SCIENCE CHINA
Information Sciences

• INSIGHT •
Special Focus on Advanced Technology of Software
May 2016, Vol. 59 050107:1–050107:3
doi: 10.1007/s11432-016-5557-1

Towards a bisimulation theory for open synchronized networks of automata

Eric MADELAINE[1]* & Min ZHANG[2]

[1]*INRIA, 2004 rte des Lucioles, Sophia-Antipolis 06902, France;*
[2]*Shanghai Key Laboratory of Trustworthy Computing, MOE Int^{al} Joint Laboratory of Trustworthy Software, ECNU, Shanghai 200062, China*

Received November 5, 2015; accepted January 26, 2016; published online April 13, 2016

Citation Madelaine E, Zhang M. Towards a bisimulation theory for open synchronized networks of automata. Sci China Inf Sci, 2016, 59(5): 050107, doi: 10.1007/s11432-016-5557-1

Dear editor,

Open parameterised Networks of synchronized automata (pNets) are new semantic objects that we propose for defining the semantics of simple or complex composition operators for distributed languages. Open pNets have an operational semantics, using open transitions that include symbolic hypotheses on the behavior of the pNets "holes". We discuss when this semantics can be finite and how to compute it symbolically. We define bisimulation equivalences between open pNets, and we explore under which hypotheses these equivalences are compatible with composition and are decidable.

In the 1990s, some research work extended the basic behavioral models based on labelled transition systems, to address value-passing or parameterised systems, using some symbolic encoding of the transitions. Lin [1] addressed value-passing calculi, for which he developed a symbolic behavioral semantics, and proved algebraic properties. Separately Rathke [2] defined another symbolic semantics for value-passing calculi, together with strong and weak bisimulation equivalences, and developed a symbolic model-checker based on a tableau method for these processes. In a different spirit, de Simone [3] used a variant of symbolic semantics to develop FH-bisimulations, as a tool to prove properties of open expressions in process calculi. Resink [4] compared many variants of open or symbolic bisimulations, aiming at expressiveness results. It appears that 30 years later, no practical verification platform uses this kind of approaches to provide proof methods for value-passing processes or open process expressions. The aim of our work is to define a new symbolic model, able to express in a finite way the behavior of both value-passing processes and open process expressions, and to decide bisimulation equivalences between expressions, with the help of external solvers.

Parameterised Networks of Synchronized Automata (pNets). pNets is a (low level) semantic model [5], built on top of labelled transition systems, extended with explicit data handling, and with a hierarchical structuring mechanism. It inherited from the work of Arnold and Nivat on synchronization vectors [6], but with the pragmatic concern of being used as the internal formalism of our specification and verification platform VerCors [7]. The parameterised and hierarchical nature of pNets allows for compact models easy to generate from high-level languages, while its static but unbounded structure allows for model-checking approaches even for dynamic/reconfigurable applications.

*Corresponding author (email: eric.madelaine@inria.fr)
The authors declare that they have no conflict of interest.

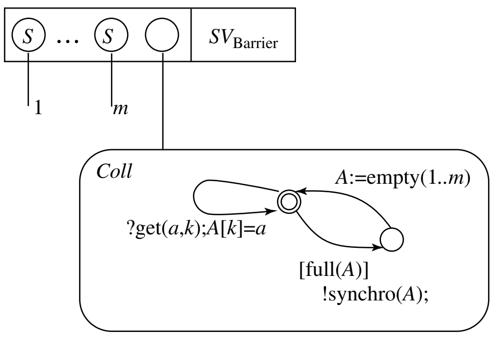

Figure 1   Exemple of a pNet structure.

PNets are tree-like structures (Figure 1), with *leaves*, *holes*, and *nodes*. Leaves are *parameterised labelled transition systems* (*pLTSs*), expressing the behavior of basic processes. Holes are placeholders for unknown processes, of which we only specify the set of possible actions, named *sort*. Nodes are synchronizing artifacts, using a set of *synchronization vectors* that express the possible synchronization between the parameterised actions of a subset of the sub-trees.

The formal definitions of pLTSs and (open) pNets would take too much space, and can be found in [5]. We give here a short example of an open pNet expressing a synchronization barrier: it has one pLTS "*Coll*", collecting messages from a (parameterised) number of holes labelled $1, \ldots, m$. Its set of synchronization vectors include m vectors synchronizing an action from each hole with the Coll pLTS, and another one exposing the collected result for synchronization in the upper level of pNets:

$\langle \ldots a \ldots$?get(a,k)$\rangle \rightarrow$?get(a,k), $\forall\ k \in [1..m]$, $a \in$ Sort(H_k)

$\langle$- . . . -!synchro(A)$\rangle \rightarrow$!synchro(A) where A contains the values.

PNets are flexible enough to express many different synchronization and communication mechanisms. In [5] we have defined a behavioral semantics for closed pNets (i.e., with no holes), in an early style, based on all possible instantiations of variables. We have defined a strong bisimulation equivalence, and proved that it is a congruence and compatible with a flattening operator. These results are used in our verification platform to build and minimize behavioral models in a compositional way. This provides a methodology for model-checking properties of distributed applications with good scalability.

What we aim at now is different: we want to address open systems, and prove their properties independently of processes that will instantiate their holes, manipulating symbolically the assumptions about the behavior of holes. To this aim, we need a behavioral semantics that does not depend on the shape of the pNets, and that keeps track of the assumptions in the transitions. The semantics of an open pNet is an *open automaton*, in which:

- States have the form $\langle s\_0, \ldots, s\_n \rangle$, in which s_i is a state of the pLTS at the *leave i* of the pNet,
- Transitions are rules specifying which leaves and which holes are involved in the transition, what is the global action, and containing a predicate relating the actions of leaves and holes with the global action, as illustrated in the example in Figure 2.

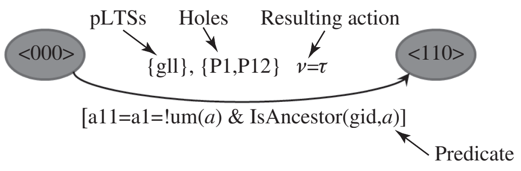

Figure 2   Structure of an open transition.

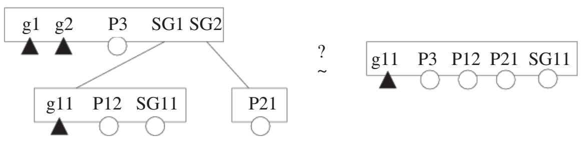

Figure 3   Bisimulation between 2 open pNets.

Building the open automaton of a pNet is done using a classical residual algorithm enumerating all possible transitions of each composite state. Open transitions are built using deduction trees in which each node is using one synchronization vector to combine the (open) transitions of some of the subnets of the pNet. By lack of space, we cannot give the full semantic rules for this construction here.

Proposition. If a pNet has a finite number of Holes and Leaves, and each pLTS at its leaves is finite-state and has finitely many (parameterised) transitions, then its open-automaton has a finite number of states and of open-transitions.

Defining equivalences. To compare open pNets, we define a strong bisimulation equivalence, between pNets having the same holes (see Figure 3), but using a flexible matching between open transitions. We name it FH-Bisimulation, "FH" being a short cut for "Formal Hypotheses" manipulated in the transitions. It also refers to the work of de Simone [3], which pioneered this line of research.

Definition 1 (FH-Bisimulation)**.** Let $A1$ and $A2$ be open automata whose set of holes are equal; let R be a symmetrical relation between states of $A1$ and $A2$. Then

R is a strong FH-bisimulation iff

$\forall\ (F,G) \in R, \forall$ open transition $F \xrightarrow{H,\mathrm{Pred}} F'$,
$\exists \{G \xrightarrow{H_i,\mathrm{Pred}_i} G'_i\}_{i=1..m}$ such that
$\mathrm{Pred} \subseteq \cup_i \mathrm{Pred}_i$ and $\forall\, i, H = H_i \wedge (F', G'_i) \in R$.

The intuition here is that open-transitions encode sets of transitions, for many possible values of the variables, so it is possible that one open-transition of F can be matched by several of G, provided that their conjunction covers the original set, and that all resulting states are equivalent (Figure 4).

Conjecture. We have not yet proven this result, but it is very likely that: (Decidability of FH-Bisimulation) Suppose $A1$ and $A2$ are finite open automata and the predicates inclusion is decidable. Then the FH-Bisimulation between them is decidable.

Discussion. To get full benefits from this approach, we still have to explore compositionality properties of the equivalence and to validate it on significant case-studies, e.g., in the domain of parallel skeletons for large-scale distributed programming. But also, it is clear that interesting properties will not be provable using strong bisimulation, because equivalence between different algorithms or implementations of a problem in a distributed context will necessarily involve different local, hidden actions. So we definitely need to study some forms of weak equivalences or weak refinements as well.

Then validation of the approach would start with proofs of algebraic laws from some specific calculi or languages; but more interestingly, we plan to study properties of some generic distributed algorithms or parallel programming skeletons that are not amenable with classical model-checking techniques. In this area, our proposal could be an interesting competitor to existing theorem-proving-based approaches. First steps will be to build a tool implementing the open-automaton generation and the bisimulation decision algorithms.

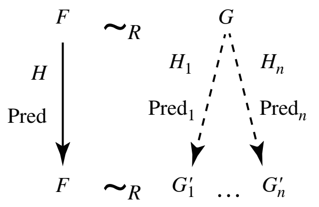

Figure 4 Schema of the FH-Bisimulation.

References

1 Lin H M. Symbolic transition graph with assignment. In: Proceedings of 7th International Conference on Concurrency Theory, Pisa, 1996. 50–65
2 Rathke J. Symbolic techniques for value-passing calculi. Dissertation for the Doctoral Degree. University of Sussex, 1997
3 de Simone R. Higher-level synchronizing devices in Meije-SCCS. Theor Comput Sci, 1985, 37: 245–267
4 Resink A. Bisimilarity of open terms. In: Proceedings of 4th Workshop on Expressiveness in Concurrency, Santa Margherita Ligure, 1997. 262–285
5 Henrio L, Madelaine E, Zhang M. pNets: an expressive model for parameterised networks of processes. In: Proceedings of 23rd Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), Turku, 2015. 492–496
6 Arnold A. Finite Transition Systems: Semantics of Communicating Systems. Hertfordshire: Prentice-Hall, 1994
7 Cansado A, Madelaine E. Specification and Verification for Grid Component-based Applications: from Models to Tools. In: Proceedings of 7th International Symposium on Formal Methods for Components and Objects, Sophia Antipolis, 2008. 180–203