Tapir: Embedding Fork-Join Parallelism into LLVM’s Intermediate Representation

Why existing compilers for parallel code suck and how to fix them

William S. Moses, Tao B. Schardl, Charles E. Leiserson

WHAT A GOOD COMPILER CAN DO

Compilers are wonderful tools that allow us to write code in high-level languages. We also depend on them to optimize our code. The difference between a good and bad compiler can be enormous. A poor compiler can have serious adverse consequences for applications that demand efficiency.

WHERE COMPILERS FAIL

Modern compilers allow programmers to easily write parallel code with high-level frameworks such as OpenMP and Cilk. In these frameworks, programmers specify tasks that may be run in parallel, such as the iterations of the loop in Fig 3.

TAPIR: PARALLELISM IN THE COMPILER

Compilers aren’t able to optimize parallel code well because they have no way of representing the parallel semantics of programs. As a result, they extract parallel programs into separate functions that they then pass as arguments to a parallel runtime. This confusing way of representing a program prevents the compiler from being able to do any of its standard analysis and optimization.

MULTICORE PROCESSORS

For the past several decades, we have been able to rely on Moore’s law to provide us with improvements in performance by roughly doubling the clock speed of processors every few years. Stems from fundamental physical limitations on power, however, the performance of individual cores is no longer improving at the rate it once was. As a way to cope with power limitations, semiconductor vendors add many processing cores to a single machine in order to continue to scale performance.

EVALUATION

Tapir allows the compiler to represent parallel programs as a natural extension to serial code without a dependency between parallel tasks. Existing compiler optimizations work with Tapir with zero or minimal modification. In fact, to add Tapir constructs to the LLVM compiler required modifying only 0.1% of the 4-million-line codebase.

Win Best Paper at the 2017 Conference on Principles and Practice of Parallel Programming