Multi2C: An Open-Source GPU compilation framework from OpenCL to Southern Islands ISA
Charu Kalra, Xiang Gong, Rafail Ubal, and David Kaeli
Department of Electrical and Computer Engineering, Northeastern University, Boston, MA

ABSTRACT
- Popularly of GPU computing has led to extensive research in the design and optimization of workloads for GPUs.
- Efficient utilization of hardware resources is the key to improve GPU application performance.
- Compiler level optimizations directly influence the architecture and provide superior control over hardware resources. Architectural complexity of the GPU makes it challenging to implement advanced optimizations at the compiler level.
- Additionally, the lack of open source tools (compilers, drivers, runtimes, etc.) from hardware vendors has limited the scope of optimizations.
- We develop an open source framework that translates OpenCL to LLVM (frontend), and translates LLVM IR to the AMD Southern Islands GPU ISA (backend).
- We have incorporated a backend transformation pass to optimize vector/scalar register usage.
- Our compiler framework shows a speedup of 1.25x and 21.7% savings in vector registers over the baseline greedy register allocation.

BACKGROUND
- The SI ISA contains two category of instructions: vector instructions, which function in a SIMD manner, and scalar instructions, which only execute once per wavefront.
- Registers associated with the scalar unit hold one value per wavefront. 256 vector general purpose registers and 104 scalar general purpose registers for AMD Radeon 7970 GPU
- Work-items are bundled together in work-groups and execute the same instruction at the same time in a SIMD manner.
- Conditional branch instructions are resolved differently for some of the work-items in a wavefront, which leads to branch divergence.
- The Southern Islands ISA utilizes a series of execution masks to address branch divergence. The execution mask is a 64-bit map, where each bit represents the active status of an individual work-item in the wavefront.

FRAMEWORK
- Frontend: Converts OpenCL to LLVM bit code
- Backend: Converts LLVM bit code to SI ISA
- Assembler: Converts SI ISA to binary
- Multi2C backend also supports Clang based OpenCL frontend

REGISTER ALLOCATION
- Liveness Analysis: Data-flow analysis is used by compilers to calculate the live range of every variable. The Live Range is the time between a write to a register followed by all the uses of that register until the next write.
- A variable is live at a certain point if it holds a value that is needed in the future.
- Vector and scalar registers follow different paths in the control flow graph (CFG).
- The multi2c framework supports Clang based OpenCL frontend

RESULTS
- Interference Graph: Each node represents the live range of a particular value.
- An edge between two nodes indicates interference of two live ranges.
- Scalar and Vector memory operations allow program instructions to fetch multiple/dwors at a time (e.g., S_LOAD_DWORD and S_BUFFER_LOAD_DWORD allow a shader program to load 1-16 dwors at a time into SRGPR).
- B: Big node size > 1
- S: Small node size = 1

CONCLUSION AND FUTURE WORK
1. Developed an open source compiler framework for AMD GPUs.
2. Register allocation improves the performance and occupancy on the GPU, except in applications that are bound by LDS usage.
3. Working on integrating LLVM R600 with the Multi2Sim heterogeneous simulation framework.