Merrimac Simulation Tools

Jung Ho Anh, Abhishek Das, Mattan Erez, Timothy Knight, Nuwan Jayasena, Francois Labonte, William J. Dally

The Merrimac streaming supercomputer project requires tools to simulate correctness performance and evaluate implementation tradeoffs. The infrastructure of the Merrimac cycle accurate simulator is inherited from the Imagine project. In addition to Merrimac specific features, the Merrimac simulator also models the scalar code running on the RISC CPU. Cycle accurate simulation of a system with many nodes is not feasible so we calibrate a higher level simulator, the SVM simulator to run simulations with many nodes.

Simulation Flow

- Will use different tools for simulations at different resolutions
- Functional simulator to develop applications
- No timing information, just check for correctness
- Cycle-accurate node simulator for accurate timings of small-scale simulations
- Accurate timing measurements, but simulations take a long time to run
- SVM simulator for large scale simulations
- Uses parameterized machine model
- Timings calibrated by cycle-accurate simulator

Merrimac SVM Machine model

- Machine Model used to characterize targets of SVM code
- Describes both
  - Physical composition, connections
  - Performance characteristics
- Allows for Hierarchies (nice to model packaging hierarchy)
- Used by:
  - Rstream compiler to generate efficient code
  - SVM simulator to estimate run-times (bandwidth, kernel runtimes)

SVM Simulator Runtime Estimates

- SVM simulator models run times of kernels
  - Use schedule from Merrimac kernel scheduler
  - Linear cost function model:
    - Fixed cost for kernel includes prologue and postscript
    - Per input stream element cost (inner loop)
  - Doesn’t account for conditionals in kernels (possible extension)
- SVM simulator models transfer times of stream memory operations
  - Cache included in simulator, gets hit-rate on cached streams
  - Outstanding memory requests take a certain amount of time based on their locations
  - SVM simulator doesn’t currently account for run-time on scalar processor and memory traffic generated by it
  - Could use thread run-time to estimate it’s run-time
  - When we are waiting on concurrent (scalar, kernel, memory) operations, the longer one dictates the wait time.

Multi-node Simulation

- Perform small multi-node simulations using the accurate node simulator
  - Feasible for up to 4-8 nodes
  - Results in fairly accurate performance measurements
  - For more nodes, cycle-accurate simulations will take a prohibitive amount of time
- Larger scale simulations with 100s or 1000s of nodes will be done with the SVM simulator
  - SVM simulator will get kernel timings directly from schedules
  - Memory timings calibrated by comparisons with the accurate node simulator

Status: October 2004