Ideas for a bachelor's thesis based on swarm model checking on the GPU

Leonard Techel, 21510495

June 26, 2021

This document proposes multiple ideas for a bachelor's thesis on the topic of swarm model checking on the GPU, shows up some important related work, starts writing down tasks and their acceptance criteria and shows a first idea on an outline.

1 Ideas

1.1 Exploration of low-connectivity models using swarm model checking on the GPU

Exploring the state space of models with *low-connectivity* using the default search strategy of the Grapple algorithm, *Parallel Deep Search* (PDS), can lead to a slow-down in coverage growth, as shown in [5]. The paper already suggests multiple countermeasures, in particular a two-phase swarm, depth-limited PDS / process-PDS and a combination of such techniques called *scatter PDS*. They differentiate in whether they are controlled by the host or if they are self-contained within a VT running on the GPU. However, these ideas still need fine-tuning, and it may be possible to improve exploration performance by combining them in new ways, for example by alternating the two-phase approach multiple times. The work on this approach could contain:

1. Finding low-connectivity models compatible with Grapple, e.g. from the BEEM database, the SPIN examples or custom-made
2. Experimental performance evaluation of said models using the approaches from [5]
3. Formulating how the algorithms could be re-combined to improve low-connectivity model exploration
4. Experimental evaluation of these new ideas

Instead of (only) experimenting with different parameters to the already existing algorithms, another idea could be to formulate the swarm exploration mathematically using the notions of graph and probability theory in order to find a theoretical solution to the exploration of low-connectivity models.

1.2 Detecting cycles using swarm model checking on the GPU

Currently, the Grapple algorithm is limited to the verification of safety and reachability properties. To verify liveness properties and, by doing so, full LTL models, the algorithm needs to be extended, so it can detect accepting cycles. In order to do so, combining Grapple with the CUDA accelerated *maximal accepting predecessors* (MAP) algorithm from [1] could be possible: Their approach is based on on-the-fly state space exploration on the host, which leaves room for replacing it with Grapple.

The work on this approach could contain:

1. Finding a way to combine the algorithms in a way that their data structures still fit into the GPU's shared memory
2. Experimental implementation of the Grapple+MAP algorithm
3. Evaluation on common LTL models, e.g. those from [1]

2 Related work that is interesting for this thesis

In [4], a good overview on the topic of model checking is given. In [6], Gerard Holzmann, the author of the famous SPIN model checker, discusses *explicit-state model checking* on which Grapple is based. In [7] and [8], Holzmann et al. invent *Swarm Verification* and discuss techniques related to it. In [3] and [2], the foundation for the main contribution [5] is laid by the same authors.

3 Draft outline

A first idea what could be part of the thesis.

- Foundations
  - Model checking
    - What is model checking, why is it necessary, and what is checked?
    - Explicit-State Model Checking: How it limits the scope of our work
    - Swarm Verification
      - Key idea 1: Diversification of search strategies may lead to higher state space coverage than exhaustive verification under hardware bounds
      - Key idea 2: Parallelization on heterogeneous hardware is easy if the model checking can be split into independent, time- and memory-bounded verification tests
  - GPU Programming: CUDA model
  - Model checking on the GPU
    - Quick recap on the difference between CPUs and GPUs

        * Why GPUs are so interesting for our problem
        * Why running model checking on the GPU is hard
        * Swarm Verification on the GPU: How Grapple works
- Theoretical formulation and experimental implementation of one of the ideas above
- Evaluation
  - Comparison to the paper's results
  - Memory efficiency: How large can the state space be?
  - Scaling behavior
  - Further optimizations?
- Related Work and Conclusion

4 Draft of the to-be-done tasks

This section specifies tasks and their acceptance criteria that should be part of the week-based timetable.

- Technical project setup
  - The LaTeXenvironment is set up and working
  - A new project based on the STS template is set up and
  - The project's source is tracked using GIT
- Develop a good understanding on the model checking foundations and write them down
  - The foundation papers are re-read
  - Key ideas are noted down
  - The model checking part of the foundations chapter is written
- Develop a good understanding in CUDA programming and write it down
  - Re-Read the *CUDA C Programming* and the *Numba for CUDA GPUs* guides
  - Key ideas are noted down
  - The CUDA programming part of the foundations chapter is written
- Develop a good understanding in Swarm Verification and Grapple
  - Re-Read the related papers
  - Key ideas are noted down
  - The Model checking on the GPU chapter of the foundations chapter is written

- Do an experimental implementation of the Grapple algorithm
  - Decide whether to extend SPIN / DIVINE or do a standalone implementation
  - The waypoints model is written down in code
  - The host setup routines are implemented
  - The CUDA kernel is implemented
  - The program can be executed
- ...

References

[1] Jiří Barnat et al. "CUDA Accelerated LTL Model Checking". In: *2009 15th International Conference on Parallel and Distributed Systems*. Dec. 2009, pp. 34–41. DOI: 10.1109/ICPADS.2009.50.

[2] Ezio Bartocci, Richard DeFrancisco, and Scott A. Smolka. "Towards a GPGPU-Parallel SPIN Model Checker". In: *Proceedings of the 2014 International SPIN Symposium on Model Checking of Software*. SPIN 2014. San Jose, CA, USA: Association for Computing Machinery, 2014, pp. 87–96. ISBN: 9781450324526. DOI: 10.1145/2632362.2632379.

[3] Shenghsun Cho, Michael Ferdman, and Peter Milder. "FPGASwarm: High Throughput Model Checking on FPGAs". In: *2018 28th International Conference on Field Programmable Logic and Applications (FPL)*. 2018 28th International Conference on Field Programmable Logic and Applications (FPL). 2018, pp. 435–4357. DOI: 10.1109/FPL.2018.00080.

[4] Edmund M. Clarke, Thomas A. Henzinger, and Helmut Veith. "Introduction to Model Checking". In: *Handbook of Model Checking*. Ed. by Edmund M. Clarke et al. Cham: Springer International Publishing, 2018, pp. 1–26. ISBN: 978-3-319-10575-8. DOI: 10.1007/978-3-319-10575-8_1.

[5] Richard DeFrancisco et al. "Swarm model checking on the GPU". In: *International Journal on Software Tools for Technology Transfer* 22.5 (Oct. 2020), pp. 583–599. ISSN: 1433-2787. DOI: 10.1007/s10009-020-00576-x.

[6] Gerard J. Holzmann. "Explicit-State Model Checking". In: *Handbook of Model Checking*. Ed. by Edmund M. Clarke et al. Cham: Springer International Publishing, 2018, pp. 153–171. ISBN: 978-3-319-10575-8. DOI: 10.1007/978-3-319-10575-8_5.

[7] Gerard J. Holzmann, Rajeev Joshi, and Alex Groce. "Swarm Verification". In: *2008 23rd IEEE/ACM International Conference on Automated Software Engineering*. 2008 23rd IEEE/ACM International Conference on Automated Software Engineering. 2008, pp. 1–6. DOI: 10.1109/ASE.2008.9.

[8] Gerard J. Holzmann, Rajeev Joshi, and Alex Groce. "Swarm Verification Techniques". In: *IEEE Transactions on Software Engineering* 37.6 (2011), pp. 845–857. ISSN: 1939-3520. DOI: 10.1109/TSE.2010.110.