Abstract. We investigate the issue of reducing the verification problem of multi-stack machines to the one for single-stack machines. For instance, elegant (and practically efficient) algorithms for bounded-context switch analysis of multi-pushdown systems have been recently defined based on reductions to the reachability problem of (single-stack) pushdown systems [10,18]. In this paper, we extend this view to both bounded-phase visibly pushdown automata (BVMPA) [16] and ordered multi-pushdown automata (OMPA) [11] by showing that each of their emptiness problem can be reduced to the one for a class of single-stack machines. For these reductions, we introduce effective generalized pushdown automata (EGPA) where operations on stacks are (1) pop the top symbol of the stack, and (2) push a word in some (effectively) given set of words $L$ over the stack alphabet, assuming that $L$ is in some class of languages for which checking whether $L$ intersects regular languages is decidable. We show that the automata-based saturation procedure for computing the set of predecessors in standard pushdown automata can be extended to prove that for EGPA too the set of all predecessors of a regular set of configurations is an effectively constructible regular set. Our reductions from OMPA and BVMPA to EGPA, together with the reachability analysis procedure for EGPA, allow to provide conceptually simple algorithms for checking the emptiness problem for each of these models, and to significantly simplify the proofs for their 2ETIME upper bounds (matching their lower-bounds).

1 Introduction

In the last few years, a lot of effort has been devoted to the verification problem for models of concurrent programs (see, e.g., [5,3,16,9,2]). Pushdown automata have been proposed as an adequate formalism to describe sequential programs with procedure calls [8,14]. Therefore, it is natural to model recursive concurrent programs as multi-stack automata. In general, multi-stack automata are Turing powerful and hence come along with undecidability of basic decision problems [13]. To overcome this barrier, several subclasses of pushdown automata with multiple stacks have been proposed and studied in the literature.

Context-bounding has been proposed in [12] as a suitable technique for the analysis of multi-stack automata. The idea is to consider only runs of the automaton that can be divided into a given number of contexts, where in each context pop and push operations are exclusive to one stack. Although the state space which may be explored is still unbounded in presence of recursive procedure calls, the context-bounded reachability
problem is NP-complete even in this case \[11\]. In fact, context-bounding provides a very useful tradeoff between computational complexity and verification coverage.

In \[16\], La Torre et al. propose a more general definition of the notion of a context. For that, they define the class of \textit{bounded-phase visibly multi-stack pushdown automata} (BVMPA) where only those runs are taken into consideration that can be split into a given number of phases, where each phase admits pop operations of one particular stack only. In the above case, the emptiness problem is decidable in double exponential time by reducing it to the emptiness problem for tree automata.

Another way to regain decidability is to impose some order on stack operations. In \[6\], Breveglieri et al. define \textit{ordered multi-pushdown automata} (OMPA), which impose a linear ordering on stacks. Stack operations are constrained in such a way that a pop operation is reserved to the first non-empty stack. In \[11\], we show that the emptiness problem for OMPA is \textit{2ETIME}-complete\[4\]. The proof of this result lies in a complex encoding of OMPA into some class of grammars for which the emptiness problem is decidable.

In this paper, we investigate the issue of reducing the verification problem of multi-stack machines to the one for single-stack machines. We believe that this is a general paradigm for understanding the expressive power and for establishing decidability results for various classes of concurrent program models. For instance, elegant (and practically efficient) algorithms for bounded-context switch analysis of multi-pushdown systems have been recently defined based on reductions to the reachability problem of (single-stack) pushdown systems \[10,18\]. We extend this view to both OMPA and BVMPA by showing that each of their emptiness problem can be reduced to the one for a class of single-stack machines. For these reductions, we introduce \textit{effective generalized pushdown automata} (EGPA) where operations on stacks are (1) pop the top symbol of the stack, and (2) push a word in some (effectively) given set of words \(L\) over the stack alphabet, assuming that \(L\) is in some class of languages for which checking whether \(L\) intersects a given regular language is decidable. Observe that \(L\) can be any finite union of languages defined by a class of automata closed under intersection with regular languages and for which the emptiness problem is decidable (e.g., pushdown automata, Petri nets, lossy channel machines, etc). Then, we show that the automata-based saturation procedure for computing the set of predecessors in standard pushdown automata \[4\] can be extended to prove that for EGPA too the set of all predecessors of a regular set of configurations is a regular set and effectively constructible. As an immediate consequence of this result, we obtain similar decidability results of the decision problems for EGPA like the ones obtained for pushdown automata.

Then, we show that, given an OMPA \(\mathcal{M}\) with \(n\) stacks, it is possible to construct an EGPA \(\mathcal{P}\), whose pushed languages are defined by OMPA with \((n - 1)\) stacks, such that the emptiness problem for \(\mathcal{M}\) is reducible to its corresponding problem for \(\mathcal{P}\). The EGPA \(\mathcal{P}\) is constructed such that the following invariant is preserved: The state and the content of the stack of \(\mathcal{P}\) are the same as the state and the content of the \(n\)-th stack of \(\mathcal{M}\) when its first \((n - 1)\) stacks are empty. Then, we use the saturation procedure for

---

\(^1\) Recall that \textit{2ETIME} is the class of all decision problems solvable by a deterministic Turing machine in time \(2^{adn}\) for some constant \(d\).