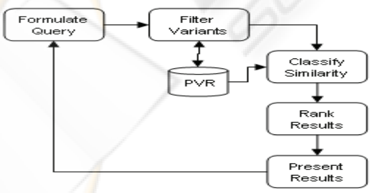

Figure 3: Approach for Structural Matching.

However, the question of degree of similarity still remains. For example, two graphs may have same task set, but arranged in different sequences (A, B, C vs. A, C, B). Should such a difference be classified as “similar”. If yes, to what degree. In the remaining section, we will present our approach to address the above question in a systematic way as summarized in Fig. 3.

4.1 Formulate Query

A *query* is a structural expression of search criteria representing partial or complete description for a process variant, or multiple process variants sharing similar features. Unlike traditional query systems however, the search criteria for process variants may also include reference to complex structural features as well as multi-dimension features *e.g.*, *tasks T1, T2 and T3 were performed by a senior engineer in sequence, and finished execution within 1 day* (*cf.* W_e in Fig. 3), or *having execution sequence <T1, T3, T4, T5, T6> and tasks T5 and T6 were in parallel branches in the process model* (*cf.* W_b in Fig. 2).

We propose that the structural query requirement be expressed in a way that is in like with the query-by-example (QBE) paradigm, where a process model W^Q is presented in the query containing the desired structural features, and the objective is to retrieve all process variants with a process model W similar to W^Q. W^Q can resemble a *complete* process model (*cf.* W_a^Q in Fig. 4), which specifies the exact structure required for the process variants to be retrieved; or a *partial* process model (*cf.* W_b^Q in Fig. 4), which contains a fragment of the process model characterizing the desired structural features to be retrieved. Based on the above discussion, we define the schema for a query as follows:

Definition 3 (Query). Let F be the set of all features in PVR. A query Q is defined by the set of query features $\{F_1^Q, \ldots, F_k^Q\}$, where $\forall F_i^Q \in F$, F_i^Q corresponds to a feature defined in schema of V.

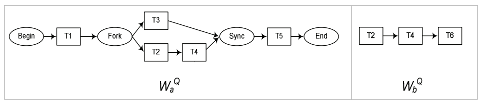

Figure 4: Example of structural query features, W_a^Q as a complete process model and W_b^Q as a partial process model.

As mentioned before, the focus of this paper is on the structural dimension, and hence in the discussion below, the query feature is assumed to represent the process model W, and set of tasks T from the variant schema (*id*, W, B, T, C, M).

4.2 Filter Variants

As variant repositories can potentially be very large, we propose a pre-processing step through which variants that are totally dissimilar to the submitted query can be filtered out of the similarity analysis and ranking steps. The filtering process consists of two steps. Firstly, task set T for each variant is compared with the task set of the submitted query, and only those variants are filtered out where the intersection of the two sets is above a certain threshold.

Upon this filtered set of variants, a method of *select reduce* (Lu & Sadiq, 2006) is applied, which allows process variant models to be reduced to graphs containing only the tasks present in the query, while preserving the structure of the original variant model.

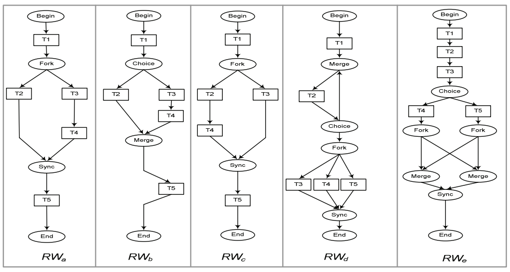

Figure 5: Reduced process models against query feature W^{Q}.

Figure 5 provides an illustration of the *select reduce* method for the variants given in Fig. 2.

The *select reduce* method provides a visual capability of identifying equivalent or similar variants. Although in large variant repositories, the visualization will not assist and hence further analysis is required as described below.

Variants may have different levels of similarity to the given query. The overview of the structural similarity types is described in Fig. 6.

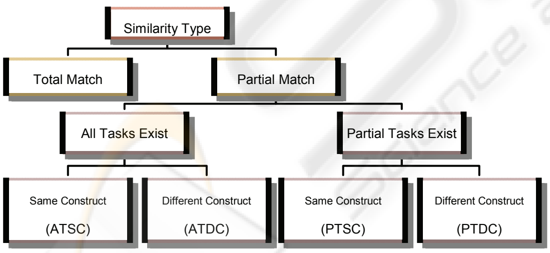

Figure 6: Types of Structural Similarity.

"Total Match" type represents the process variant that is totally similar to the process query (e.g. RW_b in Fig 5). Meanwhile, "Partial Match" type represents the process variant that is *similar* to the process query.

The partial match is split into two categories where "All Tasks Exist" and "Partial Tasks Exist". "All Tasks Exist" represents the reduced process variant that consists of all of the query tasks. Meanwhile "Partial Tasks Exist" represents the reduced process variant that contains only some of the query tasks.

Subsequently, each of these two categories of the partial match type is divided into two more specific categories where SC indicates "Same Construct" and DC "Different Construct". The ATSC category signifies that the reduced process variants are exactly similar to the process query. The PTSC category denotes that although the reduced process variant does not contain all the tasks existed in the process query, however, the structural constructs are similar to the process query. Similarly ATDC and PTDC refer to the different structural constructs of the reduced process variant against the process query.

4.3 Rank Results

Except for the case of the "Total Match" all other cases, namely ATSC, ATDC, PTSC and PTDC will need to be somehow ranked to determine the degree of structural similarity with the submitted query. Before we proceed with the structural similarity rank computation for partial match, some structural elements of process variants should be considered and should be assigned a dissimilarity weight/degree (DD) in order to distinguish the difference between the types of structural elements and to specifically formulate the computation.

In this paper, we only focused on selected structural elements in order to illustrate the rank computation, namely (1) extra task, (2) extra fork or (3) synchronizer, (4) missing task and (5) missing fork or (6) synchronizer within process variants in comparison to a process query.

The following sections present a justification or rationale of the dissimilarity degree (DD) of various structural elements representing the dissimilarity of the process variant. DD will be applied in the structural similarity rank computation algorithm for partial matches later.

(1) <u>Dissimilarity Element: **Extra Task:**</u> The DD of 0.5 is given to every extra task that exists in the process variant under the intuition that if the task is extra, the process variant might be a bit less effective (i.e. the process will have to run more tasks or unnecessary tasks, thus it will consume more resources).

(2) <u>Dissimilarity Element: **Extra Fork:**</u> The DD of 0.8 is given to every fork split that exists in the process variant. An extra fork split contributes more DD than the extra task because each fork split involves a different strategy in process execution.

(3) <u>Dissimilarity Element: **Extra Synchronizer:**</u> The DD of 1.0 is given to every extra Synchronize

Coordinator existing in the process variant. The higher weight is assigned to the extra synchronizer as it contributes more DD than the extra task and fork Coordinator since rationally the synchronizer may involve more than a task with several incoming transitions.

(4) Dissimilarity Element: **Missing Task:** The DD of 1.5 is given to every missing task in process variant because we believe that if a task is missing, it contributes more DD than the extra task and the extra fork/synchronizer coordinator because reasonably if a task is missing, the process variant will not execute the task which was deemed important for the query formulation.

(5) Dissimilarity Element: **Missing Fork:** The DD of 1.8 is given to every missing fork split in process variant since a missing fork contributes more DD than the missing task as logically some important strategies in process execution are missing that might lead to a different result.

(6) Dissimilarity Element: **Missing Synchronizer:** The DD of 2.0 is given to every missing synchronizer in process variant because if any synchronizer is missing, it contributes more DD than the missing task and fork split since logically the synchronizer may involve dissimilarity of due to other branches in addition to the differences found in above two cases.

To compute the structural similarity rank of partial match, we assume that the filtering (based on task sets and *select reduce*) as well as the classification of similarity type (i.e. ATSC, ATDC, PTSC, PTDC), has been completed. Then, we formulate a different computation formula for every partial match category by combining the DD calculation and flow match count to provide a reasonable structural similarity rank for every partial match category. Intuitively, it can be observed that the ATSC rank should be the highest partial match rank, the ATDC and PTSC rank will be in the next and the PTDC rank should be lowest rank amongst the partial match categories.

This algorithm is used to compute the rank of structural similarity between process variants. The total match is computed based on the flow counting of the same task type. For partial match, instead of calculating only the matching flows between the process variant, the DD of different structural elements as introduced earlier should be included to specifically formulate the computation in order to distinguish the different types and different levels of structural similarity between the process variants.

For ATSC computation presented in the algorithm, the DD calculation for extra tasks, forks and synchronizers are included. The ATSC computation is enhanced and added with the DD calculation for missing fork and synchronizer to compute ATDC partial match. Meanwhile, the rank computation for PTSC and PTDC has included the DD calculation for all missing and extra tasks, forks and synchronizers.

Structural_Similarity_Rank_Computation

```
Input reduced process graph P, query graph Q
Output Structural Similarity Rank
    rank ← 0
    totalMatch ← 0
    count ← 0
    for each task t ∈ T[P], taskType[t] ∈ {task, coordinator} do
    if InFlows[t] ∈ F[Q] then
    count ← count + 1
    end if
    if OutFlows[t] ∈ F[Q] then
    count ← count + 1
    end if
    matchFlow = 100% ∗ (count / |F[P]| )
    if matchFlow =100% then
    totalMatch = matchFlow
    else if T[P] ∩ T[Q] > 0
        matchTask = (#(T[P] ∩ T[Q]) / T[P] )* 100%
    end if
                if taskType[t] = task
    extraTask = (#(t[P]-T[Q]) *0.5 / T[P]
    end if
                if taskType[t] = coordinator and
              coordinatorType[t] = fork
    extraFork = (#(t[P]-T[Q]) *0.8 / T[P]
    end if
                if taskType[t] = coordinator and
              coordinatorType[t] = Synchronizer
    extraSync = (#(t[P]-T[Q]) *1.0 / T[P]
    end if
    missingTask ← 0
    missingFork ← 0
    missingSync ← 0
        for T[Q] - T[P] do
                if taskType[t] = task
        missingTask = (#(t[Q]-T[P]) *1.5 / T[Q]
        end if
                if taskType[t] = coordinator and
              coordinatorType[t] = fork
        missingFork = (#(t[Q]-T[P]) *1.8 / T[Q]
        end if
                if taskType[t] = coordinator and
              coordinatorType[t] = Synchronizer
            missingSync = (#(t[Q]-T[P]) *2.0 / T[Q]
    end if
    return
    (matchFlow + matchTask) – ((extraTask + extraFork + extraSync) *
(# (T[P]-T[Q])/ (T[P] *100%)) – ((missingTask + missingFork +
missingSync) * (# (T[Q]-T[P])/ (T[Q] *100%))
```

4.4 Present Results

Using the above approach, the user can be presented with a concrete set of results from the process variant repository. In Table 1, we provide an example based on the variants presented in Fig 2 and the query W_a^Q presented in Fig 4. It is assumed that the threshold for task set intersection between the variants and query graph is set at 5 (i.e. $\#(T[P] \cap T[Q]) \geq 5$, thus all variants listed in Fig 2 will be included in the first filtering step (see section 4.2).

Table 1: Result of Structural Similarity Rank Computation.

Variant	Similarity Rank
W_c	80.47%
W_a	74.22%
W_d	35.79%
W_b	28.1%
W_e	8.63%

Based on the rank result in Table 1, process variant W_c (ATSC) carries the highest structural similarity rank which is 80.47%. The reasons for the high rank can be visually observed from Figs 2 & 5. A more subtle difference exists between W_d and W_b. The constructs of W_b seems visually more similar to W_a^Q than the process variant W_d. However, if we look closer, the matching flows between process variant W_d and the process query W_a^Q are higher. For example, there is a matching flow from task T4 to synchronizer in process variant W_d and there is also a matching flow from fork coordinator to task T3 but there is no such matching flows in process variant $W_{b.}$, and thus the higher structural similarity rank for $W_{d.}$

5 CONCLUSIONS

It is a challenging issue to find the degree of structural similarity between process variants and a given process query due to the complexity of the process graph semantics and different levels of structural similarity and partial match criteria that need to be taken into account. We have proposed a means to facilitate the search and retrieval of process variants that satisfy the structural criteria of a given process query. The dissimilarity degree rationalization introduced in this paper gives an intuitive weighting scheme to compute the different rankings between the process variant.

The results of the proposed method can be enhance the capability of process designers in their instance adaptation and process improvement endeavours due to the additional knowledge of precedent preferred and successful work practice embedded in process variants. In our future work we intend to utilize the proposed algorithm within a larger framework of multi-criteria. Although these extensions hold several challenges, it is envisaged that by providing querying capabilities across various properties of the variants will further improve the experience of process designers.

REFERENCES

Aalst, W. M. P. van der, Alves de Madeiros, A.K. and Weijters, A. J. M. M., *Process Equivalence: Comparing Two Process Models Based on Observed Behaviour*, BPM 2006, vol. 4102, 2006, pp. 129-144.

Chen, L., Gupta, A., Kurul, M. E., *Efficient Algorithms for Pattern Matching on Directed Acyclic Graphs*, IEEE Int. Conf. on Data Engineering (ICDE), Tokyo, Japan, March. 2005.

Koudas, N., Marathe , A., Srivastava D., *Flexible string matching against large databases in practice*. Proc. of the 13th VLDB, Morgan Kaufmann, 2004.

Koschmider, A., and Oberweis, A., *How To Detect Semantic Business Process Model Variants*?, In Proc. Of SAC '07, Korea, 2007.

Leymann, F. and Altenhuber W., *Managing Business Processes as an Information Resource*, IBM Systems Journal, vol. 33, 1994, pp. 326-348.

Lu, R., and Sadiq, S., *On the Discovery of Preferred Work Practice through Business Process Variants*. 26th International Conference on Conceptual Modeling (ER 2007) Nov 05-09, 2007.

Lu, R., and Sadiq, S., *A Reference Architecture for Managing Business Process Variants*. Proceedings of the 9th International Conference on Enterprise Information Systems (ICEIS2007) Funchal, Portugal, 2007.

Lu, R., and Sadiq, S., *Managing Process Variants as an Information Resource*. 4th International Conference on Business Process Management (BPM2006), Vienna, Austria, 2006.

Sadiq, S., Sadiq, W., Orlowska, M., *A Framework for Constraint Specification and Validation in Flexible Workflows*. Information Systems Vol.30/5, Jul 2005.

Sadiq,W. and Orlowska, M.E., *Analyzing Process Models using Graph Reduction Techniques*, Information System, vol.25,no.2, 2000, pp.117-134.

Shen, H.T., Zhou, X., Huang, Z., Shao, J., and Zhou, E., *UQLIPS: A Real-time Near-duplicate Video Clip Detection System*, In Proceedings of 33rd VLDB, 2007. (demo)

Weber , B., Rinderle, S., Reichert, M., *Change Patterns and Change Support Features in Process-Aware Information Systems*. CAiSE 2007: 574-588.