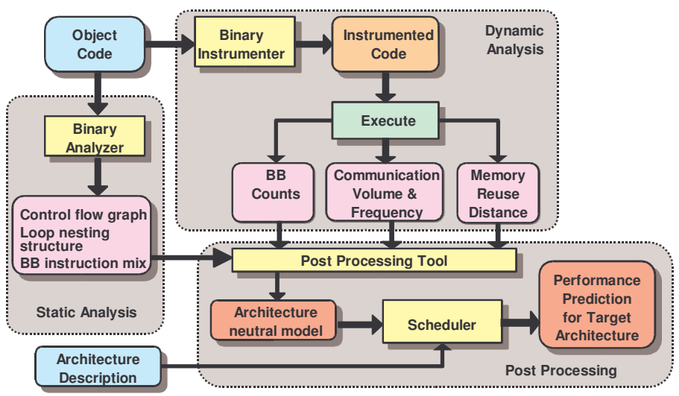
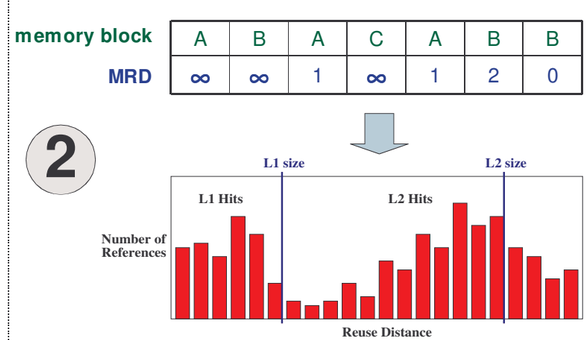
# Cross-Architecture Performance Predictions for Scientific Applications Using Parameterized Models


Gabriel Marin
mgabi@cs.rice.edu

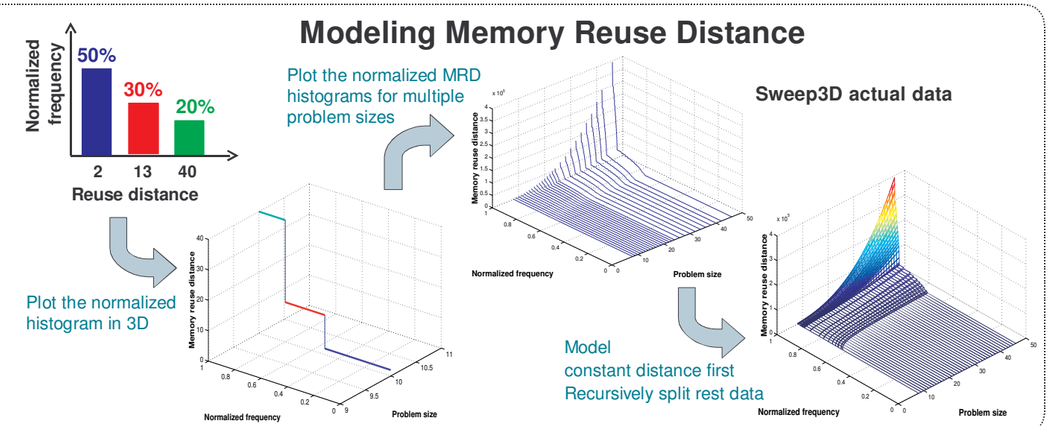
John Mellor-Crummey
johnmc@cs.rice.edu


## Measuring Memory Reuse Distance




## Modeling Memory Reuse Distance




## Why Performance Prediction?

1

- Expected execution time
- Causes of inefficiency
- Estimate payoff of optimizations
- Execution characteristics scaling
- Resource selection at launch-time
- Future architecture design

## Performance Prediction Challenges

- Performance depends on:
  - architecture specific factors
  - application characteristics
  - input data parameters
- Difficult to model execution time
- Collecting data at scale is expensive

## Approach

Separate contribution of application characteristics

- Measure the application-specific factors
  - static analysis
  - dynamic analysis
- Construct scalable models
- Explore interactions with hardware

## Challenges of Program Analysis

- Application issues
  - multiple languages, programming models, and communication models
  - heterogeneous applications
  - source code unavailable

Use binary analysis and instrumentation

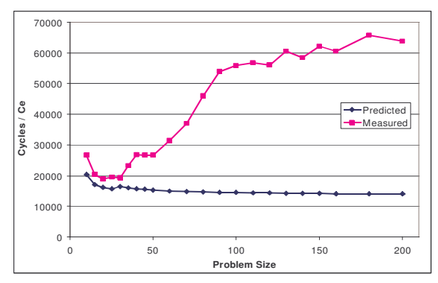
## Binary Instrumentation Process

- Two measurement modes:
  - block execution frequency + communication characteristics
  - memory reuse distance
- Build control flow graph
- Compute loop nesting structure
- Block execution frequency instrumentation
  - assign weights to edges
  - build maximum spanning tree
  - place counters on remaining edges
- Memory instrumentation at each load/store
- Instrument each communication call
- Dynamic library routine handles instrumentation events
  - change on-line processing independently

## Example: Predict execution time of Sweep3D on IA-64

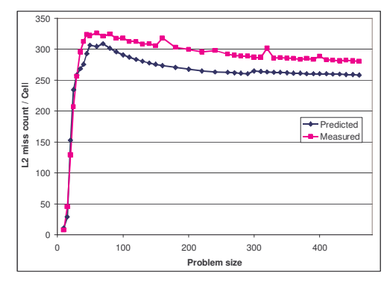
3

### Predicted execution time in absence of cache misses

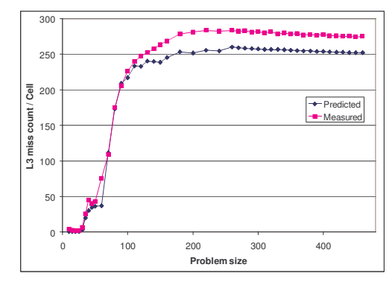



### Cache Miss Predictions for Sweep3D

L2 miss count




L3 miss count



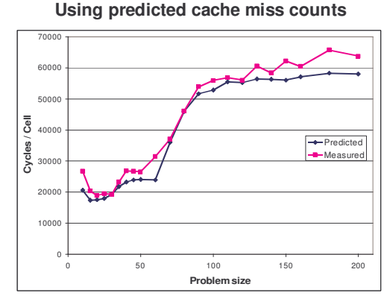

### Consider memory hierarchy penalty

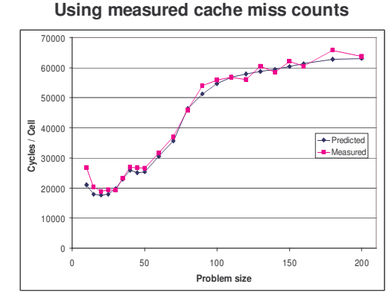
- Minimum access time to L2: 6 cycles
- Minimum access time to L3: 15 cycles
  - L2 miss penalty: 9 cycles
- Access to local memory: 180 cycles
  - L3 miss penalty: 165 cycles
- Itanium2 is an in-order processor
  - memory accesses are not-blocking
  - data dependencies stall the CPU

Predictions are based on data collected for a set of problem sizes between 10 and 57

### End-to-end prediction for Sweep3D

Using predicted cache miss counts




Using measured cache miss counts