Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 201 (2008) 255–264

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Refinement in the Presence of Unknowns

Heike Wehrheim[1],[2]

Institut für Informatik
Universität Paderborn
33098 Paderborn, Germany

Abstract

The standard theory of refinement assumes the systems/specifications/programs under consideration to be completely known: states and transitions are explicitly or implicitly given. This assumption fails to hold for systems that dynamically evolve over time, changing their composition as well as their components due to new requirements or new environmental conditions.
In this paper, we study refinement for such *partially known* systems. In our setting, partiality refers to transitions: transitions between states may be present or not present (the standard case) as well as unknown. This third possibility is expressed by using a three-valued logic for reasoning. We define a simulation relation on such partial transition systems, and show that the simulation problem on partial transition systems can be rephrased in terms of two simulation problems on complete transition systems, employing an *optimistic* and a *pessimistic* completion.

Keywords: Partial transition systems, simulation, three-valued logic, completions.

1 Introduction

Refinement is the central concept in a formal model-based development of systems involving a stepwise design on different levels of abstraction. In this setting, refinement takes on the role of a *correctness criterion*: a transformation of a model needs to preserve the overall behaviour; for a user the same functionality should be available after the change. Refinement is usually shown via simulation relations between the models/specifications before and after the transformation.

[1] This work was partially funded by the German Research Council DFG under grant WE 2290/6-1.
[2] Email: wehrheim@upb.de

doi:10.1016/j.entcs.2008.02.023

While there are different forms of refinement depending on the specification formalism used (e.g. data refinement for state-based specifications [5], process refinement or (bi-)simulation for behaviour-oriented formalisms [12],[10]), all these notions of refinement assume to have *complete* specifications of the system. This general assumption fails to hold for some classes of systems, for instance for certain kinds of self-* systems (e.g. self-organising, self-adapting, self-healing). Self-organising systems [8] can adapt to changing system states (e.g. failures) or environmental conditions, and thus may dynamically evolve over time reaching initially unknown states. This adaption concerns both single components as well as the composition of systems. It may lead to system models or specifications, which are only partially known. The partiality may concern both states (unknown values of some of the variables) as well as transitions (unknown transitions between states). This kind of partiality differs from the one investigated in [1], which refers to *views*. A view is describing one part or aspect of a system; views themselves are however complete in that everything is known about their states and/or transitions.

The phenomenon of partial specifications has already been studied in the area of model checking [4,2]. The underlying system models for model checking are (usually) Kripke structures, and partiality in Kripke structures might concern both the atomic propositions (in [2]) and the transitions (in [4]). Model checking algorithms on partial Kripke structures use three-valued logics (most often Kleene's logic $\mathcal{L}_3$ [6]) to formalise the unknown value: $\mathit{true}, \mathit{false}$ plus $\perp$ for unknown. In [4,3] this setting is even generalised to arbitrary multi-valued logics.

While model checking partial models is thus already a well-researched area, this is different for refinement. Bruns and Godefroid [2] define completeness preorders among Kripke structures via (bi)simulation-like relations in order to be able to compare partial structures (where partiality concerns atomic propositions only), with the objective to preserve temporal logic properties. However, refinement aiming at deriving an implementation out of an abstract specification is not their target. A bisimulation relation over multi-valued transition systems is also given in [7]. Refinement for partial specifications is studied in [11]. There, refinement relations are defined for the modal transition systems of [9] containing two kinds of transitions: *must* and *may* transitions. In contrast to the partial specifications treated here, may transitions do not correspond to unknown transitions, but to transitions which are allowed but do not have to be present in an implementation. Unknown transitions are transitions which – once their actual value becomes known - are either definitely there or definitely not there.

In this paper, we define a simulation relation between transition systems with unknown transitions. Since we do not know whether unknown transitions will actually be present, we might not succeed in showing that a fixed relation is or is not a simulation relation. Thus the outcome of a test for simulation also takes values of a three-valued logic: *true*, *false* or $\perp$ (unknown). We furthermore show that a test for simulation between partial transition systems can be reduced to two tests on complete transition systems, employing one completion which makes all unknown transitions *true* (an *optimistic completion*) and another one making all unknowns *false* (a *pessimistic completion*). This in in line with corresponding results about reducing multi-valued model checking problems to two (in case of three-valued logics) or more standard model checking problems.

2 Background

Before treating partial specifications we give the corresponding definitions for the complete case. /The behaviour of) our specifications is given by transition systems over a fixed set of actions *Act*. Transitions describe execution of actions, viz. operations of the system.

Definition 2.1 A *(complete) transition system* $T = (S, s_0, \longrightarrow)$ consists of

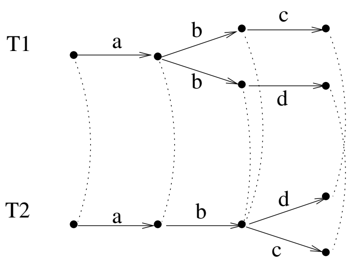
- a set of states S,
- an initial state s_0,
- and a transition relation $\longrightarrow \subseteq S \times Act \times S$.

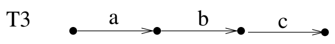
We use the notation T_i to stand for $(S_i, s_{0,i}, \longrightarrow_i)$. Simulation between transition systems is simply defined as one half of a *bisimulation* [10]: if T_2 simulates T_1 then it should be able to follow every step that T_1 is making. The similarity of T_1 and T_2 is recorded in a simulation relation $\mathcal{S}$ between the states of T_1 and T_2.

Definition 2.2 Let T_1, T_2 be complete transition systems, $\mathcal{S} \subseteq S_1 \times S_2$ a relation connecting states of T_1 with states of T_2.
$\mathcal{S}$ is a *simulation relation* between T_1 and T_2, $T_1 \leq_{\mathcal{S}} T_2$, if

- $(s_{0,1}, s_{0,2}) \in \mathcal{S}$ and
- $\forall (s_1, s_2) \in \mathcal{S}, a \in Act$:
  if $s_1 \xrightarrow{a}_1 s_1'$ then $\exists\, s_2' : s_2 \xrightarrow{a}_2 s_2' \wedge (s_1', s_2') \in \mathcal{S}$.

As an example, consider the following three transition systems T_1, T_2 and T_3 representing the behaviour of the CCS terms $a.(b.d + b.c)$, $a.b.(d + c)$ and $a.b.c$, respectively.





T_2 can simulate T_1 (but not vice versa) using the simulation relation depicted by dotted lines. Simulation is different from failures refinement [12]: T_2 is also a (stable) failures refinement of T_1 but T_3 being a failures refinement of T_1 as well cannot simulate T_1.

A given relation $\mathcal{S} \subseteq S_1 \times S_2$ can be checked for the simulation property: it either is or is not a simulation relation. We can thus rephrase the last definition as follows: for a pair $(s_1, s_2) \in \mathcal{S}$ we define

$$[s_1 \leq_{\mathcal{S}} s_2] = \begin{cases} \mathit{true} & \text{if } \forall\, a \in \mathit{Act} : s_1 \xrightarrow{a}_1 s_1 \Rightarrow \exists\, s_2' : s_2 \xrightarrow{a}_2 s_2' \wedge (s_1', s_2') \in \mathcal{S} \\ \mathit{false} & \text{else} \end{cases}$$

and for transition systems we define

$$[T_1 \leq_{\mathcal{S}} T_2] = \mathit{min}\{[s_1 \leq_{\mathcal{S}} s_2] \mid (s_1, s_2) \in \mathcal{S}\} \wedge (s_{0,1}, s_{0,2}) \in \mathcal{S}$$

Here, min refers to the minimal value in the ordering $\mathit{false} < \mathit{true}$. Thus min computes the conjunction over a set of boolean values which is then conjoined with the evaluation of the predicate $(s_{0,1}, s_{0,2}) \in \mathcal{S}$. In the next section we will generalise this to three-valued logics.

3 Simulation on partial transition systems

Next, we generalise this definition to partial transition systems. In a partial transition system we in addition have unknown transitions. This is described by assigning values $\{\mathit{false}, \bot, \mathit{true}\}$ from a three-valued logic to triples (s, a, s'). The third value $\bot$ stands for unknown: the transition might or might not be present in the real system, we just do not know. This is different to the symbol $\bot$ used as "undefinedness" in data refinement. There, $\bot$ stands for (the result of) a not yet specified operation which can be refined into a more specific one (even refined into an undefined operation again). Here, transitions labelled $\bot$ are in reality either present or not present, just from our point of view they are currently unknown (and in particular, unknown transitions cannot be simulated by unknown transitions, see below).

$\wedge$	1	$\perp$	0
1	1	$\perp$	0
$\perp$	$\perp$	$\perp$	0
0	0	0	0

$\vee$	1	$\perp$	0
1	1	1	1
$\perp$	1	$\perp$	$\perp$
0	1	$\perp$	0

$\Rightarrow$	1	$\perp$	0
1	1	$\perp$	0
$\perp$	1	$\perp$	$\perp$
0	1	1	1

Fig. 1. Truth tables for $\mathcal{L}_3$

Definition 3.1 A *partial transition system* $T = (S, s_0, R)$ consists of

- a set of states S,
- an initial state s_0,
- and a partial transition relation $R : (S \times Act \times S) \to \{true, false, \perp\}$.

For simplicity, we write $s \xrightarrow{a} s'$ for $R(s, a, s') = true$ and $s \overset{a}{\dashrightarrow} s'$ for $R(s, a, s') = \perp$ (and omit arrows for $R(s, a, s') = false$). Figure 1 gives the truth tables for conjunction, disjunction and implication in the three-valued logic (using 0 for *false* and 1 for *true*). It can be seen that conjunction computes the *minimal* value of its arguments and disjunction the *maximal* value using the ordering *false* $< \perp <$ *true*. Thus we also write *min* and *max* for conjunction and disjunction, respectively.

Partial transition systems can be completed by making decisions for unknown transitions. There are two such completions we will be particularly interested in: an *optimistic* completion assuming that there are indeed transitions wherever we have suspected so, and a *pessimistic* completion assuming the contrary.

Definition 3.2 Let $T = (S, s_0, R)$ be a partial transition system.
Its *optimistic completion* is $T^0 = (S, s_0, \longrightarrow)$ with $(s, a, s') \in \longrightarrow \Leftrightarrow R(s, a, s') \in \{true, \perp\}$, its *pessimistic completion* is $T^p = (S, s_0, \longrightarrow)$ with $(s, a, s') \in \longrightarrow \Leftrightarrow R(s, a, s') = true$.

The simulation relation should now - like in the complete case - guarantee that steps of T_1 can be matched by T_2. In case of unknown transitions, we can only have *true* as a result if under *all possible values for the unknowns* simulation is possible. Since this cannot always be guaranteed, one possible outcome of evaluating a given relation for simulation is also $\perp$. The outcome *false* on the other hand signals that the given relation can never be a simulation relation.

Figure 2 shows some simple examples of different forms of simulation relationships.

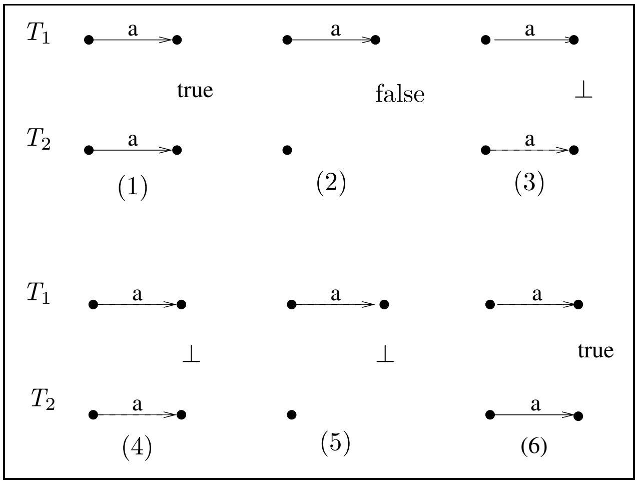


Fig. 2. Examples of different simulation relationships

- Cases (1) and (2) are the standard ones: a *true*-transition in T_1 can be matched by a *true*-transition in T_2 and cannot be matched by a *false*-transition.
- Case (3): if a *true*-transition in T_1 only has a $\perp$-transition in T_2, a simulation relationship is unknown (since we do not know whether the unknown transition is actually there).
- Case (4): an unknown transition in T_1 can also not be truely matched by an unknown transition in T_2 since it might turn out later that the transition in T_1 is present while that in T_2 is not. Thus the result is undetermined ($\perp$).
- A similar argument holds for case (5): we cannot definitely say that there is no simulation relation here, since the unknown transition in T_1 might not be present and then T_2 is able to simulate it (hence outcome $\perp$ again).
- Finally, case (6) is the positive outcome: an unknown transition can be simulated by an existing one.

Note that the outcome of evaluating for simulation is the *implication* between the truth value of the transition in T_1 and the transition in T_2.

This is now to be formally captured. We start with evaluating pairs of states under a given relation $\mathcal{S} \subseteq S_1 \times S_2$. Informally, given a pair (s_1, s_2) we have to check the matching for every transition (s_1, a, s_1'). The result of these checks have to be conjoined (we take the minimal value of the outcomes). While checking for one transition (s_1, a, s_1') we have to see what the best match is that we can find under all transitions (s_2, a, s_2'). Thus, here we take the maximal value of all results.

Definition 3.3 Let T_1, T_2 be partial transition systems, $\mathcal{S} \subseteq S_1 \times S_2$ a proposed matching relation and (s_1, s_2) a pair in $\mathcal{S}$.

$$\begin{aligned}&[s_1 \leq_{\mathcal{S}} s_2] = \\ &\quad min\ \{max\ \{R_1(s_1, a, s_1') \Rightarrow (R_2(s_2, a, s_2') \wedge (s_1', s_2') \in \mathcal{S}) \mid s_2' \in S_2\} \\ &\qquad\quad \mid a \in Act, s_1' \in S_1\}\end{aligned}$$

Lifting this to transition systems we get

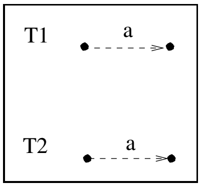
$$[T_1 \leq_{\mathcal{S}} T_2] = min\{[s_1 \leq_{\mathcal{S}} s_2] \mid (s_1, s_2) \in \mathcal{S}\} \wedge (s_{0,1}, s_{0,2}) \in \mathcal{S}$$

The result of evaluating for simulation on partial transition systems can also be obtained on complete transition systems, by looking at their optimistic and pessimistic completions.

Theorem 3.4 *Let T_1, T_2 be two partial transition systems. Then the following holds:*

$$[T_1 \leq_{\mathcal{S}} T_2] = \begin{cases} true & iff\ [T_1^o \leq_{\mathcal{S}} T_2^p] = true \\ false & iff\ [T_1^p \leq_{\mathcal{S}} T_2^o] = false \\ \perp & else \end{cases}$$

A simple reasoning can show that the simultaneous holding of $[T_1^o \leq_{\mathcal{S}} T_2^p] = true$ and $[T_1^p \leq_{\mathcal{S}} T_2^o] = false$ is not possible. We can thus use ordinary simulation to check for simulation relations on partial transition systems. Note that this result does not hold if we would let unknowns be simulated by unknowns. As an example consider the following two transition systems:



If unknowns could be simulated by unknowns, T_2 could simulate T_1 ($[T_1 \leq T_2]$ would be true). Using the above completions, the optimistic completion of T_1 contains an a-transition, the pessimitic completion of T_2 does not, hence $[T_1^o \leq T_2^p] = false$ which would then contradict the above result.

Proof of 3.4. 1. $[T_1 \leq_{\mathcal{S}} T_2] = true$ iff $T_1^o \leq_{\mathcal{S}} T_2^p$.

(i) $(s_{0,1}, s_{0,2}) \in \mathcal{S}$ has to hold for both sides.

(ii) The only transitions of interest are the unknown ones. From the left side we get for every $(s_1, s_2) \in \mathcal{S}$, $a \in Act$ and $s_1 \xrightarrow{a}_1 s_2$ that there is some s_2'

with $s_2 \xrightarrow{a}_2 s_2'$ and $(s_1', s_2') \in \mathcal{S}$. Hence there is a transition $s_1 \xrightarrow{a} s_1'$ in T_1^o and a transition $s_2 \xrightarrow{a} s_2'$ in T_2^p.
Reverse direction: if $s_1 \xrightarrow{a} s_1'$ in T_1^o is matched by $s_2 \xrightarrow{a} s_2'$ in T_2^p, then there is either a transition $s_1 \xrightarrow{a}_1 s_1'$ or $s_1 \overset{a}{\dashrightarrow}_1 s_1'$ in T_1 and $s_2 \xrightarrow{a}_2 s_2'$ in T_2. This holds for all $(s-1, s_2) \in \mathcal{S}$ and all $a \in Act$. Hence $[T_1 \leq_{\mathcal{S}} T_2]$.

2. $[T_1 \leq_{\mathcal{S}} T_2] = \mathit{false}$ iff $T_1^p \leq_{\mathcal{S}} T_1^o$.
   (i) $(s_{0,1}, s_{0,2}) \in \mathcal{S}$ either holds for both sides or no side.
   (ii) From left to right: $[T_1 \leq_{\mathcal{S}} T_2] = \mathit{false}$ means: $\exists (s_1, s_2) \in \mathcal{S}$, $a \in Act$ such that $s_1 \xrightarrow{a}_1 s_1'$ and $\neg \exists\, s_2' : (s_2 \xrightarrow{a}_2 s_2' \vee s_2 \overset{a}{\dashrightarrow}_2 s_2') \wedge (s_1', s_2') \in \mathcal{S}$. Then there is a transition $s_1 \xrightarrow{a} s_1'$ in T_1^p but still no corresponding transition in T_1^o.
   The reverse direction follows with a similar reasoning.

□

Finally, we come to the general definition of simulation, independent of particular representation relations.

Definition 3.5 Let T_1, T_2 be two partial transition systems. *Simulation* between T_2 and T_1 is defined as

$$[T_1 \leq T_2] = \mathit{max}\{T_1 \leq_{\mathcal{S}} T_2 \mid \mathcal{S} \subseteq S_1 \times S_2\}$$

Again, we use conjunction for composing the results of evaluating different relations for simulation: if one is suitable, T_2 can simulate T_1. Thus the question about simulation between partial transitions systems can be answered in three different ways: "definitely yes", "definitely no" or "don't know".

4 Example

Finally, we take a look at some examples. First of all, we have to see where the "unknowns" are actually coming from. There are several possibilities which can be thought of:

(i) A transition might be unknown because the specification/model of the system consists of different views, and these are contradictory as far as the specific transition is concerned: in one view it might be present, in another one it might not. The transition system for the complete system can thus only mark this transition as unknown.

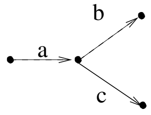
(ii) The transition system describes a self-adapting system. Initially, when the system is started, its behaviour can be described by a complete transition system, but afterwards it evolves (according to certain rules). This evolution might change certain transitions, but since we have no control over this change, we can only mark these transitions as unknown.

(iii) Our system is a parallel composition of a large number of components but we only know one component. This component shares variables and/or communication channels with other components. The behaviour of our component thus depends on other components (e.g. the possibility of a communication), and hence we can only partially determine its transition system.

We take the latter view as starting point for the following example. Our system consists of 3 components, out of which we only know one. The components communicate over channels using a CSP-like synchronisation. The first system T_1 is

$$[?_1 \;||_{\{a\}}\; a.(b+c) \;||_{\{b\}}\; ?_2]$$

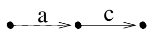
The transition system of the known component, *in the context* of the other two unknown components $?_1, ?_2$ is depicted below.



Next consider the following system T_2

$$[?_3 \;||_{\{a\}}\; a.c \;||_{\{b\}}\; ?_4]$$

The semantics of the known component in the context of $?_3$ and $?_4$ is



The check for a simulation relation gives us $\perp$ as result: we cannot definitely say that the second component can, nor that it cannot simulate the first (and a similar result is obtained if we just keep the first specification). This is different, if we instead take T_3 as system

$$[?_5 \;||_{\varnothing}\; a.(b+c) \;||_{\varnothing}\; ?_4]$$

Now synchronisation can never fail (since there is essentially no action to synchronise over), and thus all three transitions a, b and c are known to be there and a comparison with the first component according to simulation can be shown to yield *true*.

5 Conclusion

In this paper, we have defined a simulation relation between partial transition systems, where partiality refers to transitions. Both the transition systems and the outcome of a simulation check are formulated via a three-valued logic. We have discussed two different possibilities for simulation of unknowns: unknowns being simulated by unknowns, or unknowns only being simulated by knowns. The latter has been adopted as the definition (due to it being more realistic), and for this definition we have shown simulation on partial transition systems to coincide with two simulation checks on complete transition systems, using an optimistic and a pessimistic completion.

References

[1] Eerke Boiten, John Derrick, Howard Bowman, and Marten Steen. Consistency and refinement for partial specification in Z. In M.-C. Gaudel and J. Woodcock, editors, *FME'96: Industrial Benefit of Formal Methods, Third International Symposium of Formal Methods Europe*, volume 1051 of *Lecture Notes in Computer Science*, pages 287–306. Springer-Verlag, 1996.

[2] Glenn Bruns and Patrice Godefroid. Model checking partial state spaces with 3-valued temporal logics. In Nicolas Halbwachs and Doron Peled, editors, *CAV*, volume 1633 of *Lecture Notes in Computer Science*, pages 274–287. Springer, 1999.

[3] Glenn Bruns and Patrice Godefroid. Model checking with multi-valued logics. In Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella, editors, *ICALP*, volume 3142 of *Lecture Notes in Computer Science*, pages 281–293. Springer, 2004.

[4] Marsha Chechik, Steve M. Easterbrook, and Victor Petrovykh. Model-checking over multi-valued logics. In José Nuno Oliveira and Pamela Zave, editors, *FME*, volume 2021 of *Lecture Notes in Computer Science*, pages 72–98. Springer, 2001.

[5] John Derrick and Eerke Boiten. *Refinement in Z and Object-Z.* Springer-Verlag, 2001.

[6] Melvin Fitting. Kleene's three valued logics and their children. *Fundam. Inform.*, 20(1/2/3):113–131, 1994.

[7] Beata Konikowska and Wojciech Penczek. On Designated Values in Multi-valued CTL* Model Checking. *Fundam. Inform.*, 60(1-4):211–224, 2004.

[8] Jeff Kramer and Jeff Magee. Self-Managed Systems: an Architectural Challenge. In *ICSE 2007 - Future of Software Engineering Track.* ACM Press, 2007.

[9] Kim Guldstrand Larsen and Bent Thomsen. A modal process logic. In *LICS*, pages 203–210. IEEE Computer Society, 1988.

[10] Robin Milner. *Communication and Concurrency.* Prentice Hall, 1989.

[11] Shiva Nejati. Refinement Relations on Partial Specifications. Master's thesis, University of Toronto, 2003.

[12] A. W. Roscoe. *The Theory and Practice of Concurrency.* Prentice Hall, 1998.