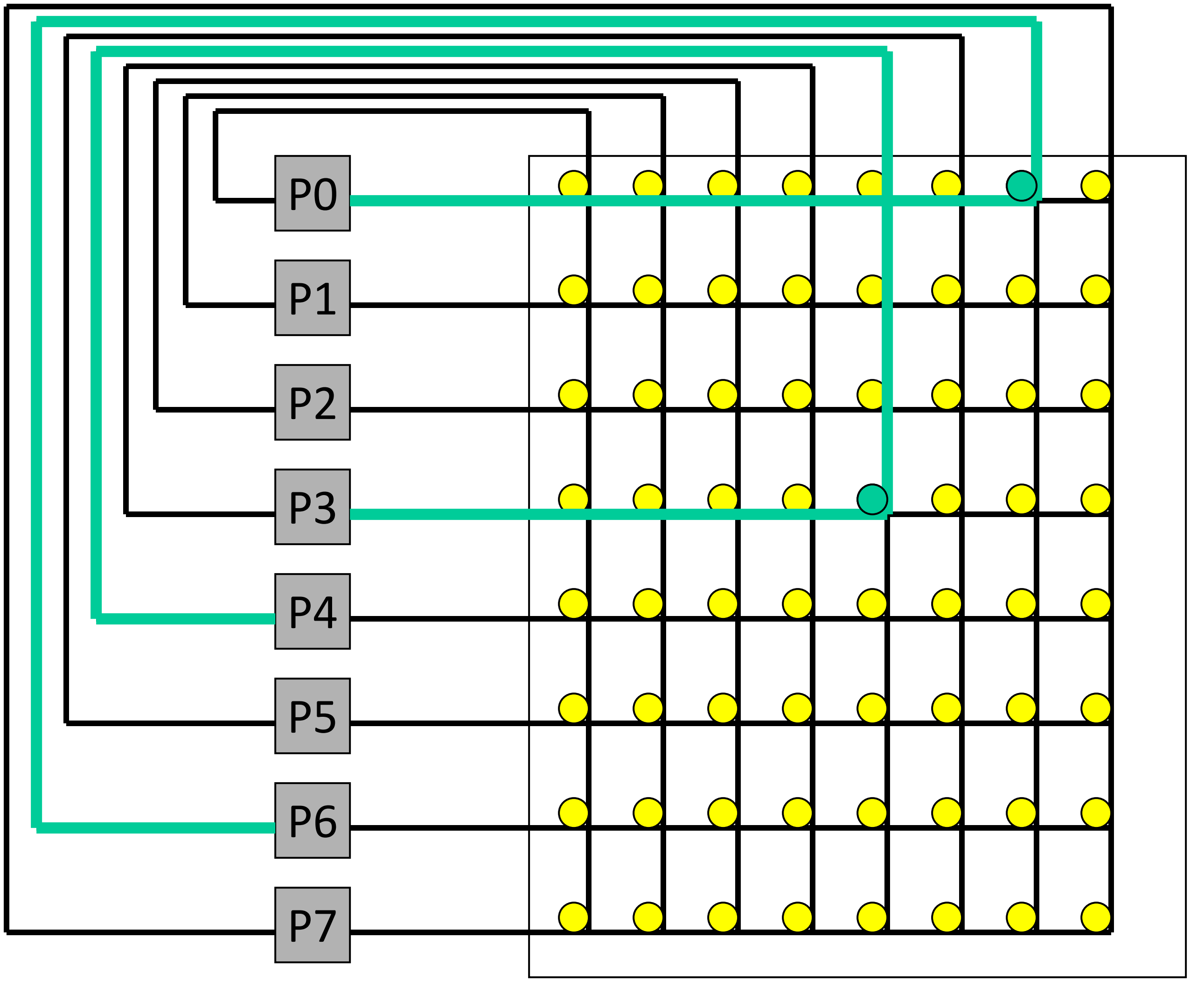
Lecture: Networks, Disks, Datacenters

- Topics: networks wrap-up, disks and reliability,
  datacenters and energy proportionality

Centralized Crossbar Switch



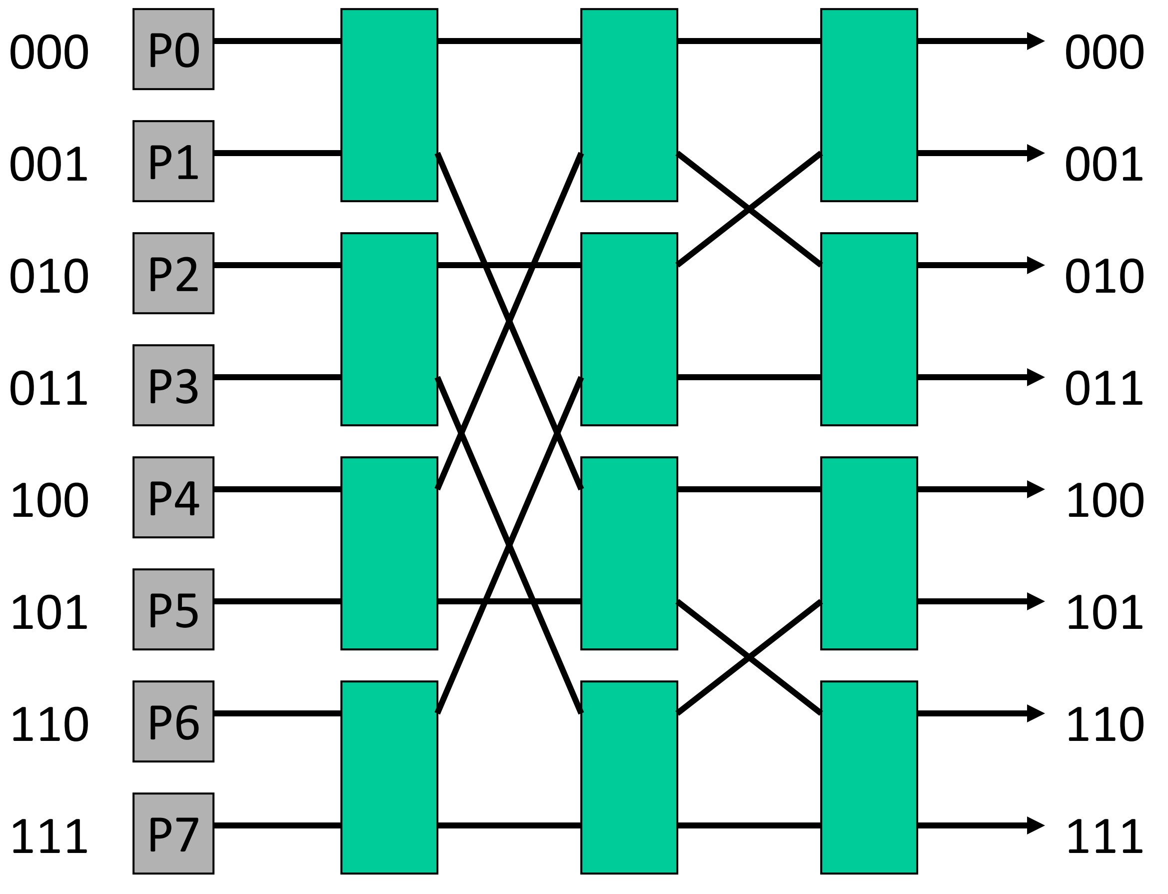
Crossbar Properties

- Assuming each node has one input and one output, a crossbar can provide maximum bandwidth: N messages can be sent as long as there are N unique sources and N unique destinations

- Maximum overhead: WN^2 internal switches, where W is data width and N is number of nodes

- To reduce overhead, use smaller switches as building blocks – trade off overhead for lower effective bandwidth

Switch with Omega Network

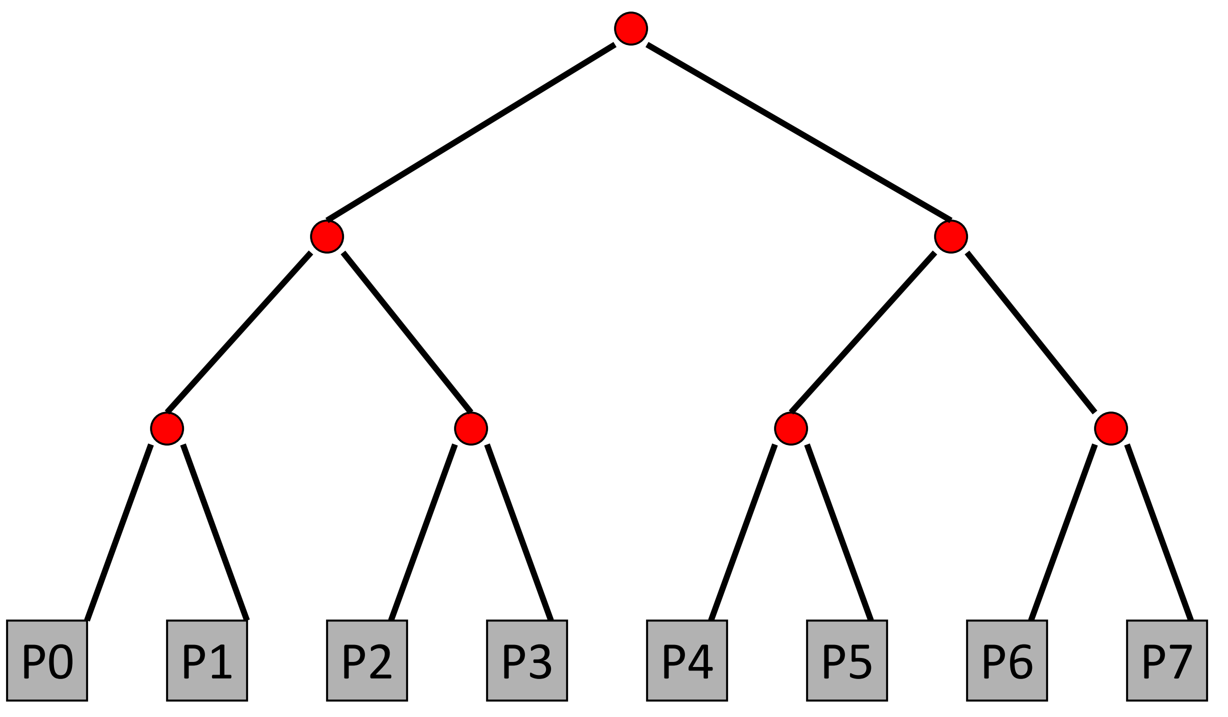


Omega Network Properties

- The switch complexity is now O(N log N)
- Contention increases: P0 → P5 and P1 → P7 cannot happen concurrently (this was possible in a crossbar)
- To deal with contention, can increase the number of levels (redundant paths) – by mirroring the network, we can route from P0 to P5 via N intermediate nodes, while increasing complexity by a factor of 2

Tree Network

- Complexity is O(N)
- Can yield low latencies when communicating with neighbors
- Can build a fat tree by having multiple incoming and outgoing links



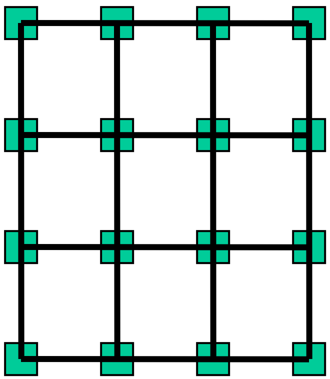
Bisection Bandwidth

- Split N nodes into two groups of N/2 nodes such that the bandwidth between these two groups is minimum: that is the bisection bandwidth

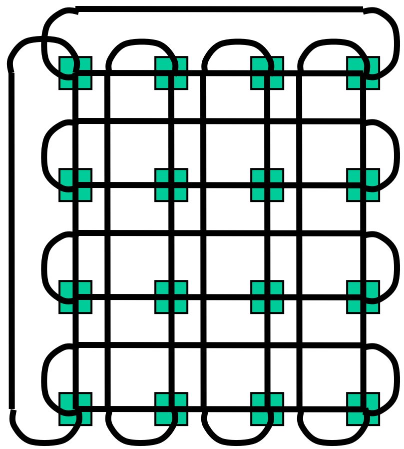
- Why is it relevant: if traffic is completely random, the probability of a message going across the two halves is ½ – if all nodes send a message, the bisection bandwidth will have to be N/2

- The concept of bisection bandwidth confirms that the tree network is not suited for random traffic patterns, but for localized traffic patterns

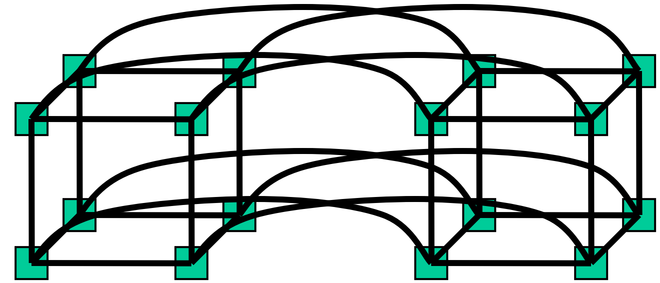
Topology Examples



Grid



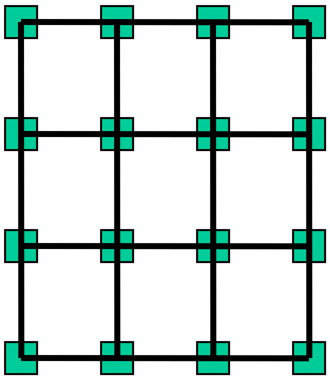
Torus



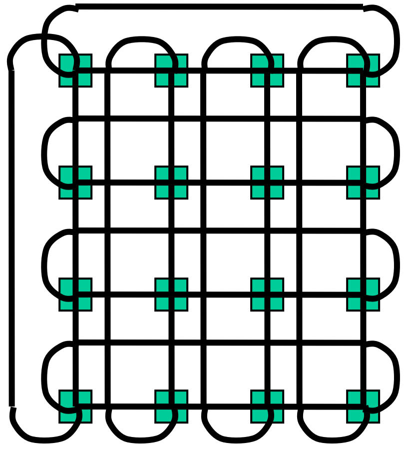
Hypercube

Criteria 64 nodes	Bus	Ring	2Dtorus	Hypercube	Fully connected
Performance Bisection bandwidth					
Cost Ports/switch Total links					

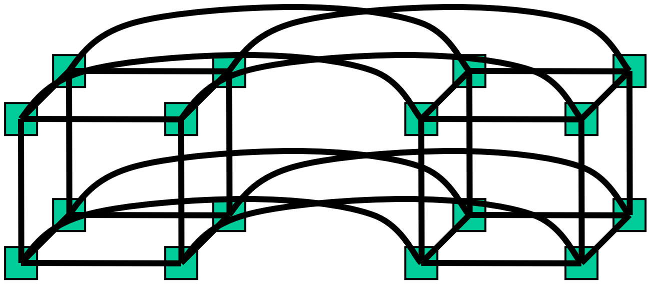
Topology Examples



Grid



Torus



Hypercube

Criteria 64 nodes	Bus	Ring	2Dtorus	Hypercube	Fully connected
Performance					
Diameter	1	32	8	6	1
Bisection BW	1	2	16	32	1024
Cost					
Ports/switch		3	5	7	64
Total links	1	64	128	192	2016

k-ary d-cube

- Consider a k-ary d-cube: a d-dimension array with k elements in each dimension, there are links between elements that differ in one dimension by 1 (mod k)

- Number of nodes $N = k^d$

Number of switches :	Avg. routing distance:
Switch degree :	Diameter :
Number of links :	Bisection bandwidth :
Pins per node :	Switch complexity :

Should we minimize or maximize dimension?

k-ary d-Cube

- Consider a k-ary d-cube: a d-dimension array with k elements in each dimension, there are links between elements that differ in one dimension by 1 (mod k)

- Number of nodes $N = k^d$

Number of switches :	N	Avg. routing distance:	$d(k-1)/4$
Switch degree :	$2d + 1$	Diameter :	$d(k-1)/2$
Number of links :	Nd	Bisection bandwidth :	$2wk^{d-1}$
Pins per node :	2wd	Switch complexity :	$(2d + 1)^2$

The switch degree, num links, pins per node, bisection bw for a hypercube are half of what is listed above (diam and avg routing distance are twice, switch complexity is $(d + 1)^2$) because unlike the other cases, a hypercube does not have right and left neighbors.

Should we minimize or maximize dimension?

Warehouse-Scale Computer (WSC)

- 100K+ servers in one WSC
- ~$150M overall cost
- Requests from millions of users (Google, Facebook, etc.)
- Cloud Computing: a model where users can rent compute and storage within a WSC; there's an associated service-level agreement (SLA)
- Datacenter: a collection of WSCs in a single building, possibly belonging to different clients and using different hardware/architecture

PUE Metric and Power Breakdown

- PUE = Total facility power / IT equipment power
  (power utilization effectiveness)

- It is greater than 1; ranges from 1.33 to 3.03, median of 1.69

- The cooling power is roughly half the power used by servers

- Within a server, the approximate power distribution is as follows: Processors (33%), DRAM memory (30%), Disks (10%), Networking (5%), Miscellaneous (22%)

CapEx and OpEx

- Capital expenditure: infrastructure costs for the building, power delivery, cooling, and servers

- Operational expenditure: the monthly bill for energy, failures, personnel, etc.

- CapEx can be amortized into a monthly estimate by assuming that the facilities will last 10 years, server parts will last 3 years, and networking parts will last 4

CapEx/OpEx Case Study

- 8 MW facility : facility cost: $88M, server/networking cost: $79M

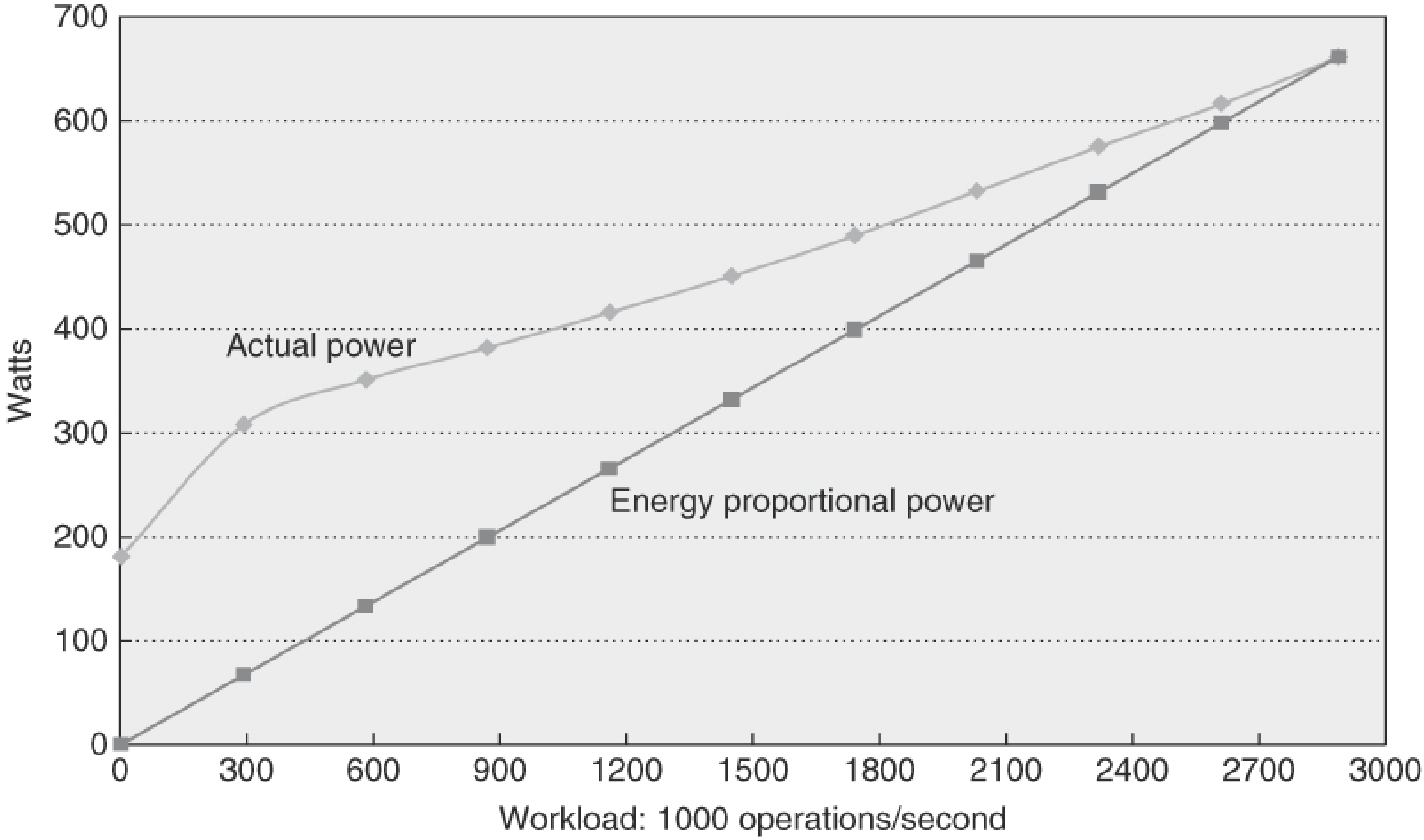
- Monthly expense: $3.8M. Breakdown:
  - Servers 53% (amortized CapEx)
  - Networking 8% (amortized CapEx)
  - Power/cooling infrastructure 20% (amortized CapEx)
  - Other infrastructure 4% (amortized CapEx)

  - Monthly power bill 13% (true OpEx)
  - Monthly personnel salaries 2% (true OpEx)

Improving Energy Efficiency

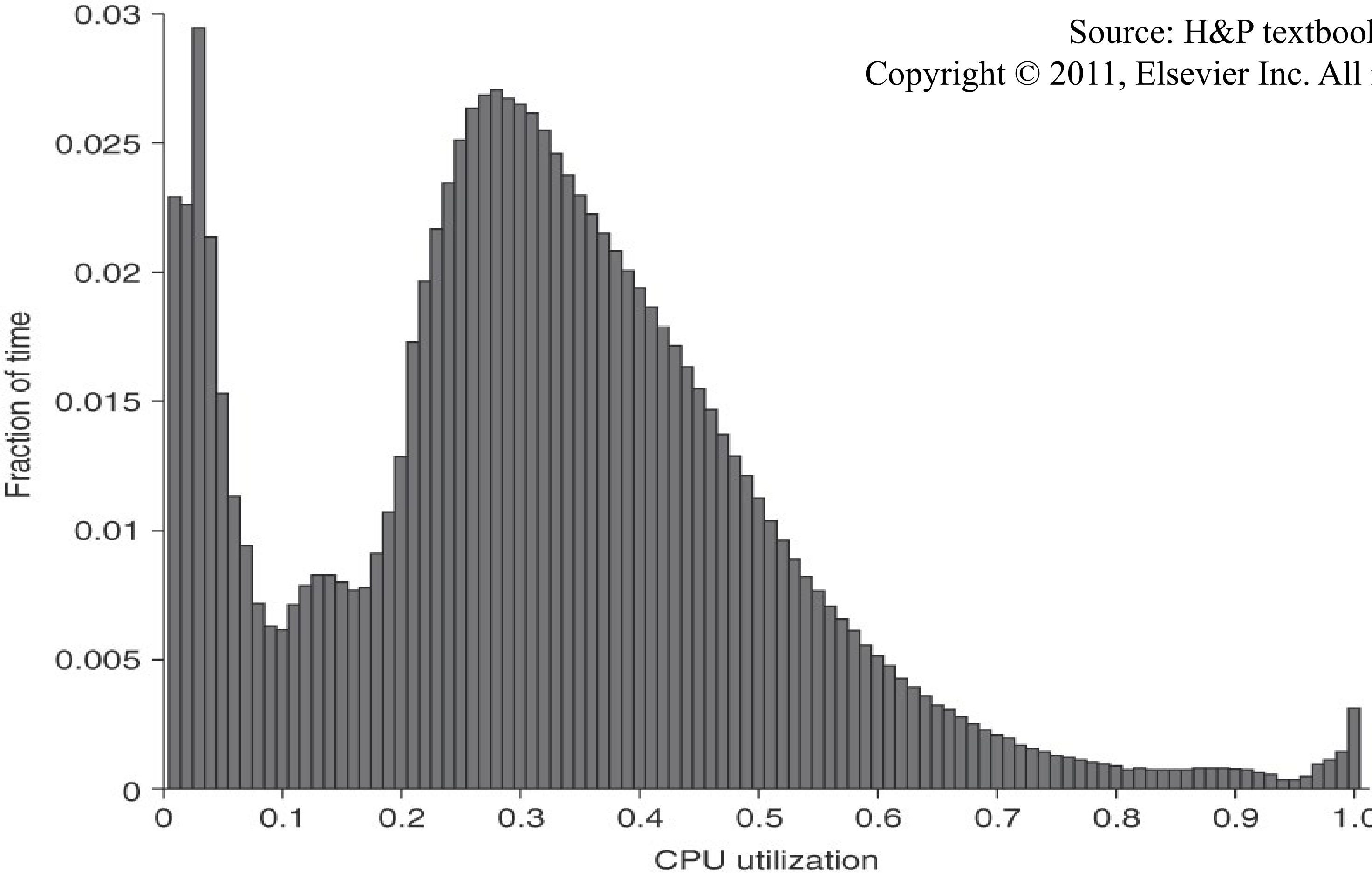
- An unloaded server dissipates a large amount of power
- Ideally, we want energy-proportional computing, but in reality, servers are not energy-proportional
- Can approach energy-proportionality by turning on a few servers that are heavily utilized
- See figures on next two slides for power/utilization profile of a server and a utilization profile of servers in a WSC

Power/Utilization Profile



Source: H&P textbook.

Server Utilization Profile



Source: H&P textbook.

Figure 6.3 Average CPU utilization of more than 5000 servers during a 6-month period at Google. Servers are rarely completely idle or fully utilized, in-stead operating most of the time at between 10% and 50% of their maximum utilization. (From Figure 1 in Barroso and Hölzle [2007].) The column the third from the right in Figure 6.4 calculates percentages plus or minus 5% to come up with the weightings; thus, 1.2% for the 90% row means that 1.2% of servers were between 85% and 95% utilized.

Problem 1

Assume that a server consumes 100W at peak utilization and 50W at zero utilization. Assume a linear relationship between utilization and power. The server is capable of executing many threads in parallel. Assume that a single thread utilizes 25% of all server resources (functional units, caches, memory capacity, memory bandwidth, etc.). What is the total power dissipation when executing 99 threads on a collection of these servers, such that performance and energy are close to optimal?

Problem 1

Assume that a server consumes 100W at peak utilization and 50W at zero utilization. Assume a linear relationship between utilization and power. The server is capable of executing many threads in parallel. Assume that a single thread utilizes 25% of all server resources (functional units, caches, memory capacity, memory bandwidth, etc.). What is the total power dissipation when executing 99 threads on a collection of these servers, such that performance and energy are close to optimal?

For near-optimal performance and energy, use 25 servers. 24 servers at 100% utilization, executing 96 threads, consuming 2400W. The 25th server will run the last 3 threads and consume 87.5~W.

Other Metrics

- Performance does matter, both latency and throughput
- An analysis of the Bing search engine shows that if a 200ms delay is introduced in the response, the next click by the user is delayed by 500ms; so a poor response time amplifies the user's non-productivity
- Reliability (MTTF) and Availability (MTTF/MTTF+MTTR) are very important, given the large scale
- A server with MTTF of 25 years (amazing!) : 50K servers would lead to 5 server failures a day; Similarly, annual disk failure rate is 2-10% ➔ 1 disk failure every hour

Important Problems

- Reducing power in power-down states
- Maximizing utilization
- Reducing cost with virtualization
- Reducing data movement
- Building a low-power low-cost processor
- Building a low-power low-cost hi-bw memory
- Low-power low-cost on-demand reliability

Magnetic Disks

- A magnetic disk consists of 1-12 *platters* (metal or glass disk covered with magnetic recording material on both sides), with diameters between 1-3.5 inches

- Each platter is comprised of concentric *tracks* (5-30K) and each track is divided into *sectors* (100 – 500 per track, each about 512 bytes)

- A movable arm holds the read/write heads for each disk surface and moves them all in tandem – a *cylinder* of data is accessible at a time

Disk Latency

- To read/write data, the arm has to be placed on the correct track – this *seek time* usually takes 5 to 12 ms on average – can take less if there is spatial locality

- *Rotational latency* is the time taken to rotate the correct sector under the head – average is typically more than 2 ms (15,000 RPM)

- *Transfer time* is the time taken to transfer a block of bits out of the disk and is typically 3 – 65 MB/second

- A disk controller maintains a disk cache (spatial locality can be exploited) and sets up the transfer on the bus (*controller overhead*)

RAID

- Reliability and availability are important metrics for disks
- RAID: redundant array of inexpensive (independent) disks
- Redundancy can deal with one or more failures
- Each sector of a disk records check information that allows it to determine if the disk has an error or not (in other words, redundancy already exists within a disk)
- When the disk read flags an error, we turn elsewhere for correct data

RAID 0 and RAID 1

- RAID 0 has no additional redundancy (misnomer) – it uses an array of disks and stripes (interleaves) data across the arrays to improve parallelism and throughput

- RAID 1 mirrors or shadows every disk – every write happens to two disks

- Reads to the mirror may happen only when the primary disk fails – or, you may try to read both together and the quicker response is accepted

- Expensive solution: high reliability at twice the cost

RAID 3

- Data is bit-interleaved across several disks and a separate disk maintains parity information for a set of bits

- For example: with 8 disks, bit 0 is in disk-0, bit 1 is in disk-1, …, bit 7 is in disk-7; disk-8 maintains parity for all 8 bits

- For any read, 8 disks must be accessed (as we usually read more than a byte at a time) and for any write, 9 disks must be accessed as parity has to be re-calculated

- High throughput for a single request, low cost for redundancy (overhead: 12.5%), low task-level parallelism

RAID 4 and RAID 5

- Data is block interleaved – this allows us to get all our data from a single disk on a read – in case of a disk error, read all 9 disks
- Block interleaving reduces thruput for a single request (as only a single disk drive is servicing the request), but improves task-level parallelism as other disk drives are free to service other requests
- On a write, we access the disk that stores the data and the parity disk – parity information can be updated simply by checking if the new data differs from the old data

RAID 5

- If we have a single disk for parity, multiple writes can not happen in parallel (as all writes must update parity info)
- RAID 5 distributes the parity block to allow simultaneous writes

Other Reliability Approaches

- High reliability is also expected of memory systems; many memory systems offer SEC-DED support – single error correct, double error detect; implemented with an 8-bit code for every 64-bit data word on ECC DIMMs

- Some memory systems offer chipkill support – the ability to recover from complete failure in one memory chip – many implementations exist, some resembling RAID designs

- Caches are typically protected with SEC-DED codes

- Some cores implement various forms of redundancy, e.g., DMR or TMR – dual or triple modular redundancy