Analytical Evaluation of Shared-Memory Systems with ILP Processors

Daniel Sorin, Vijay Pai, Sarita Adve,
Mary Vernon, and David Wood

Computer Science	Electrical & Computer Engr.
Univ. of Wisconsin	Rice University

July 1, 1998

Overview

Problem: Detailed simulation of multiprocessors with modern (ILP) processors takes hours or even days

Solution: Develop highly accurate analytic model that can explore memory system design space in seconds

Side benefit: Insight into application behavior

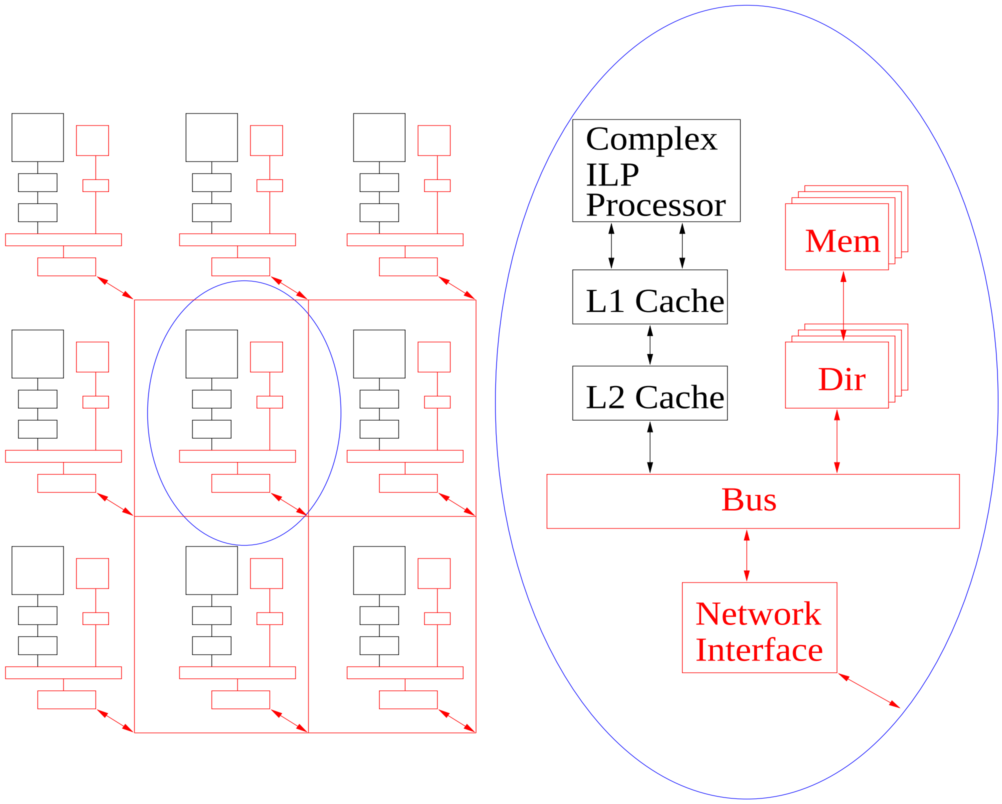
Key: Application parameters invariant to modeled architecture parameters.

Outline

- What exactly are we modeling?
- How are we modeling it?
- How well are we modeling it?
- What can we learn from the model?
- What's next?

What We Are Modeling

Hardware:



- Proc/cache + memory subsystem
- Memory subsystem = buses, memories & directories, mesh interconnect
- 3-state invalidation directory protocol
- Release consistency

What We Are Modeling

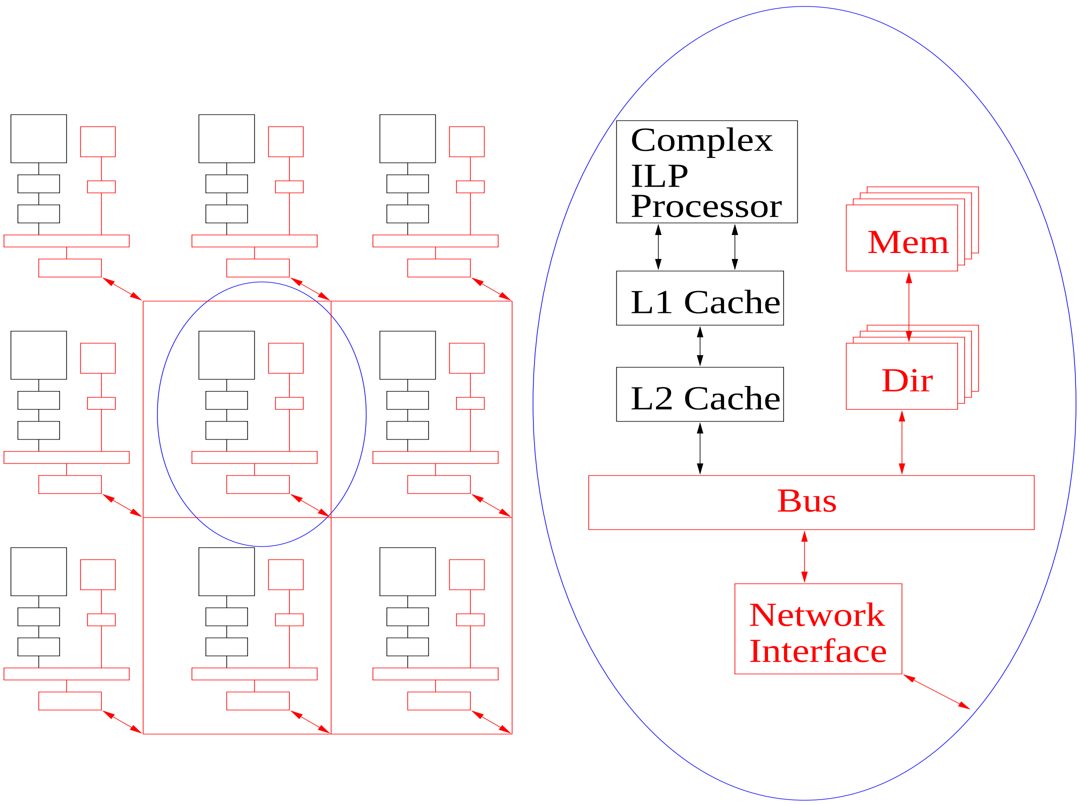
Software:

- Homogeneous applications
  $\rightarrow$ all procs have same statistical behavior
- Many SPMD programs are homogeneous

Model Parameters

3 Types of Parameters:

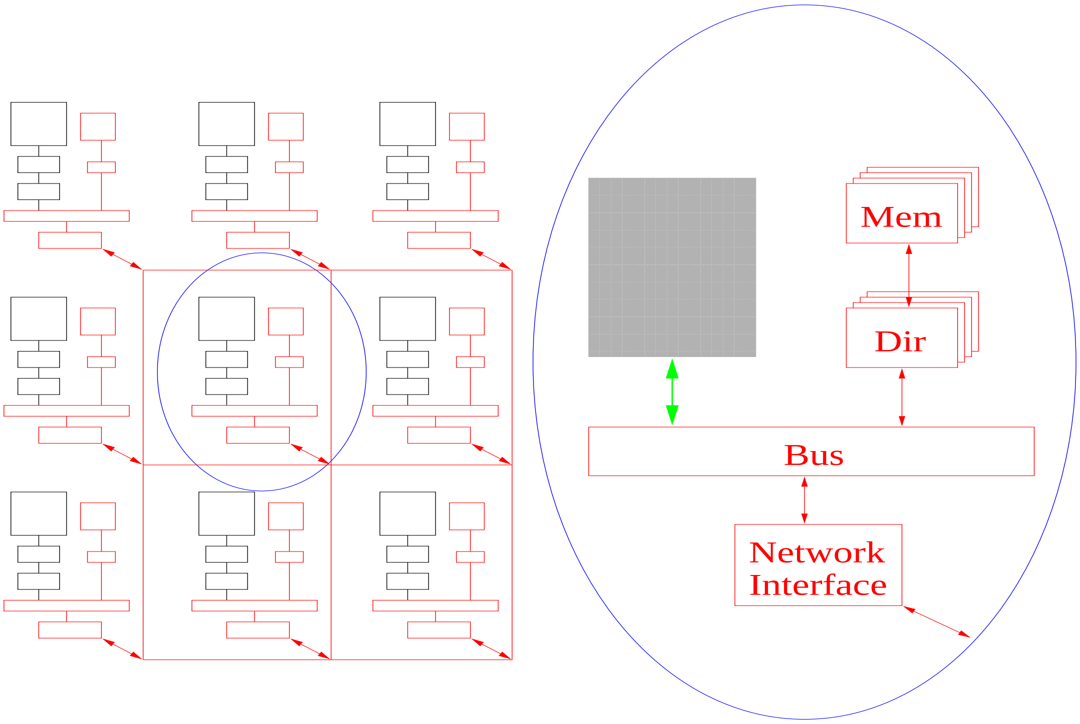
(1) Processor/cache characterization



Model Parameters

3 Types of Parameters:

(1) Processor/cache characterization



- Request rate
- Burstiness
- Dynamic # outstanding requests
- Prob. read coalesces with write

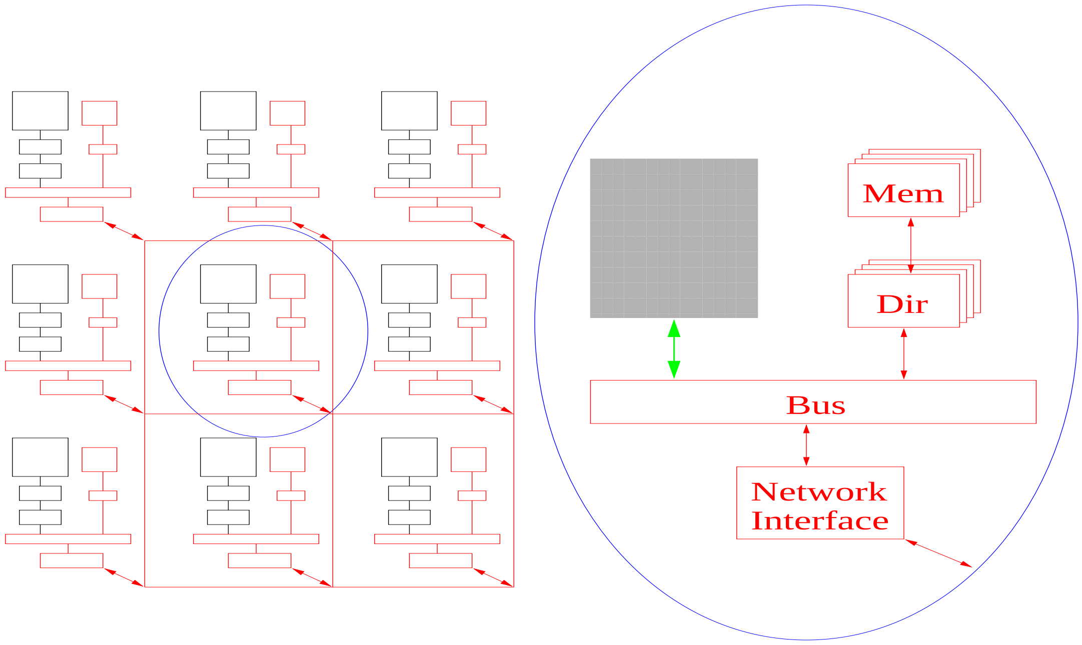
→ Obtain from new, very fast FastILP simulator

Model Parameters

3 Types of Parameters:

(1) Processor/cache characterization

(2) Memory request characterization



- Read/write
- Local/remote
- Cache/memory

  Etc.

→ Obtain from conventional fast MP simulator with simple processor (no ILP)

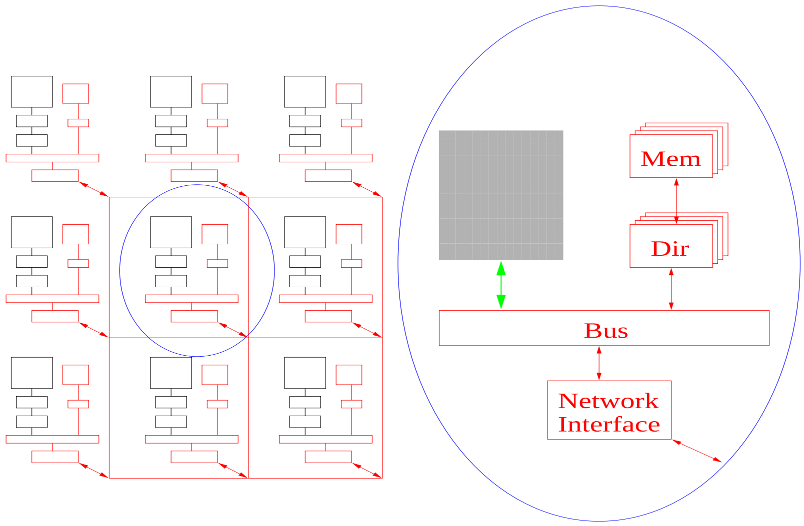
Model Parameters

3 Types of Parameters:

(1) Processor/cache characterization

(2) Memory request characterization

(3) Memory and interconnect architecture



- bus latency
- DRAM latency
- NI latency
- memory interleaving, directory organization
  Etc.

Key: Parameters of type (1) and (2) are not sensitive to type (3) parameter values

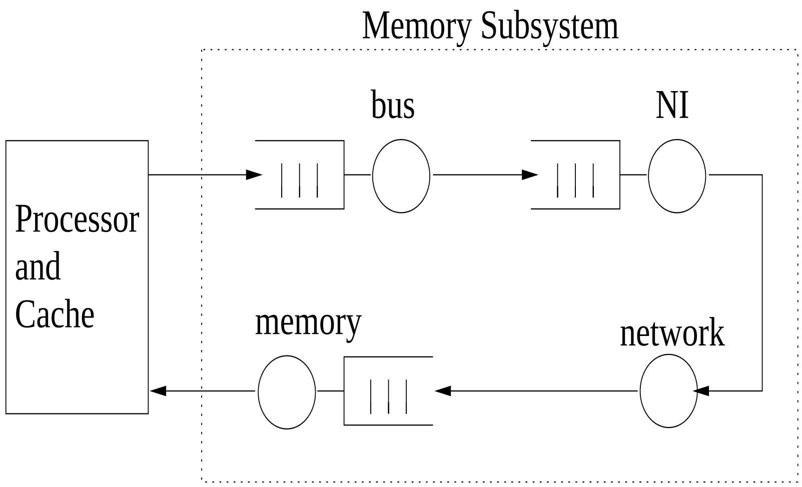
FastILP

- Obtains type (1) parameters
- Two orders of magnitude faster than detailed simulator
- Speedup techniques:
  - Trace driven
  - Uses single processor trace
    - Assumes application homogeneity
  - Abstract memory system
    - Long memory access latencies

Overview of Analytic Model

Low-level model (easy):

- Mean value analysis (MVA) equations for contention delays at bus, memory, network interface
- 1,000 standard MVA equations



$$R = R_{bus} + R_{NI} + R_{network} + R_{memory}$$

$$R_{bus} = R_{bus-local} + R_{bus-remote}$$

High level model (hard):

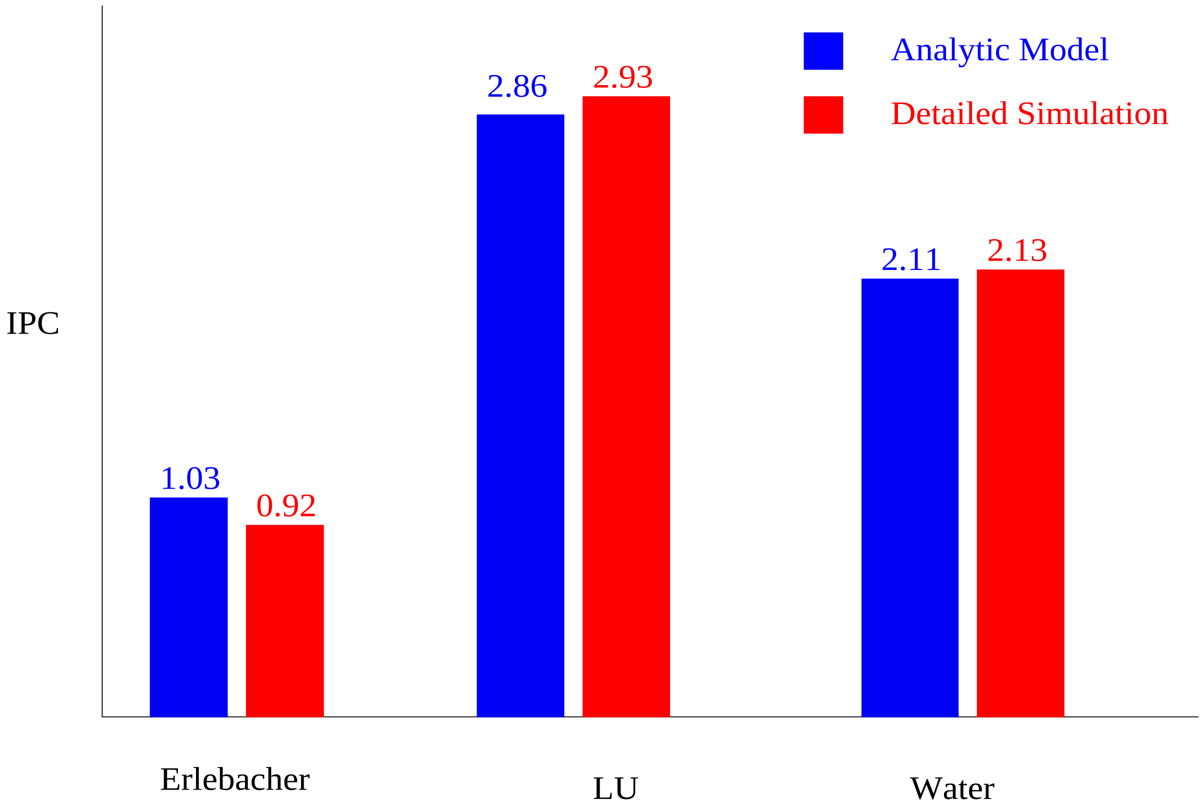
- Two types of blocking behavior:
  - Reads that cannot retire
  - Full MSHRs
- Varying number of outstanding requests
- Mis-speculated reads
- Reads coalescing with writes

Validation of the Model

- Validation performed against RSIM, a detailed simulator of an ILP multiprocessor
- Validated with 5 benchmarks: Erlebacher, FFT/FFTopt, LU/LUopt, Radix, and Water
- Wide variety of application behavior
- Different system configurations:
  - Number of processors
  - Instruction window size
  - Fast and slow components

Sample Validation Results

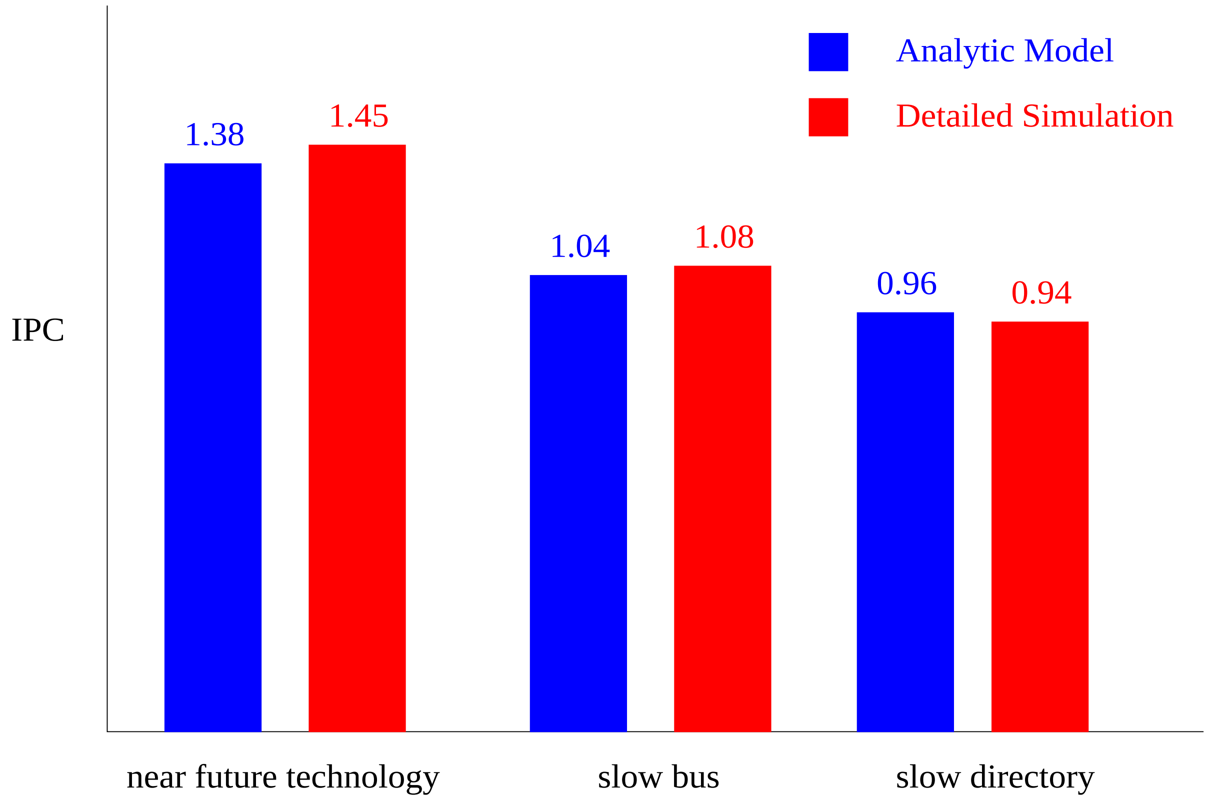
Different applications:



- Max error: 12% (Erlebacher)
- IPC ranges from 0.9 to 2.9

Sample Validation Results

Erlebacher with different memory subsystem designs:



- Input parameters measured once
- Accurate absolute and relative estimates

Example Applications of the Model

- Insights from input parameters for each application

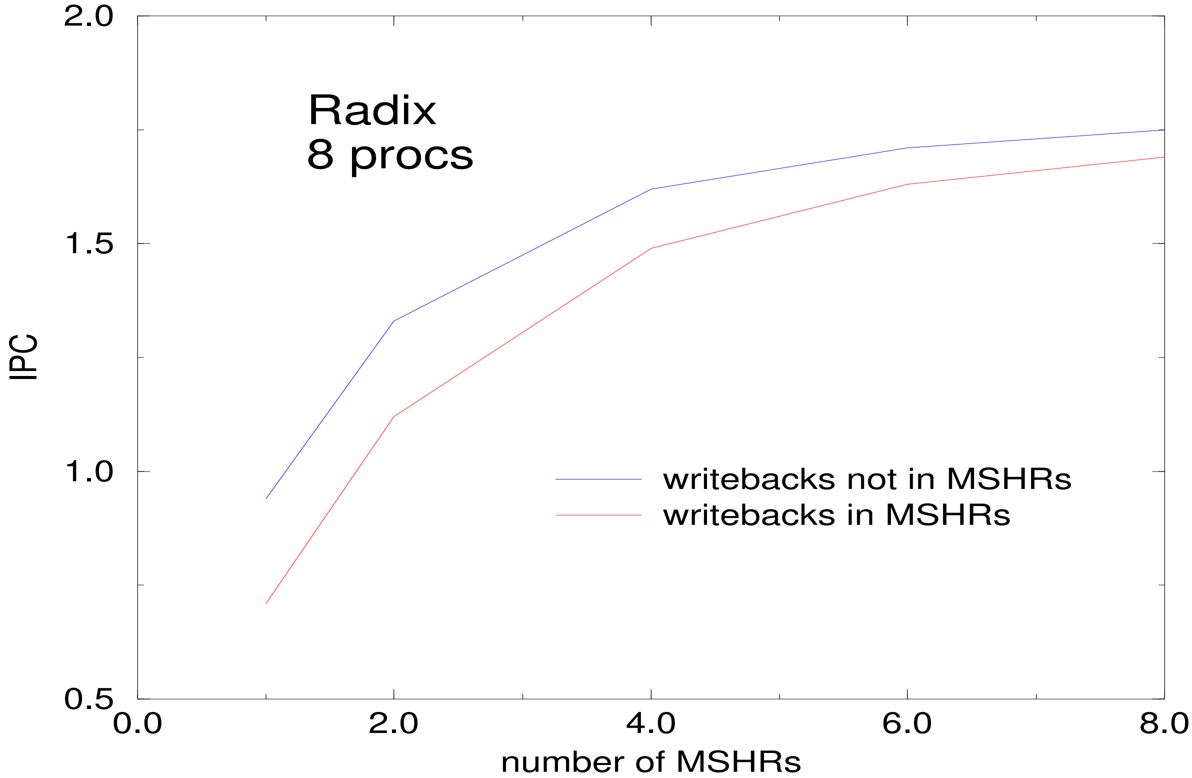
- Memory system design
  - Varying number of MSHRs
  - Programmable coherence controller
  - Different directory organizations

Insights from Application Parameters

Why isn't IPC higher?

- Different reasons for each application
  - FFT and Erle have frequent misses (small time between requests)
  - Radix never has more than one outstanding read request
  - LU has high overlap of reads, but most reads are to remote memory
  - Water has poor overlap and requires many 3-hop transfers
- Use these insights to improve applications and architecture

Varying Number of MSHRs

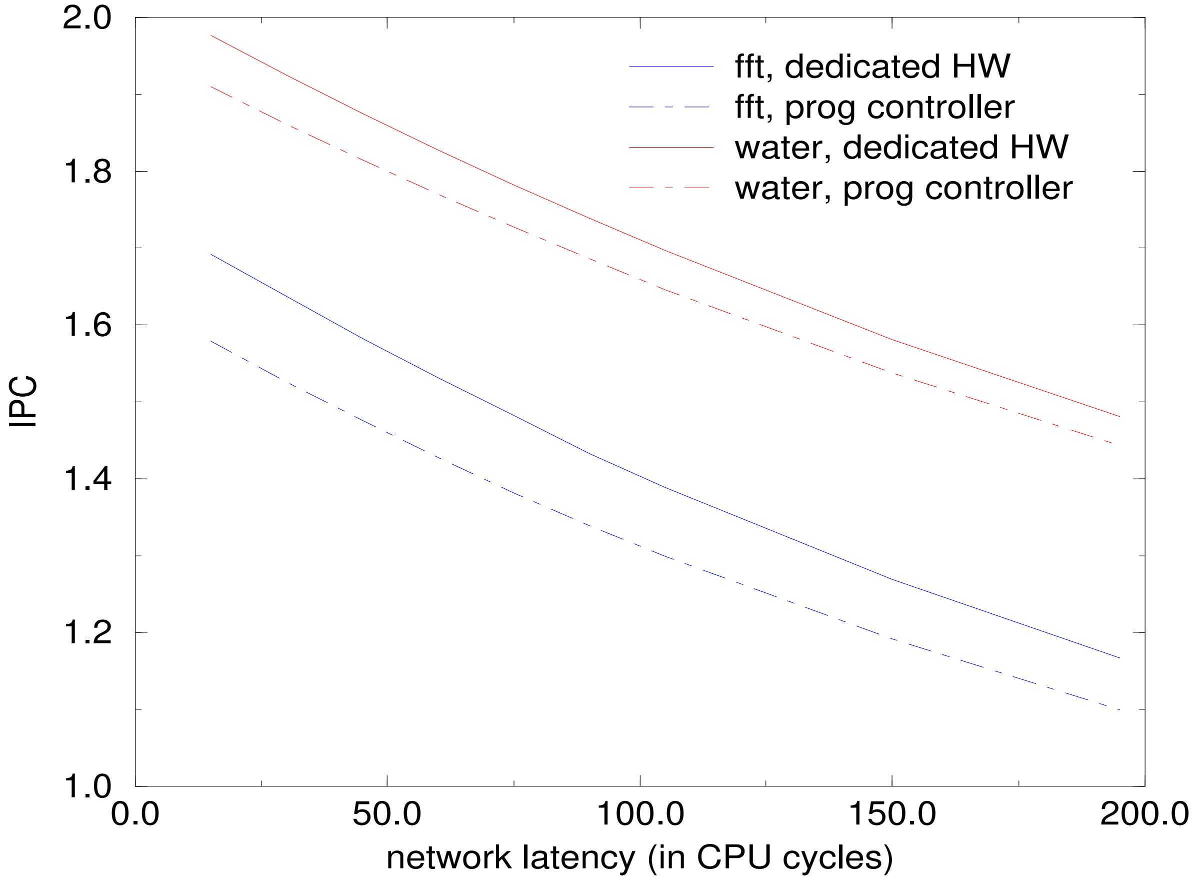


- Greater than 6 MSHRS: not much improvement

- Writebacks in MSHRs causes 5-10% degradation (if #MSHRS $\geq$ 6)

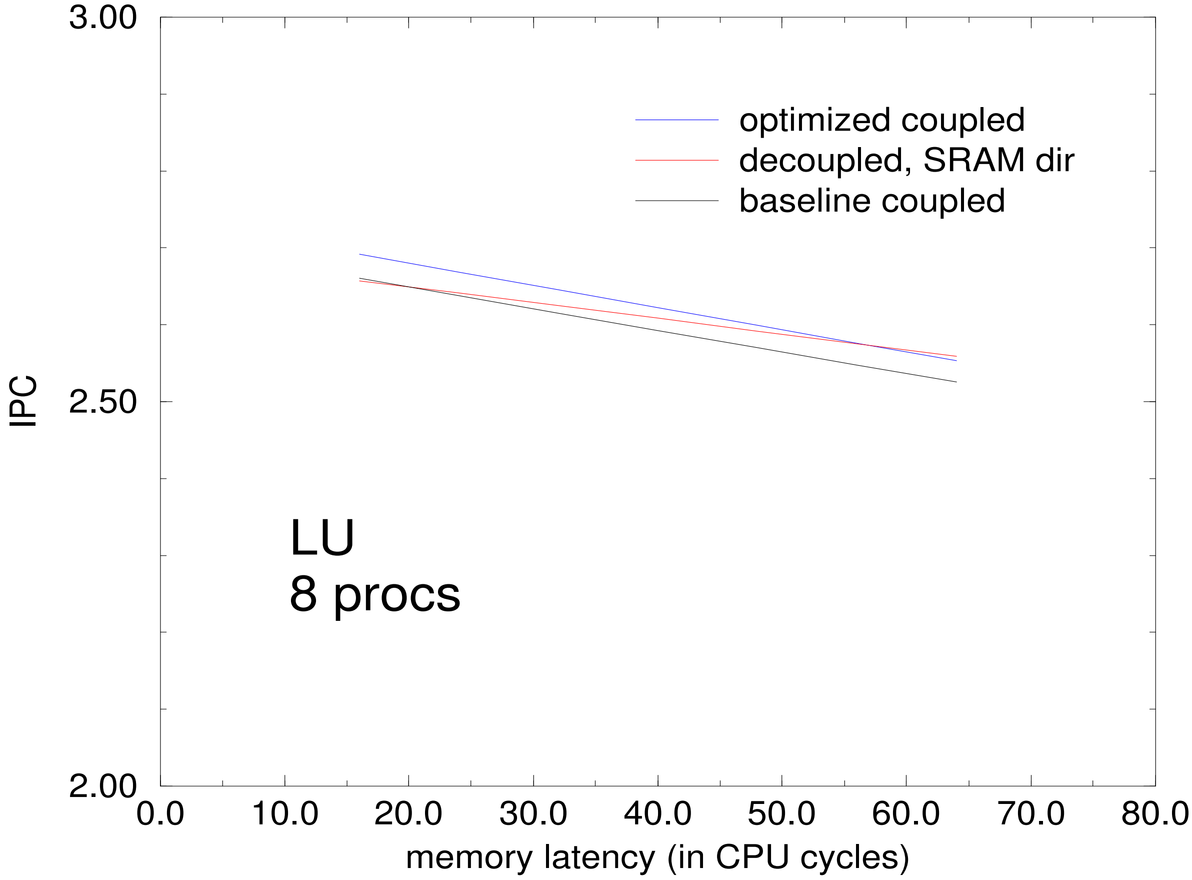
  (preliminary conclusions, Radix input parameters)

Programmable Coherence Controller



- Programmable coherence controller assumed to be twice as slow as memory
- IPC reduction is 3-7%.

Different Directory Organizations



- Performance on LU is similar for all configurations.

Conclusions

- Model is many orders of magnitude faster than detailed simulation
- Within 12% of detailed simulation (or less)
- Parameters can be obtained fairly quickly
- Input parameter values yield insight into application behavior
- Model can easily be adapted for other memory system architectures

On-Going Work

- Extensions to the model
  - Non-homogeneous applications (e.g., MP3D)
  - More accurate modeling of burstiness
  - Model synchronization more precisely (with LogP model)
- Further applications of the model
  - Study memory system architectures
  - Improve application performance