Some Perspectives of Infinite-State Verification

Wolfgang Thomas

RWTH Aachen, Lehrstuhl Informatik 7, 52056 Aachen, Germany
thomas@informatik.rwth-aachen.de

Abstract. We report on recent progress in the study of infinite transition systems for which interesting properties (like reachability of designated states) can be checked algorithmically. Two methods for the generation of such models are discussed: the construction from simpler models via operations like unfolding and synchronized product, and the internal representation of state spaces by regular sets of words or trees.

1 Introduction

The method of model-checking has developed largely in the domain of finite system models, and its success in industrial applications is built on highly efficient data structures for system representation. Over infinite models, the situation is different, and for practical applications the field is still in its beginnings. Even simple properties may be undecidable over infinite state spaces, and thus a careful preparatory analysis is necessary in order to determine the possible range of fully automatic verification.

The purpose of the present short survey is to report on some techniques which yield classes of infinite models such that the model-checking problem is decidable for interesting properties. Our presentation is far from complete; it is biased towards results which were obtained in the author's research group and collaborations with other groups (mostly that of D. Caucal, Rennes). We focus on system models in the form of edge-labelled transition graphs; thus a central aspect is the investigation of structural properties of infinite graphs. An alternative and equally fundamental approach for introducing infinite models, which is not discussed in this paper, is to extend finite transition graphs by infinite data structures, for example over the natural or real numbers (as in timed systems).

Transition graphs are considered in the format $G = (V, (E_a)_{a \in \Sigma})$ where V is the set of states (vertices) and where E_a (for a symbol a from a finite alphabet Σ) is the set of a-labelled edges. We write E for the union of the E_a. State-properties may be introduced by subsets V_a of V, where a is from a second label alphabet Γ.

The logics we consider allow to express the reachability relation E^*, the reflexive transitive closure of E, since reachability is the most fundamental property arising in verification. A prominent logic of this kind is monadic second-order logic MSO. It encompasses most standard temporal logics. On the other end,

D.A. Peled and Y.-K. Tsay (Eds.): ATVA 2005, LNCS 3707, pp. 3–10, 2005.

as a kind of minimal logic in this context, we consider FO(R) ("first-order logic with reachability"), the extension of first-order logic by a relation symbol for E^*.

We shall address two methods for constructing infinite transition graphs where model-checking (with respect to MSO or FO(R)) is decidable. First we review the effect of fundamental model constructions – namely, interpretation, unfolding, and synchronized product – on the existence of model-checking procedures. Secondly, we discuss model-checking as based on "regular" internal representations of infinite transition graphs, using finite automata over strings or trees, respectively.

2 Operations on Graphs

2.1 Interpretations

Rabin's Tree Theorem [19] states that the MSO-theory of the infinite binary tree T_2 is decidable (or in other terminology: that model-checking the binary tree with respect to MSO-properties is decidable). We can view T_2 as a graph $(\{1,2\}^*, S_1, S_2)$, where $\{1,2\}^*$ is the set of vertices and S_1, S_2 the successor relations with $S_i = \{(v, vi) \mid v \in \{1,2\}^*\}$. Many other theories were shown decidable (already in [19]) using interpretations in the tree T_2. To show that the model-checking problem for a structure S with respect to formulas of a logic L is decidable one proceeds as follows: One gives an MSO-description of S within the binary tree T_2, and using this one provides a translation of L-formulas φ into MSO-formulas φ' such that $S \models \varphi$ iff $T_2 \models \varphi'$. Taking $L =$ MSO, we see that an MSO-interpretation (i.e., a model description using MSO-formulas) preserves decidability of model-checking with respect to MSO-formulas.

As a simple example of interpretation consider the n-ary branching tree T_n (for $n > 2$), with vertices in the set $\{1, \ldots, n\}^*$ rather than $\{1,2\}^*$ as for T_2. We may represent the vertex $i_1 \ldots i_r$ of T_n by $1^{i_1}2 \ldots 1^{i_r}2$ in T_2. It is easy to give an MSO-definition of the range of this coding in T_2 and to supply the translation $\varphi \mapsto \varphi'$ as above. As a second example, consider a pushdown automaton A with stack alphabet $\{1, \ldots, k\}$ and states $q_1, \ldots, q_m$. Let $G_A = (V_A, E_A)$ be its configuration graph; here V_A consists of A-configurations $(q_j, i_1 \ldots i_r)$ (with state q_j and stack content $i_1 \ldots i_r$, reading i_1 as top symbol), and we restrict to those configurations which are reachable from the initial one (say $(q_1, 1)$). The edge relation E_A is the one-step transition relation of A between configurations. Choosing $n = \max(k, m)$, we can exhibit an MSO-interpretation of G_A in T_n: Just represent configuration $(q_j, i_1 \ldots i_r)$ by the vertex $i_r \ldots i_1 j$ of T_n. Note that then the A-steps lead to local moves in T_n, from one T_n-vertex to another, e.g. in a push step from vertex $i_r \ldots i_1 j$ to a vertex $i_r \ldots i_1 i_0 j'$. These moves are easily definable in MSO, and reachability (from the initial vertex 11) as well. Due to this interpretation, we obtain the fundamental result of Muller and Schupp ([18]): *For the configuration graph of a pushdown automaton, checking MSO-properties is decidable.*

It is known that the ε-closures of the pushdown transition graphs capture precisely those graphs which are MSO-interpretable in T_2 (or equivalently in T_n);

see Section 3 below. We do not consider here a slightly more general version of MSO-interpretation, the "MSO-definable transduction" in the sense of Courcelle [7]; such a transduction from S to T involves a description of S in a k-fold copy of T rather than T itself.

2.2 Unfoldings

In the previous section we explained how to generate a model "within" a given one, via defining formulas. A more "expansive" way of model construction is the unfolding of a graph $(V, (E_a)_{a\in\Sigma})$ from a given vertex v_0, yielding a tree $T_G(v_0) = (V', (E'_a)_{a\in\Sigma})$: V' consists of the vertices $v_0a_1v_1 \dots a_rv_r$ with $(v_{i-1}, v_i) \in E_{a_i}$, and E'_a contains the pairs $(v_0a_1v_1 \dots a_rv_r, v_0a_1v_1 \dots a_rv_rav)$ with $(v_r, v) \in E_a$. The unfolding operation has no effect in bisimulation invariant logics, but is highly nontrivial for MSO. Consider, for example, the singleton graph G_0 over $\{v_0\}$ with a 1-labelled and a 2-labelled edge from v_0 to v_0. Its unfolding is the infinite binary tree. While checking MSO-formulas over G_0 is trivial, this is quite difficult over T_2. A powerful result due to Courcelle and Walukiewicz [8] says: *If model-checking MSO-formulas over G is decidable and v_0 is an MSO-definable vertex of G, then model-checking MSO-formulas over $T_G(v_0)$ is decidable.* The result holds also for a slightly more general construction ("tree iteration") which can also be applied to relational structures other than graphs (see [1,24]).

MSO-interpretations and unfoldings are two operations which preserve decidability of MSO model-checking. Caucal [4] studied the structures generated by applying both operations, alternating between unfoldings and interpretations. (In [4] a more special type of interpretation was used; the link to MSO was supplied by Carayol and Wöhrle in [9]; for a detailed treatment see [25].) Starting with the class of finite graphs, one first obtains the regular infinite trees by unfoldings, then a class of graphs containing all pushdown transition graphs by interpretations, then the algebraic trees by unfoldings, and so on. The process yields many more complicated structures, all with a decidable MSO-theory. It is known that this "Caucal hierarchy" of graphs and trees is strict and quite rich, but we do not really have an overview which structures belong to it. An introduction with some examples is given in [23]. We also know of a few infinite graphs outside the Caucal hierarchy which still have a decidable MSO-theory (see [9]).

A related problem is to find more extensive classes of transition graphs for which the unfolding operation also preserves decidability of model-checking, but now for suitably chosen weaker logics than MSO. Note that MSO covers more than reachability properties (for example, one can express the existence of global colorings satisfying local constraints) and thus is more expressive than needed for many practical purposes.

2.3 Products

Products of transition graphs with different synchronization constraints are ubiquitous in system modelling, in particular for representing distributed systems.

While this construction causes fundamental complexity problems when the components are finite-state ("state space explosion"), undecidability may arise over infinite state spaces.

As an example, consider the successor structure $(\mathbb{N}, S)$ over the natural numbers with $S = \{(i, i+1) | i \in \mathbb{N}\}$, whose MSO-theory is known to be decidable (Büchi's Theorem; see [21]). The asynchronous product of $(\mathbb{N}, S)$ with itself is the structure $(\mathbb{N} \times \mathbb{N}, E)$ where $((i,j),(k,l)) \in E$ iff either $i = k$ and $l = j + 1$, or $j = l$ and $k = i + 1$. This is the infinite grid, where the model-checking problem with respect to MSO-properties is undecidable (see e.g. [21]). Thus, if product formation should preserve decidability of model-checking, then MSO is too strong.

If products should preserve decidability of model-checking, the task is to compose model-checking algorithms for the component structures to a corresponding algorithm for the product. Such composition results have a long tradition in logic, starting with the work of Feferman and Vaught [11] in first-order model theory. The situation is more complicated when second-order aspects enter (as involved in reachability properties).

Builing on the approach of [11], a preservation result on decidability of model-checking is shown in [26] for the logic FO(R) (first-order logic with reachability). In each component graph G_i $(1 \leq i \leq n)$, synchronizing and local actions are distinguished by a partition of the label alphabet Σ_i. Transitions may be executed locally via local labels, or else via a "synchronization constraint" $(c_1, \ldots, c_n)$ where each c_i is either a synchronizing label or ε. A corresponding execution leaves the states identical in the components with entry ε and involves a c_i-transition for each of the other components G_i. We speak of a finitely synchronized product if for each constraint $(c_1, \ldots, c_n)$ and each $c_i \neq \varepsilon$, only from finitely many vertices in G_i a c_i-labelled transition exists. This assumption applies to products of infinite systems where synchronization can only be realized within finite parts of the components. In [26], the following is shown: *If the graphs $G_1, \ldots, G_n$ have a decidable model-checking problem with respect to FO(R)-specifications, then this holds also for any finitely synchronized product of the G_i.*

This result is sharp in several ways. First, the assumption on finite synchronization cannot be weakened. If there is just one component which shares infinitely many synchronized transitions, the result fails. Also it is not possible to generalize the logic in any essential way; for example, the result fails if the reachability operator is restricted to regular sets of label sequences or if universal path quantification enters (see [26,20]).

In all the decidability results mentioned above, very high lower bounds for the complexity are known. One of the main tasks in the field is to single out cases which are both practically significant and at the same time allow more efficient procedures than those derived from the first decidability proofs.

3 Regular Presentations

Automata provide a natural framework for finite representations of infinite structures. For graphs (V, E), the idea is to represent the vertex set as a regular

language and the edge set by some sort of "regular relation". Since there are many versions of finite-state transducers, there are several options for the latter; for an introduction see e.g. [22]. One choice, leading to the "automatic structures", is based on the "automatic" (or "synchronized rational") relations. Here an edge relation E is defined by an automaton which processes a given word pair (u, v) synchronously in both components letter by letter (and one assumes that, if necessary, a dummy letter is used to extend the shorter word to the same length as the longer word). An automatic structure has a decidable first-order theory (see [2]); however, already the point-to-point-reachability problem ("Given vertices u, v, is there a path from u to v?") may be undecidable for an automatic structure. As an example, one can use the transition graph U of a universal Turing machine: Its configuration space is a regular language, and the one-step relation between configurations is clearly automatic. The halting problem for Turing machines can be reduced to the point-to-point reachability problem over U.

The one-step transition relation over Turing machine configurations is an infix rewriting relation. Restricting to prefix rewriting, as it occurs in pushdown transition graphs, the reachability problem becomes decidable. This follows already from classical work of Büchi [3] on his "regular canonical systems". If for the graph $G = (V, (E_a)_{a \in \Sigma})$ the vertex set is presented as a regular language, and the edge relations E_a by finite prefix-rewriting systems, then G has a decidable MSO-theory; this is shown by an interpretation in T_2 as in Section 2.1 above. As observed by Caucal [5], the prefix-rewriting rules can even be generalized to the form $U_1 \to U_2$ for regular sets U_1, U_2, meaning that a prefix $u_1 \in U_1$ can be replaced by any $u_2 \in U_2$. The "prefix-recognizable" graphs arising this way coincide with those which can be obtained from the binary tree T_2 by an MSO-interpretation (see, for example, [15]).

The idea of prefix-rewriting underlies many decidability results in infinite-state model-checking. It can be generalized in several ways while keeping (at least some of) the mentioned decidability properties. We present two such generalizations, the higher-order pushdown systems, and the ground tree rewriting graphs.

3.1 Higher-Order Pushdown Systems

Higher-order pushdown automata are a classical model of computation which arises in the evaluation of higher-order recursion schemes (see [10,14]). The idea is to generalize the stack symbols of a pushdown automaton to be again of stack format, and so on iteratively, which yields stacks of stacks of stacks etc. If k levels of stacks occur, we speak of a level-k pushdown automaton. For example, in a transition of a level-2 pushdown automaton, one can access the topmost symbol of the topmost stack, can modify the topmost stack in the usual way, or can execute global operations on the topmost stack, by deleting it or adding a copy of it as new topmost stack.

The configuration graphs of higher-order pushdown automata, called higher-order pushdown graphs, are of bounded out-degree (since only finitely many successor configurations can be reached directly from a given one). When we

consider the ε-closure, i.e. we allow ε-moves and compress sequences of ε-moves into a single transition, then transition graphs of infinite degree are generated. Surprisingly, the hierarchy of these transition graphs (for increasing level k) coincides with the Caucal hierarchy of graphs mentioned in Section 2.2: In [9] (and with full proof in [25]) it is shown that *a graph can be generated from finite graphs by k applications of unfolding and MSO-interpretation iff it is the transition graph of the ε-closure of a level-k pushdown automaton.* Of course, it follows that model-checking a higher-order pushdown graph with respect to MSO-properties is decidable.

3.2 Ground Term Rewriting Graphs

The transition graphs generated by higher-order pushdown automata are still tightly connected with infinite trees – in fact, they can be generated for a given level k from a single tree structure via MSO-interpretations. So these graphs are too restricted for many purposes of verification (excepting applications on the implementation of recursion).

A more flexible kind of model is generated when the idea of prefix-rewriting is generalized in a different direction, proceeding from word rewriting to tree rewriting (which we identify here with term rewriting). Instead of modifying the prefix of a word by applying a prefix-rewriting rule, we may rewrite a subtree of a given tree, precisely as it is done in ground term rewriting. A ground term rewriting graph (GTRG) has a vertex set V which is given by a regular tree language, and each edge relation E_a is defined by a finite ground term rewriting system.

A simple example of a GTRG is the infinite grid: It is generated from the tree $f(c, d)$ by applying the rules $c \to g(c)$ and $d \to g(d)$, which produces the trees $f(g^i(c), g^j(d))$ in one-to-one correspondence with the elements (i, j) of $\mathbb{N} \times \mathbb{N}$. Thus over GTRG's, model-checking MSO-properties is in general undecidable.

In work of C. Löding (see [16,17]), the structural and logical properties of GTRG's are investigated. As it turns out, *the model-checking problem over GTRG's is decidable for a logic which covers reachability and even recurrent reachability.* The atomic formulas of this logic refer to regular state properties (specified by finite tree automata), and the connectives are, besides the boolean ones, EX_a, EF, and EGF (in CTL-like notation). This result is optimal in the sense that adding universal quantification (for example, when adjoining the operator AF) leads to undecidability of the model-checking problem. On the other hand, it is possible – as for pushdown graphs – to generalize the rewriting rules without affecting the decidability results: Instead of allowing replacement of a single subtree by another one, one may use rules of the form $T \to T'$ for regular tree languages T, T', meaning that an occurrence of subterm $t \in T$ can be replaced by any $t' \in T'$. More results, also connecting GTRG's with asynchronous products of pushdown graphs, are shown in [6].

4 Conclusion

The above-mentioned results are as yet mosaic pieces of a picture which hopefully will grow into an esthetically pleasing and practically useful algorithmic theory

of infinite models (which the author would call "algorithmic model theory"). It seems that the two approaches mentioned – global model construction and local descriptions based on automata theoretic concepts – can be developed much further and also be combined in new ways.

There is, of course, a different approach for infinite-state model-checking, based on the admission of infinite data structures (like counters over the natural numbers, or addition and inequalities over the real numbers). An interesting direction of current work aims at establishing bridges between that approach and the results treated in the present paper. As an example, we mention the recent paper [13] where transition graphs arising from monotonic counters are discussed.

A dual track of research is to destillate efficient model-checking procedures from the general decidability results mentioned above, by restricting both the models and the logics to simple but relevant cases.

References

1. D. Berwanger, A. Blumensath, The monadic theory of tree-like structures, in: [12], 285-302.
2. A. Blumensath, E. Grädel, Automatic structures, in: *Proc. 15th LICS*, IEEE Comput. Soc. Press 2000, 51-62.
3. J.R. Büchi, Regular canonical systems, *Z. Math. Logik Grundl. Math.* 6 (1964), 91-111.
4. D. Caucal, On infinite terms having a decidable theory, in: *Proc. 27th MFCS*, Springer LNCS 2420 (2002), 265-176.
5. D. Caucal: On infinite transition graphs having a decidable monadic theory., *Theor. Comput. Sci.* 290 (2003), 79-115.
6. Th. Colcombet, On families of graphs having a decidable first order theory with reachability, in: *Proc. 29th ICALP*, Springer LNCS 2380 (2002), 98-109.
7. B. Courcelle, Monadic second-order graph transductions: a survey, *Theor. Comput. Sci.* 126 (1994), 53-75.
8. B. Courcelle, I. Walukiewicz, Monadic second-order logic, graph coverings and unfoldings of transition systems, *Ann. Pure Appl. Logic* 92 (1998), 51-65.
9. A. Carayol, S. Wöhrle, The Caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata, in: *Proc. 23rd FSTTCS*, Springer LNCS 2914 (2003), 112-123.
10. W. Damm, A. Goerdt, An automata theoretical characterization of the OI-hierarchy, *Inf. Contr.* 71 (1986), 1-32.
11. S. Feferman, R. Vaught, The first-order properties of products of algebraic systems, *Fund. Math.* 47 (1959), 57-103.
12. E. Grädel, W. Thomas, Th. Wilke (Eds.), *Automata, Logics, and Infinite Games*, Springer LNCS 2500 (2002).
13. W. Karianto, Adding monotonic counters to automata and transition graphs, *Proc. 9th Conf. on Developments in Language Theory*, Springer LNCS 3572 (2005), 308-319.
14. T. Knapik, D. Niwinski, P. Urzyczyn, Higher-order pushdown trees are easy, in: *Proc. 5th FOSSACS*. Springer LNCS 2303 (2002), 205-222.
15. M. Leucker, Prefix recognizable graphs and monadic logic, in: [12], 263-284.

16. C. Löding, *Infinite Graphs Generated by Tree Rewriting*, Dissertation, RWTH Aachen 2002.
17. C. Löding, Reachability problems on regular ground-tree rewriting graphs, *Theory of Computing Systems* (to appear).
18. D. Muller, P. Schupp, The theory of ends, pushdown automata, and second-order logic, *Theor. Comput. Sci.* 37 (1985), 51-75.
19. M.O. Rabin, Decidability of second-order theories and automata on infinite trees, *Trans. Amer. Math. Soc.* 141 (1969), 1-35.
20. A. Rabinovich, On compositionality and its limitations, *ACM Trans. on Computational Logic* (to appear).
21. W. Thomas, Automata on infinite objects, in: *Handbook of Theoretical Computer Science, Vol. B* (J.v. Leeuwen, Ed.), Elsevier, Amsterdam 1990, 133-191.
22. W. Thomas, A short introduction to infinite automata, in: *Proc. 5th Conf. on Developments in Language Theory* Springer LNCS 2295, 130-144
23. W. Thomas, Constructing infinite graphs with a decidable MSO-theory, in: *Proc. 28th MFCS*, Springer LNCS 2747 (2003), 113-124.
24. I. Walukiewicz, Monadic second-order logic on tree-like structures, *Theor. Comput. Sci.* 275 (2002), 311-346.
25. S. Wöhrle, *Decision Problems over Infinite Graphs: Higher-Order Pushdown Systems and Synchronized Products*, Dissertation, RWTH Aachen 2005.
26. S. Wöhrle, W. Thomas, Model checking synchronized products of infinite transition systems, in: *Proc. 19th LICS*, IEEE Comp. Soc. 2004, 2-11.