Computer Systems Performance Analysis: An Introduction

Dr. John Mellor-Crummey

Department of Computer Science
Rice University

johnmc@cs.rice.edu

Course Objectives

- Learn techniques to approach performance problems
  - —compare two systems
  - —determine the optimal value of a parameter
  - —identify performance bottlenecks
  - —characterize the load on a system
  - —select the number and size of system components
  - —predict the performance of future workloads
- Understand the use of different analysis strategies
  - —measurement, simulation, analytical modeling
- Learn mathematical techniques for performance analysis
- Develop skill applying these techniques in practice
  - —homework assignments and final project

Topics

- **Introduction**
- **Performance Measurement**
- **Workload Selection and Characterization**
- **Fundamentals of Probability Theory and Statistics**
- **Analysis of Sample Data including Regression Analysis**
- **Performance Modeling**
- **Experimental Design and Analysis**
- **Simulation including Random Number Generation**
- **Queuing Theory**

Approach Problems Intelligently

Learn to select appropriate evaluation techniques, performance metrics and workloads for analyzing a system

- **System**
  - —collection of hardware, software and firmware under study
- **Evaluation techniques**
  - —measurement, simulation, analytical modeling
- **Metrics**
  - —criteria used to quantify system performance
    - e.g. system throughput, network bandwidth, response time
- **Workloads**
  - —scheduler: job mix
  - —network: packet mix
  - —database: query set

Measure and Analyze Appropriately

- Understand hardware and software monitoring capabilities and limitations
- Use proper statistical techniques to compare alternatives: which is best?
  - —measurements often have variability
  - —comparing the average is often not enough, especially with high variability

File size	Link A	Link B
50	0	1
1000	5	10
1200	7	3
1300	3	0

Design Effective Experiments

Quantify effects of different factors

- **Example**
  - —2 CPU scheduling algorithms
  - —4 workloads (e.g. database server, WWW server, compilation)
  - —three CPU types
- **Questions**
  - —how can one estimate the performance impact of each factor?
  - —how many experiments are needed?

Simulation

- Types of simulations
- Model verification and validation
- Random number generation
- Testing random number generators
  - —what is the period of a generator?
  - —is randomness global, or local as well?
  - —are bit subsets equally random?
- Random variate generation
  - —how to generate random numbers with a particular distribution

To compare two cache replacement algorithms

- —what type of simulation is appropriate?
- —how long should it be run?
- —how can the same accuracy be achieved with shorter run?

Queueing Theory

- **Example**
  - —In a network gateway, packets arrive at a mean rate of 125/s and the gateway takes 2ms to forward them.
- **Questions**
  - —what is the gateway utilization?
  - —what is the probability of n packets in the gateway?
  - —what is the mean number of packets in the gateway?
  - —what is the mean time a packet spends in the gateway?
  - —how big a buffer does the gateway need to keep losses < 10^{-6}?

Performance Evaluation as Art

- **Evaluation requires knowledge of system being modeled**
  - —understand potential interactions between system components
  - —understand what kind of model of system behavior is appropriate for parameter fitting
  - —which system is better?

System	Workload 1	Workload 2
A	20	10
B	10	20

  - —Earth Simulator vs. ASCI Q, which is better?

Common Mistakes and How to Avoid Them

No Goals

- A model must be developed with a purpose in mind
  - —there are no “general-purpose” models
- Metrics, model and workload depends on goals
  - —must understand problem to be solved to pick them appropriately
- Identifying goals may not be easy
  - —the key problem may not be readily apparent
  - —example
    - problem of choosing a timeout algorithm for packet retransmission later became one of how to adjust network load under packet loss

Biased Goals

- **Implicit or explicit bias in goals can slant selection of metrics and workloads**
  - **—biased goal:**
    - **show message passing parallel programming yields better performance than shared-memory programming**
  - **—biased workload choice:**
    - **statically-decomposed problems that require only coarse-grain interprocess communication**
- **Aim for objectivity: compare**
  - **—select appropriate workloads and metrics for fair comparison**

Common Mistakes - I

- **Unsystematic approach**
  - —arbitrary selection of system parameters, factors, metrics, workloads leads to irrelevant conclusions
- **Incorrect performance metrics**
  - —appropriate metrics depends on the services provided
  - —what metrics should be used to compare the following?
    - two disk drives
    - two transaction processing systems
    - two packet retransmission algorithms
- **Unrepresentative workload**
  - —example: HPL for evaluating parallel systems
  - —workload studied should represent actual system use

Common Mistakes - II

- **Wrong evaluation technique**
  - —three choices: measurement, simulation, analytical modeling
  - —understand the applicability of each and recognize when one is unnatural
- **Overlooking important parameters**
  - —useful analysis results are unlikely without any of them
- **Ignoring significant factors**
  - —study all parameters likely to be relevant to performance
  - —avoid studying parameters representing infeasible alternatives
- **Inappropriate experimental design**
  - —naïve designs change factors one at a time
  - —require extra experiments, miss interactions

Common Mistakes - III

- Inappropriate level of detail
  - —detailed model vs. high-level model
- No analysis
  - —measurement data without analysis is of limited value
- Erroneous analysis
  - —e.g. too short simulations, fitting inappropriate model
- No sensitivity analysis
  - —sensitivity to workload or system parameters is often ignored
- Ignoring errors in input
  - —if input parameters are uncertain or biased, so may be the results
- Improper treatment of outliers
  - —if not representative of system phenomenon, ignore
  - —if outliers are possible in the real system, consider them

Common Mistakes - IV

- Ignoring variability
  - —if variability is high, mean performance may be misleading
- Too complex analysis
  - —use the simplest technique appropriate for the job
  - —complex models are rarely applied outside academia
- Analysis without understanding
  - —modeling is not an end in itself; must draw relevant conclusions
- Improper presentation of results
  - —use words, pictures, and graphs to clearly convey results
- Ignoring social aspects
  - —know your audience: analysts and decision makers are different
- Omitting assumptions and limitations
  - —may lead user to apply result where it is not appropriate

A Systematic Approach to Performance Evaluation

Systematic Approach - I

- State goals and define system
  - —define boundaries of system under study: what can be ignored
- List services and outcomes
  - —service: network transmits packets from source to destination
  - —outcomes: packets delivered correctly, corrupted, lost
- Select metrics for quantitatively evaluating system behavior
  - —e.g. speed, accuracy, availability of service
- List system and workload parameters
  - —system parameters generally immutable, e.g. CPU speed
  - —workload parameter types
    - analytical: probability distribution
    - simulation: traces

Systematic Approach - II

- Select factors to study
  - —factors = parameters that vary
  - —factor values = levels
  - —select a short list likely to be important
  - —consider the implications of varying a factor before studying it!
- Select evaluation technique
  - —measurement, simulation, analytical modeling
  - —appropriate technique depends upon
    - – feasibility of measurement
    - – time and resources available, accuracy required
- Select workload: list of service requests
  - —analytical modeling: probability distributions of requests
  - —simulation: web access traces
  - —measurement: scripts or program executions

Systematic Approach - III

- **Design experiments for effectiveness**
  - —phase 1: identify important factors (many factors, few levels)
    - fractional factorial experimental designs are effective
  - —phase 2: fewer factors, more levels of significant factors
- **Analyze and interpret data**
  - —use statistical techniques to consider impact of variability
  - —results and conclusions are separate things!
- **Communicate results effectively (high level, no jargon)**
  - —graphs and charts help distill results but do not stand alone!

Selecting Evaluation Techniques and Metrics

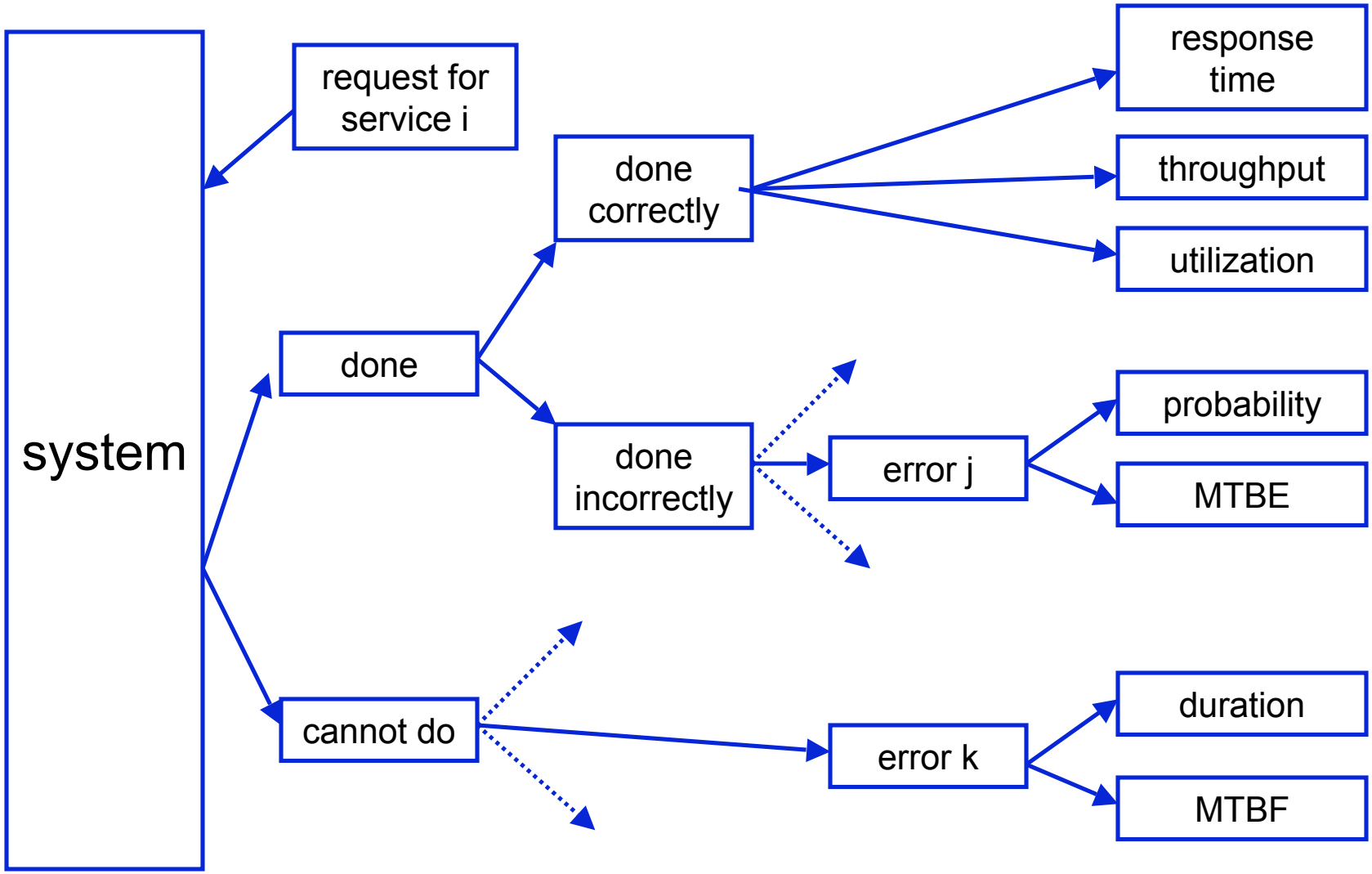
Selecting an Evaluation Technique

Criterion	Analytical Modeling	Simulation	Measurement
Stage	any	any	post-prototype
Time required	small	medium	varies
Tools	analysts	programs	instrumentation
Accuracy	low	moderate	varies
Trade-off evaluation	easy	moderate	difficult
Cost	low	medium	high
Saleability	low	medium	high

Rules of Thumb

- **Until validated, all evaluation results are suspect!**
  - —always validate one analysis modality with another
  - —measurements are as susceptible to errors as other techniques
    - experimental error, program bugs
  - —beware of counterintuitive results!
- **Combining evaluation techniques is useful**
  - —analytical model: find interesting range of parameters
  - —simulation: study performance within parameter range

Selecting Performance Metrics



Commonly Used Metrics

- Nominal capacity: maximum achievable under ideal conditions
  - networks: nominal capacity = bandwidth
- Throughput: requests / unit time
- Usable capacity: max throughput for given response time limit
- Efficiency: usable capacity / nominal capacity
- Utilization: fraction of time resource busy servicing requests
- Idle time
- Reliability: probability of error, MTBE
- Availability: fraction of time system servicing requests
- Mean uptime: MTBF

Types of Performance Metrics

- Higher better (HB), e.g. throughput
- Lower better (LB), e.g. response time
- Nominal is best (NB), e.g. utilization
  - —too high: bad response time; too low: resources underused

Setting Performance Requirements

(for a design or procurement)

- **Requirements should be SMART**
  - —specific (quantitative)
  - —measurable (can verify that system meets requirements)
  - —acceptable (high enough to be useful)
  - —realizable (low enough to be achievable)
  - —thorough (requirements should be specified for all outcomes)