Information Systems 47 (2015) 82–97

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

A new verification technique for large processes based on identification of relevant tasks

Richard Mrasek*, Jutta Mülle, Klemens Böhm

Karlsruhe Institute of Technology, KIT Institute for Program Structures and Data Organization, 76131 Karlsruhe, Germany

ARTICLE INFO

Article history:
Received 16 August 2013
Received in revised form
22 May 2014
Accepted 14 July 2014
Recommended by: G. Vossen
Available online 24 July 2014

Keywords:
Business process management
Business process modeling
Workflow modeling
Verification
Model checking
Petri net

ABSTRACT

Verification recently has become a challenging topic for business process languages. Verification techniques like model checking allow to ensure that a process complies with domain-specific requirements, prior to the execution. To execute full-state verification techniques like model checking, the state space of the process needs to be constructed. This tends to increase exponentially with the size of the process schema, or it can even be infinite. We address this issue by means of requirements-specific reduction techniques, i.e., reducing the size of the state space without changing the result of the verification. We present an approach that, for a given requirement the system must fulfill, identifies the tasks relevant for the verification. Our approach then uses these relevant tasks for a reduction that confines the process to regions of interest for the verification. To evaluate our new technique, we use real-world industrial processes and requirements. Mainly because these processes make heavy use of parallelization, full-state-search verification algorithms are not able to verify them. With our reduction in turn, even complex processes with many parallel branches can be verified in less than 10 s.

1. Introduction

Verification recently has become a challenging topic for business processes. The objective is to ensure their compliance with domain-specific requirements, i.e., rules and regulations. In industry, this is particularly important for processes specified in high-level modeling languages [1,2]. High-level process languages are languages like BPEL, BPMN, EPC or OTX that let the users design processes in a comfortable way. Verification allows to check characteristics of the behavior of a process schema prior to its execution. Verification can prove the presence of characteristics of a process, e.g., soundness [3], or their absence, like deadlocks [4] or irreducible cancellation regions [5]. More general approaches to specify complex requirements are worthwhile and do exist, notably model checking [6]. Users can find compliance violations in the process schema before they cause high financial costs. Tassey [7] reports that errors in software systems in general cause a financial damage of 58.5 billion US-Dollars per year only in the US, and companies could reduce the costs by 22.2 billion US-Dollars by using verification techniques. Verification techniques are quite mature but in some complex cases, as for example in the application domain we have studied, have performance issues.

For verification techniques like model checking to take place, the verification algorithm needs to construct the state space of the process. Most high-level process languages lack formal semantics that hinder the direct construction of the state space. But it is possible to transform these processes into a formal representation, e.g., Petri Nets that does allow the construction. However, the state space can increase exponentially with the size of the process

* Corresponding author.
E-mail addresses: richard.mrasek@kit.edu (R. Mrasek), jutta.muelle@kit.edu (J. Mülle), klemens.boehm@kit.edu (K. Böhm).

http://dx.doi.org/10.1016/j.is.2014.07.001
0306-4379/

schema, or it can even be infinite. This is well-known as state-space explosion [8]. It leads to unacceptable runtimes or renders the verification not executable. This is often caused by parallel branches in the schema. To overcome this problem, reduction techniques can be used, either (a) during construction of the state space or (b) on the level of the process schema already. Approaches like stubborn set reductions [9] fall into the first category. However, many of the industrial processes to be analyzed in our evaluation are too large to be verified only with stubborn set reductions. Even with stubborn set reduction, there are more than 1 million states in 78% of the processes we have evaluated; thus, verification has not been possible in reasonable time. Regarding (b), only few proposals exist, although preprocessing of the process schema is promising to achieve a significant reduction of the state space. An example is given in [10]. They specify the requirements in BPMN-Q. BPMN-Q is a visual language to query business process models. The approach of Reference [10] however is not sufficient to express all requirements from our real-world application scenario, see Section 6.3 for details. Furthermore, they apply reduction rules on the process schema in an iterative way. After each reduction step, another reduction rule may become again applicable. Thus, a rescan of the whole process may be necessary after each step, rendering this kind of approach expensive. In the industrial setting envisioned here, it is necessary to verify hundreds of requirements per process, in short time. In our approach, the requirements are dynamically generated at verification time from a database with context information on the testing processes, see Section 4. Many requirements are sequential and parallel ordering constraints that certain tasks need to fulfill. The processes are from a German car manufacturer. They contain between 200 and 1000 elements, arranged in up to 14 parallel lanes. Conventional techniques without reduction cannot verify these requirements for these processes. Compared to the processes dealt with by others [11], ours are much larger and more complex, leading to an exploding state space, see Section 6.1 for a comparison.

In this paper we present a new approach that verifies a process efficiently by exploiting the structure of the high-level process schema, see Section 4. The new algorithm traverses the process structure tree and identifies the regions of the process that are relevant for verification of a given complex requirement, e.g., defined in a temporal logic like CTL. Identifying the relevant regions of a process is far from trivial, see for a discussion Section 4.1. Even an elementary task cannot be removed in all cases, as we will explain, see Section 4.4. Our approach features a criterion for process-graph reduction, which we refer to as relevance function. The algorithm proposed creates a formal reduced representation of the process for each requirement, see Section 4.5. In particular, the reduction of parallel regions helps to decrease the size of the state space and hence the runtime of the verification, see Section 4.6. The approach is evaluated with industrial processes for testing newly produced vehicles in the factories of a German car manufacturer, see Section 2. One result is that even complex processes with many parallel branches can be verified in less than 10 s on a commodity PC, see Section 5.2. To demonstrate that the approach is not limited to one specific scenario we have applied our algorithm to a second use case. There, we have analyzed whether a process contains data-flow anti-patterns. The evaluation shows that our algorithm has been able to reduce the state space for this use case significantly as well. Thus, it also allows to analyze data-flow correctness for large processes, see Section 2.2.

The structure of the paper is as follows. Section 2 describes the use cases and the requirement collection phase. In Section 3 we introduce the notation, namely RPST (Refined Process Structure Tree), Petri Nets, and CTL (Computation Tree Logic). Section 4 describes our verification approach, in particular the notion of relevance for tasks, reducing the process tree and verifying the optimized process model. We describe the implementation and evaluate our approach in Section 5. In Section 6 we discuss related work. Section 7 concludes.

2. Scenarios and collection of requirements

The main use case of this paper is the verification of testing processes in the automobile industry. In Section 2.1.1 we briefly introduce this scenario. Section 2.1.2 contains the requirements and how they have been collected. In Section 2.2 we describe the second use case *Data-Flow Errors*.

2.1. Use case 1: testing processes

2.1.1. Scenario

In cooperation with a German automobile manufacturer, we use real processes that specify the sequential and parallel ordering of commissioning and testing tasks of an automobile, in short testing tasks, right after its assembly. These processes are the ones that the manufacturer does carry out in its factories worldwide. The processes are described in OTX notation (*O*pen *T*est sequence e*X*change) [12], a standard to specify testing workflows. The industrial partner has provided us with 40 processes. They contain between 6 and 813 elementary tasks. Because of the massive use of parallelization, the state space often (with 78% of the processes in our case) exceeds the size which is practically computable, see Section 5 for more details. The processes are executed at different testing stations in a factory. For each station, each vehicle series and each factory, process designers need to specify a process by hand. A new vehicle variation again requires a modification of the process. The processes for the same station are quite similar in size and complexity.

2.1.2. Collection of requirements

To collect the requirements on testing processes, we have conducted a series of about 10 interviews with the process modelers of our industrial partner during three months. The interviews have comprised a wide range of experiences from more common points up to detailed issues. Such detailed issues may result from concrete test processes in vehicle development, to give an example. The goal of the interviews has been to identify typical requirements that testing processes must fulfill. Example 1 illustrates the outcome of such an interview.