Towards Certified Data Flow Analysis of Business Processes

Thomas S. Heinze

Friedrich Schiller University Jena, 07743 Jena, Germany
t.heinze@uni-jena.de

Abstract. Data flow analysis allows for the static analysis of business processes. Certified data flow analysis would even allow for a trustwhorty analysis, as the analysis comes with a machine-checkable correctness proof. In this paper, we argue for a certified analysis of business processes.

1 Introduction and Motivation

Data flow analysis has shown to be a utility for business process modeling and analysis, be it for supporting formal verification [5,6], for analyzing data- or control-flow related process properties [2,11], or for optimization and re-engineering [8]. In particular two aspects make it such a useful method: (1) Data flow analysis provides a general framework which can be easily adjusted to new domains and analysis problems, (2) data flow analysis is grounded on well-founded theories like Kildall's lattice-theoretic formulation [7] or Cousot's *abstract interpretation* [4]. While the latter aspect offers a way to formally show an analysis to be correct, until now, only a small number of data flow analyses for business processes have been proven correct, mostly using pencil-and-paper proofs and just considering termination [2,5]. This is suprising considering the recent advances made in the verification of static analysis. State-of-the-art approaches define not only an analysis but rather add a mechanized, that is machine-checkable proof of its correctness. The term *certified analysis* has therefore been coined, as a user does not need to trust in such an analysis but can automatically check its conjoined correctness proof. Most notably, the verification of an optimizing C compiler [9] documented the power of the certified analysis approach.

In this position paper, we argue for the idea of a *certified analysis of business processes*. We believe the advantages to be manifold. For instance, the complexity of business process modeling languages like BPEL and BPMN make not only the design and implementation of processes, but also of analyses error-prone. An approach for proving analysis correctness thus helps in regaining confidence. This can be of vital importance considering, e.g., business processes in healthcare. Further, a mechanized correctness proof of a compliance analysis augments process auditing scenarios [1] with an orthogonal check that the audit itself can be trusted. Synthesizing an analysis from its correctness proof, even relieves the analysis designer from the implementation burden. In the following, we will introduce the concept of certified analysis based upon abstract interpretation, sketch challenges which arise when applied to business processes and finally outline our first steps towards a certified data flow analysis of business processes.

O. Kopp, J. Lenhard, C. Pautasso (Eds.): 9th ZEUS Workshop, ZEUS 2017, Lugano, Switzerland, 13-14 February 2017, published at http://ceur-ws.org/

2 Abstract Interpretation and Certified Analysis

Data flow analysis can be seen as *abstract interpretation* [4], where programs are evaluated using abstract values instead of concrete values. As an example, instead of performing arithmetic operations on integers when executing a program, an analysis may only track whether an integer has abstract value *odd* or *even*. In this way, e.g., addition boils down to four cases: $even + even = even$, $even + odd = odd$, $odd + even = odd$, and $odd + odd = even$. The domains of abstract and concrete values are connected by the concretization function γ, which maps abstract values to the represented concrete ones, e.g., $\gamma(odd) = \{1, 3, 5, \dots\}$. In addition, the abstract domain A should form a partially-ordered set, reflecting precision among abstract values such that $\forall a, b \in A\colon a \leq b \leftrightarrow \gamma(a) \subseteq \gamma(b)$, and contain a distinct *top* element with $\forall a \in A\colon a \leq top$, representing all concrete values. Executing a program's statements on concrete values can be formalized using Hoare's weakest precondition calculus, as in [3], which yields the concrete semantics. The effect of statements on abstract values is defined by the analysis in terms of a monotone function $f\colon A \to A$ mimicing, e.g., above's addition example, which in this way provides the abstract semantics. An analysis based upon abstract interpretation then calculates a fixpoint abstract value, i.e., $f(a) = a$, for a given program by continuously applying the abstract semantics to an initial abstract value until the fixpoint is reached (optionally using accelerators like widening/narrowing [4]).

An analysis can then be shown correct based upon abstract interpretation, iff:

$$\forall a \in A\colon c_{initial} \in \gamma(a) \to f(a) = a \to \forall c'\colon c_{initial} \rightsquigarrow c' \to c' \in \gamma(a)$$

meaning that a fixpoint a of the abstract semantics, that includes the initial concrete value $c_{initial}$ of the program in its concretization, also includes the concrete values c' of all reachable program states, $c_{initial} \rightsquigarrow c'$, in its concretization. Note that correctness here refers to soundness, i.e., the abstract value a is guaranteed to (over-)approximate the concrete value. The *Coq proof assistent*[1] has been shown of use for proving analysis correctness [3,9]. Its inductive types provide a way for encoding programs and its recursive functions for encoding concrete and abstract semantics. Proving analysis correctness in Coq may still require manual work, yet Coq allows for automatic proof validation and even to synthesize the analysis implementation[1], which justifies the notion of a *certified analysis*.

3 Research Challenges and First Steps

While the automated analysis for certifying business processes has been a recent research topic [1], we are not aware of any prior work on the certification of such an analysis for business processes itself. As indicated above, though, the area of static program analysis provides a rich spectrum of methods for proving analysis correctness. Yet, business processes and business process modeling languages feature certain peculiarities which challenge a certified analysis approach:

[1] `https://coq.inria.fr/` (link was last followed on February 1, 2017)

- First, as parallelism is essential to business processes, the concrete and abstract semantics need to reflect parallel executions. Application of *(concurrent) separation logic* [10] can thus be of advantage, extending the Hoare calculus with a rule for independently reasoning about parallel processes.
- Second, the phletora of language constructs in languages like BPEL or BPMN make formalizing the concrete semantics a tedious task. We thus opt for an approach, where we consider a *kernel* of basic language constructs, mapping all other constructs onto the kernel. This is a common method to boil down the complexity of a language [2] and furthermore provides a starting point for coping with the different and often mixed types of data flow definitions (XPath, Java, scripting languages, etc.) used in BPMN processes.
- Third, unstructured processes. While unstructuredness in case of sequential control flow alone can be handled by basing semantics on low-level jump operations instead of high-level ifelse or loop statements, mixed use of gateways as possible in BPMN, even more in the presence of the notoriously difficult to formalize (inclusive) OR gateway, provides for a real research challenge.

Currently, we are formalizing a toy language for business processes in Coq, supporting basic workflow patterns, e.g, sequence, parallel split/synchronize, exclusive choice/merge, and a data flow analysis for computing communication dependencies (see [2]). This language shall be enriched in future steps. Future work will also address other analyses, where we want to follow a modular approach using a general analysis framework, which is instantiated for a specific analysis.

References

1. Accorsi, R., Lowis, L., Sato, Y.: Automated Certification for Compliant Cloud-Based Business Processes. BISE 3(3), 145–154 (2011)
2. Amme, W., Martens, A., Moser, S.: Advanced verification of distributed WS-BPEL business processes. IJBPIM 4(1), 47–59 (2009)
3. Bertot, Y.: Structural Abstract Interpretation. In: Language Engineering and Rigorous Software Development, LNCS, vol. 5520, pp. 153–194. Springer (2008)
4. Cousot, P., Cousot, R.: Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs. In: POPL'77, Proc. pp. 238–252. ACM (1977)
5. Heinze, T.S., Amme, W.: Sparse Analysis of Variable Path Predicates Based upon SSA-Form. In: ISoLA'16, Proc. (1). LNCS, vol. 9952, pp. 227–242. Springer (2016)
6. Heinze, T.S., Amme, W., Moser, S.: Compiling More Precise Petri Net Models for an Improved Verification of Service Implementations. In: SOCA 2014, Proc. pp. 25–32. IEEE (2014)
7. Kildall, G.A.: A Unified Approach to Global Program Optimization. In: POPL'73, Proc. pp. 194–206. ACM (1973)
8. Kopp, O., Khalaf, R., Leymann, F.: Deriving Explicit Data Links in WS-BPEL Processes. In: SCC 2008, Proc. (2). pp. 367–376. IEEE (2008)
9. Leroy, X.: Formal Verification of a Realistic Compiler. Comm. of the ACM 52(7), 107–115 (2009)
10. O'Hearn, P.W.: Resources, Concurrency and Local Reasoning. In: CONCUR 2004, Proc. LNCS, vol. 3170, pp. 49–67. Springer (2004)
11. Saad, C., Bauer, B.: Data-Flow Based Model Analysis and Its Applications. In: MODELS 2013, Proc. LNCS, vol. 8107, pp. 707–723. Springer (2013)