A Business Process Metric Based on the Alpha Algorithm Relations

Fabio Aiolli, Andrea Burattin*, and Alessandro Sperduti

Department of Pure and Applied Mathematics
University of Padua, Italy

Abstract. We present a metric for the comparison of business process models. This new metric is based on a representation of a given model as two sets of local relations between pairs of activities in the model. In order to build this two sets, the same relations defined for the Alpha Algorithm [2] are considered. The proposed metric is then applied to hierarchical clustering of business process models and the whole procedure is implemented and made publicly available.

1 Introduction

Process mining algorithms [1], designed for real world data, typically cope with noisy or incomplete logs. Because of that, many process models corresponding to different parameters settings can be generated, and the analyst very easily gets lost in such a variety of process models. In [6] a technique for the automatic discretization of the space of the values of the parameters and a technique for selecting one among all the possible models have been proposed. Presenting just a single output model, however, could not be enough informative for the analyst, so the problem is how to find a way of presenting only a small set of informative results, so that the analyst can either point out the one that better fits the actual business context, or extract general knowledge about the business process from a set of relevant extracted models. In this work, we propose a model-to-model metric that allows the comparison between business processes.

2 Comparing processes

The comparison of two business processes is not trivial as it requires to select those perspective that should be considered relevant for the comparison. For example, we can have two processes having same structure but different activity names: a human will detect the underlying similarity easily, while a machine will hardly be able to capture it.

Comparison of processes has been the focus of several papers, especially in the context of process composition (e.g. in the case of web services), process

* Andrea Burattin (*burattin@math.unipd.it*) is supported by SIAV S.p.A.

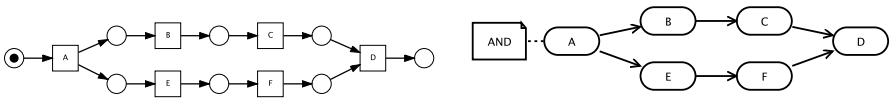
diagnosis and conformance between a reference model and the result of a process mining control-flow discovery algorithm. In the context of business process mining, the first papers to propose a process metric are [3, 12], where the underpinning idea is that models that differ on infrequent traces should be considered much more similar than models that differ on very frequent traces. In [10], the authors address the problem of detection of synonyms and homonyms that can occur when two business processes are compared and structural similarity is based on the hierarchical structure of an ontology. The work by Bae et al. [5] proposes to represent a process via its corresponding dependency graph. The paper [9] presents an approach for the comparison of models on the basis of "causal footprints", i.e. collections of the essential behavioral constraints that process models impose. The idea behind [8] tries to point out the differences between two processes so that a process analyst can understand them. The proposed technique exploits the notion of complete trace equivalence in order to determine differences. The work by Wang et al. [15] focusses on Petri nets, which are converted into corresponding coverability trees. The comparison is performed on the principal transition sequences. The paper [17] describes a process in terms of its "Transition Adjacency Relations" (TAR). The set of TARs describing a process is the set of pairs of activities that occur one directly after the other. The similarity measure is computed between the TAR sets of the two processes. It is defined as the ratio between the cardinality of the intersection of the TARs and the cardinality of the union of them. A recent work [16] proposes to measure the consistency between processes representing them as "behavioral profiles" that are defined as the set of strict order, exclusiveness and interleaving relations. The approach for the generation of these sets is based on Petri nets (their firing sequences) and the consistency of two processes is calculated as the amount of shared holding relations, according to a correspondence relation, that maps transitions of one process into transitions of the other.

The first step of our approach is to convert a process model into another formalism where we can easily define a similarity measure. We think that the idea of [17] can be refined to better fit the case of business processes. In that work, a process is represented by a set of TARs. Specifically, given a Petri net P, and its set of transitions T, a TAR $\langle a, b\rangle$ (where $a, b \in T$) exists if and only if there is a trace $\sigma = t_1 t_2 t_3 \dots t_n$ generated by P and $\exists i \in \{1, 2, \dots, n-1\}$ such that $t_i = a$ and $t_{i+1} = b$.

The main problem with this metric is that, for example, even if from a "trace equivalence" point of view two processes are the same, from a structural point of view (i.e., business processes) they are not.

2.1 Process representation and proposal for a metric

The idea here is to convert a given process model into two sets: one set of relations between activities that *must* occur, and another set of relations that *cannot* occur. In order to better understand the representation of business processes we are introducing, it is necessary to give the definition of *workflow trace*, i.e. the sequence of activities that are executed when a business process is performed. For



(a) Petri net representation (b) Dependency graph representation

Fig. 1: Example of process presented as a Petri net and as a dependency graph.

example, considering the process in Fig. 1 (it is the same process presented as a Petri net and as a dependency graph), the set of all the possible traces that can be observed is $\{ABCEFD, ABECFD, ABEFCD, AEBCFD, AEBFCD, AEFBCD\}$. We propose to represent such kind of processes by using relations $>$ and $\not>$ introduced in the Alpha Algorithm [2].

More formally, if a relation $A > B$ holds, it means that, in at least one of the workflow traces that the model can generate, activities A and B are adjacent: let W be the set of all the possible traces of a model, then there exists at least one trace $\sigma = t_1 \dots t_n \in W$, where $t_i = A$ and $t_{i+1} = B$ for some $i \in \{1, \dots, n-1\}$. The other relation $A \not> B$ is the negation of the previous one: if it holds, then, for any $\sigma = t_1 \dots t_n \in W$, there is no i for which $t_i = A$ and $t_{i+1} = B$. It is important to note that the above relations describe only local behaviors (i.e., they do not consider activities that occur far apart). Moreover, it must be noticed that our definition of $>$ is the same as the one used in [17]. These relations have been presented in [1, 11, 2] and are used by the Alpha Algorithm for calculating the possible causal dependency between two activities. In the case of mining, given a workflow log W, the algorithm finds all the $>$ relations and then, according to some predefined rules, these relations are combined to get more useful derived relations. The particular rules which are mined starting from $>$ are: *(i)* $A \rightarrow B$, iif $A > B$ and $B \not> A$; *(ii)* $A \# B$, iif $A \not> B$ and $B \not> A$; *(iii)* $A \| B$, iif $A > B$ and $B > A$. Here, the relations $>$ and $\not>$ will be called *primitive relations*, while $\rightarrow$, $\#$ and $\|$ will be called *derived relations*. The basic ideas underpinning these three rules are that (1) if two activities are observed always adjacent and in the same order, then there should be causal dependency between them ($\rightarrow$); (2) if two activities are never seen as adjacent activities, it is possible that they are not in causal dependency ($\#$) (3) if two activities are observed in no specific order, it is possible that they are in parallel branches ($\|$). The idea of this work is to perform a "reverse engineering" of a process in order to discover which relations must be observed and which relations cannot be observed in an ideal "complete log" (a log presenting all the possible behaviors). The Alpha Algorithm starts from the log (i.e. the set of traces) and extracts the primitive relations that are then converted into derived relations and finally into a Petri net model. In our approach that procedure is reversed: starting from a given model, derived relations are first extracted and then converted into primitive ones; the comparison between business process models is actually performed at this level. The main difference with respect to other approaches in the literature

(e.g. [16, 17]), is that our approach can be applied on every modeling language and not only Petri net or Workflow net. This is why our approach cannot rely on Petri net specific notions (such as firing sequence). We prefer to just analyze the structure of the process from a "topological" point of view. In order to face this problem, we decided to consider a process in terms of composition of well known patterns. Right now, a small but very expressive set of "workflow patterns" [14] are taken into account. When a model is analyzed, the following derived relations are extracted: *i)* a sequence of two activities A and B (pattern WCP-1[1]), will generate a relation $A \rightarrow B$; *ii)* every time an AND split is observed and activities A, B and C are involved (WCP-2) the following rules can be extracted: $A \rightarrow B$, $A \rightarrow C$ and $B \| C$; a similar approach can handle the AND join (WCP-3), generating a similar set of relations: $D \rightarrow F$, $E \rightarrow F$, $D \| E$; *iii)* every time an XOR split is observed (pattern WCP-4) and activities A, B and C are involved, the following rules can be extracted: $A \rightarrow B$, $A \rightarrow C$ and $B \# C$; a similar approach can handle the XOR join (WCP-5), generating a similar set of relations: $D \rightarrow F$, $E \rightarrow F$, $D \# E$. For the case of dependency graphs, this approach is formalized in Algorithm 1 of [4]: the basic idea being that given two activities A and B, directly connected with an edge, the relation $A \rightarrow B$ must hold. If A has more than one outgoing or incoming edges $(C_1, \ldots, C_n)$ then the following relations will also hold: $C_1 \rho C_2, \ldots, C_1 \rho C_n, \ldots, C_{n-1} \rho C_n$ (where ρ is $\#$ if A is a XOR split/join, ρ is $\|$ if A is an AND split/join). Once the algorithm has completed the generation of the set of holding relations, this can be split in two sets of positive an negative relations, according to the type of the "derived relations".

Given two processes $P_1 = (R^+, R^-)$ and $P_2 = (R^+, R^-)$, expressed in terms of positive and negative constraints, they are compared according to the amount of shared "required" and "prohibited" behaviors. A possible way to compare these values is the Jaccard similarity J and the corresponding distance J_δ, that are defined as $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$ and $J_\delta(A, B) = 1 - J(A, B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}$. In [13] it is proven that the Jaccard is actually a distance measure over sets. Our new metric is built considering the convex combination of the Jaccard distance for the set of positive and negative relations of two processes: $d(P_1, P_2) = \alpha J_\delta\left(R^+(P_1), R^+(P_2)\right) + (1-\alpha) J_\delta\left(R^-(P_1), R^-(P_2)\right)$ where $0 \leq \alpha \leq 1$ is a weighting factor that allows the user to calibrate the importance of the positive and negative relations. Since this metric is defined as a linear combination of distances (J_δ), it is a distance itself. It is important to note that there are couples of relations that are not "allowed" at the same time, otherwise the process is ill-defined and shows problematic behaviors, e.g. deadlocks[2]. Incompatible couples are defined as follows: *(i)* if $A \rightarrow B$ holds then $A \| B$, $B \| A$, $A \# B$, $B \# A$, $B \rightarrow A$ should not hold; *(ii)* if $A \| B$ holds then $A \# B$, $B \# A$, $A \rightarrow B$, $B \rightarrow A$, $B \| A$ should not hold; *(iii)* if $A \# B$ holds then $A \| B$, $B \| A$, $A \rightarrow B$, $B \rightarrow A$, $B \# A$ should not hold. Similarly, considering primitive relations, if $A > B$ holds then

[1] The pattern names are the same as in [14].

[2] It must be stressed that a process may be ill-defined even if no such couples of relations are present at the same time.

$A \not> B$ represents an inconsistency so this behavior should not be allowed.
Theorem 1. Two processes composed of different patterns, that do not contain duplicated activities and that do not have contradictions into their set of relations (either derived or primitive), have distance measure greater than 0.
Proof. See [4].
Since the sets of relations are generated without looking at the set of traces, but just starting from the local structure of the process model, if it is not sound (considering the Petri net notion of soundness) it is possible to have "contradictions". There is an important aspect that needs to be pointed out: in the case of contradictions, there may be an unexpected behavior of the proposed metric. For example, the two processes shown in Fig. 2 are "structurally different", but have distance measure 0. This is due to the contradictions contained in the set of primitive relations that are generated because of the contradictions on the derived relations (in both processes $B \| C$ and $B \# C$ hold at the same time). A comparison of values of the current metric and TAR is proposed in [4].

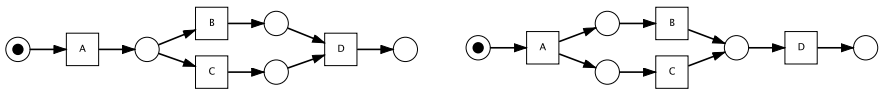


Fig. 2: Two processes that are different and contain contradictions in their corresponding set of relations: they have 0 distance measure.

Once the metric on business processes is available, it is possible to perform clustering. Since in general it is difficult to discover how many clusters are present in a set of items, we decided to use an agglomerative hierarchical clustering algorithm with, in this first stage, an average linkage (or average inter-similarity). The entire procedure has been implemented in PLG[3] [7], a software for the generation of random business processes.

3 Conclusions and future work

This work presented a new approach for the comparison of business processes. This approach relies on the conversion of a process model into two sets of relations: local relations that must hold; and local relations that must not hold. These two sets are generated starting from the relations of the Alpha Algorithm but, instead of starting from a log, the input is a process model. The proposed metric is based on the comparison of these two sets.

Future work will include further study about the case of contradictory relations as well as considering not only sets of primitive relations, but multisets of relations, eventually considering the distance between the labels of the activities.

[3] The PLG software is a free and open source software and can be downloaded at `http://www.processmining.it/sw/plg`.

References

1. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
2. van der Aalst, W.M.P., van Dongen, B.: Discovering Workflow Performance Models from Timed Logs. In: Engineering and Deployment of Cooperative Information Systems. pp. 45–63. Springer Berlin / Heidelberg (2002)
3. van der Aalst, W.M.P., de Medeiros, A.K.A., Weijters, A.J.M.M.: Process Equivalence: Comparing Two Process Models Based on Observed Behavior. In: Business Process Management. pp. 129–144 (2006)
4. Aiolli, F., Burattin, A., Sperduti, A.: A Metric for Clustering Business Processes Based on Alpha Algorithm Relations. Tech. rep. (2011), `http://www.processmining.it`
5. Bae, J., Liu, L., Caverlee, J., Zhang, L.J., Bae, H.: Development of Distance Measures for Process Mining, Discovery, and Integration. International Journal of Web Services Research 4(4), 1–17 (2007)
6. Burattin, A., Sperduti, A.: Automatic determination of parameters' values for Heuristics Miner++. In: IEEE Congress on Evolutionary Computation. pp. 1–8. IEEE, Barcelona, Spain (Jul 2010)
7. Burattin, A., Sperduti, A.: PLG: a Framework for the Generation of Business Process Models and their Execution Logs. In: 6th International Workshop on Business Process Intelligence Proceedings. pp. 214–219. Springer (2010)
8. Dijkman, R.: Diagnosing Differences between Business Process Models. In: Proceedings of the 6th International Conference on Business Process Management. pp. 261–277. Springer (2008)
9. van Dongen, B., Dijkman, R., Mendling, J.: Measuring Similarity between Business Process Models. Advanced Information Systems Engineering 5074, 450–464 (2008)
10. Ehrig, M., Koschmider, A., Oberweis, A.: Measuring Similarity between Semantic Business Process Models. In: Proceedings of the Fourth Asia-Pacific Conference on Conceptual Modelling. pp. 71–80 (2007)
11. Maruster, L., Weijters, A.J.M.M., van den Bosch, A., van der Aalst, W.M.P.: Process mining: discovering direct successors in process logs. Discovery Science pp. 364–373 (2002)
12. de Medeiros, A.K.A., van der Aalst, W.M.P., Weijters, A.J.M.M.: Quantifying process equivalence based on observed behavior. Data & Knowledge Engineering 64(1), 55–74 (Jan 2008)
13. Rajaraman, A., Ullman, J.D.: Mining of Massive Datasets (2010)
14. Russell, N., Ter Hofstede, A.H.M., van der Aalst, W.M.P., Mulyar, N.: Workflow control-flow patterns: A revised view. BPM Center Report BPM-06-22, BPMcenter. org (2006)
15. Wang, J., He, T., Wen, L., Wu, N., Ter Hofstede, A.H.M., Sun, J.: A Behavioral Similarity Measure between Labeled Petri Nets Based on Principal Transition Sequences. In: On the Move to Meaningful Internet Systems. pp. 394–401 (2010)
16. Weidlich, M., Mendling, J., Weske, M.: Efficient Consistency Measurement Based on Behavioral Profiles of Process Models. IEEE Transactions on Software Engineering 37(3), 410–429 (May 2011)
17. Zha, H., Wang, J., Wen, L., Wang, C., Sun, J.: A workflow net similarity measure based on transition adjacency relations. Comp. in Industry 61(5), 463–471 (2010)