Experimental Approaches in Computer Science

Dror Feitelson
Hebrew University

Lecture 7 – Observations about Workloads
In algorithm analysis, performance depends on the input

In systems analysis, the input is the workload
The main requirement from workloads is that they be representative

- Lead to *exactly the same* performance evaluation results as will occur with real production workloads
  - Include all and only the important features
  - Need iterative evaluations to find what is important
- Or lead to *qualitatively similar* performance evaluation results
  - Reliably conclude that approach A is better than B
- Or at least exhibit the same *general behavior*
  - Include known features because they might be important
Example: packing parallel jobs for execution depends on the distribution of sizes. A uniform distribution suffers the worst fragmentation.
Sources of workload data

- **Active instrumentation**
  - Network sniffers to record packets
  - Instrument an I/O library to record operations
  - Collect data from architecture counters

- **Use available data**
  - Many systems collect data for accounting
  - Web server access logs
  - Parallel Workloads Archive
    www.cs.huji.ac.il/labs/parallel/workload/
Workload Statistics
• Typical way to characterize or model a workload is using statistics
• Distributions of workload attributes
• Correlations among workload attributes
• All this is based on experimental observations
Distributions may be modal

- File sizes
- Parallel job sizes
- Network packet sizes
Distributions may be heavy tailed

- File sizes
- Process runtimes
- Web page popularity

(more on this later)
Arrival processes tend to be bursty
• Not well-modeled by a Poisson process
• Do not average out when aggregated
• Fluctuations in load at many different time scales

(more on this later)
Many workloads tend to display locality

- Not well-modeled by a random sampling from a distribution

- Significant short-range correlations
  - Repetitions of the same activity
  - Repetitions of the same sequences

- Adaptation and evolution over longer ranges
  (more on this later)
Data Cleaning
• Workload data may be multiclass
  – A mixture of different workloads
• We may be interested in only part of them
  – Real user work as opposed to system administrator activity
  – User applications as opposed to the OS
• Especially if one class is actually junk
  – Errors in tabulating the data
  – Unique and unrepresentative activity
• Undesired data should be filtered out
Example: weekends and holidays are different
Example: process runtime data including multiple processes running `ps` as a result of a bug in an OS exercise
Example: the Welchia worm caused a change in Internet traffic composition that lasted 4 months.
Example: half of NASA Ames iPSC workload was system administrators running pwd on one node to verify that the system was responsive.
Example: SDSC Paragon has a suspicious peak of activity at 3:30 AM (probably daily cleanup)
Need to set the resolution right to see this.
Example: workload may include flurries of intense activity by specific users
Flurries affect distributions of workload attributes
Different flurries cause different effects

- Graph 1: Cumulative % of jobs vs. job size
  - Red dashed: all jobs '95
  - Green dashed: w/o flurries '95
  - Solid red: all jobs '96
  - Green solid: w/o flurries '96

- Graph 2: Cumulative % of jobs vs. memory per node [MB]

- Graph 3: Cumulative % of jobs vs. job runtime [s]

- Graph 4: Cumulative % of jobs vs. interarrival time [s]
Example: robot activity has different characteristics than humans

450 root processes at 4:15 AM

~200 processes every 10 minutes by user 1301
Example: impossibly long sessions created by staff that leave windows connected to a server open for several days.
Heavy Tails
• Distributions of workload attributes are typically positive
  – No negative file sizes, runtimes, etc.
• There are typically many small items and few large ones
• The large ones can be very large
  – And therefore important in terms of resource usage
• This is the tail of the distribution
  – Technically, the "right" tail
The large items can be **so large** that they dominate the whole distribution.

Consider the following discrete distribution:

- 2 with probability of 1/2
- 4 with probability of 1/4
- 8 with probability of 1/8
- 16 with probability of 1/16

and so on

...The mean of this distribution is \( \infty \)
If we look at the running average of samples from a Pareto distribution, it grows in jumps whenever a large sample is seen.
Perhaps the most important attribute of heavy-tail distributions is mass-count disparity: most of the items are small, but most of the mass is concentrated in a few items

- Most processes are short, but most CPU seconds are used by long processes
- Most files are small, but most disk space is used to store large files
- Most files on a web server are seldom requested, while most requests target a small subset of the files
Mass-count disparity can be quantified by the joint ratio: here 11% of the files account for 89% of the disk space, and 89% of files are only 11% of space.

Generalization of the Pareto principle (the 20/80 rule)
Also quantified by the 0-50 rule: 50% of the items together are practically 0 of the mass, and 50% of the mass comes from essentially 0 items.
The formal definition of a heavy tail is that the survival function decay according to a power law

$$\bar{F}(x) = Pr(X > x) = x^{-\alpha}$$

By taking the log from both sides, we get

$$\ln(\bar{F}(x)) = \ln(x^{-\alpha}) = -\alpha \ln(x)$$

This serves both to identify heavy tails and to assess the tail index $\alpha$