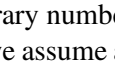
version: 20101009
(minor corrections)

Run Graphs of Communicating Processes and MSO

Alexander Heußner[1]
Université Bordeaux 1, LaBRI

Current trends in automata and concurrency theory, like the (re-)emergence of domain theory and topology, subliminally entail a lot of diagrammatic and graph based concepts. Nevertheless, a direct, more "graphical" approach to communicating transition and event systems seems more intuitive to us, and allows to avoid a lot of the algebraic machinery.
In the following, we propose a perspective onto the decidability of monadic second-order properties over graph representations of runs of distributed transition systems—in the following called *run graphs*. We combine well-known results from different fields (from Courcelle's theorem via graph grammars to reachability results for communicating pushdown systems) to gain new insights into the decidability/undecidability frontier for the verification of distributed transitions systems.
Run graphs seem to be an emerging—or better: *re*-emerging [16]—research topic as both this year's publications [10; 13] and recent articles [3; 4; 14] indicate. Our research on the interplay between decidability and graph grammars originated from our previous work on the decidability frontier for communicating pushdown systems [9] but received some fresh impulses from recent ideas of Madhusudan & Parlato [13].
Our approach's unique ingredient is the application of hyperedge replacement grammars to connect satisfiability of monadic second-order logic (MSO) to run graphs via treewidth. This allows, for example, to avoid the direct and cumbersome calculation of treewidth by graph-/tree-decomposition as, for example, fundamentally required in [13].
In the following, we will give a brief introduction to hyperedge replacement grammars, as well as their connection to treewidth, and decidability. Next, we present either different proofs for known decidability results, e.g., for message sequence charts and communicating pushdown systems, or new ones, e.g., for lock graphs (i.e., run graphs of processes that communicate via shared locks [10]). We will complete this paper with an outlook to our current research based on these ideas as well as some open questions.

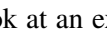
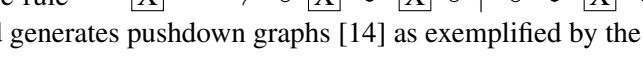
Hyperedge Replacement Grammars

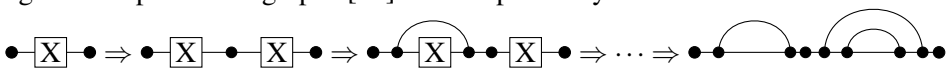
In a nutshell, a *hypergraph* $\mathscr{H} = \langle V, E, Ext \rangle$ is a generalization of directed graphs where an edge connects an arbitrary number of vertices of V. Such a *hyperedge* $X \in E \subseteq 2^V \setminus \{\emptyset\}$ will be depicted as 

where we assume a fix labeling of the "connectors" by $1, \cdots, n$. (Connectors are placeholders for vertices that allow to discuss a hyperedge without directly referring to a concrete context of vertices.) The *arity* of an edge will be the number of vertices it connects to, e.g., in the previous case the arity of X would be n. Further, *Ext* is a linearly ordered set of "external" vertices. Vertices that are not in *Ext* are called "internal".
A *hyperedge replacement Grammar* (HRG) is a tuple $\mathscr{G} = \langle N, T, P, S \rangle$ where both N and T are disjoint, finite sets of hyperedges—non-terminals and terminals, $S \in N$ is the start symbol, and P is a finite set of production rules. The latter map one hyperedge X to a choice of hypergraphs $\mathscr{H}_i$ $(1 \leq i \leq n)$ written $X \hookrightarrow \mathscr{H}_1 | \mathscr{H}_2 \cdots | \mathscr{H}_n$ where the number of external nodes of the $\mathscr{H}_i$ match the arity of X. The *width* of a HRG is the maximal number of vertices of the graphs at the right-hand sides of the rules minus one.

Submitted to:

[1] work of author supported by ANR AVeriSS;

We will define a HRG-*derivation* as follows: $\mathscr{H} \Rightarrow_{\mathscr{G}} \mathscr{H}'$ if there exists an $X \in N$ in $\mathscr{H}$ as well as a rule in P of the form $X \hookrightarrow \mathscr{H}_1 | \cdots | \mathscr{H}_n$ such that we can generate $\mathscr{H}'$ from $\mathscr{H}$ by replacing X with one of these $\mathscr{H}_i = \langle V_i, E_i, Ext_i \rangle$ by bijectively mapping Ext_i onto the connectors of X respecting the order on connectors/external vertices. Let $\mathscr{L}_{\mathscr{G}}(\mathscr{H}_0)$ be the language of hypergraphs that can be derived in a finite number of steps from the initial graph $\mathscr{H}_0$ which, w.l.o.g., will only contain one occurrence of S.
We depict internal vertices by $\bullet$ and external ones by $\circ$. Further, we will assume that T consists only of "unlabeled binary edges"; hence, our HRGs generate only classical graphs. We will leave out the numbering of a hyperedge's connectors if it is clear from the context.
Let us take a look at an example. Let $\mathscr{G}_{pd} = \langle N, T, P, S \rangle$ be an HRG, $\mathscr{H}_0 =$  , and P consisting of the single rule  . Obviously, $\mathscr{G}_{pd}$ is of width 3 and generates pushdown graphs [14] as exemplified by the derivation:



HRG — Treewidth — MSO

The *treewidth* of a (classical) graph measures how close it is to a tree. Historically, treewidth is defined and calculated via tree-decompositions [15], but we use the following result of Lautemann to draw the bridge between HRG, their derivation trees, tree-decomposition of graphs, and treewidth [11; 12]:

Theorem 1 *Every graph generated by a HRG of width k has treewidth at most k.*

It is well known that satisfiability of *monadic second order logic* (MSO) over Σ-labeled graphs is strongly tied to bounded treewidth following the results of Courcelle [5], Seese [17], et al. Our point of departure is the following theorem based on [13]:

Theorem 2 *Given a class $\mathscr{C}$ of Σ-labeled graphs defined by an HRG $\mathscr{G}$, and an MSO formula φ, the problem of testing whether there exists a graph in $\mathscr{C}$ that satisfies φ is decidable.*

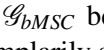
Based on the work of Courcelle and others [6], we can give an additional upper complexity bound for the previous result by the tower of exponentials (of base two) whose height equals the quantifier height of the formula, and whose exponent is the width of $\mathscr{G}$.

Run Graphs of Distributed Systems

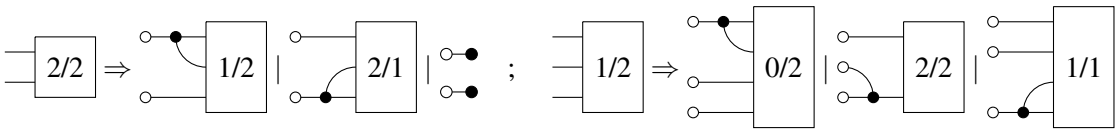
Our goal is to verify an MSO property φ over an automata-based model of communicating processes. If we can show how to generate the class of these models by an HRG (whereas we maybe need to additionally encode some of the models' properties into an MSO formula φ' over the labeled graph generated by the HRG), testing satisfiability of φ ($\wedge\varphi'$) becomes decidable. In the following, we present the generation of the bare structure of the run graph by HRG which can easily extended by (vertex-)labels to the generation of Σ-labeled graphs.

➪ *Pushdown Graphs*
Our first example already introduced $\mathscr{G}_{pd}$ which generates pushdown graphs, i.e., the run graphs of classical pushdown automata. These are known to have decidable MSO satisfiability [14].

➪ *Bounded MSC / Communicating Fifo Systems*

The behaviour of *communicating fifo systems* (or communicating finite state automata) is often described by *message sequence charts* (MSC). Bounded MSC are an important subclass of MSC whose channel capacity is restricted. Let a *n-bounded MSC* be an MSC where each channel does not hold more than n messages at the same time. This local restriction allowed to derive basic decidability results [8].
Let $\mathscr{G}_{bMSC}$ be the HRG of non-terminals labeled "i/j" with $0 \leq i, j \leq 2$, and $\mathscr{H}_0 =$ 

. We will exemplarily show the principle behind the rules:



The rules generate a 2-bounded MSC over 2 processes left-to-right starting with empty channels (with full capacity or "2/2"). We will use the non-terminals to buffer the unreceived messages and store the remaining capacity via the type of the non-terminal. For example, the second rule above encodes that if there is already one message in channel 1 ("1/2"), it is either received in the following step, a second message is added or the other process can send a message. Additional (data) properties, like the matching of messages, are encoded directly in MSO and added, to the formula we want to verify.
Obviously, we can generalize $\mathscr{G}_{bMSC}$ to any finite number p of processes and of channel bounds b. Hence, bounded MSC have treewidth bounded by $\mathscr{O}(p \cdot b)$.

➪ *Lock Graphs*

Lock graphs (or lock causality graphs) depict the causality involved when different processes battle for the access to a critical resource by locks. Again, we will code the current state of the locks (free, requested by process i, held by process i) in non-terminals. Our rules ensure that each lock can only be held by one process and that the choice among the processes waiting to enter a lock is fair. Additionally, we generate the causal "back edges" from [10] between the last liberation of a lock and the current entering analogous to the buffering applied for the message passing in $\mathscr{G}_{bMSC}$. The width of the HRG and therewith the treewidth remains bounded by $\mathscr{O}(p \cdot l)$ for p processes and l locks. Lock graphs are a relatively new notion, and only atomicity and race conditions were proven to be verifiable [10].

➪ *Communicating Pushdown Systems*

Extending existentially one-bounded MSC leads to *eagerly communicating pushdown systems (eager CPS)* [9] on which MSO is obviously undecidable: with at least two processes, one can easily generate a run graph that includes an unbounded grid as minor, and hence has unbounded treewidth. Assuming the additional constraint of non-confluency (no two processes can synchronize with both their pushdowns non-empty), these systems were shown to have decidable reachability by a direct, constructive proof [9]. We can derive an HRG $\mathscr{G}_{CPS}$ for these systems by applying the fact that one can simulate any eager and non-confluent run on only one pushdown. Our HRG approach arrives at an exponential upper bound which coincides with our already known completeness result.
Further, it is also possible to generate the run graph of a k-bounded CPS by a HRG. The latter restrict the considered runs of CPS to those of a certain form described by a finite sequence of phases. Despite the exponential explosion of the number of non-terminals needed to encode the different combinations of phases, and the additional complexity from encoding synchronization constraints into the rules, we arrive at a HRG that has bounded width. As shown in [9], this notion of bounded phase subsumes the one presented in [19].

Recent results from Atig [1] as well as Seth [18] generalize the idea behind the decidability proofs for multi-pushdown systems which are based on the reduction to only one pushdown, as well as introduce more general clases of decidable systems. Consequently, we could describe the latter—which only require recursion on one "parameter" at the time—also with the help of HRG.

Summary & Outlook

The decidability/undecidability frontier for communicating processes seems to depend on causal patterns that allow to simulate Turing machines (e.g., unbounded crossing of message for MSC, or the grid-minor seen earlier). The presented approach allows to describe the "good" patterns as those that can be generated by a HRG whereas we can directly exclude patterns whose tree-width is unbounded. Hence, HRG—extended with tree-width or decidability preserving graph transformations—serve as useful tool when searching for classes of models that allow to verify MSO properties. Nevertheless, we reached for the stars by demanding MSO decidability whereas in most practical situations safety verification (i.e., reachability) would already be satisfying. However, there are *no* known results for the connection of this simpler question with grammar descriptions of run graphs yet. Currently, we are also applying HRGs as a tool to find restrictions to dynamic QCP (i.e., that can generate/spawn processes) that allow decidability of MSO (extending known results like [2] with additional pushdowns).

Acknowledgements: I want to thank A. Muscholl and G. Parlato for fruitful and pertinent discussions as well as to the referees for their apposite remarks.

References

[1] M. F. Atig (2010): *From Multi to Single Stack Automata*. In: *Proc. of CONCUR'10* (to be published).
[2] B. Bollig,L. Hélouët (2010): *Realizability of Dynamic MSC*. In: *Proc. of CSR*, *LNCS* 6072, Springer, pp. 48–59.
[3] T. Cachat (2003): *Higher Order Pushdown Automata, the Caucal Hierarchy of Graphs and Parity Games*. In: *Proc. of ICALP*, pp. 556–569.
[4] D. Caucal (2003): *On infinite transition graphs having a decidable monadic theory*. *Theor. Comput. Sci.* 290(1), pp. 79–115.
[5] B. Courcelle (1990): *The monadic second-order logic of graphs, I: Recognisable sets of finite graphs*. *Inf. and Comp.* 85, pp. 12–75.
[6] B. Courcelle (in preparation): *Graph Algebras and Monadic Second-order Logic*. Cambridge Univ. Press. (draft at author's webpage).
[7] E. A. Emerson & C. S. Jutla (1999): *The Complexity of Tree Automata and Logics of Programs*. *SIAM J. Comput.* 29(1), pp. 132–158.
[8] B. Genest, D. Kuske & A. Muscholl (2007): *On communicating automata with bounded channels*. *Fundamenta Informaticae* 80, pp. 147–167.
[9] A. Heußner, J. Leroux, A. Muscholl & G. Sutre (2010): *Reachability Analysis of Communicating Pushdown Systems*. In: *Proc. of FOSSACS 2010*, *LNCS* 6014, Springer, pp. 267–281.
[10] V. Kahlon & C. Wang (2010): *Universal Causality Graphs: A Precise Happens-Before Model for Detecting Bugs in Concurrent Programs*. In: *Proc. of CAV 2010*, *LNCS* 6174, Springer, pp. 434–449.
[11] C. Lautemann (1988): *Decomposition Trees: Structured Graph Representation and Efficient Algorithms*. In: *Proc. of CAAP 1988*, *LNCS* 299, Springer, pp. 28–39.
[12] C. Lautemann (1990): *Tree Automata, Tree Decomposition and Hyperedge Replacement*. In: *Graph-Grammars and Their Application to Computer Science*, *LNCS* 532, Springer, pp. 520–537.
[13] P. Madhusudan & G. Parlato (2010): *The Tree Width of Automata with Auxiliary Storage.* (under submission, preliminary version at author's webpage).
[14] D. E. Muller & P. E. Schupp (1985): *The Theory of Ends, Pushdown Automata, and Second-Order Logic*. *Theor. Comput. Sci.* 37, pp. 51–75.
[15] N. Robertson & P. D. Seymour (1986): *Graph minors III: Planar tree-width*. *Journal of Combinatorial Theory, Series B* 41, pp. 92–114.
[16] G. Rozenberg, editor (1997): *Handbook of Graph Grammars and computing by Graph Transformation*, I/II/II. World Scientific.
[17] D. Seese (1991): *The Structure of Models of Decidable Monadic Theories of Graphs*. *Ann. Pure Appl. Logic* 53(2), pp. 169–195.
[18] A. Seth (2010): *Global Reachability in Bounded Phase Multi-stack Pushdown Systems*. In: *Proc. of CAV'10*, *LNCS* 6174, Springer, pp. 615–628.
[19] S. La Torre, P. Madhusudan & G. Parlato (2008): *Context-Bounded Analysis of Concurrent Queue Systems*. In: *TACAS 2008*, *LNCS* 4963, Springer, pp. 299–314.