Internship proposal
Distributed model-checking: efficient multi-core algorithms

Supervisors: Sami Evangelista, Laure Petrucci
Location: LIPN, CNRS UMR 7030
Université Paris 13
99 avenue Jean-Baptiste Clément
93430 VILLETANEUSE
Contacts: {Sami.Evangelista,Laure.Petrucci}@lipn.univ-paris13.fr

Model-checking aims at checking whether a software or hardware system meets its specification. The analysis of LTL (Linear Temporal Logic) properties boils down to checking the emptiness of the language of a Büchi automaton, which is the synchronised product of the system and an automaton modelling the negation of the formula. Verification then consists in finding an accepting cycle in a graph, i.e. a cycle containing at least one accepting state.

This problem has been widely studied, and depth-first search (DFS) techniques allow for linear time verification. Such techniques are split in two main families:

- *Nested* DFS (NDFS) [CVWY90] involves two procedures: the first one searches accepting states, while the second, embedded in the first one, checks if these states are part of an accepting cycle ;
- Strongly Connected Components (SCC) based algorithms [Cou99, GV04] take advantage of reachability of a SCC containing an accepting state, from the initial state, being a necessary and sufficient condition to the existence of a counterexample.

Verification often has to face the "state space explosion problem", with impact both on space (limited memory) and time (too long computation times). Different techniques have been proposed to overcome these problems.

Memory management may be improved using appropriate coding of states [Hol95] or applying reductions to the set of transitions [CGMP99].

Computation time can be reduced using distributed algorithms, for distributed-memory architectures [BBC03, BBS01, BCKP01, BCMS04, CP03] or shared memory [BBR07, HB07].

Recent works have designed and shown the efficiency of multicore NDFS algorithms [LLvdP$^+$11, EPY11, LvdP11], MC-NDFS.

Objectives of the internship: The use of reduction techniques in a multicore environment will allow for pushing further the limits of analysis both in time and memory. It seems obvious that some approaches tackling states representation, such as bitstate hashing [Hol95] are independent of the searching algorithm, and these algorithms should easily combine. On the other hand, the use of transitions based approaches, such as partial order reduction [CGMP99] is not straightforward.

Implementing these techniques typically use two components: a selection mechanism, independent of the search algorithm, and thus compatible with MC-NDFS ; and an *ignoring problem* solver guaranteeing that a transition will not be systematically ignored by the selection function. This second component depends on the model-checking algorithm used.

The internship will study combinations of such reduction techniques with the multi-core algorithm MC-NDFS, will prove the algorithms thus obtained, and will implement them within the LTSMIN tool suite (`http://fmt.cs.utwente.nl/tools/ltsmin/`), hence allowing for comparisons.

References

[BBC03] J. Barnat, L. Brim, and J. Chaloupka. Parallel Breadth-First Search LTL Model-Checking. In *ASE'03*, pages 106–115. IEEE Computer Society, 2003.

[BBR07] J. Barnat, L. Brim, and P. Rockai. Scalable Multi-core LTL Model-Checking. In *SPIN'07*, volume 4595 of *LNCS*, pages 187–203. Springer, 2007.

[BBS01] J. Barnat, L. Brim, and J. Stríbrná. Distributed LTL Model-Checking in SPIN. In *SPIN'01*, volume 2057 of *LNCS*, pages 200–216. Springer, 2001.

[BCKP01] L. Brim, I. Cerná, P. Krcál, and R. Pelánek. Distributed LTL Model Checking Based on Negative Cycle Detection. In *FSTTCS'01*, volume 2245 of *LNCS*, pages 96–107. Springer, 2001.

[BCMS04] L. Brim, I. Cerná, P. Moravec, and J. Simsa. Accepting Predecessors Are Better than Back Edges in Distributed LTL Model-Checking. In *FMCAD'04*, volume 3312 of *LNCS*, pages 352–366. Springer, 2004.

[CGMP99] Edmund M. Clarke, Orna Grumberg, Marius Minea, and Doron Peled. State Space Reduction Using Partial Order Techniques. *STTT*, pages 279–287, 1999.

[Cou99] J.-M. Couvreur. On-the-Fly Verification of Linear Temporal Logic. In *FM'1999*, volume 1708 of *LNCS*, pages 253–271. Springer, 1999.

[CP03] I. Cerná and R. Pelánek. Distributed Explicit Fair Cycle Detection (Set Based Approach). In *SPIN'03*, volume 2648 of *LNCS*, pages 49–73. Springer, 2003.

[CVWY90] C. Courcoubetis, M. Y. Vardi, P. Wolper, and M. Yannakakis. Memory Efficient Algorithms for the Verification of Temporal Properties. In *CAV'1990*, volume 531 of *LNCS*, pages 233–242. Springer, 1990.

[EPY11] S. Evangelista, L. Petrucci, and S. Youcef. Parallel nested depth-first searches for LTL model checking. In *Proceedings of the 9th International Symposium on Automated Technology for Verification and Analysis (ATVA11), Taipei, Taiwan*, LNCS. Springer Verlag, October 2011.

[GV04] J. Geldenhuys and A. Valmari. Tarjan's Algorithm Makes On-the-Fly LTL Verification More Efficient. In *TACAS'04*, volume 2988 of *LNCS*, pages 205–219. Springer, 2004.

[HB07] G.J. Holzmann and D. Bosnacki. The Design of a Multi-Core Extension of the Spin Model Checker. *IEEE Trans. on Software Engineering*, 33(10):659–674, 2007.

[Hol95] G.J. Holzmann. An Analysis of Bistate Hashing. In *PSTV'1995*, pages 301–314, 1995.

[LLvdP+11] A. Laarman, R. Langerak, J. van de Pol, M. Weber, and A. Wijs. Multi-core nested depth-first search. In T. Bultan and P.-A. Hsiung, editors, *ATVA 2011*, LNCS. Springer Verlag, 2011.

[LvdP11] A. Laarman and J. van de Pol. Variations on multi-core nested depth-first search. In *PDMC 2011*, LNCS. Springer Verlag, 2011.