


Concurrent Algorithms and Data Structures for Model Checking

Jaco van de Pol 

Aarhus University, Denmark

University of Twente, The Netherlands

<http://www.cs.au.dk/~jaco>

jaco@cs.au.dk

Abstract

Model checking is a successful method for checking properties on the state space of concurrent, reactive systems. Since it is based on exhaustive search, scaling this method to industrial systems has been a challenge since its conception. Research has focused on *clever data structures and algorithms*, to reduce the size of the state space or its representation; *smart search heuristics*, to reveal potential bugs and counterexamples early; and *high-performance computing*, to deploy the brute force processing power of clusters of compute-servers. The main challenge is to combine these approaches – brute-force alone (when implemented carefully) can bring a linear speedup in the number of processors. This is great, since it reduces model-checking times from days to minutes. On the other hand, proper algorithms and data structures can lead to exponential gains. Therefore, the *parallelization bonus* is only real if we manage to speedup clever algorithms.

There are some obstacles though: many linear-time graph algorithms depend on a depth-first exploration order, which is hard to parallelize. Examples include the detection of strongly connected components (SCC) and the nested depth-first-search (NDFS) algorithm. Both are used in model checking LTL properties. Symbolic representations, like binary decision diagrams (BDDs), reduce model checking to “pointer-chasing”, leading to irregular memory-access patterns. This poses severe challenges on achieving actual speedup in (clusters of) modern multi-core computer architectures.

This talk presents some of the solutions found over the last 10 years, which led to the high-performance model checker LTSmin [2]. These include parallel NDFS (based on the PhD thesis of Alfons Laarman [3]), the parallel detection of SCCs with concurrent Union-Find (based on the PhD thesis of Vincent Bloemen [1]), and concurrent BDDs (based on the PhD thesis of Tom van Dijk [4]).

Finally, I will sketch a perspective on moving forward from high-performance model checking to *high-performance synthesis algorithms*. Examples include parameter synthesis for stochastic and timed systems, and strategy synthesis for (stochastic and timed) games.

2012 ACM Subject Classification Software and its engineering → Model checking; Theory of computation → Verification by model checking; Theory of computation → Parallel algorithms

Keywords and phrases model checking, parallel algorithms, concurrent datastructures

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2019.4

Category Invited Talk

References

- 1 Vincent Bloemen. *Strong Connectivity and Shortest Paths for Checking Models*. PhD thesis, University of Twente, Enschede, Netherlands, 2019. URL: <https://research.utwente.nl/en/publications/strong-connectivity-and-shortest-paths-for-checking-models>.
- 2 Gijs Kant, Alfons Laarman, Jeroen Meijer, Jaco van de Pol, Stefan Blom, and Tom van Dijk. LTSmin: High-performance language-independent model checking. In *TACAS*, volume 9035 of *Lecture Notes in Computer Science*, pages 692–707. Springer, 2015.
- 3 Alfons Laarman. *Scalable multi-core model checking*. PhD thesis, University of Twente, Enschede, Netherlands, 2014. URL: <http://eprints.eemcs.utwente.nl/25673/>.
- 4 Tom van Dijk. *Sylvan: multi-core decision diagrams*. PhD thesis, University of Twente, Enschede, Netherlands, 2016. URL: <http://purl.utwente.nl/publications/100676>.



© Jaco van de Pol;

licensed under Creative Commons License CC-BY

30th International Conference on Concurrency Theory (CONCUR 2019).

Editors: Wan Fokkink and Rob van Glabbeek; Article No. 4; pp. 4:1–4:1

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany