Performance Measurement and Workload Models

Daniel Zappala

CS 360 Internet Programming
Brigham Young University

Performance Measurement

Motivation: Content Creators

- goal: maximize profit
- how do people use my web site?
    - what led people to buy this book? recommendation list? ad on another site?
    - what led people to *not* buy this book? negative reviews? ordering process too complicated?
- how many people use my web site?
    - ad revenue
    - where do they come from?
    - where are *not* coming from?

Motivation: Web Hosting and ISPs

- goal: maximize profit and performance
- accounting
  - bill people for the bandwidth they use
- system performance
  - replicate content to balance load
  - evaluation of a server to find and examine bottlenecks
  - effect of server parameters, e.g. number of active threads
  - measure the effectiveness of caching: number of hits, bandwidth saved

Motivation: Network Researchers

- goal: understand the network and how well it is working
- understanding
  - determine how many hosts have deployed a new protocol or feature (e.g. how many caches use Etag for validation?)
  - measure and model Internet traffic
- protocol development: improve performance
  - led to HTTP/1.1 persistent connections
  - major role in determining TCP performance
  - evaluate cache replacement, validation, and prefetching algorithms

Types of Measurement

Passive Measurement

- examine log files
- place a monitor "inside" the network
  - promiscuous mode on Ethernet means you get all packets
  - modify host to log all IP packets to a very large disk
- advantages
  - can capture all HTTP and TCP behavior
  - can examine a detailed timeline of all events
  - non-invasive for the web server and clients
- limitations
  - hard to capture all packets at line speed
  - limited to examining what happened, not what *may* happen

Active Measurement

- generate user requests on your own to examine web server and user performance
- locating user agents
  - wide variety of connection speeds
  - wide variety of network locations
  - what distributions?
- generating requests
  - which web pages?
  - how many requests per client?
  - use distributions collected from past experiments
- collecting data
  - DNS time
  - TCP connection time
  - HTTP request time

Simulation

- model a system and examine its behavior in a simulator
- requires modeling
  - the Internet (connectivity, bandwidth, delay)
  - Internet traffic
- complexity depends on level of simulation
  - packet-level simulation: treat each packet as an event, which must be handled at each router and host in the network
  - session-level simulation: treat each file transfer as an event, which must be handled only at the hosts in the network

Simulation vs Measurement

- advantages of simulation
  - network can be tightly controlled delay, bandwidth, competing traffic
  - can often simulate at a larger scale than a measurement study
  - experiments in a simulated network run in simulated time (much faster than real time)
  - can often vary more factors for a more complete study
- advantages of measurement
  - you learn different things when you actually have to build it
  - more convincing results: it works in the "real world"
  - can use real users and real traffic

Deciding Between Simulation and Measurement

- can you implement your solution?
  - time allocated to design, spec, testing, debugging
  - is there an existing framework or library that would make your job easier?
- can you deploy your solution?
  - do you have access to enough computers or friends?
  - what operating system and libraries will you use?
  - can you use PlanetLab?
- how many factors do you want to study?

Methodology

Experimental Methodology

- important components of an experimental methodology for network performance measurement
  1. topology
  2. workload
  3. data collection
  4. metrics

Topology

- what does the network look like?
  - LAN
  - campus
  - PlanetLab
  - Internet
- simulated networks must be generated
  - how many nodes?
  - how many links?
  - link and node characteristics - bandwidth, delay, queue size
  - network characteristics - organizations, ISPs

Workload

- what load will you put on your solution and therefore on the network?
- *example: web server*
    - how often do clients make a request to the web server?
        - constant arrival rate
        - exponential arrival rate
    - how many requests does a client make and what is the time between requests?
    - what size are the objects that are available on the web server?
    - what is the popularity of each of the objects?
    - what is the available bandwidth to each client?

Data Collection

- collect a log of important events in your system
- **important: collect data, not metrics**
  - if you collect metrics, you will invariably find that you need to change a metric or add another one and now you need to re-run all of your experiments
  - collecting data takes more room but if you generalize it then you can compute any metrics you want later on
- *example: web server*
  - thread time threadID start
  - thread time threadID stop
  - queue time size

Metrics

- how do you know whether your solution is solving the problem?
- how do you know it is doing it better than other solutions?
- *hint*: look at previous solutions and see what metrics they use
- *example: web server*
  - response time
  - client bandwidth
  - percentage of time each thread is busy
  - queue size over time
  - average, standard deviation, percentiles, distributions, etc.
- note: knowing whether a difference is significant requires a good statistics background

Workload

Two Approaches

- trace-driven workload
  - use a log of actual requests to a server
  - reproduces an actual workload
  - avoids statistical analysis
  - no flexibility
  - may not be representative of other situations
  - mingles load with performance (the rate at which a user can make requests depends on how fast it can download a page)
  - hard to collect, requires extensive storage
- stress testing
  - send requests as fast as possible to induce high load
  - can quickly identify performance bottlenecks
  - not realistic

Synthetic Workloads

- characterize the load placed on a system using a mathematical/statistical model
- want it to be repeatable, realistic, parameterized
- advantages
  - derived from a model that can be analyzed, validated, and compared to other models
  - can capture general properties of real traffic without being limited to a particular trace
  - can explore performance in a controlled setting by adjusting the parameters of the model

Probability Distributions

Modeling Workload

- *example: web server log - how can we characterize the size of a response message?*
- mean can be skewed by high variance
  - response messages: 4100, 4700, 4200, 20,000, 4000 bytes
  - mean size = 7400 bytes
- median doesn't represent variance
  - median size = 4200 bytes
  - response messages: 4100, 4700, 4200, 4800, 4000 bytes has same median
- variance, standard deviation provide more information but they are also relatively simple
- need probability distributions to provide a more complete picture

Probability Density Function (pdf)

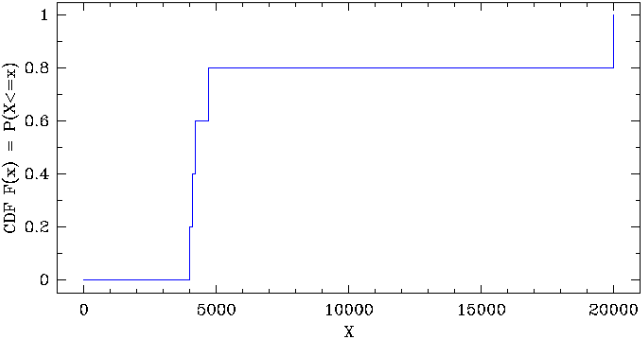
- describes the likelihood for a continuous random variable to occur over an interval
- defined by a function $f(x)$ such that
  - $f(x) \geq 0$ for all values of x
  - the total area under the graph is 1

$$\int_{-\infty}^{\infty} f(x)\, dx = 1 \tag{1}$$

- probability of the interval $(a, b]$ is given by $\int_a^b f(x)dx$ for any two numbers a and b

Cumulative Distribution Function (CDF)

- describes the likelihood that a continuous random variable will have a value less than or equal to x
- $F(x) = P(X \leq x)$

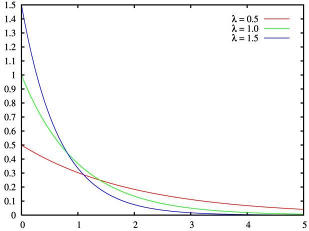


Common Distributions

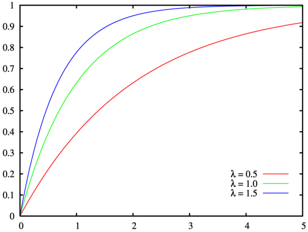
Exponential Distribution

- $F(x) = e^{-\lambda x}$
- frequently used to model the time interval between successive random events that happen at a constant average rate
- mean $\mathbf{E}[X] = \frac{1}{\lambda}$

pdf



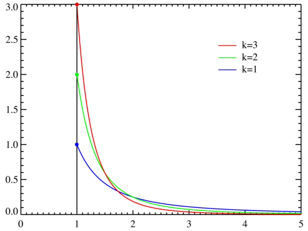
cdf



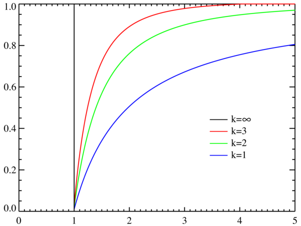
Pareto Distribution

- frequently used to model resource sizes
  - Italian economist Vilfredo Pareto (1848 - 1923)
  - 1906 observation: *20% of the population own 80% of the property*
- $P(X > x) = (\frac{x}{b})^{-a}, x \geq b > 0,\ b =$ minimum possible value
  - also called a "heavy-tailed" distribution
  - Wikipedia uses $a = k, b = 1$ in graphs below

pdf



cdf



Generating Random Variates

- want to be able to generate a random workload that follows a particular distribution
  - *generate the arrival of the next web request using an exponential distribution*
- exponential function
  - $x = -\frac{1}{\lambda} ln(1 - U)$
  - U is a uniform random variable [0,1]
  - choose a random number between 0 and 1, and calculate the corresponding value of x
- Pareto
  - $x = \frac{b}{(1-U)^{\frac{1}{a}}}$

Discrete Probability

Discrete Probability

- probability mass function (pmf)
  - probability that a discrete random variable is exactly equal to some value
  - rank resources from highest to lowest probability and plot probability for each rank
- typically use Zipf's Law to model resource popularity

Zipf's Law

- in a corpus of natural language, the frequency of a word is inversely proportional to its rank
    - Harvard linguist George Kingsley Zipf
    - *example: in the Brown Corpus, "the" accounts for 7% of all words, "of" accounts for about 3.5% of all words, etc*
- $P(r) = kr^{-1}$, where r is the rank of an object and k is constant such that all probabilities sum to 1
    - more generally, $P(r) = kr^{-c}$ for some constant c
    - heavy-tailed
- similar to Pareto
    - Zipf-like distributions can be viewed as Pareto distributions if you exchange some variables

Modeling Workload

Determining the Right Distribution

1. determine workload parameters
2. measure some real traffic
3. analyze measurement data to construct a statistical model of the workload
4. validate the model against additional real workloads

Generating a Web Workload

1. generate session (user) arrivals using an exponential distribution
2. for each session, generate the number of pages and the number of objects per page using Pareto distributions
3. for object size, use a Pareto distribution
4. for object popularity use a Zipf distribution