

Comparing Disjunctive Modal Transition Systems with an One-Selecting Variant*

Heiko Schmidt and Harald Fecher

Christian-Albrechts-Universität zu Kiel, Germany
{hsc|hf}@informatik.uni-kiel.de

1 Introduction

Refinement and abstraction are common concepts for the specification, verification and analysis of programs. The two notions are dual to each other: Whenever a system \mathcal{C} refines another system \mathcal{A} , we call \mathcal{A} an abstraction of \mathcal{C} . Abstraction is a common technique in model checking to fight the state explosion problem [2]. Refinement is frequently used in model-driven software development, where the process of development starts with an abstract model, which is then refined in later design phases [5]. Formally, a refinement/abstraction setting consists of a class of models (e.g., labelled transition systems) and a refinement preorder on its elements. The minimal elements of the refinement preorder are called *concrete*.

An expressive refinement/abstraction setting is that of disjunctive modal transition systems (DMTS) [3]. Their concrete systems correspond to labelled transition systems. These systems allow hypertransitions, which are transitions having a single starting point, but several targets (combined with possibly different labels), as illustrated by the example in Figure 1(a). In DMTSs, hypertransitions are interpreted *disjunctively* (OR), i.e., *at least* one of the targets must appear in a concrete refinement. Thus systems (b), (c) and (d) are concrete refinements of (a).

This disjunctive approach lacks a straightforward possibility to express that exactly one of several implementations should be used in any refinement. For example, consider a model for a soda machine, in which it is not yet specified whether it dispenses 0.4 liters or 0.5 liters of soda. A concrete machine however should always give 0.4 liters or always give 0.5 liters and not allow randomly both possibilities, which could lead to overflowing or not completely filled glasses.

Contribution. We develop in Section 2 a variant of DMTSs called 1-selecting modal transition systems (1MTS) that interpretes hypertransitions *exclusively* (XOR), i.e., *exactly* one of the targets must appear in a concrete refinement. Thus, if (a) is interpreted as an 1MTS, systems (b) and (c) are concrete refinements, but (d) is not. Furthermore, in Section 3, we give transformations between DMTSs and 1MTSs that preserve the sets of concrete systems. These transformations enable the reuse of tools and yield the result that the two settings are equally expressive with respect to the sets of concrete systems. Additionally we show in Section 3 that 1MTSs have strictly more expressive power, if the whole refinement preorder is taken into account.

For technical details and proofs, we refer the reader to the first author’s diploma thesis [4].

2 DMTS and 1MTS

DMTSs and 1MTSs have the same syntax and differ only in their refinement definitions. For DMTSs, it is possible to implement any subset of the targets of a hypertransition. Thus, a refinement step may discard any number of targets, as long as *at least* a single one remains to be

* This work is in part financially supported by the DFG project *Refism* (FE 942/1-1).

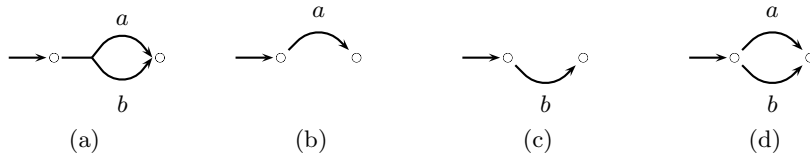


Fig. 1. Some DMTSs. The DMTSs (b), (c) and (d) are all concrete refinements of the DMTS (a). The systems can also be regarded as 1MTSs; then (b) and (c) are concrete refinements of (a), but (d) is not.

implemented. Refinement steps in 1MTSs should in principle treat hypertransitions similarly, but it must be made sure that in concrete refinements *exactly one* single alternative is implemented. We achieve this requirement using choice functions, i.e., functions mapping sets to elements inside the set (in this case, mapping outgoing hypertransitions to targets contained in the hypertransitions). Then for every choice function in the concrete state a choice function in a related abstract state has to be found such that the chosen targets are related. Consequently, (d) from Figure 1 is not an 1-selecting refinement of (a), because the concrete system (d) allows only a single choice function (every transition has exactly one target to choose) and we cannot find a suitable choice function in (a); we can only choose action a (thus finding a counterpart for the action a in (d)) or action b (thus finding a counterpart for the action b in (d)), but not both, as it would be required.

In the first author's diploma thesis [4], the two refinement definitions are illustrated by full characterisations of all disjunctive and 1-selecting refinements of the simple system shown in Figure 1(a) (up to refinement equivalence, where two systems are called refinement equivalent if they refine each other in both directions). It turns out that the simple DMTS has surprisingly many (disjunctive) refinements; besides concrete ones equivalent to (b), (c) and (d) there are ten more equivalence classes of non-concrete refinements. If interpreted as 1MTS, the system (a) only has four equivalence classes of refinements, two of which have representants (b) and (c).

3 Comparison

We consider two approaches to compare DMTSs and 1MTSs, the first of which is based on the the sets of concrete refinements:

Definition 1 (Implementation-based comparison). *A function f from one setting A to another setting B is an implementation-based embedding, if it preserves the sets of concrete refinements, i.e., the set of concrete refinements of each abstraction a in A equals the set of concrete refinements of $f(a)$ in B .*

These kinds of transformation are satisfactory in many cases. For example, if an algorithm for generalised model checking [1] is established for one setting, satisfaction in the other setting can be checked as well, by first applying the transformation. This works for generalised model checking, because in this technique an abstraction satisfies a given formula iff each concrete refinement satisfies the formula.

Theorem 2. *There are implementation-based embeddings from DMTS to 1MTS and vice versa.*

This implies that the two settings can express the same sets of concrete refinements. However, note that the approach based on implementations does not take the full structure of the refinement preorder into account: Only refinement to concrete systems, i.e., the minimal elements,

has to be preserved, whereas the relations between larger elements are ignored. However, the structure of the refinement preorder could be of interest, for example, when a compositional satisfaction relation is given that approximates the validity of formulae in abstractions (an abstraction might not satisfy a formula, although all its concrete refinements do). A richer structure could imply better approximation without an increase in the complexity of algorithms. Thus we consider a finer comparison approach by comparing the refinement preorders by use of order homomorphisms (if a refines b , then $f(a)$ refines $f(b)$) and embeddings (a refines b if and only if $f(a)$ refines $f(b)$) and the additional requirement that any homomorphism should keep concrete systems fixed.

Definition 3 (Preorder-based comparison). *A function f from one setting A to another setting B is an preorder-based homomorphism (resp. embedding), if*

1. *f induces an order homomorphism (resp. embedding) on the refinement equivalence classes, and*
2. *for all concrete systems a of A , we have that a and $f(a)$ are refinement equivalent.*

Note that every preorder-based embedding is also an implementation-based embedding, which however need not hold for preorder-based homomorphisms.

Theorem 4. *There is a preorder-based embedding from DMTS to 1MTS.*

The idea is to turn a DMTS-hypertransition with n targets into n 1MTS-hypertransitions, each with all n targets. Then it becomes possible to choose any subset of targets, even when using the “exactly one”-interpretation of 1MTSs.

Theorem 5. *There is a preorder-based homomorphism from 1MTS to DMTS.*

Here, the idea is to turn every 1MTS-state s into several DMTS-states (s, γ) , one for each possible choice function γ . Then (s, γ) plays the role of s in case hypertransition targets in the 1MTS are chosen according to choice function γ . This homomorphism even preserves the sets of concrete systems, i.e., it is an implementation-based embedding. However, the homomorphism is not a preorder-based embedding. We even show the following:

Theorem 6. *There is no preorder-based embedding from 1MTS to DMTS.*

This is done by proving that the refinement structure of the 1MTS shown in Figure 1(a) cannot be embedded in the refinement structure of any DMTS.

To sum up, 1MTSs and DMTSs have the same expressive power when compared using the implementation-based approach, but 1MTSs are more expressive when compared using the preorder-based approach. Thus 1MTSs not only support more straightforward modelling of exclusive alternatives, they also have a richer refinement structure. This has the potential to be useful when considering compositional, approximative logics, which is however future work and out of the scope of this paper.

References

- [1] G. Bruns and P. Godefroid. Generalized model checking: Reasoning about partial state spaces. *LNCS*, 1877:168–182, 2000.
- [2] E. M. Clarke, O. Grumberg, and D. E. Long. Model checking and abstraction. *ACM Transactions on Programming Languages and Systems*, 16(5):1512–1542, September 1994.
- [3] K. G. Larsen and L. Xinxin. Equation solving using modal transition systems. In *LICS '90*, pages 108–117. IEEE Computer Society Press, 1990.
- [4] H. Schmidt. Comparing disjunctive modal transition systems with their one-selecting variant, 2006. Diploma thesis. Available at <http://www.informatik.uni-kiel.de/~hsc/diplomarbeit.pdf>.
- [5] N. Wirth. Program development by stepwise refinement. *Communications of the ACM*, 14(4):221–227, 1971.