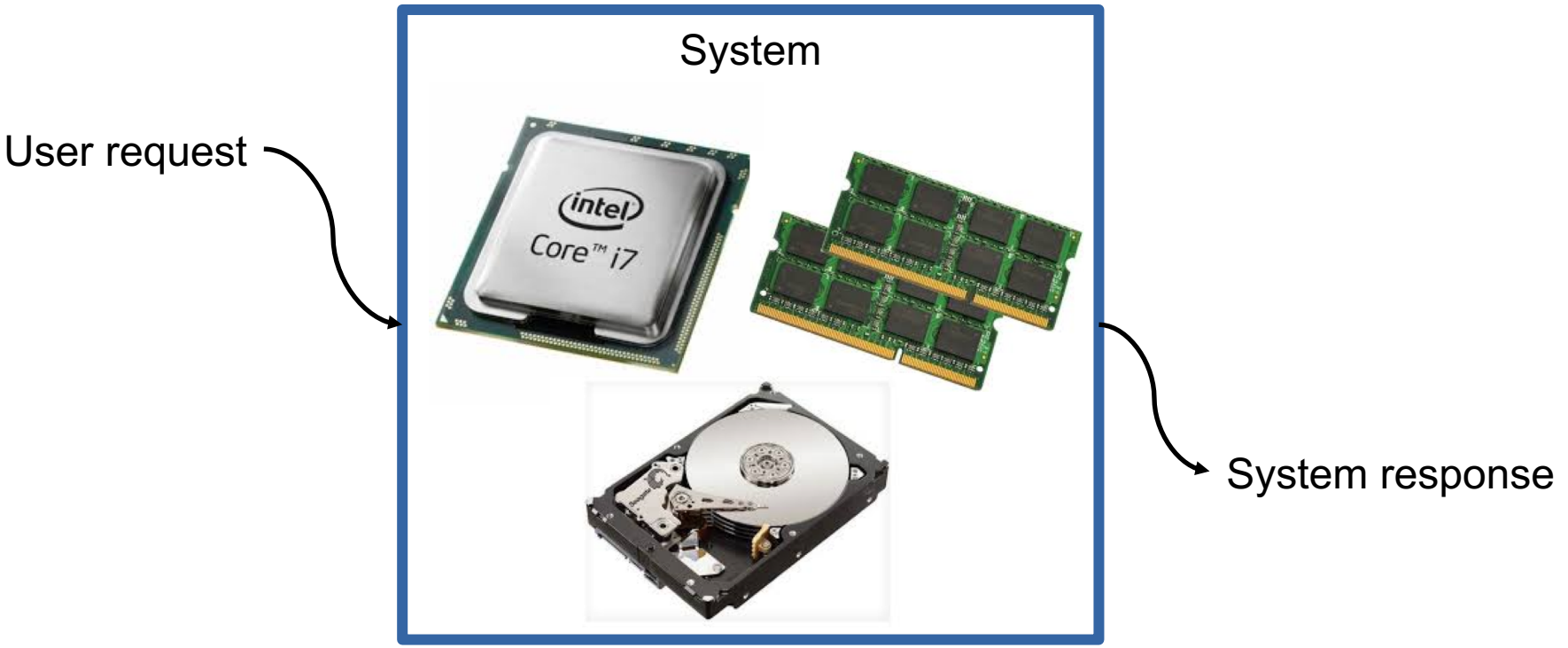
System Deployment & Benchmarking

Context

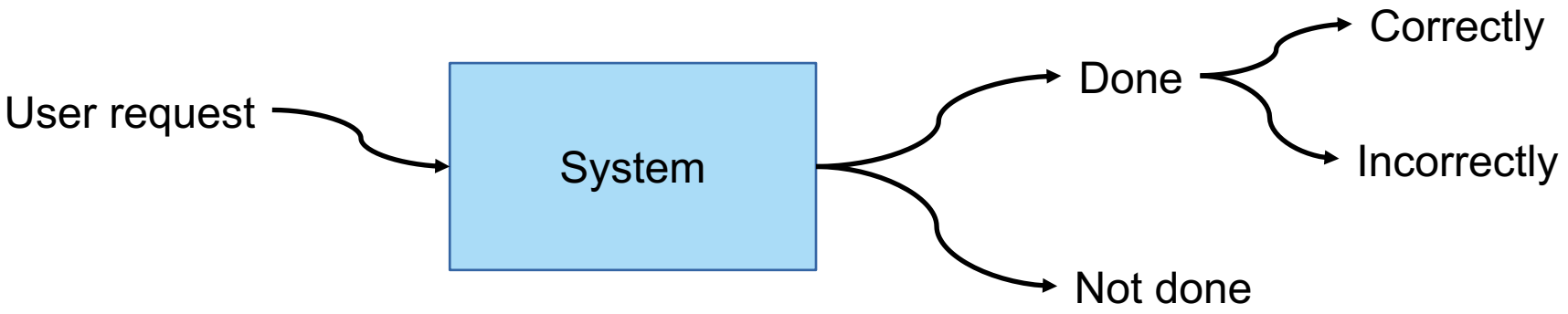
- What is performance?
- How to measure performance?

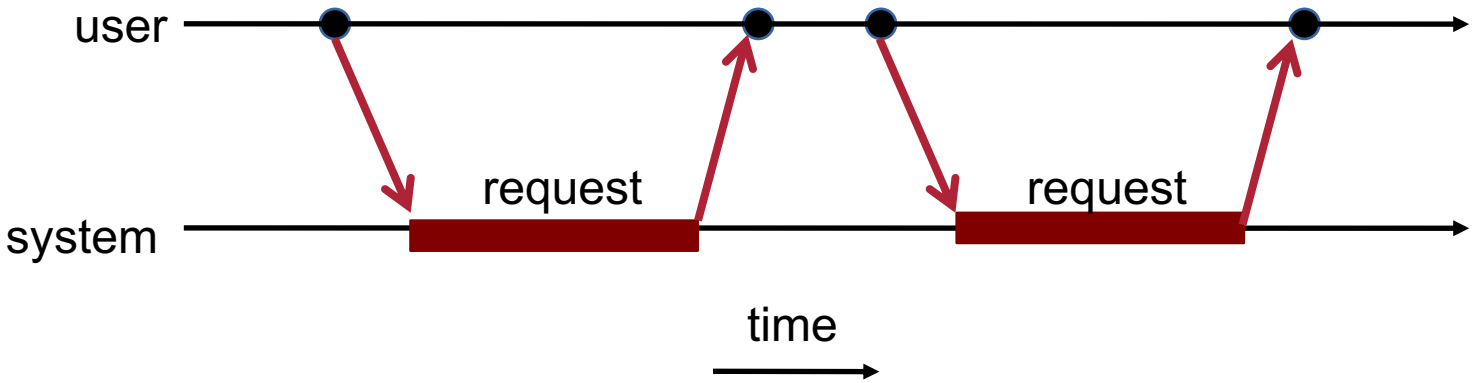
Computer systems

- Performing useful work for users
- Using resources with limited capacity:
  - Physical: CPU, memory, disk, network,...
  - Logical: locks, pools, ...



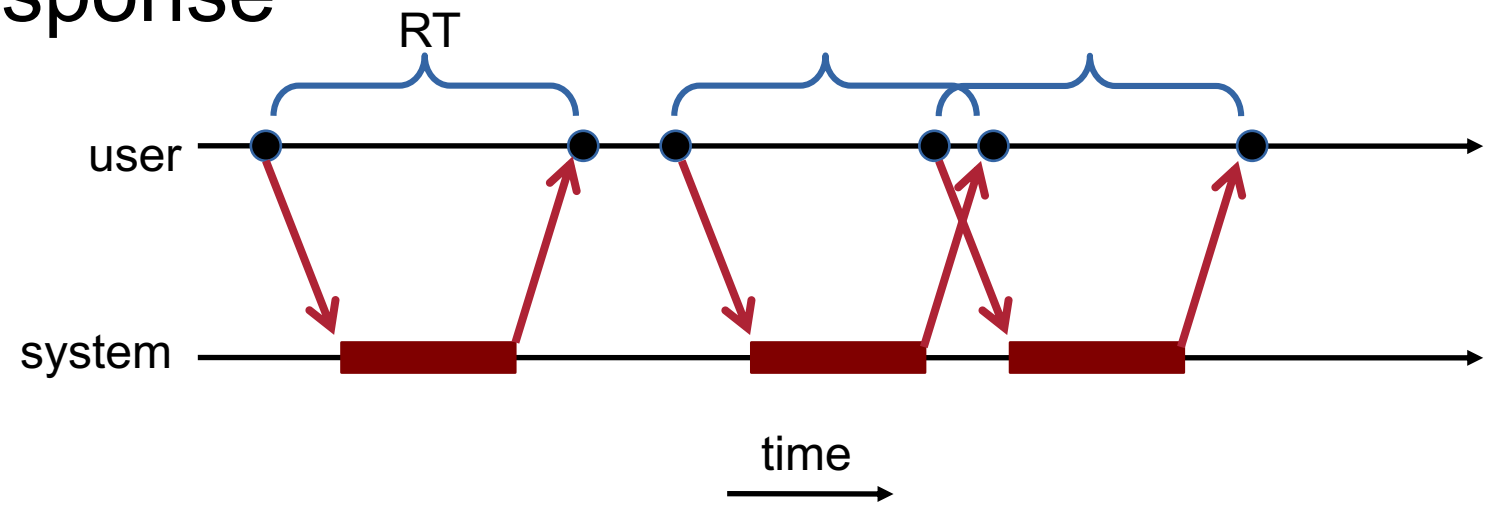
Performance: What and When

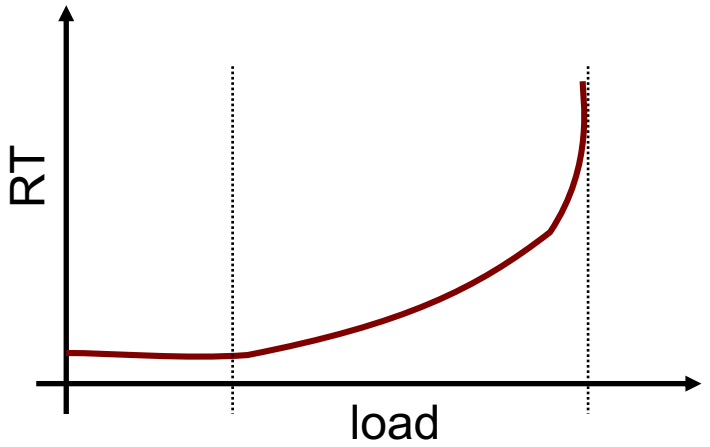




Metric: Response time

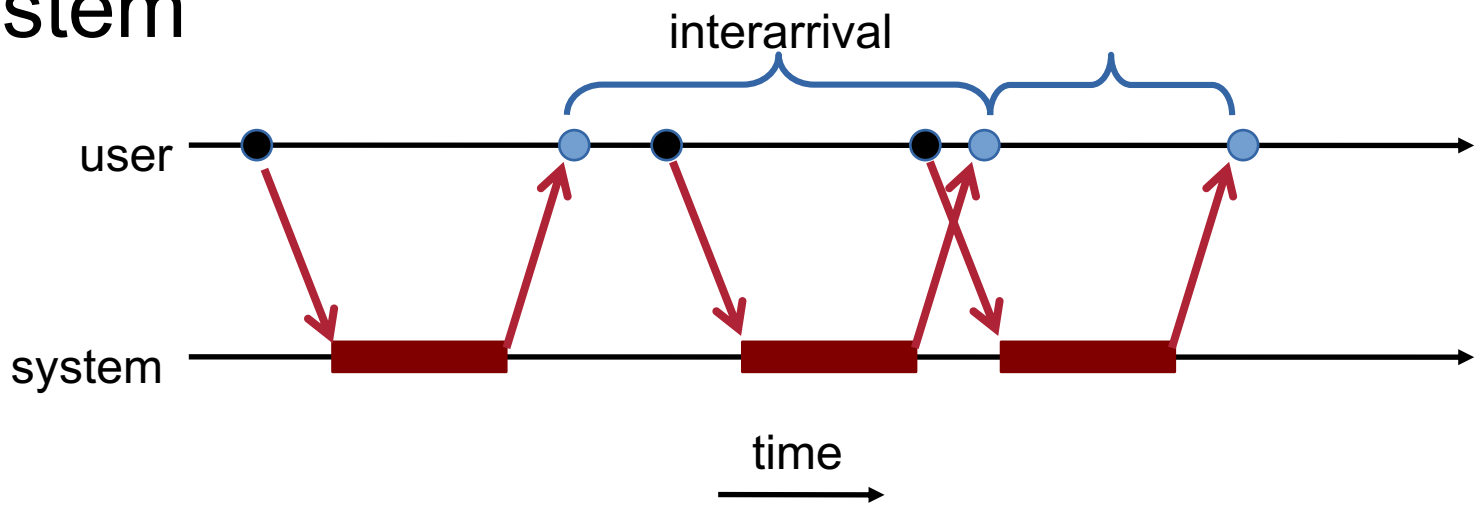
- Interval between user request and system's response

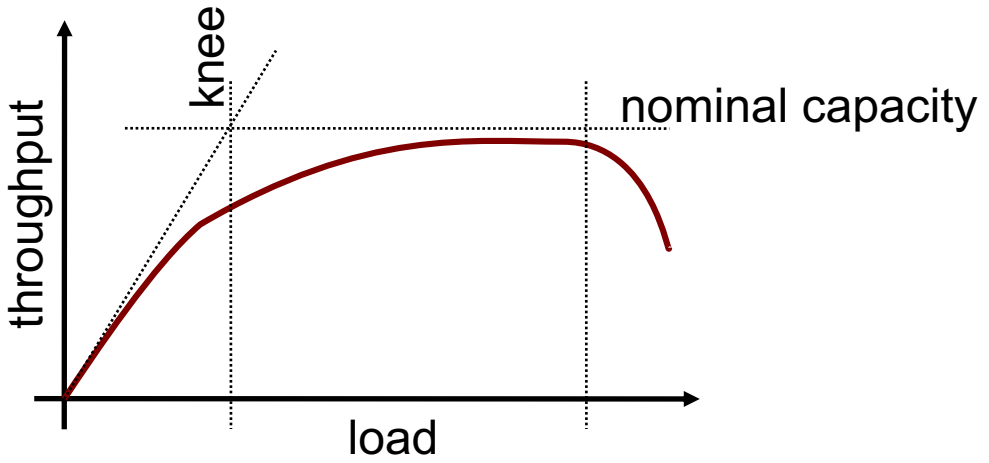




Metric: Throughput

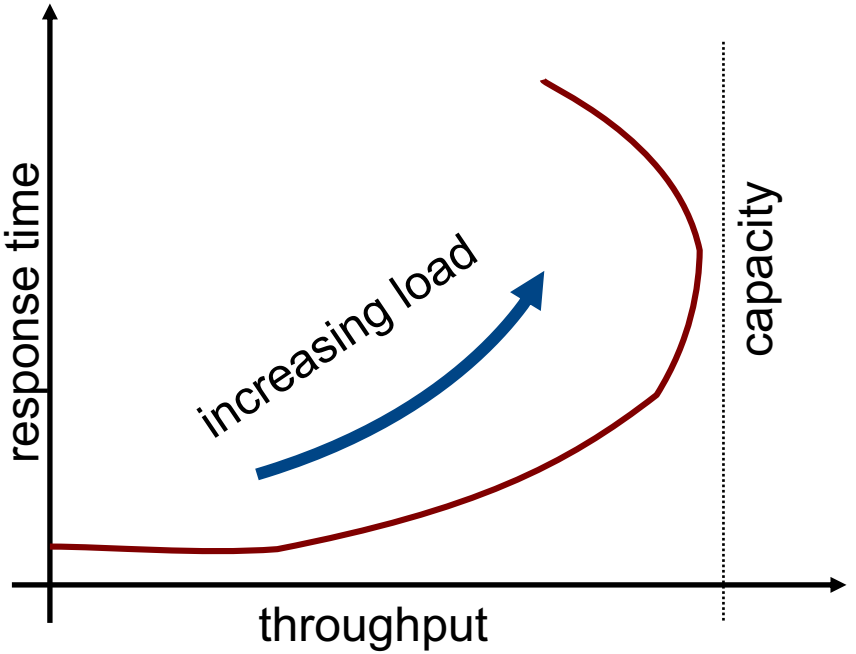
- Rate at which requests are serviced by the system





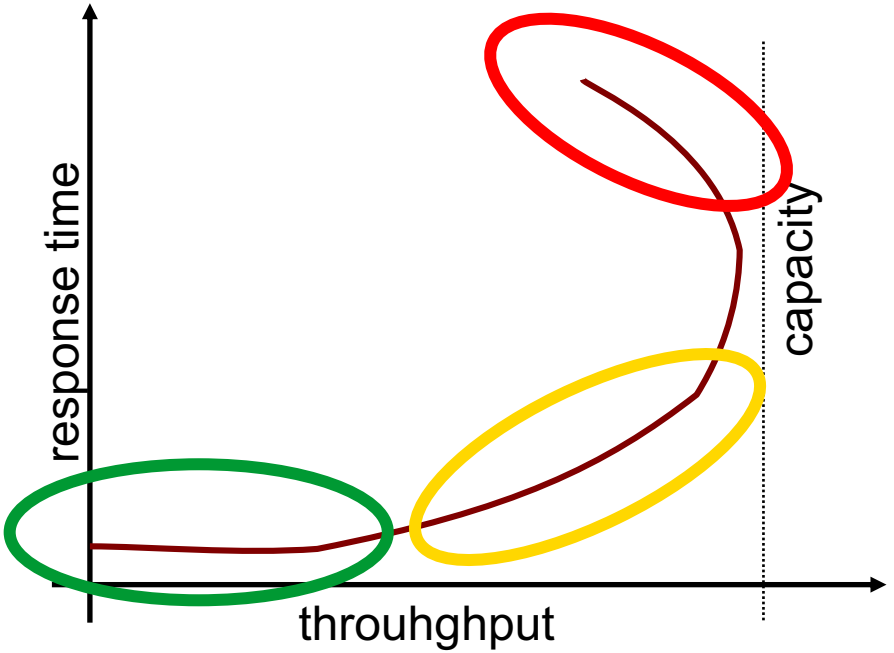
RT vs Throughput

- Naive view (RT = 1 / Throughput) is false!
  - Only true when the system is busy 100% of time executing exactly 1 request!
- The relation between them characterizes system performance:



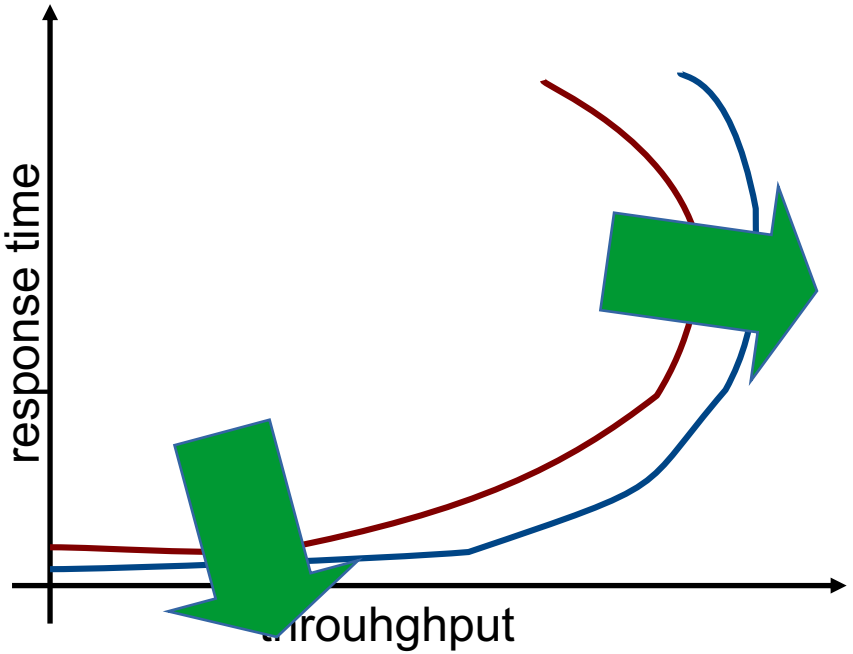
RT vs Throughput

- Idle: requests are immediately handled as the system has a lot of spare capacity
- Requests are handled after a brief wait
- Overload: (some) resources are not optimally used

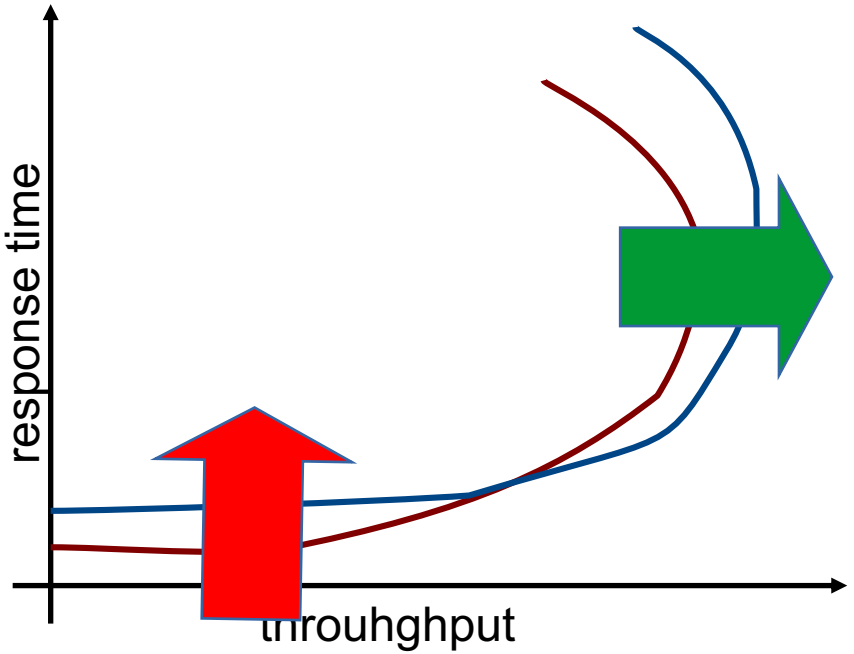


RT vs Throughput

- Optimization:



- Tradeoff:



Other metrics

- Utilization:
  - Resources actually used
- Efficiency
  - Ratio between throughput and utilization
- Reliability
  - Errors
- Availability
  - Uptime/Downtime

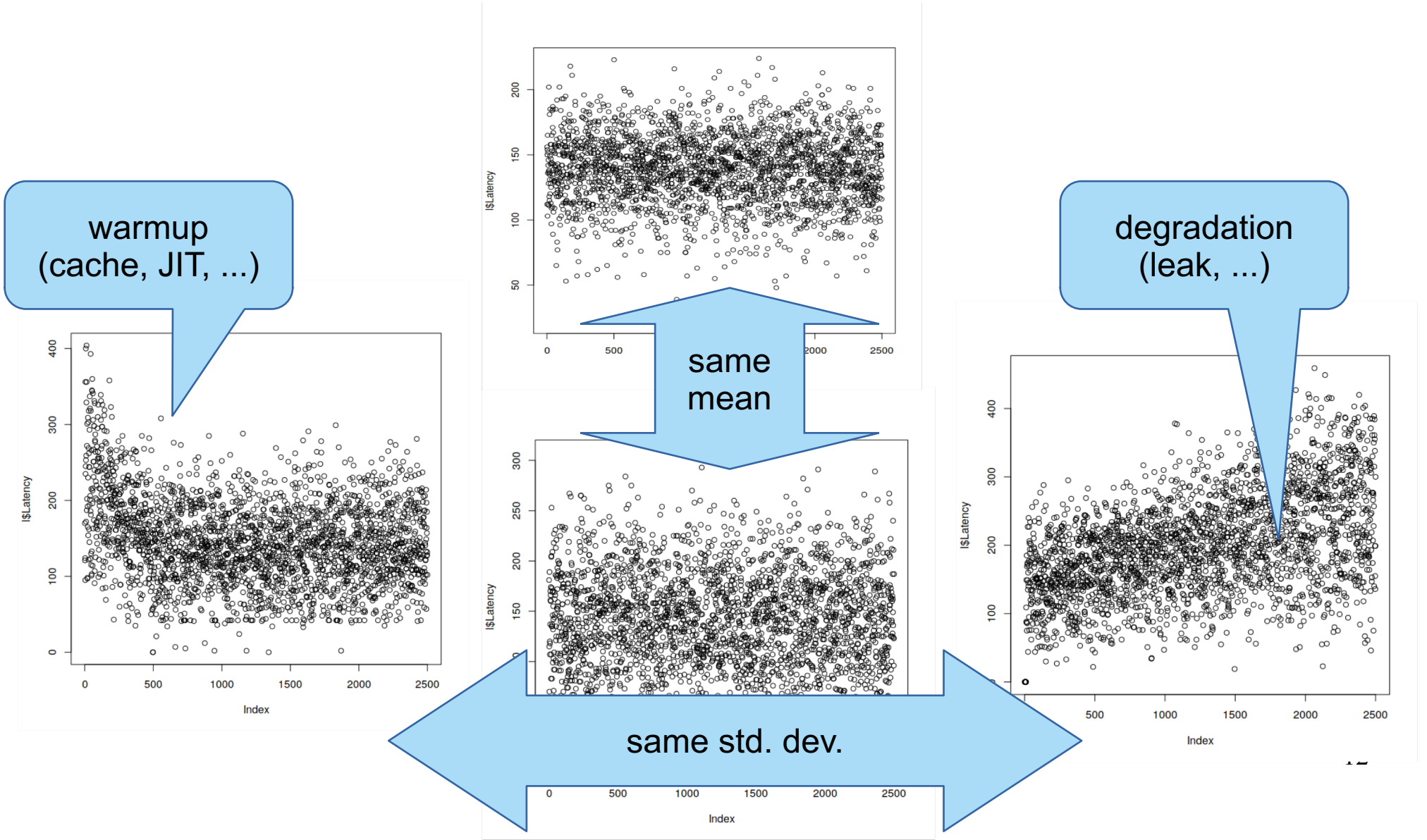
Measuring

- Collect samples
- Summarize with mean…
- … and standard deviation?

- Can we express it as a single number?

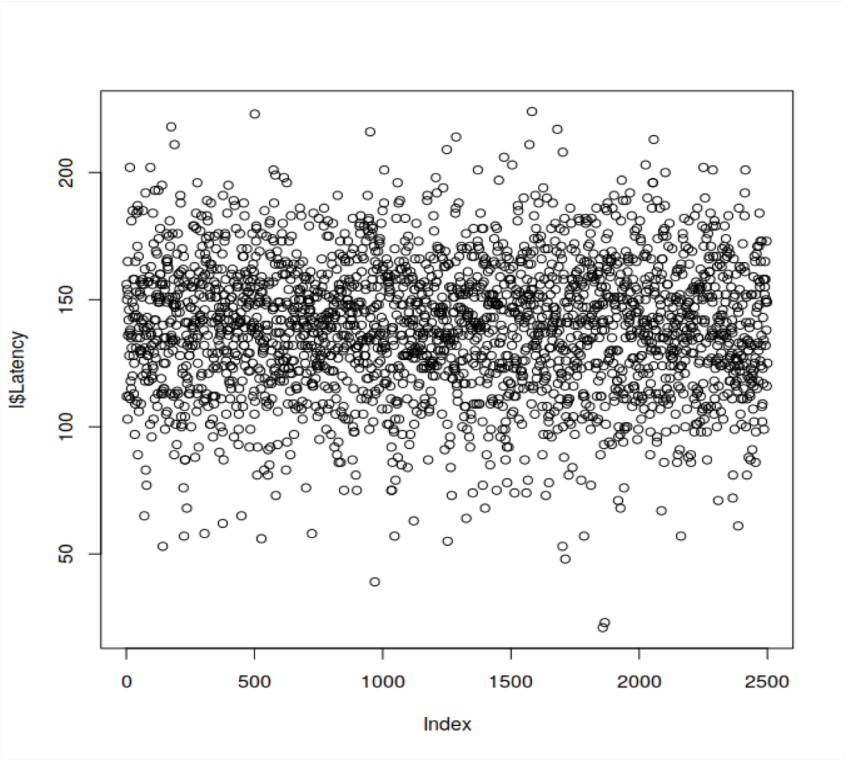
Samples vs Time

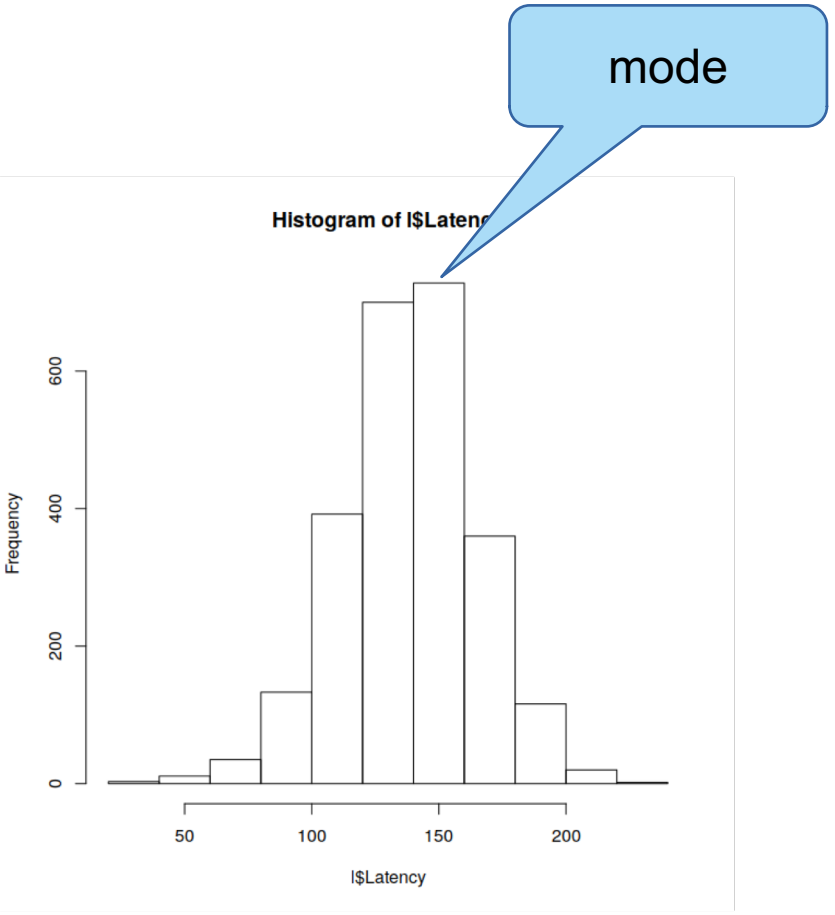
- Represent each sample individually



Samples vs Frequency

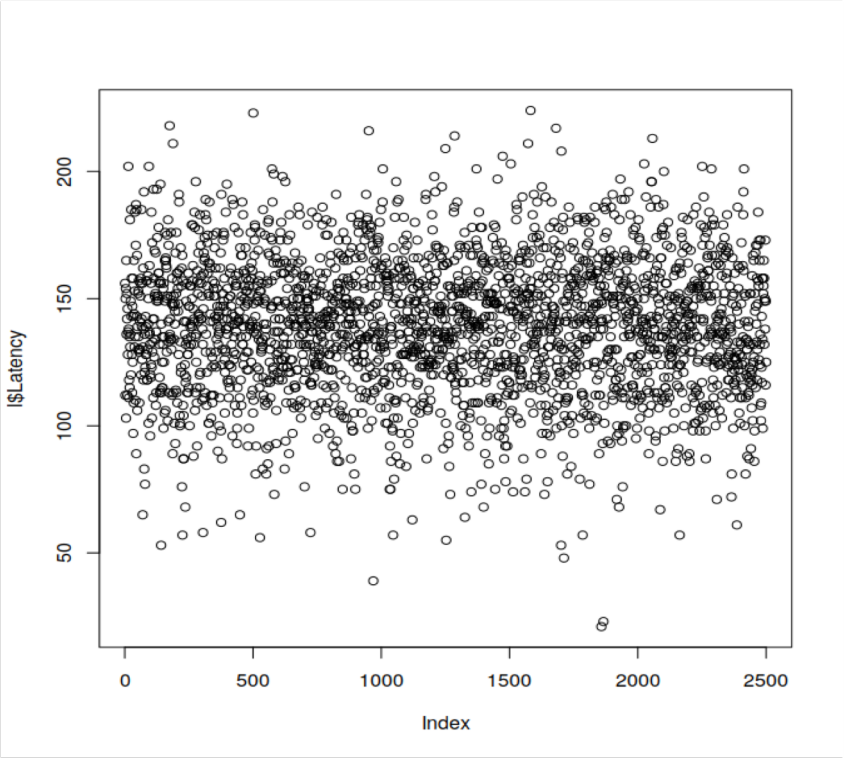
- Represent the frequency of each result
- Histogram
  - Mode, symmetry

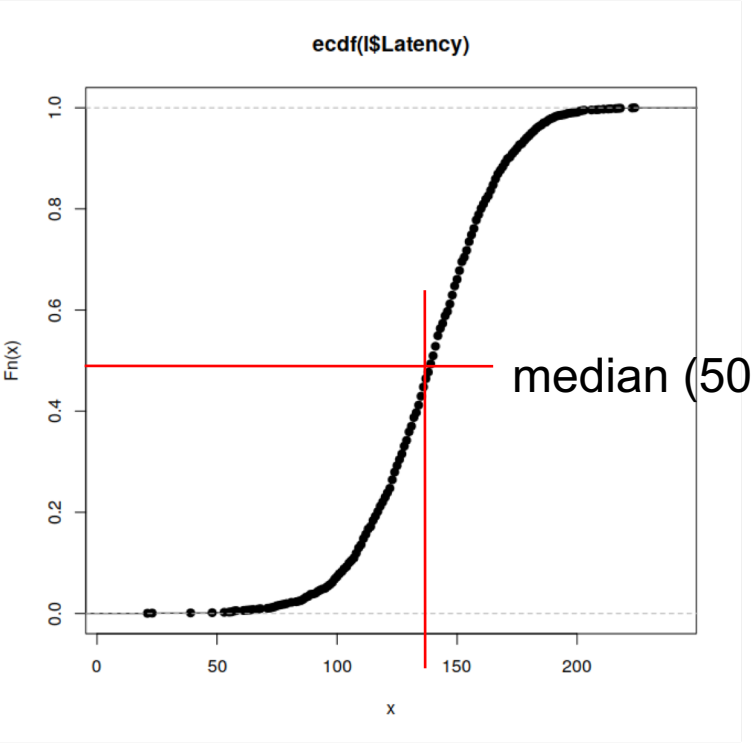




Samples vs Frequency

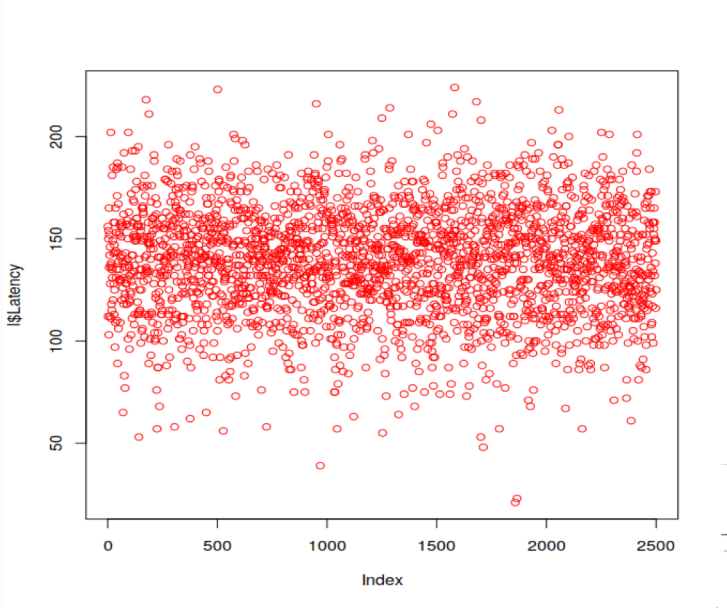
- Empirical Cumulative Distribution Func. (ECDF)
  - Median, percentiles, quartiles, ….
- E.g. 95% RT in Service Level Agreements (SLA)

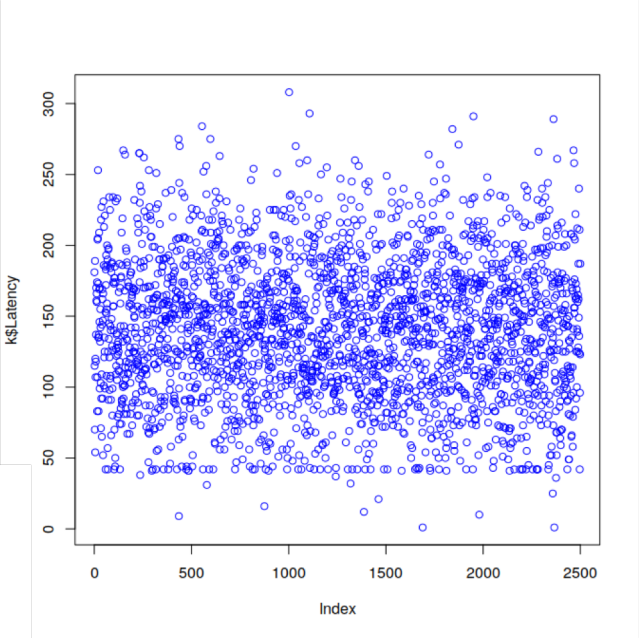


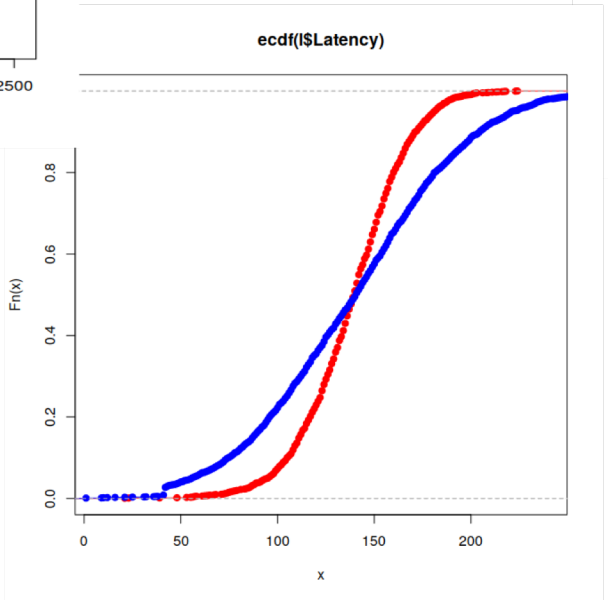


Samples vs Frequency

- Direct comparison of distributions

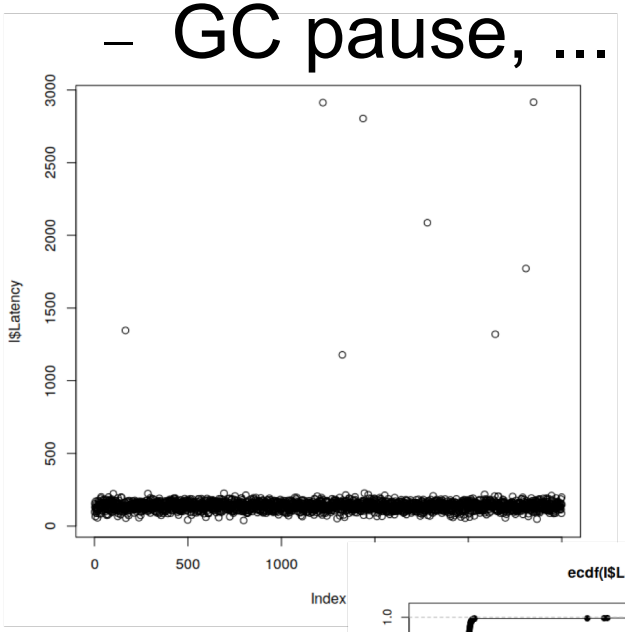


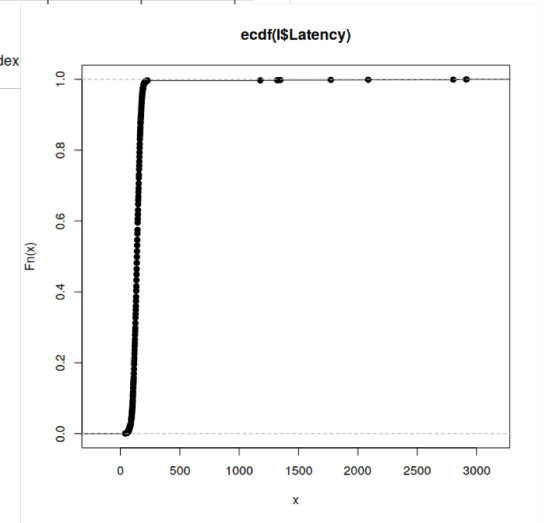




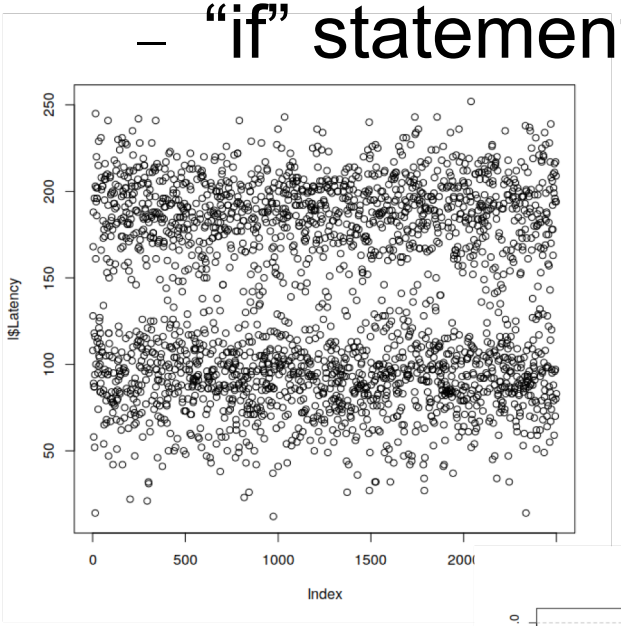
Samples vs Frequency

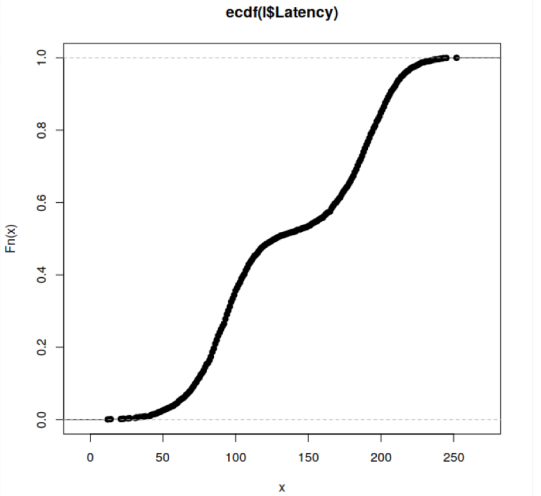
- Long tail:
  - GC pause, ...





- Bimodal:
  - “if” statement, ...

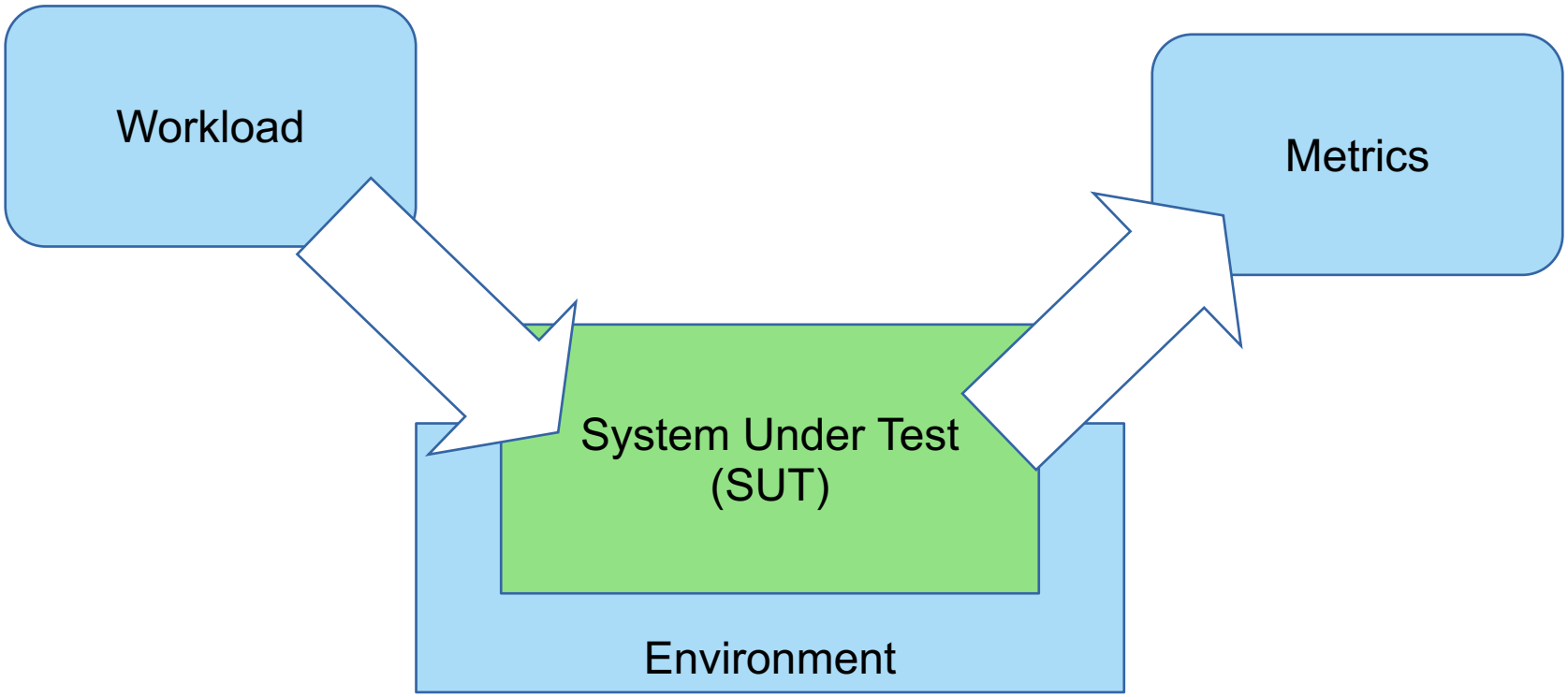




Summarizing samples

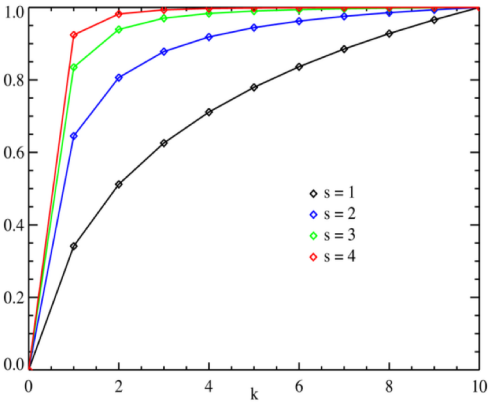
- Mean, mode, median or high percentile
- Confidence interval (CI)
- Coefficient of variation (C.O.V)
  - std. dev. / mean, usually expressed as %

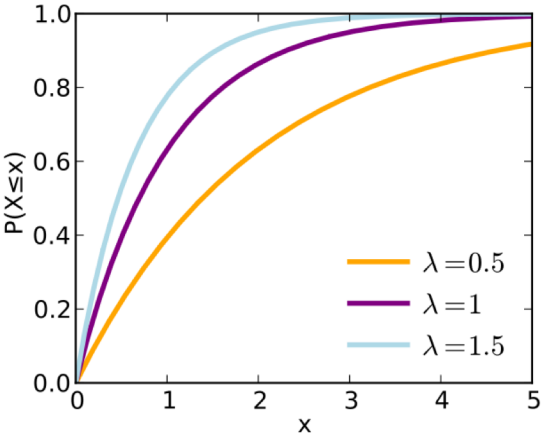
Benchmark



Workload

- Trace from a real system:
  - Hard to scale
  - Hard to get
- Generate synthetic requests:
  - Select subset of operations
  - Generate random parameters
    - Non-uniform random (Zipf)
  - Schedule requests:
    - Concurrent requests
    - Inter-arrival time (Exponential)

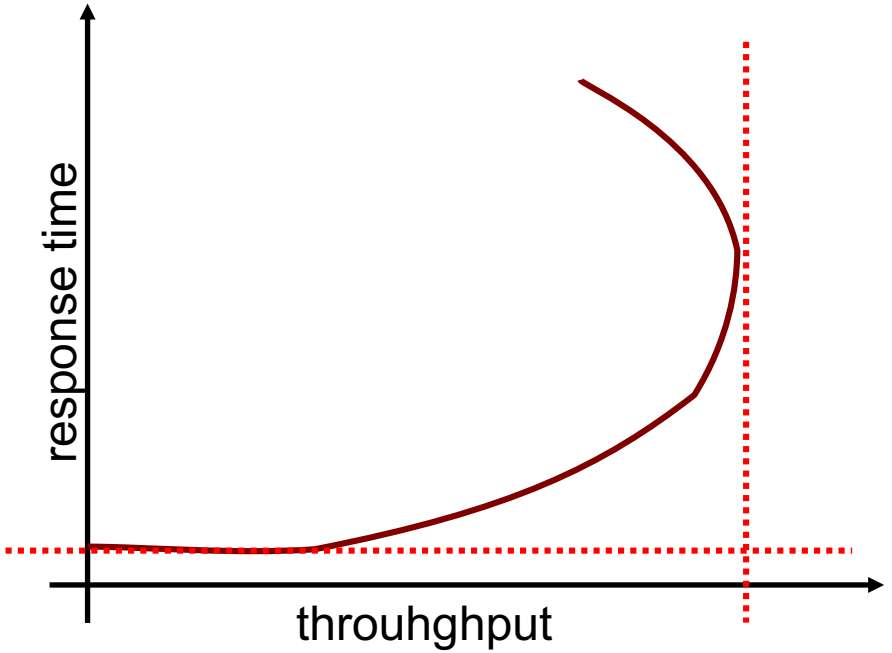




(Source: Wikipedia)

Metrics

- Consider warm-up and cool-down
- Define the performance envelope with:
  - Rate-limited response time
  - Maximum throughput



Tools

- Workload generator and sampling:

- Data analysis

R Cheat Sheet

- Load CSV from JMeter:

  > d<-read.table("file.csv",header=TRUE,sep=",")

- Plot, histogram and ECDF:

  > plot(d$Latency)

  > hist(d$Latency)

  > plot(ecdf(d$Latency))

- Summary:

  > summary(d$Latency)

  > sd(d$Latency)/mean(d$Latency)

Common Mistakes

- No goals or biased ones
- Unsystematic approach
- Unrepresentative workloads and metrics
- Wrong evaluation technique (analytical modeling, simulation, measurement) and experimental design
- Wrong analysis and presentation of results

Conclusion

- Multiple dimensions to system performance:
  - Response time vs throughput
- Avoid misunderstanding measurements:
  - Inspect samples in time for stability
  - Inspect ECDF for distribution
  - Then summarize…
- Benchmarks for repeatable performance evaluation

More...

- R. Jain, *"The Art of Computer Systems Performance Analysis."* Wiley, 1991.
  - Chapters 1 to 5 and 12
  - Further reading:
    - Chapter 6, 9, 10, 11 and 13