Appendix D: Storage Systems (Cont)

Instructor: Josep Torrellas

CS433

Reliability, Availability, Dependability

- Dependability: deliver service such that reliance can be placed on this service
- Actual vs specified behavior
- System failure: actual and specified behavior deviate
- A fault creates a latent error, which becomes effective when it is activated
- When the error actually affects the delivered service, a failure occurs
- Time between the occurrence of an error and the resulting failure is the error latency.
- THEREFORE: given a fault:
  - its manifestation in the system: an error
  - manifestation of the error on the service: failure

Examples

- Example of a fault: a programming mistake
- Its consequence: a latent error (or an effective error)
- When the effective error causes erroneous data that affects service: a failure

- Example of a fault: alpha particle hitting a DRAM
- If it changes the memory: creates an error
- If the memory word is read and the error affects the delivered service: failure (if EEC corrected the error, a failure would not occur).

- Example of a fault: a mistake by a human operator

Notes

- An effective error often propagates from one component to another, creating new errors
- Service can be in two states:
  - Service accomplishment
  - Service interruption
- Transitions between these two states:
  - failures
  - restoration
- Module reliability: measure of the continuous service accomplishment. Often given as **Mean Time To Failure (MTTF).**
- Reciprocal of MTT: failure rate
- Service interruption: measured as **Mean Time To Repair (MTTR).**

Module Availability

- Availability: measure of the service accomplished with respect to the alternation between the two states

- Module Availability: MTTF / (MTTF + MTTR)
- Mean time Between Failure (MTBF) = MTTF + MTTR

Example: find system MTTF

- 10 disks, each 1,000,000 hour MTTF
- 1 SCSI controller: 500,000 hour MTTF
- 1 power supply: 200,000 hour MTTF
- 1 fan: 200,000 MTTF
- 1 SCSI cable: 1,000,000 MTTF
- Assume: component lifetimes are exponentially distributed (age of the component is not important in the prob. Failure)
- Assume: failures are indep
- Failure rate system= $10 \times \frac{1}{1M} + \frac{1}{0.5M} + \frac{1}{0.2M} + \frac{1}{0.2M} + \frac{1}{1M}$

$= 23/1M$

- MTTF = 1/Failure_rate = 1M hours / 23 = 43,500 hours

Classifications

- Faults:
  - Hardware faults: devices that fail
  - Design faults: faults in the software or hardware design
  - Operation faults: mistakes by maintenance personnel
  - Environmental faults: fire, flood…

- Faults:
  - Transient: exist for a limited time and are not recurring
  - Intermittent: system oscillate between faulty and fault-free
  - Permanent: remain

Classifications

- Error Recovery:
  - Backward: returns to a previous correct state, such as with checkpoint and restart
  - Forward: constructs a new correct state: TRM (triple module redundancy)

- Reliability Improvements:
  - Fault avoidance: prevents the occurrence of the fault
  - Fault tolerance: provides service with redundancy

Benchmarks of Disk Performance

1. Transaction processing benchmarks (TP or OLTP)

- Concerned w/ I/O rate : disk accesses/second
- Involves change to a large body of info (database) from many terminals
- Need to guarantee proper behavior on a failure
  e.g. bank transactions from ATM
  airline reservation systems
- Several benchmarks : TPC-A, TPC-B,TPC-C,TPC-D

measure # transactions per second (TPS) or per minute

Example: TPC-C

- Simulates an order-entry environment for a wholesale supplier
- Includes transactions to enter and deliver orders, record payments, check the status of orders, etc
- Runs 5 concurrent transactions of varying complexity
- Measured in transactions per minute (tpmC) and price of the system
- TPC benchmarks (Figure 6.12):
  - the higher the throughput, the better, but price included in benchmark results
  - however , benchmark requires that for throughput ↑ the size of the files ↑
  - this scaling is necessary to ensure that benchmark measures I/O ; else large memory w/small files
  - Throughput is the performance metric, but response times are limited
  - Benchmark results are audited.

Benchmarks of Disk Performance

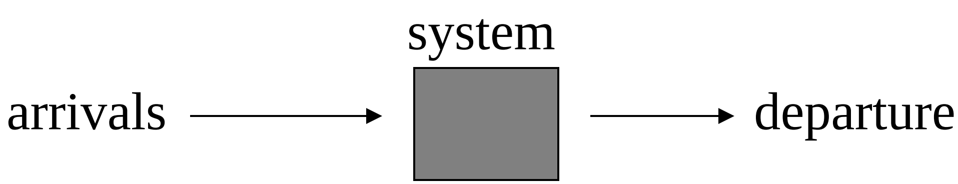
2. Spec system level File Server, Mail and Web Benchmarks

- System Level File Server:
  - synthetic benchmark to evaluate NFS performance
  - contains a mix of rd/wr/file ops
  - scales the size of the file system according to the reported throughput: for every 100 NFS ops/second , capacity must increase by 1GB
- SpecMail: evaluate mail servers
- SpecWeb: evaluate web servers

Queuing Theory

- Since equilibrium ⇒ input rate = output rate

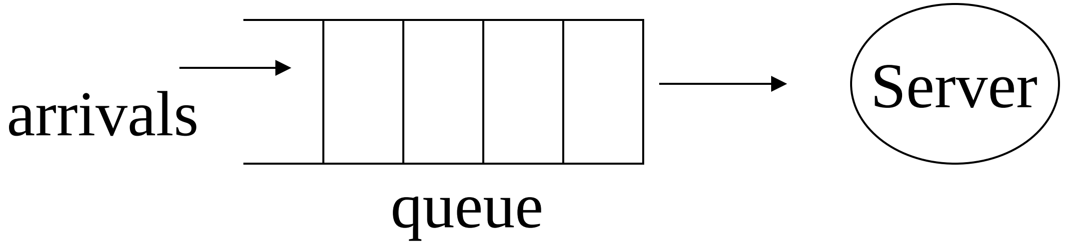
-Little's Law :



Mean Number of tasks in system = Arrival rate of tasks * Mean response time

Queuing Theory

The system is



- $\text{Time}_{\text{server}}$ = avg time to service a task
- Service rate = $1/\text{Time}_{\text{server}}$ = μ
- $\text{Time}_{\text{queue}}$ = avg time per task in the queue
- $\text{Time}_{\text{system}} = T_{\text{queue}} + T_{\text{server}}$ = response time
- Arrival rate = # Tasks arriving / second = λ
- $\text{Length}_{\text{server}}$ = Avg # tasks in service
- $\text{Length}_{\text{queue}}$ = Avg length queue

Queuing Theory

Little's law applied to each component

$Length_{queue}$ = Arrival rate * $Time_{queue}$

$Length_{server}$ = Arrival rate * $Time_{server}$

Example : time to service a disk request 50 ms

system requires 200 I/O req/second

On average , how many I/O req at the server ?

Length server = Arrival rate * $Time_{server}$

200 req/sec * 0.05 sec = 10 req

10 req on average at the server

Queuing Theory

$Length_{system} = Length_{queue} + Length_{server}$ = #tasks in system

Therefore , Little's law:

$$Length_{system} = \text{Arrival Rate} * Time_{system}$$

Server utilization :

$$\rho = \frac{\text{Arrival rate}}{\text{Service Rate}}$$

Needs to be between 0 and 1

also called **traffic intensity**

Example : Disk gets 10 I/O req/second
time to service 1 request = 50ms } sever util ?

$$\text{server util} = \frac{\text{Arrival rate}}{\text{Service Rate}} = \frac{\text{10 IOPS}}{1/0.05} = 0.5$$