Specification and Verification of Modal Properties for Structured Systems[⋆]

Andrea Vandin

IMT Institute for Advanced Studies Lucca, Italy
{andrea.vandin}@imtlucca.it

1 Problem Statement

System specification formalisms should come with suitable property specification languages and effective verification tools. We sketch a framework for the verification of quantified temporal properties of systems with dynamically evolving structure. We consider visual specification formalisms like graph transformation systems (GTS) where program states are modelled as graphs, and the program behaviour is specified by graph transformation rules. The state space of a GTS can be represented as a graph transition system (GTrS), i.e. a transition system with states and transitions labelled, respectively, with a graph, and with a partial morphism representing the evolution of state components. Unfortunately, GTrSs are prohibitively large or infinite even for simple systems, making verification intractable and hence calling for appropriate abstraction techniques.

2 State-of-the-art in GTS Logics

After the pioneering works on monadic second-order logic (MSO) [7], various graph logics have been proposed and their connection with topological properties of graphs investigated [8]. The need to reason about the evolution of graph topologies has then led to combining temporal and graph logics in propositional temporal logics using graph formulae as state observations (e.g. [4]). However, due to the impossibility to interleave the graphical and temporal dimensions it was not possible to reason on the evolution of single graph components. To overcome this limitation, predicate temporal logics were proposed (e.g. [2, 16]), where edge and node quantifiers can be interleaved with temporal operators.

More recent approaches [2] propose quantified μ-calculi combining the fixpoint and modal operators with MSO. These logics fit at the right level of abstraction for GTSs, allowing to reason on the topological structure of a state, and on the evolution of its components. We refer to § 8 of [11] for a more complete discussion. Unfortunately, the semantical models for such logics are less clearly cut. Current solutions are not perfectly suited to model systems with dynamic structure, where components might get merged [2, 16], or (re)allocated [2]. These problems are often solved by restricting the class of admissible models or by reformulating the state transition relation, hampering the meaning of the logic.

⋆ Partly supported by the EU FP7-ICT IP ASCENS and by the MIUR PRIN SisteR.

3 State-of-the-art in GTS Verification

Various approaches have been proposed for the verification of GTSs, often adopting traditional techniques (e.g. model checking) to the area of graph transformation. We mention two research lines that have integrated the proposed techniques in verification tools, namely GROOVE [5, 12, 15, 17] and AUGUR [1–4, 13][1].

The model checking problem for GTSs is in general not decidable for reasonably expressive logics, since GTSs are Turing complete languages. Pragmatically, GTSs are often infinite-state and it is well known that only some infinite-state model checking problems are decidable. Several approximation techniques inspired to abstract interpretation have thus been proposed, the main idea being to consider a *finite-state* system approximating an *infinite-state* one. In order to be meaningful, those approximations may be related with the original systems via behavioural relations. The above mentioned research lines developed approximation techniques: namely *neighbourhood abstractions* [5], and *unfoldings* [1–4].

4 Current Contributions

In [10, 11] we introduced a novel semantics for quantified μ-calculi. We defined *counterpart models*, generalizing GTrSs, where states are algebras and the evolution relation is given by a family of partial morphisms. One of the main characteristics of our approach is that formulae are interpreted over sets of pairs (w, σ_w), for w a state and σ_w an assignment from formula variables to components of w. This allows for a straightforward interpretation of fixed points and for their smooth integration with quantifiers, which often asked for a restriction of the class of admissible models. Our proposal avoids the limitations of existing approaches, in particular in what regards merging and name reuse. Moreover it dispenses with the reformulation of the transition relation, obtaining a streamlined and intuitive semantics, yet general enough to cover the alternatives we are aware of.

In [14] we presented a first step towards a tool support for our approach, preparing the ground for an efficient tool framework. We first presented a Maude [6] implementation of graph rewriting as conditional rewrite rules on object multisets. Then we introduced a prototypal model checker for finite counterpart models. Our tool allows to analyze the evolution of individual components, and, as far as we know, it is one of the few model checkers for quantified μ-calculi.

Finally, [9] proposes a general formalization of similarity-based counterpart model approximations, and a technique for approximated verification exploiting them. We extended and generalized in several directions the type system of [4], proposed within the unfolding technique to classify formulae as preserved or reflected by a given approximation: (i) our type system is *technique-agnostic*, meaning that it does not require a particular approximation technique; (ii) we consider counterpart models, a generalization of GTrSs; (iii) our type system is parametric with respect to a given simulation relation (while the original one

[1] See groove.cs.utwente.nl and www.ti.inf.uni-due.de/research/tools/augur2.

considers only those with certain properties); (iv) we use the type system to reason on all formulae (rather than just on closed ones); and (v) we propose a technique that exploits approximations to estimate properties more precisely, handling also part of the untyped formulae.

References

1. Baldan, P., Corradini, A., König, B.: A static analysis technique for graph transformation systems. In: Larsen, K., Nielsen, M. (eds.) CONCUR. LNCS, vol. 2154, pp. 381–395. Springer (2001)
2. Baldan, P., Corradini, A., König, B., Lluch Lafuente, A.: A temporal graph logic for verification of graph transformation systems. In: Fiadeiro, J.L., Schobbens, P.Y. (eds.) WADT. LNCS, vol. 4409, pp. 1–20. Springer (2007)
3. Baldan, P., König, B.: Approximating the behaviour of graph transformation systems. In: Corradini, A., Ehrig, H., Kreowski, H.J., Rozenberg, G. (eds.) ICGT. LNCS, vol. 2505, pp. 14–29. Springer (2002)
4. Baldan, P., König, B., König, B.: A logic for analyzing abstractions of graph transformation systems. In: Cousot, R. (ed.) SAS. LNCS, vol. 2694, pp. 255–272. Springer (2003)
5. Bauer, J., Boneva, I., Kurbán, M.E., Rensink, A.: A modal logic based graph abstraction. In: Ehrig, H., Heckel, R., Rozenberg, G., Taentzer, G. (eds.) ICGT. LNCS, vol. 5214, pp. 321–335. Springer (2008)
6. Clavel, M., Durán, F., Eker, S., Lincoln, P., Martí-Oliet, N., Meseguer, J., Talcott, C.L.: All about Maude, LNCS, vol. 4350. Springer (2007)
7. Courcelle, B.: The expression of graph properties and graph transformations in monadic second-order logic. In: Rozenberg, G. (ed.) Handbook of Graph Grammars and Computing by Graph Transformation, pp. 313–400. World Scientific (1997)
8. Dawar, A., Gardner, P., Ghelli, G.: Expressiveness and complexity of graph logic. Information and Computation 205(3), 263–310 (2007)
9. Gadducci, F., Lluch Lafuente, A., Vandin, A.: Exploiting over- and under- approximations for infinite-state counterpart models. This volume
10. Gadducci, F., Lluch Lafuente, A., Vandin, A.: Counterpart semantics for a second-order μ-calculus. In: Ehrig, H., Rensink, A., Rozenberg, G., Schürr, A. (eds.) ICGT. LNCS, vol. 6372, pp. 282–297. Springer (2010)
11. Gadducci, F., Lluch Lafuente, A., Vandin, A.: Counterpart semantics for a second-order mu-calculus. Fundamenta Informaticae 118(1-2) (2012)
12. Ghamarian, A.H., de Mol, M., Rensink, A., Zambon, E., Zimakova, M.: Modelling and analysis using GROOVE. STTT 14(1), 15–40 (2012)
13. König, B., Kozioura, V.: Counterexample-guided abstraction refinement for the analysis of graph transformation systems. In: Hermanns, H., Palsberg, J. (eds.) TACAS. LNCS, vol. 3920, pp. 197–211. Springer (2006)
14. Lluch Lafuente, A., Vandin, A.: Towards a Maude tool for model checking temporal graph properties. In: Gadducci, F., Mariani, L. (eds.) GT-VMT. ECEASST, vol. 42. EAAST (2011)
15. Rensink, A.: Towards model checking graph grammars. In: Leuschel, M., Gruner, S., Lo Presti, S. (eds.) AVOCS. DSSE-TR, vol. 2003-2. University of Twente
16. Rensink, A.: Model checking quantified computation tree logic. In: Baier, C., Hermanns, H. (eds.) CONCUR. LNCS, vol. 4137, pp. 110–125. Springer (2006)
17. Rensink, A., Zambon, E.: Neighbourhood abstraction in GROOVE. In: de Lara, J., Varró, D. (eds.) GraBaTs. ECEASST, vol. 32. EAAST (2010)