EDAN70 Projects in Computer Science

Jonas Skeppstedt

March 6, 2025

Projects with Modelon

1. Performance evaluation of LLVM abstract data types similar types in ISO C++ standard library

   For example LLVM SmallVector

2. Evaluation of the custom RTTI in LLVM vs ISO C++

3. To reduce time to produce an executable program (actually a shared library), are there simple — and `fast` — optimizations in LLVM that reduce the time to produce the program, and can these likely be done faster by a program that produces LLVM IR?

4. Evaluate the options when producing LLVM IR from Modelon Impact in parallel?

5. Benchmark JastAdd both if possible with a Java profiler but also with Valgrind/Cachegrind, preferably in the part of Modelon Impact implemented in Java with JastAdd.

6. Benchmark the new Modelon Impact backend written in C++ — what takes time and why?

7. Investigate the effects of adding compiler directives such as for ISO C `restrict` in the LLVM IR?

   See also: `https://lup.lub.lu.se/lupStat/record/4941274`

8. What is the best approach to inline C functions from a library? Are there restrictions on how the library functions already have been partially optimized?

9. Sequential and parallel bipartite graph matching in C or C++

Slice compiler optimization hardware support

If you are interested in multicores and know VHDL or Verilog there is an exciting opportunity to explore how to hide memory latency.

In one completed and one ongoing exjobb, a new method to hide memory latency of indirect memory accesses is explored. The idea is to use compiler analysis to slice a loop into two threads, where the first inserts addresses in a new hardware structure with a queue which prefetches data, and the second thread reads from this queue. The loop is basically duplicated into the two threads and dead code elimination is used to remove unneeded instructions.

The completed exjobb used a simulator of an IBM Power multicore, and the ongoing is currently implementing this hardware in an open source Power processor using FPGA from IBM, called IBM Microwatt.

Recently IBM released a new version of Microwatt with support for multiple cores.

The project in computer science is to study the multicore Microwatt, and getting it to boot Linux (assumed to be working already), and then integrate the slice hardware support in this new multicore Microwatt. If this then looks interesting, it can be the basis for a very fun exjobb in the autumn to explore both this and possibly also hardware support to make LWSYNC faster as a possible additional part of the exjobb. Slice is based on two papers from Princeton and MIT [1, 2]. A link to the completed exjobb is here:
`https://lup.lub.lu.se/student-papers/search/publication/9175439`

References

[1] Aninda Manocha, Tyler Sorensen, Esin Tureci, Opeoluwa Matthews, Juan L. Aragón, and Margaret Martonosi. Graphattack: Optimizing data supply for graph applications on in-order multicore architectures. *ACM Trans. Archit. Code Optim.*, 18(4), sep 2021.

[2] Quan M. Nguyen and Daniel Sánchez. Phloem: Automatic acceleration of irregular applications with fine-grain pipeline parallelism. In *IEEE International Symposium on High-Performance Computer Architecture, HPCA 2023, Montreal, QC, Canada, February 25 - March 1, 2023*, pages 1262–1274. IEEE, 2023.