# ECE 259 / CPS 221
# Advanced Computer Architecture II
# (Parallel Computer Architecture)

## Evaluation – Metrics, Simulation, and Workloads



Duke University

# Outline

- **Metrics**

- **Methodologies**
  - **Modeling**
  - **Simulation**

- **Workloads**

# Performance Metrics

- **How do we tell if our design is good?**
- **Performance metrics**
  - **Clock speed (gigahertz)?  No!  Why not?**
  - **Instructions per cycle?  No!  Why not?  (tougher question)**
  - **Database transactions per second?**
- **What's important?  Depends on workload …**
  - **Latency → interactive computing**
  - **Throughput → batch jobs/queries**
  - **Availability → enterprise applications**
  - **Power → mobile computing … and everything else now, too**
  - **Cost-efficiency → everything but perhaps supercomputing**

# Metrics and Units

- **Latency (an aspect of performance)**
  - **Response time**
- **Throughput (another aspect of performance)**
  - **Transactions per cycle (e.g., TPM-C or TPM-H)**
- **Availability**
  - **How many “nines”  (e.g., 5 nines = 99.999% available)**
- **Power: watts**
- **Energy: joules**
- **Hybrid metrics capture more than one aspect**
  - **Cost-efficiency: dollars-seconds**
  - **Power-delay (energy-delay): watts-seconds (joules-seconds)**
  - **Performability: combines performance with availability**

# Secondary Metrics

- **Metrics that we can use for insight, debugging, etc.**
  - **Quantify specific aspects of system, not holistic behavior**

- **Examples**
  - **Instructions per cycle (IPC)**
  - **Cache hit rates**
  - **Average memory request latency**
  - **Average network link utilization**
  - **Fraction of directory requests that require 3 hops**
  - **Etc.**

- **You can use these metrics to explain results**
  - **Otherwise, results are just inscrutable, unjustified numbers**

# Comparing to Prior Work

- How does your idea compare to prior work?
  - This is how we show that our idea is worthy of publication
  - E.g., 50% better throughput on TPC-C, but with 20% more power
- Why is comparison difficult?
  - Impossible to exactly reproduce experimental setup
- Example differences in experimental setup
  - Different system model
    - » Different ISA, microarchitecture, network, etc.
  - Different workloads (or same workloads compiled differently)
    - » Different OS
    - » Even for same exact application, can have different jobs running in the background (e.g., kernel daemons)
  - Different simulator (or different configuration of simulator)
    - » Assumptions about latencies, bandwidths, etc.

# Fair Comparisons

- **Ideally, we'd make perfectly fair comparisons**
  - **Compare "apples and apples"**
- **If impossible, then give benefit of doubt to prior work**
  - **Assumptions about prior work should be optimistic**
- **Assumptions about our work should be pessimistic**
  - **Don't assume that our 4MB cache can be accessed in 1 cycle**
  - **Find the worst-case scenario for our system**
  - **Assume that future trends will be less favorable than is likely**
- **Show that, even in our worst case, we still do well**
  - **Otherwise, readers will be less convinced**

# Cost Effective Computing (Wood & Hill)

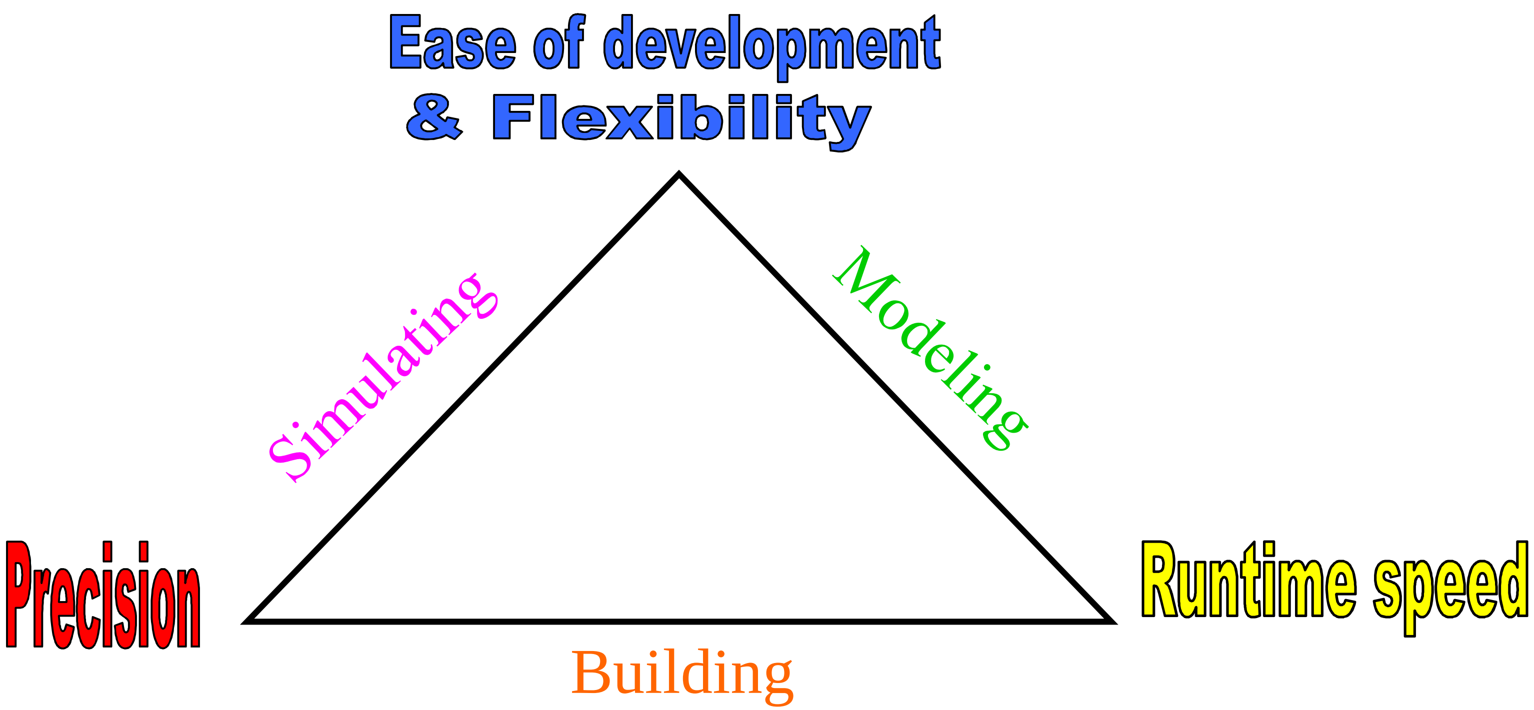
- **DISCUSSION**

# Outline

- **Metrics**
- **Evaluation Methodologies**
  - **Modeling**
  - **Simulation**
- **Workloads**

# Ways to Evaluate New Architectures

Tradeoff between three desired features



# Building

- **Construct a hardware prototype**
  - **ASIC vs. FPGA**


- **Advantages**
  - \+ **Way cool to show off hardware to friends**
  - \+ **Runs quickly**
- **Disadvantages**
  - **Takes long time (grad student time!) to build**
  - **Expensive**
  - **Not flexible (esp. ASIC)**

**ASICs generally too labor intensive for research studies, but FPGAs are viable options in many cases**

# Modeling

- **Mathematically model the system**
  - **Use probabilities and/or queuing models (see ECE 255/257)**

- **Advantages**
  - \+ **Very flexible**
  - \+ **Very quick to develop**
  - \+ **Runs quickly**
- **Disadvantages**
  - **Cannot capture effects of system details**
  - **Architects are skeptical of models**

**Generally OK for back of the envelope estimates**

# Simulating

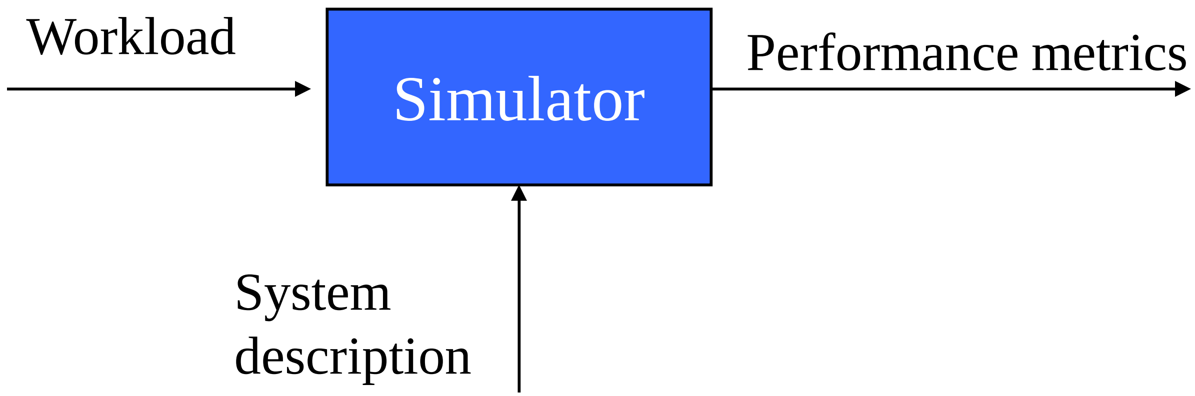
- **Write a program that mimics system behavior**

- **Advantages**
  - + **Very flexible**
  - + **Relatively quick to develop**

- **Disadvantages**
  - – **Runs slowly (e.g., 30,000 times slower than hardware)**

**Method of choice for most architectural research**

# Simulation Challenges




Tough problems associated with each arrow!

# Applications to Simulate

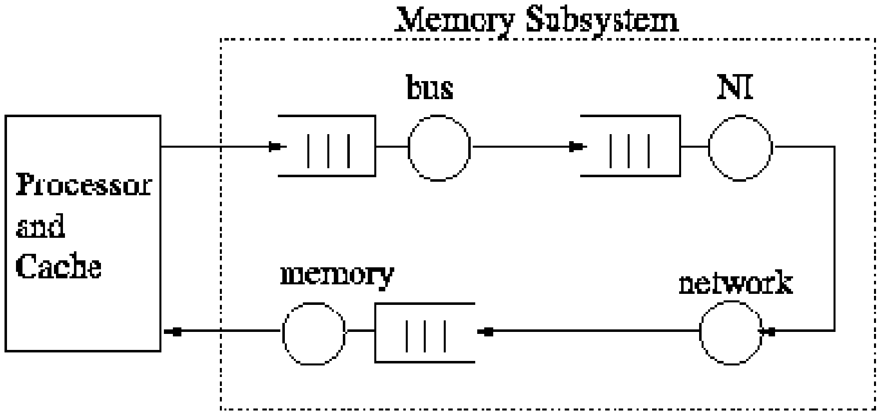
- We care how system does on important applications
- We'll talk about this in a few slides …

# Describing Simulated System

- How detailed must our simulator be?
- Model every transistor in the processor?
  – Would take too long
- Abstract away details of processor organization?
  – Could miss important effects of processor features
  – Could achieve wrong conclusion
- Need balance
  – Model in detail only where necessary
  – E.g., model memory system in detail, but abstract disks

# Analytic Model of Shared Memory System

- **Queuing model can capture behavior of system**
- **Optional reading from ISCA 1998: *"Analytic Evaluation of Shared-Memory Parallel Systems with ILP Processors"***
  - **Models processor cores as request generators**
  - **Models cache coherent memory system (directory protocol) as queuing system where requests (customers) access**
  - **Outputs average utilizations, throughputs, waiting times, etc.**



# Some Processor Simulators

| | Not full-system | Full-system |
|---|---|---|
| uniproc | SimpleScalar | Simics (Virtutech)<br>Simics+GEMS (Wisconsin)<br>Simics + Flexus (CMU)<br>M5 (Michigan)<br>PTLsim (Suny-Binghamton)<br>ASIM (Intel)<br>SimOS (Stanford) |
| MP | Liberty (Princeton)<br>SESC (UCSC/Illinois)<br>RSIM (Rice)<br>Wisconsin Wind Tunnel | |

# Simics

- **Simics is a full-system commercial simulator**
- **PRESENTATION**

# RAMP

- **PRESENTATION**

# Outline

- **Metrics**
- **Methodologies**
  - **Modeling**
  - **Simulation**
- **Workloads**

# Workloads

- We care how system does on important applications
- But who defines "important"? (I do!)
- Types of applications
  - Scientific (genomics, weather simulation, protein folding)
  - Commercial (database, web serving, application serving)
  - Desktop (office productivity software, multimedia)
  - Portable (voice recognition)
  - ???

# DEC/Compaq/Intel (?) Workload Analysis

- **Commercial workloads are different from scientific**
- **PRESENTATION**

# “Simulating $2M Server on $2K PC”

- **Commercial workloads are different from scientific**
- **Simulating them requires extra work**

- **PRESENTATION**