Performance Evaluation

Chapter 1
Introduction

Overview

- Goals of Performance Evaluation
- Basic Notions: System and Model
- Quality of Service and Typical Performance Measures
- Main Performance Evaluation Techniques
- Pitfalls
- Performance Optimization
- Outline of a Performance Study
Goals

- General goals:
  - Determine certain performance measures for existing systems or for models of (existing or future) systems
  - Develop new analytical and methodological foundations, e.g. in queuing theory, simulation etc.
  - Find ways to apply theoretical approaches in creating and evaluating performance models
    - We will be mostly concerned with the first and third point

- Typical specific goals:
  - Bottleneck analysis and optimization
  - Comparison of alternative systems / protocols / algorithms
  - Capacity planning
  - Contract validation
  - Pure (academic) interest
  - Also: performance analysis is often a tool in investment decisions or mandated by other economic reasons

Making Goals Explicit and Precise

- The goal should be precisely specified, because a precise question:
  - Often carries half of the answer
  - Forces you to understand the system thoroughly
  - Allows you to select the right level of detail / abstraction
  - Allows you to select the right workload

- The methods, workloads, performance measures etc. should be relevant and objective; examples:
  - To determine the maximum network throughput it is appropriate to use a high load instead of a (typical) low load
  - To test your system under errors you have to force these errors

- The obtained performance results should clearly answer the question

- The limitations of the performance results should be clear
  - Consider for example web server workloads: performance results obtained for a workload composed of mostly static pages does not necessarily allow performance prediction under a highly dynamic workload (with lots of PHP, Perl or database accesses) – why???
Meta Goals

- Results should be communicated in a clear, concise and understandable manner to the persons setting the goals (often suits :o)
- The performance assessment should be fair and careful, e.g. when comparing OUR system to THEIRs
- The results should be complete, e.g. you should not restrict to a single workload favoring your system
- The results should be reproducible:
  - Often not easy, e.g. measurements in wireless systems

Basic Notions: System and Model

- Goal: clarify which types of systems / models we look at
- General notion of a system [KB71]:
  - A system denotes a collection of elements which have mutual relations and which act together to jointly solve some given task (its function).
  - This task could not be solved by any element on its own.
  - A system could be made out of materials (material system) or from notions, statements, theorems, etc. (ideal system)
  - The elements of a system can itself be systems (subsystems)
- Often we will use the term “system” somewhat sloppily
Features of a system for performance evaluation purposes

Elements of a System: Input (1)

- **Workload**: specifies the arrival of requests which the system is supposed to serve; examples:
  - Arrival of packets to a communication network
  - Arrival of processes to a computer system
  - Arrival of instructions to a processor
  - Arrival of read/write requests to a database

- Workload characteristics:
  - Request type (e.g. TCP packet vs. UDP packet vs. . . . )
  - Request size / service time / resource consumption (e.g. packet lengths)
  - Inter-arrival times of requests
  - (statistical) dependence between requests
Elements of a System: Input (2)

- **Configuration or parameters:** in general all inputs influencing the systems operation; examples:
  - Maximum # of retransmissions in ARQ schemes
  - Time slice length in a multitasking operating system
  not all of these can be (easily) controlled

- **Factors:** a subset of the parameters, which are purposely varied during a performance study to assess their influence

- **Error model:** specifies the types and frequencies of failures of system components or communication channels:
  - Persistent vs. transient errors:
    - Component failures are often persistent
    - Channel errors are often transient
  - System malfunctioning vs. malicious behavior caused by an adversary
  - Occurrence of “chain reactions” or “alarm storms”

Elements of a System: Others

- The system generates an output, some parts of which are presented to the user
- It can also have an internal state, which determines its operations together with the input
- There could be feedback: some parts of the output serve as input
- To obtain desired performance measures, the output or the observable system state may have to be processed further, by some “function” $f$
There are a number of classifications of systems [1]

**Static vs. dynamic systems:**
- In a static system the output depends only on the current input, but not on past inputs or the current time.
- A dynamic system might depend on older inputs (“memory”) or on the current time (a system needs internal state to have memory).

**Time-varying vs. time-invariant systems:**
- Time-invariant: the output might depend on the current and past inputs, but not on the current time.
- In a time-varying system this restriction is removed.

**Open systems vs. closed systems:**
- Open systems have an “outside world” which is not controllable and which might generate workloads, failures, or changes in configuration.
- In a closed system everything is under control.

**Stochastic systems vs. deterministic systems:**
- In a stochastic system at least one part of the input or internal state is a random variable / random process ⇒ outputs are also random.
- Almost all “real” systems are stochastic systems because of
  - “true” randomness, or
  - because the system is so complex / so susceptible to small parameter variations that predictions are hardly possible (in theory the roulette ball is predictable, but in practice roulette can be considered a random game).
Classification of Systems – State Based Systems (1)

- Definition 1 [CL99, p. 9]:
  The state of a system at time $t_0$ is the information required at $t_0$ such that the output $y(t)$ for all $t \geq t_0$ is uniquely determined from this information and from the inputs $u(t)$ ($t \geq t_0$).
- In computer systems / communication networks state is typically captured in variables.
- Continuous time systems (CTS) vs. discrete time systems (DTS):
  - In CTS state changes might happen at any time, even uncountable often within any finite time interval.
  - In DTS there is at most a countable number of state changes within any finite time interval.
    - At arbitrary times, or
    - Only at certain prescribed instants (e.g., equidistant).
  - We refer to discrete time systems also as discrete-event systems (DES).

Classification of Systems – State Based Systems (2)

- Continuous state systems vs. discrete state systems:
  - In a continuous state system the state space (= set of possible system states) is uncountable.
  - In a discrete-state the state space is finite or countably infinite.

  - Computer and communication systems are mostly viewed as dynamic and stochastic discrete state / discrete time systems.
Models of Systems

- According to [KB71] a model is an “object” used by an individual for its behavioral, structural or functional similarity to a given original object or system, in order to solve a given task or for a particular purpose.
- Models are formed because the original is not available or its manipulation is too complicated, a model is itself a system.
- A model always has a specific purpose for which it is built, and which determines its structure and representation.
- The models purpose also determines which of the aspects of the original object are considered and which are not.
Important Points about Models

- A model is made from a carefully chosen *model substrate*
- Model substrates could be material or immaterial / (formal) languages (e.g. mathematical formulae, programming / specification languages)
- A model is a result of a mapping process from the given original to the model substrate
- A model does not necessarily have any structural similarity to the given original, it does not need to consume the same inputs nor generate the same outputs; still it should be appropriate
  - An astrophysical simulation model of a black hole has (fortunately) not the properties of a real one

Types of Models: Physical vs. Mathematical

- Physical models:
  - Scale representation of system
  - Example: build a little airplane and a wind tunnel to study aerodynamics with experiments
- Mathematical models:
  - Represent system with appropriate mathematical or logical formalism
  - An airplane is described by the laws of aerodynamics
  - Manipulate this representation, e.g., introduce external stimuli
  - Move the rudders of the airplane = use different laws to represent it
  - Try to deduce how the real system would react (provided model is valid)
  - Would such an airplane turn left or right?
- We will only consider mathematical models!
Types of Models: Static vs. Dynamic

- **Static model:**
  - A model where only a certain, fixed state of a system is considered, state changes are not taken into account
  - Evidently appropriate for static systems (one without state changes)
  - Sometimes appropriate even for dynamic systems, if only system properties in certain, fixed states are of interest

- **Dynamic model:**
  - Reflect the system’s state changes as it evolves over time
  - How to handle continuous or discrete systems?

---

Types of Models: Continuous vs. Discrete (1)

- **(Time-)Continuous models:**
  - A continuous model describes the system such that the state variables are a continuous function of time
  - Typical description: differential equation(s), describing relationships between interdependence of the rate of change of certain state variables with each other and with time

- **(Time-)Discrete models:**
  - Change of state only happens at discrete, well separated instances of time (the set of points in time where the state changes is at most countably infinite)
  - In between such times, all state variables maintain their values, the state does not change
  - At such points in times, events of the model occur, i.e., the state of the systems can only change when an event occurs (but need not necessarily change at every event)
Types of Models: Continuos vs. Discrete (2)

- Remark:
  - The type of system and does not automatically determine the type of an appropriate model of the system

- For continuous systems, continuous models are not necessarily used:
  - E.g., voice data is a continuous system, but is – for purposes of transmission – modeled as a discrete system (quantization, sampling)

- For discrete systems, discrete models are not necessarily used:
  - E.g., traffic flow on a highway – with discrete events of cars entering and leaving the highway – can be modeled continuously if only the behavior of large numbers is interesting

- Choice depends on intentions, objectives, feasibility

Types of Models: Deterministic vs. Stochastic

- A model where the evolution of state is completely described such that it only depends on the initial state is a **deterministic model**:
  - E.g., a set of differential equations describing concentration of different substances in a chemical reaction

- A model where the evolution of state depends on random events (random in both time of occurrence or nature) is a **stochastic model**:
  - E.g., model of a highway where the times when cars enter the highways are described by a random variable
  - Output/results for such models do not only depend on initial state, but also on the values of random variables -> no fixed or single result for such models

- Again note difference between system and its model:
  - Sometimes, stochastic systems are modeled deterministically
  - Example: chemical processes are actually random by their very nature (quantum mechanics), yet they are usually modeled deterministically (appropriate because of the large number of particles involved)
Model Criteria

- Good models should be:
  - Appropriate representation of the system (depending on purpose of investigation)
  - As simple as possible without impeding appropriateness
  - Reusable for similar systems / models, as a part in other models
  - Parameterizable
  - Amenable to appropriate investigation method:
    - In acceptable time, with acceptable effort, with desired accuracy
    - Method also depends on desired results

Model Overview

```
Model
  - Physical
  - Mathematical
    - Static
    - Dynamic
      - Continuous
      - Discrete
        - Deterministic
        - Stochastic
```

- Deterministic
- Stochastic
- Deterministic
- Stochastic
Quality of Service and Typical Performance Measures

- Ultimately, users are interested in their applications to run with “satisfactorily” or even “good” performance.
- “Good performance” is subjective and application-dependent:
  - Frame rate / level of detail / resolution of an online-game,
  - Sound / speech quality,
  - Network bandwidth and latency, etc.
- **Objective measures** (we only deal with these):
  - Can be measured
  - Typically expressed as numerical values
  - Other persons reproducing the experiment would obtain (nearly) the same values
- **Subjective measures**:
  - Are influenced by individual judging, e.g. speech quality, video quality
  - Can sometimes be “objectified”; example: the ITU has specified a method for judging the output of audio codecs, the model aggregates the results of numerous listening tests

Types of Service Guarantees (1)

- A service is provided by a service provider upon receiving service requests and under certain conditions.
- For example, an ISP might grant Internet access when you:
  - Pay your monthly fee
  - Behave according to specified policies (do not send spam mails, do not offend other netizens, etc.)
  - Obey certain technical standards (right modem, dial-in numbers, etc.)
  - And no serious breakdown of network infrastructure happens
Types of Service Guarantees (2)

Given that these conditions are fulfilled, the provider might give one of the following promises:

- Guaranteed quality of service (QoS):
  - Service provider claims that a certain service level will be provided under any circumstances
  - Anything worse is a contract violation
  - Sometimes additional requirements may be posed, e.g.: to guarantee a certain end-to-end delay in a network, you must not exceed some given sending rate; example:
    - An ISDN B-channel guarantees 64 kbit/sec; anything in excess is dropped

Types of Service Guarantees (3)

- Expected QoS or statistical QoS:
  - Service provider promises “more or less” the service
  - Problem: how to specify this exactly? examples:
    - At most x out of N consecutive data packets will get lost
    - At most x / N * 100% of data packets will get lost
    - The probability that a packet is lost is x / N
  - The first two examples make the difference between short term service and long term service guarantees:
    - 1st case: “at most 1 packet out of 100”
    - 2nd case: “at most 10.000 packets out of 1.000.000”

- Best effort service:
  - “I will do my very best, but I promise nothing”
Typical Performance Measures

- **System-oriented vs. application-oriented measures:**
  - System-oriented measures are independent of specific applications
  - Application-oriented measures might depend in complex ways on system-oriented measures

- **Example – video conferencing over Internet:**
  - Application-oriented measures: frame rate, resolution, color depth, SNR, absence of distortions, turnaround times, lip synchronization
  - System-/network-oriented measures: throughput, delay, jitter, losses, blocking probabilities, . . .

- Often there is no simple way to predict application measures from system-oriented measures:
  - A video conferencing tool might be aware of packet losses and can provide mechanisms to conceal them, providing the user with slightly degraded video quality instead of dropouts or black boxes

Measures for Communication Networks

- **Delay:**
  - Processing and queueing delays in network elements
  - Medium access delay in MAC protocols

- **Jitter:** delay variation

- **Throughput:** number of requests / packets which go through the network per unit time

- **Goodput:** similar to throughput, but without overhead (e.g. control packets, retransmissions)

- **Loss rate:** fraction of packets which are lost or erroneous

- **Utilization:** fraction of time a communications link / resource is busy

- **Blocking probability:** probability of getting no (connection-oriented) service; sometimes you get no line when you pick up the phone

- **Dropping probability:** in cellular systems; probability that an ongoing call gets lost upon handover
Measures for Different Types of Computing Systems

- **Desktop systems** (single user):
  - Response times, graphics performance (e.g. level of detail)
- **Server systems** (multiple users):
  - Throughput (processor, I/O bandwidth)
  - Reliability (MTBF, mean time between failures)
  - Availability (fraction of downtime per year / unit time)
  - Utilization
- **Embedded systems**:
  - Energy consumption
  - Memory consumption
  - Realtime performance:
    - Number of deadline misses
    - Jitter for periodic tasks
    - Interrupt latencies
  - System utilization

Performance Figures

- For scalar-valued measures (e.g. packet round-trip delay in a network) we might be interested in:
  - Mean
  - Variance, higher moments
  - Minimal and maximal values, quantiles / percentiles
  - The whole distribution
  - Correlation between subsequent samples
Main Performance Evaluation Techniques

- When the system under study already exists and is accessible, we can make measurements.
- When the system does not exist or is too clumsy to deal with (e.g. the system is the whole Internet), a performance model must be developed:
  - Analytical models use mathematical concepts and notations
  - Simulation models are computer programs
    both kinds of performance models restrict to the most important aspects and leave out many details
- The choice of a technique depends on:
  - The type of system to be investigated
  - Its availability
  - The familiarity of the modeler with the techniques
  - Time and resource constraints
  - Desired accuracy and “saleability” of results

Measurements

- The system under test (SUT) can be a single computer or a collection of networked computers / network elements
- The system is instrumented with probes (HW, SW); this is not always possible, e.g. normal people cannot instrument an MS Windows kernel
- The system is subjected to a well-specified workload
- The probes capture values and store them in a buffer
- A monitor system collects the values, computes performance figures and presents them to the user
Interpretation of Measurement Results

- ... is often tricky, since many factors influence the results
  - Example: when measuring the end-to-end delay of voice-over-IP packets between two Internet hosts, several factors can influence the delay:
    - Speech coder latency (includes A/D conversion, compression)
    - Operating system / networking stack at the sender
    - Network in between (propagation delays, bandwidth, queueing delays in routers, losses)
    - Operating system and networking stack at the receiver
    - Size of playout buffer (required to compensate jitter)
    - Speech decoder latency
  - How to figure out the individual contributions of the components?

Pros & Cons of Measurements

- Advantages:
  - Saleability: you can always claim that your numbers are “real” and not based on some “suspicious”, “arbitrary” or “unjustified” model
  - You do not need to find “reasonable parameters” for intermediate elements as in model-based studies (e.g.: what could be a “reasonable” queuing delay for a router in the VoIP example?)

- Disadvantages:
  - Sometimes (often :o): hard to interpret, unreproducible, substantial time/effort needed to set up
  - You have to consider all details; in model-based techniques you can neglect some
  - Workload selection can be tricky (how to find “representative” workloads?)
  - Amount of “material” needed to perform experiments of significant size (e.g. mobile communications: handover studies might need a very high number of devices moving + significant amount of infrastructure components)
Analytical Modeling

- Uses mathematical notions and models describing certain aspects of a system
- For modeling of computer systems and communication networks often probabilistic models are used:
  - Task arrival times to a computer are random
  - User inputs are random
  - Packet arrival times to a network are random
  - Errors on communication links are random
  - ...
- In this course, we are mostly concerned with stochastic models of discrete-state systems

Pros & Cons of Analytical Modeling

- **Disadvantages:**
  - Many systems are way too complex for analytical modeling, requiring simplifications / approximations
  - Solid background in mathematics / probability theory needed (can also be interpreted as an advantage :o)
- **Advantages:**
  - Thorough understanding of the system required (hey: why is this not counted as a disadvantage :o)
  - Can often be quickly set up and evaluated
  - Even as approximations, analytical models often provide qualitative insights into systems
Simulation Modeling

- A simulation model:
  - is a computer program, written in a general-purpose or specific simulation language
  - implements the most important aspects of the system under study, often in a simplified manner
  - allows for a greater level of detail than analytical modeling

- Random input data produces random output data ⇒ proper statistical evaluation needed
  - Oops, solid mathematical background also needed... :o)

Tradeoffs in Simulation Modeling

- Accuracy of results often specified in terms of confidence intervals
- High statistical accuracy (small intervals) needs long simulation times
- Higher variability of output leads to longer runtimes to reach a given accuracy target

→ Simulation runtimes are an issue!
### Comparison of Techniques

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Analytical M.</th>
<th>Simulation M.</th>
<th>Measurements</th>
</tr>
</thead>
<tbody>
<tr>
<td>stage</td>
<td>any</td>
<td>any</td>
<td>needs existing system</td>
</tr>
<tr>
<td>time required</td>
<td>small</td>
<td>medium</td>
<td>varies</td>
</tr>
<tr>
<td>tools</td>
<td>human</td>
<td>computer</td>
<td>instrumentation / probes</td>
</tr>
<tr>
<td>accuracy</td>
<td>low</td>
<td>moderate</td>
<td>varies</td>
</tr>
<tr>
<td>trade-off evaluation</td>
<td>easy</td>
<td>moderate</td>
<td>difficult</td>
</tr>
<tr>
<td>cost</td>
<td>small</td>
<td>medium</td>
<td>high</td>
</tr>
<tr>
<td>saleability</td>
<td>low</td>
<td>medium</td>
<td>high</td>
</tr>
</tbody>
</table>

### Common Pitfalls

- In [Jain91, Chapter 2] a list of common mistakes of performance evaluation studies is compiled, which we paraphrase here freely, since it is full of wisdom:
  - Have clearly specified goals; no model or measurement setup can answer all questions
  - Have unbiased goals, i.e. have no preconception on “desired” results or “results to prove”
  - Be systematic; find all relevant parameters / factors and their relevant values
  - Understand the system first
  - Use the right performance metrics
  - Use the right workload
  - Use the right performance evaluation technique (not always your favorite)
  - Find a good experimental design, which reduces the number of simulations/measurements needed to produce meaningful results
More Pitfalls

- Find the right level of detail
- Use the right method for data analysis (example: time series methods often assume equidistant sampling times)
- Avoid errors in analysis (e.g. too short simulation times, models often have transients)
- Find the sensitivity of results against parameter / factor changes
- Find the right treatment of outliers (popular approach: remove all samples having more than $x \times \sigma$ distance to mean; $\sigma$ is standard dev.)
- Do not expect your results to be valid far in the future
- Do not ignore variability of results
- Obey Occams razor (and Einsteins corollary):
  - Occam: Try to explain things as simply as possible
  - Einstein: ... but not simpler
- Present your results properly
- State clearly your assumptions and omissions

A Case Study in Performance Comparison

- (From: [HP03, Section 1.5]) three different computer systems A, B, C are compared for the execution times of different programs $P_1$ and $P_2$
- Measurements give the following results:

<table>
<thead>
<tr>
<th></th>
<th>Computer A</th>
<th>Computer B</th>
<th>Computer C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program $P_1$ (sec)</td>
<td>1</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>Program $P_2$ (sec)</td>
<td>1000</td>
<td>100</td>
<td>20</td>
</tr>
<tr>
<td>Total time (sec)</td>
<td>1001</td>
<td>110</td>
<td>40</td>
</tr>
</tbody>
</table>
Which computer is “the best”?
Standard answer: it depends!!
We look at different measures:

- If the measure is the total execution time, then C is best
- If in a “real workload” program P₁ runs 500 times, and program P₂ runs 5 times, the weighted total time is appropriate:
  - Total time for system A = 500 \times 1 \text{ sec} + 5 \times 1000 \text{ sec} = 5500 \text{ sec}
  - Total time for system B = 500 \times 10 \text{ sec} + 5 \times 100 \text{ sec} = 5500 \text{ sec}
  - Total time for system C = 500 \times 20 \text{ sec} + 5 \times 20 \text{ sec} = 10100 \text{ sec}

We could “normalize” the results by selecting one system as a “baseline system”, which always takes time 1; the other results are related to the baseline system by computing the ratios of run times

It does not need a big imagination to see that by proper selection of the baseline system and by taking “advantage” of using ratios:

- System A could be the winner
- System B could be the winner
- System C could be the winner

[Jain91] devotes a whole chapter to “ratio games”
General Rules for Performance Optimization (1)

- Focus optimization on the most common / most frequent case:
  - For computer programs: use profiling information
  - The effect of your optimization depends on two factors:
    - How much have you improved the piece of code?
    - How often is the improved piece of code called?
  - With Amdahl's Law [HP03, Section 1.6] you get an estimate of the actual improvement

- Exploit locality:
  - For example code locality: after reading an instruction from memory cell \(x\) with high probability the next instruction will be read from cell \(x + 1\), so you could fetch \(x + 1\) ahead while processing \(x\).
  - Similar: read ahead from files

- Exploit correlation / predictability:
  - Example: inter-frame coding of video sequences to reduce bandwidth
  - Example: postponing retransmissions over wireless channels

General Rules for Performance Optimization (2)

- Take advantage of parallelism:
  - On the architectural level (multiprocessor machines, multiple web servers)
  - On the algorithmic / protocol level
  - On the instruction level / level of digital circuitry

(source of this list: [HP03])
General Outline of a Performance Study (1)

- Define the goals and specify the system under study as completely as possible, including the system boundaries
- List the services a system provides and their outcomes; example:
  - In the network layer of a communication network the offered service is packet delivery
  - Outcomes for best effort service: packet is delivered or not
- Select the desired performance metrics and the desired statistical accuracy (confidence interval width)
- Identify all inputs (workload, parameters, errors) that may affect performance of the system.
- Select the factors and their levels
- Decide on the performance evaluation technique(s) to use

General Outline of a Performance Study (2)

- Define the workload and the error model properly:
  - For analytical models: as probability distributions / stochastic processes
  - For simulations: either traces or probability distributions / stochastic processes
  - For measurements: many methods
- Experimental design:
  - Usually the factor space spanned by all factor/level combinations is too large
  - This means you have to work on a subset of the whole space
  - There are some techniques which allow to choose “meaningful” subsets
- Develop the experiment setup, the simulation or analytical model
- Run the experiments or the simulations:
  - Measurements: may be nasty due to “hard to control” external influences
  - Analytical models: compute the results from the derived equations
  - Simulations: this can take lllllllloooooooooooonnnnnnngggg time
Analyze and interpret data; this includes not only the selected performance measures, but also their statistical properties like:

- Significance,
- Confidence intervals
- etc.

Interpretation of data means to draw conclusions from them.

Be warned:
- This is no waterfall model, you often have to go back some steps

(sources of this list: [Jain91, Section 2.2]

References


