Behavioral Comparison of Acyclic Business Process Models

Abel Armas-Cervantes[1], Paolo Baldan[2] and Luciano García-Bañuelos[1]

[1] Institute of Computer Science, University of Tartu, Estonia
{abel.armas,luciano.garcia}@ut.ee
[2] Department of Mathematics, University of Padova, Italy.
baldan@math.unipd.it

1 Introduction

This work presents an approach for behavioral comparison of acyclic process models – specifically acyclic ones as a starting point. Concretely, given two acyclic process models, we want to determine if they are behaviorally equivalent. If they are not equivalent, we want to highlight their differences using simple statements. For instance, if we consider the process models in Figure 1, we aim at providing the following statement: *"In the first process model (a) tasks a and e never appear in the same run; whereas, in the second process (b) there exists a run where both tasks occur"*. In order to derive such statements, we need to compare the behavior signature of each input process model, expressed in its more basic form: binary behavioral relations (e.g., causality, conflict, concurrency, etc.)

One such representation is given by Behavioral profiles [5]. In this representation, the behavior of a process model is encoded in a $n \times n$ matrix, where n represents the number of tasks in the process. Each cell in the matrix stores the behavior relation observed over the corresponding pair of tasks. However, behavioral profiles have major issues still to be resolved: a) mishandle the cases when there are duplicate tasks, b) they do not correspond to any well-accepted notion of behavioral equivalence – i.e., two models can have the same matrix representation even if their behavior is different–, and c) as a consequence of the above, it fails to diagnose various types of behavioral differences.

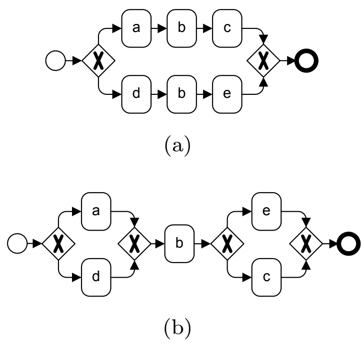


Figure 1: Running example

In light of the above, our research goals are the following. First, we aim at producing a representation of process behavior based on binary relations and producing accurate difference diagnostics under a well-accepted notion of equivalence. Secondly, we aim at producing diagnostics as short and intuitive as possible. We consider that the amount of statements in a diagnostic may be a factor compromising its understandability. Therefore, the representation adopted has to be as concise as possible, because the larger the representation is, the larger is the amount of difference diagnostic statements. Thus, an ideal representation would be one that is as close as possible to an $n \times n$ matrix but ensuring a well-accepted notion of equivalence.

2 Main definitions and results

As a starting point, we use event structures for representing the behavior of process models. Generally speaking, event structures are models of concurrency consisting of a set of events and behavioral relations between events. More specifically, our work relies on *prime* [3] and *flow event structures* [2], hereinafter abbreviated as PES and FES correspondingly, which are formally defined as follows:

Let E be a set of events and λ be a labeling function, such that λ associates each event to a label. The tuples $\mathbb{P} = \langle E, \leq, \#, \lambda \rangle$ and $\mathbb{F} = \langle E, \prec, \#, \lambda \rangle$ are **prime event structures** and **flow event structures**, respectively, where:

- $\leq$ is a partial order, known as *causal relation*, such that $\lfloor e \rfloor = \{e' \in E \mid e' \leq e\}$ is finite for all $e \in E$.
- $\#$ is a symmetric conflict relation. In the case of PES, $\#$ is irreflexive and hereditary with respect to causality, i.e., for all $e, e', e'' \in E$, if $e \# e' < e''$ then $e \# e''$.
- $\prec$ is an irreflexive non-transitive relation, known as the flow relation.

Figure 2 depicts the prime event structures of the processes in Figure 1. The arrows represent *causal* relations and the annotated dotted lines represent *conflict* relations. For the sake of simplicity, both transitive and hereditary relations were omitted in Figure 2. It can be noted that the PES of Figure 2(b), which corresponds to the process in Figure 1(b), contains two events with label c. The multiple occurrences of task c in the PES is due to the properties of the PES.

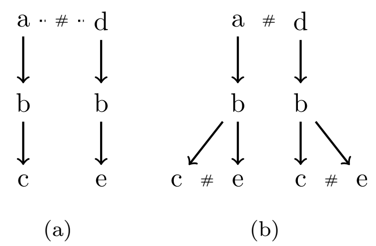


Figure 2: PES of processes in Fig. 1

FES are a more general type of event structures than PES. Indeed, every PES is also a FES [2]. The transformation of a PES into a FES is straightforward: the flow relation corresponds with transitive reduction of causality. Moreover, the conflict relation needs to be explicitly represented, as it is not longer hereditary because of the lack of transitive causality.

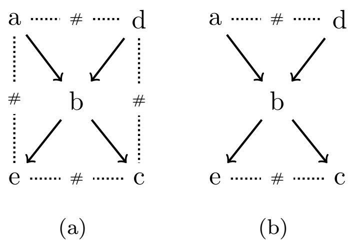


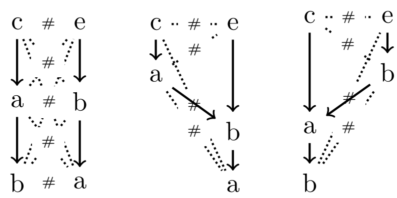
Figure 3: FES of processes in Fig.1

Due to the expressive power of flow event structures, multiple FES may exist to represent the same behavior, under *hp-bisimilarity* [4], a well-known notion of true concurrent equivalence for event structures. For instance, the FES depicted in Figure 3 represent the same behavior as those in Figure 2. In this context, our main contribution is the definition of a behavior-preserving folding operator, that allows us to reduce the size of the structure.

The aim of the operator is to find occurrences of the same task in a FES and replace them for only one occurrence, we refer to this set as *combinable set of events*. The defined folding operator establishes a set of restrictions that the combinable sets shall fulfill. Such restrictions are aligned to ensure the equivalence between a FES and its folded version according to hp-bisimilarity. The details about the folding operator, the restrictions for considering a set of events as combinable and proofs showing that the operator preserves hp-bisimulation are published in a technical report [1].

The folding operator can be applied repeatedly until no more combinable set of events is found. Ideally, the smaller is the event structure the more concise diagnostic would be. As a way of example, a diagnostic derived from the event structures in Figure 2 would contain at least 8 differences statements whereas the diagnostic derived from the event structures in Figure 3 would consist of only two statements. Indeed, from Figure 3 we can clearly see that *"tasks a and e never appear in the same run"*, whereas there is no restriction for a and e to appear in the same computation in (b), it can be interpreted as *"there is at least one run where tasks a and e can occur"*.

Although minimality seems an important property, canonicity is itself crucial for the purpose of model comparison. Unfortunately, the FES shown in Figure 4 provides a negative result in that respect: FES in Figure 4(b) and (c) are both hp-bisimilar to FES in Figure 4(a) and are minimal in size, but they are not isomorphic themselves. Therefore, further work is required to 1) determine if a canonical form can be properly defined and 2) determine the conditions to be observed during the folding to achieve the canonical form of an input FES.



(a) (b) (c)

Figure 4: FES and two of its possible foldings

We foresee three major axes for further work: 1) Fully characterizing a canonical form for FES and refining our method to reduce the size of any input FES to its canonical form, 2) Extending our approach to cover the cases of process models with cycles, and 3) evaluating empirically the usefulness of diagnostics derived with our method with business analysts using real-world process models.

References

[1] Abel Armas-Cervantes, Luciano Garca-Bauelos, and Paolo Baldan. Behavioral comparison of acyclic business process models. Internal report, Institute of Computer Science, University of Tartu, 2013.

[2] Gérard Boudol and Ilaria Castellani. Permutation of transitions: An event structure semantics for ccs and sccs. In *Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, School/Workshop*, pages 411–427, London, UK, UK, 1989. Springer-Verlag.

[3] Mogens Nielsen, Gordon D. Plotkin, and Glynn Winskel. Petri Nets, Event Structures and Domains, Part I. *Theoretical Computer Science*, 13:85–108, 1981.

[4] Rob van Glabbeek and Ursula Goltz. Equivalence notions for concurrent systems and refinement of actions. In Antoni Kreczmar and Grazyna Mirkowska, editors, *Mathematical Foundations of Computer Science 1989*, volume 379 of *Lecture Notes in Computer Science*, pages 237–248. Springer Berlin Heidelberg, 1989.

[5] Matthias Weidlich, Jan Mendling, and Mathias Weske. Efficient Consistency Measurement Based on Behavioral Profiles of Process Models. *IEEE Transactions on Software Engineering and Methodology*, 37(3):410–429, 2011.