Verification of Graph Transformation Systems based on Approximations of the Unfolding*
(EXTENDED ABSTRACT)

Paolo Baldan[1], Andrea Corradini[2], and Barbara König[3]

[1] Dipartimento di Informatica, Università Ca' Foscari di Venezia, Italy
[2] Dipartimento di Informatica, Università di Pisa, Italy
[3] Institut für Informatik, Technische Universität München, Germany

`baldan@dsi.unive.it` `andrea@di.unipi.it` `koenigb@in.tum.de`

Abstract. In this extended abstract we report on a research activity aimed at developing a framework for the analysis and verification of Graph Transformation System (GTS). The approach is based on the construction of suitable finite approximations of the *unfolding* of a GTS, as defined in [4], which provide a partial description of the computations of the original system. Both over- and under-approximations are considered. The presentation is based on the papers [1, 5] and [2]

Graph transformation systems (GTSs) [15] are a powerful specification formalism for concurrent and distributed systems [7], generalising Petri nets. Along the years their concurrent behaviour has been deeply studied and a consolidated theory of concurrency is now available [14, 8]. Although several semantics of Petri nets, like process and unfolding semantics, have been extended to GTSs (see, e.g., [6, 13, 3, 4]), concerning automated verification, the literature does not contain many contributions to the static analysis of GTSs (see [10, 11]).

Most of the mentioned semantics for GTSs define an operational model of computation, which gives a concrete description of the behaviour of the system in terms of non-effective (e.g., infinite, non-decidable) structures. In this research activity, we propose a technique for building finite approximations of the behaviour for a class of (hyper)graph transformation systems. We show how one can construct under- and over-approximations of the behaviour of the system. The "accuracy" of such approximations can be fixed and arbitrarily increased in a way that the corresponding chain of (both under- and over-) approximations converges to the exact behaviour.

We concentrate on the unfolding semantics of GTSs, one reason for referring to a concurrent semantics being the fact that it allows to avoid to check all the interleavings of concurrent events. The unfolding construction for GTSs produces a static structure which fully describes the concurrent behaviour of the system, including all possible rewriting steps and their mutual dependencies, as well as all reachable states [13, 4]. However, as already mentioned, the unfolding, being infinite for any non-trivial system, cannot be used directly for verification purposes. Given a *graph grammar*, i.e., a GTS with a start hypergraph, we show how to construct finite structures which can be seen as approximations of the full unfolding of the grammar, at a chosen level k of accuracy.

Under-approximations (k-truncations). The unfolding of a graph grammar $\mathcal{G}$ can be defined categorically as the colimit of its prefixes of finite causal depth. Hence "under-approximations" of the behaviour of $\mathcal{G}$ can be trivially produced by stopping the construction of the unfolding at a finite causal depth k, thus obtaining the so-called *k-truncation* $\mathcal{T}^k(\mathcal{G})$ of the unfolding of $\mathcal{G}$. In the case of Petri nets this is at the basis of the *finite prefix approach*: if the system is finite-state and if the stop condition is suitably chosen, the prefix turns out to be *complete*, i.e., it contains the same information as the full unfolding [12, 9]. In general, for infinite-state systems, any truncation of the unfolding will be just an under-approximation of the behaviour of the system, in the sense that any computation in the truncation can be really performed in the original system, but not

* Research partially supported by the MIUR project COFIN 2001013518 CoMeta and by the FET-GC Project IST-2001-32747 AGILE.

vice versa. Nevertheless, finite truncations can still be used to check interesting properties of the grammar, e.g., some liveness properties of the form "eventually A" for a predicate A.

Over-approximations (k-coverings). A more challenging issue is to provide (sensible) over-approximations of the behaviour of a grammar $\mathcal{G}$, i.e., finite approximations of the unfolding which "represent" all computations of the original system (but possibly more). To this aim we propose an algorithm which, given a graph grammar $\mathcal{G}$, produces a finite structure, called *Petri graph*, consisting of a hypergraph and of a P/T net (possibly not safe or cyclic) over it, which can be seen as an over-approximation of the unfolding. One can require the approximation to be exact up to a certain causal depth k, thus obtaining the so-called *k-covering* $\mathcal{C}^k(\mathcal{G})$ of the unfolding of $\mathcal{G}$.

The covering $\mathcal{C}^k(\mathcal{G})$ over-approximates the behaviour of $\mathcal{G}$ in the sense that every computation in $\mathcal{G}$ is mapped to a valid computation in $\mathcal{C}^k(\mathcal{G})$. Moreover every hypergraph reachable from the start graph can be mapped homomorphically to (the graphical component of) $\mathcal{C}^k(\mathcal{G})$ and its image is reachable in the Petri graph. Therefore, given a property over graphs reflected by graph morphisms, if it holds for all graphs reachable in the covering $\mathcal{C}^k(\mathcal{G})$ then it also holds for all reachable graphs in $\mathcal{G}$. Important properties of this kind are the non-existence and non-adjacency of edges with specific labels, the absence of certain paths (for checking security properties) or cycles (for checking deadlock-freedom). Temporal properties, such as several safety properties of the form "always A", can be proven directly on the Petri net component of the coverings.

The fact that the unfolding can be approximated with arbitrary high accuracy is formalised by proving that both under- and over-approximations of the unfolding, converge to the full (exact) unfolding. In categorical terms, the unfolding $\mathcal{U}(\mathcal{G})$ of a graph grammar $\mathcal{G}$ can be expressed both as the colimit of the chain of k-truncations $\mathcal{T}^k(\mathcal{G})$ and as the limit of the chain of k-coverings $\mathcal{C}^k(\mathcal{G})$:

$$\begin{tikzcd}
\mathcal{T}^0(\mathcal{G}) \arrow[r] \arrow[drr] & \mathcal{T}^1(\mathcal{G}) \arrow[r, dotted, no head] \arrow[dr] & \mathcal{T}^k(\mathcal{G}) \arrow[r] \arrow[d] & \mathcal{T}^{k+1}(\mathcal{G}) \arrow[dl] & \cdots \\
 & & \mathcal{U}(\mathcal{G}) \arrow[dll] \arrow[dl] \arrow[d] \arrow[dr] & & \\
\mathcal{C}^0(\mathcal{G}) & \mathcal{C}^1(\mathcal{G}) \arrow[l] \arrow[r, dotted, no head] & \mathcal{C}^k(\mathcal{G}) & \mathcal{C}^{k+1}(\mathcal{G}) \arrow[l] & \cdots
\end{tikzcd}$$

The idea that finite under- and over-approximations can be used for checking properties of a graph grammar $\mathcal{G}$ is enforced by identifying significant fragments of the μ-calculus for which the validity of a formula in some approximation implies the validity of the same formula in the original grammar. Nicely, this is done by viewing our approach as a special case of the general paradigm of abstract interpretation.

References

1. P. Baldan, A. Corradini, and B. König. A static analysis technique for graph transformation systems. In *Proc. of CONCUR '01*, pages 381–395. Springer-Verlag, 2001. LNCS 2154.
2. P. Baldan, A. Corradini, and B. König. Static analysis of distributed systems with mobility specified by graph grammars—a case study. In *Proc. of IDPT '02 (World Conference on Integrated Design & Process Technology)*, 2002. to appear.
3. P. Baldan, A. Corradini, and U. Montanari. Concatenable graph processes: relating processes and derivation traces. In *Proc. of ICALP'98*, volume 1443 of *LNCS*, pages 283–295. Springer Verlag, 1998.
4. P. Baldan, A. Corradini, and U. Montanari. Unfolding and Event Structure Semantics for Graph Grammars. In W. Thomas, editor, *Proc. of FoSSaCS '99*, volume 1578 of *LNCS*, pages 73–89. Springer Verlag, 1999.
5. P. Baldan and B. König. Approximating the behaviour of graph transformation systems. In A. Corradini, H. Ehrig, H.-J. Kreowski, and G. Rozemberg, editors, *Proceedings of ICGT'02*, volume 2505 of *Lecture Notes in Computer Science*, pages 14–30. Springer Verlag, 2002.
6. A. Corradini, U. Montanari, and F. Rossi. Graph processes. *Fundamenta Informaticae*, 26:241–265, 1996.

7. H. Ehrig, H.-J. Kreowski, U. Montanari, and G. Rozenberg, editors. *Handbook of Graph Grammars and Computing by Graph Transformation, Vol.3: Concurrency, Parallellism, and Distribution*. World Scientific, 1999.
8. H. Ehrig, J. Kreowski, U. Montanari, and G. Rozenberg, editors. *Handbook of Graph Grammars and Computing by Graph Transformation, Vol. 2: Concurrency, Parallelism and Distribution*. World Scientific, 1999.
9. J. Esparza. Model checking using net unfoldings. *Science of Computer Programming*, 23(2–3):151–195, 1994.
10. M. Koch. *Integration of Graph Transformation and Temporal Logic for the Specification of Distributed Systems*. PhD thesis, Technische Universität Berlin, 2000.
11. B. König. A general framework for types in graph rewriting. In *Proc. of FST TCS 2000*, volume 1974 of *LNCS*, pages 373–384. Springer-Verlag, 2000.
12. K.L. McMillan. *Symbolic Model Checking*. Kluwer, 1993.
13. L. Ribeiro. *Parallel Composition and Unfolding Semantics of Graph Grammars*. PhD thesis, Technische Universität Berlin, 1996.
14. G. Rozenberg, editor. *Handbook of Graph Grammars and Computing by Graph Transformation, Vol. 1: Foundations*. World Scientific, 1997.
15. G. Rozenberg, editor. *Handbook of Graph Grammars and Computing by Graph Transformation. Volume 1: Foundations*. World Scientific, 1997.