Reviews

Reviewer #1

Summary

The authors present an extension of the ESBMC tool, to verify CUDA GPU kernels by means of bounded model checking. This is an interesting and relevant topic, since GPUs are used increasingly often to speed up applications significantly. However, programming GPU software can be very tedious and error-prone. A formal verifier of GPU code can make the development process much more efficient.

Comments

- The present submission provides an ok overview of the capabilities of the tool, in my opinion, but the authors never really address the details to fully understand what the tool can and cannot do. For instance, when discussing the experimental evaluation, it is not explained what the source of the bugs is in the various benchmarks.
  - Are there clear differences in the types of bugs that the different tools being compared can detect?
  - Which types could ESBMC-GPU handle in particular better than the other tools, and why?
  - To what extent does ESBMC-GPU support CUDA?
  - Can ESBMC-GPU handle variables in different types of memory?
  - (Shared, global, thread registers?), and inter-warp communication, for instance?
  - In that respect, what does COM cover, and what remains to be added?
  - Is the memory model for the different types of memory complete?
- Another paragraph that raises questions is the second on page 3. It seems to say that COM uses pthread functions to simulate GPU kernels. Is that so, and why is it necessary to simulate them? Is this safe to do, because some features in CUDA are not available in C/C++. How do the authors handle those, or do they not?

- At some point, the authors mention in passing that Microsoft C++ AMP projects have been used for the evaluation. Does this mean that this language is also supported, or are the benchmarks translated?
- Concluding, I would like to see an overall more detailed discussion of the capabilities of the tool.

Detailed comments:

- Page 2
  - space-state exploration → state-space exploration;
  - ESBMC-GPU builds on top of ESBMC → ESBMC-GPU is built on top of ESBMC;
- Page 3
  - consists in → consists of;
  - Besides, in: remove "Besides";
  - reduce state space → reduce the state space;
- Page 4
  - if multiple threads perform unsynchronized 90 access to shared-memory locations: to the same memory locations, it is crucial to mention that;
  - has been systematically explored → have been systematically explored;
  - Therefore: the next conclusion does not follow logically from the previous statements, so "therefore" should be removed here;
- Page 5
  - Figure 2: between lines 17 and 18, the contents of `dev_a` should be copied to a with a `cudaMemcpy`, otherwise the subsequent check is quite useless;
- Page 6
  - In addition → On the other hand;

– Besides, it further simplifies verification models: further compared to what? How does it make it easier?

– if compared: remove “if”;

– stream interleaving: what is this?

– to reduce the number of thread interleavings: in what respect? Not reducing the number of threads, right? You already use the 2-thread simplification;