Parameterized verification of asynchronous shared-memory systems

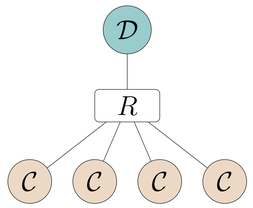
Marie Fortin, supervised by Anca Muscholl and Igor Walukiewicz
Technical University of Munich

Context

Verification of shared-memory systems is often undecidable. For instance, systems consisting of two pushdown processes with a shared boolean variable can already simulate a Turing machine. Here we consider a parametric version of this model, where the number of pushdown processes is arbitrary. Suprisingly, this model enjoys various decidable verification problems, due to the lack of process identities.

$(\mathcal{C}, \mathcal{D})$-systems

- One pushdown **leader** process $\mathcal{D}$
- Arbitrarily many anonymous and identical pushdown **contributors** $\mathcal{C}$
- Processes communicate via a bounded shared register R, without lock



Specification

We consider regular, action-based properties of traces of $(\mathcal{C}, \mathcal{D})$-systems, containing both finite and infinite traces.

$$\boldsymbol{w}(b)\ \ \overline{\boldsymbol{r}}(b)\ \overline{\boldsymbol{r}}(b)\overline{\boldsymbol{w}}(a)\boldsymbol{r}(a)\boldsymbol{w}(c)\ldots$$

Leader write — Contributor read — ...

Fig. 1: A trace of a $(\mathcal{C}, \mathcal{D})$-system

Since contributor runs can always be duplicated, it is natural to consider $\mathcal{C}$-*expanding* properties only: properties $\mathcal{P}$ such that for every trace σ in $\mathcal{P}$, any trace σ' obtained by repeating some of the contributor actions is also in $\mathcal{P}$.

General problem

Input: a $(\mathcal{C}, \mathcal{D})$-system and a regular or LTL $\mathcal{C}$-expanding property $\mathcal{P}$.
Question: is there a maximal (finite or ininite) trace of the $(\mathcal{C}, \mathcal{D})$-system that belongs to $\mathcal{P}$?

Special instances

Reachability: is there a run where the leader perform a special action $\top$?
Repeated reachability: is there a run where the leader does $\top$ infinitely many times?
Universal reachability: does the leader do $\top$ in all maximal runs?

Previous results

- The reachability problem is PSPACE-complete [2].
- The repeated reachability problem is PSPACE-hard and in NEXPTIME [1].

Approach

We first study repeated reachability and universal reachability.

Theorem

The repeated reachability problem is PSPACE-complete.
The universal reachability problem is NEXPTIME-complete.

We then use the results about these problems to design an algorithm for the verification of regular $\mathcal{C}$-expanding properties. The difficulty is that such properties talk about both the actions of the leader, and of the contributors (instead of actions of the leader only). The idea is to simulate the $(\mathcal{C}, \mathcal{D})$-system by one in which all actions of the contributors are reflected in actions of the leader.

Key properties of $(\mathcal{C}, \mathcal{D})$-systems

Finite-state contributors

Both for repeated reachability, and universal reachability, we use a crucial result from [1], showing that the contributors can be assumed to be finite-state. The idea is to "distribute" contributor runs into smaller runs that only use a bounded part of the stack.

Replicating runs

Since the number of contributors is arbitrary, it is possible to replicate any of the contributor runs by adding a new contributors copying the actions of another.

$$\begin{array}{ll}\mathcal{D} & \boldsymbol{w}(b)\\ \mathcal{C} & \overline{\boldsymbol{r}}(b)\ \overline{\boldsymbol{w}}(a)\end{array} \quad\rightarrow\quad \begin{array}{lll}\mathcal{D} & \boldsymbol{w}(b) & \\ \mathcal{C} & \overline{\boldsymbol{r}}(b) & \overline{\boldsymbol{w}}(a)\\ \mathcal{C} & \overline{\boldsymbol{r}}(b) & \overline{\boldsymbol{w}}(a)\end{array}$$

This can be used to abstract away the number of contributors in a given state, instead focusing on the set of register values they can produce.

Results

- We improved the known NEXPTIME upper bound for the repeated reachability problem, showing that it is PSPACE-complete.
- We investigated a new problem, universal reachability, and showed that it is NEXPTIME-complete. The main difference with previous work is that we consider both finite and infinite runs, which require to look into deadlocks.
- We showed that the more general problem of verifying regular $\mathcal{C}$-expanding properties – talking about both leader and contributor actions – is decidable, and also NEXPTIME-complete.

References

[1] A. Durand-Gasselin, J. Esparza, P. Ganty, and R. Majumdar. Model checking parameterized asynchronous shared-memory systems. In *CAV'15*, 2015.
[2] J. Esparza, P. Ganty, and R. Majumdar. Parameterized verification of asynchronous shared-memory systems. *J. ACM*, 63(1):10, 2016.
[3] M. Fortin, A. Muscholl, and I. Walukiewicz. On parametrized verification of asynchronous, shared-memory pushdown systems. *CoRR*, abs/1606.08707, 2016.
[4] M. Hague. Parameterised pushdown systems with non-atomic writes. In *FSTTCS'11*, 2011.
[5] S. La Torre, A. Muscholl, and I. Walukiewicz. Safety of parametrized asynchronous shared-memory systems is almost always decidable. In *CONCUR'15*, 2015.