Queueing Networks models

15/11/09

outline

- workload characterization
- types of queueing network models
  - resource types, service requests, open and closed models
- case study: cache and DRAM models
- case study: paging, DRAM and disk I/O models
- case study: http download (parallel connections)
- case study: client side and web server side models
- resource saturation (bottleneck)
  - asymptotic analysis, bottlenecks identification

workload characterization

workload

service requests submitted by users to hw and sw resources (CPU, disks, ..., objects, files, data, …, server, proxy, URL, ...) in a given time interval (observation interval)

workload characterization is fundamental for the construction of workload models

• to drive a system under measurement (in order to have a controlled and reproducible environment)

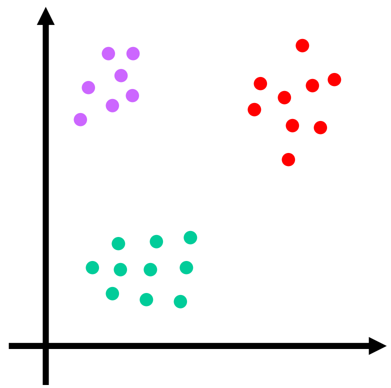
• constituted by parameters to be used in system models (e.g., queueing networks models)

component classes

obtained using clustering algorithms

a component is a point in a **n**-dimensions space (**n**: number of parameters used to describe each component)

points are assigned to a cluster according to several criteria (min. distance, min.variance, ...)



different types of workload

workload intensity specified by one of:

- λ arrival rate of requests (customers), population varies over time (open models), for transaction workloads

- **N** avg number of customers in the system, fixed population (closed models), for batch workloads

- **N** avg number of interactive customers, **Z** think time, avg time that customers use terminals (think) between interactions, (closed models) ,for interactive workloads

type of QN models

- Resource and models
- Case study: CPU, cache, RAM
- Memory hierarchy
- Case study: CPU, paging, I/O

analytical models

interactions between resources and service demands are described through matematical relations

the deterministic nature of the problem is modelled probabilistically

processing of a service request: sequence of utilization and queueing phases

many requests compete for few resources

queueing network models are well suited for the simulation of such a situations

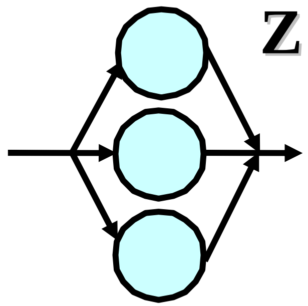
queueing network models

- iterative
  - easy to implement and parameterize
  - facilitate answering many “what if “ questions
  - require a limited set of parameters
  - high accuracy at low computational cost
  - minimum processing time

- their accuracy is consistent with that achievable in other components of the system evaluation process:
  - collected data
  - workload characterization and forecasting

- do not require low level parameters

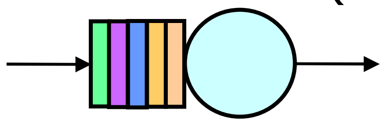
resources and models



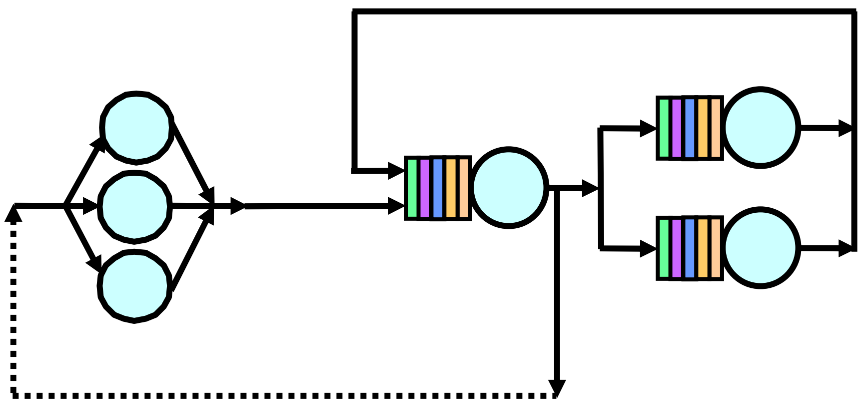
Z (think time)

delay resource

S (service time)

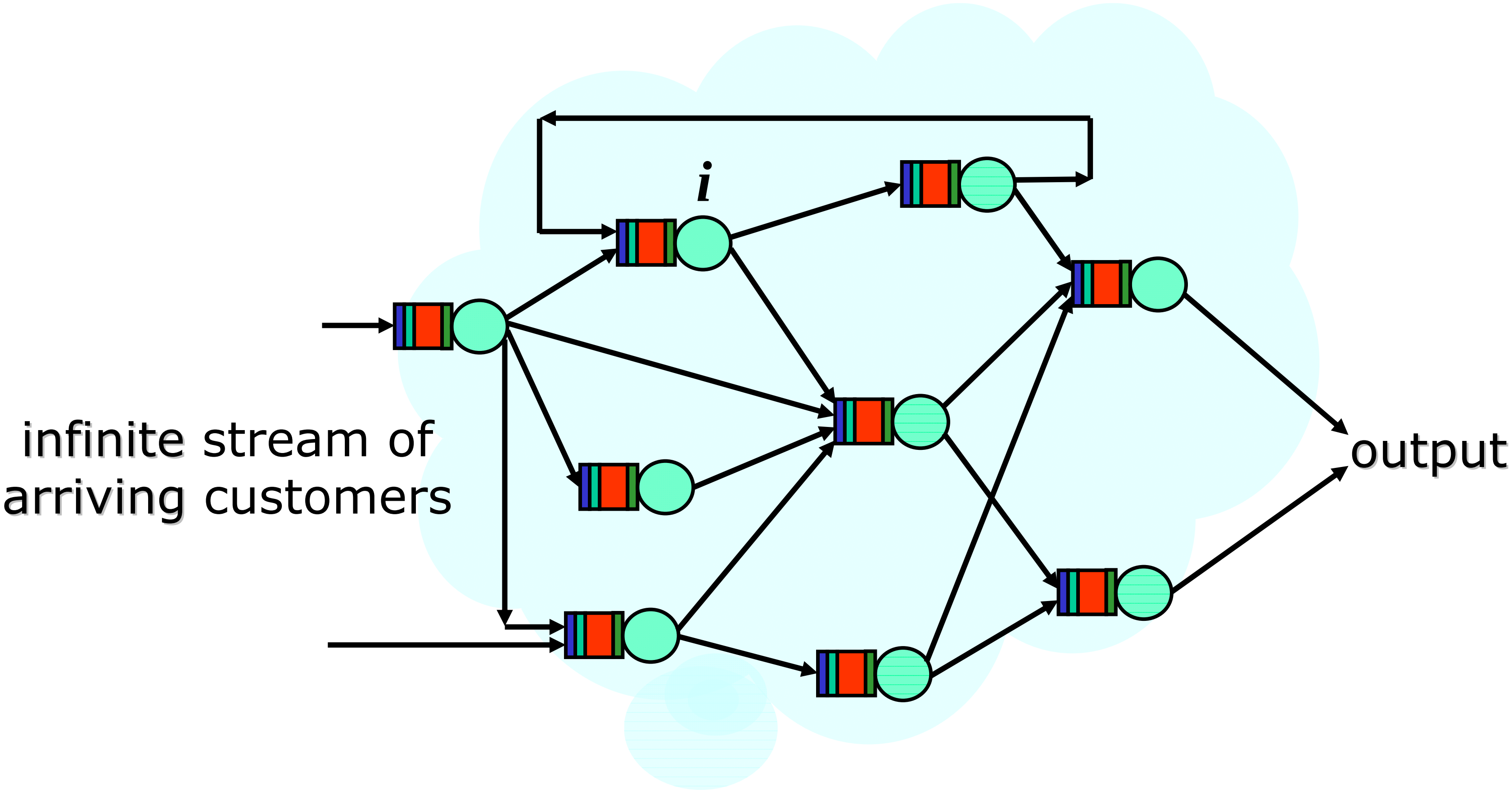


queueing resource



open models
closed models

open model queueing network



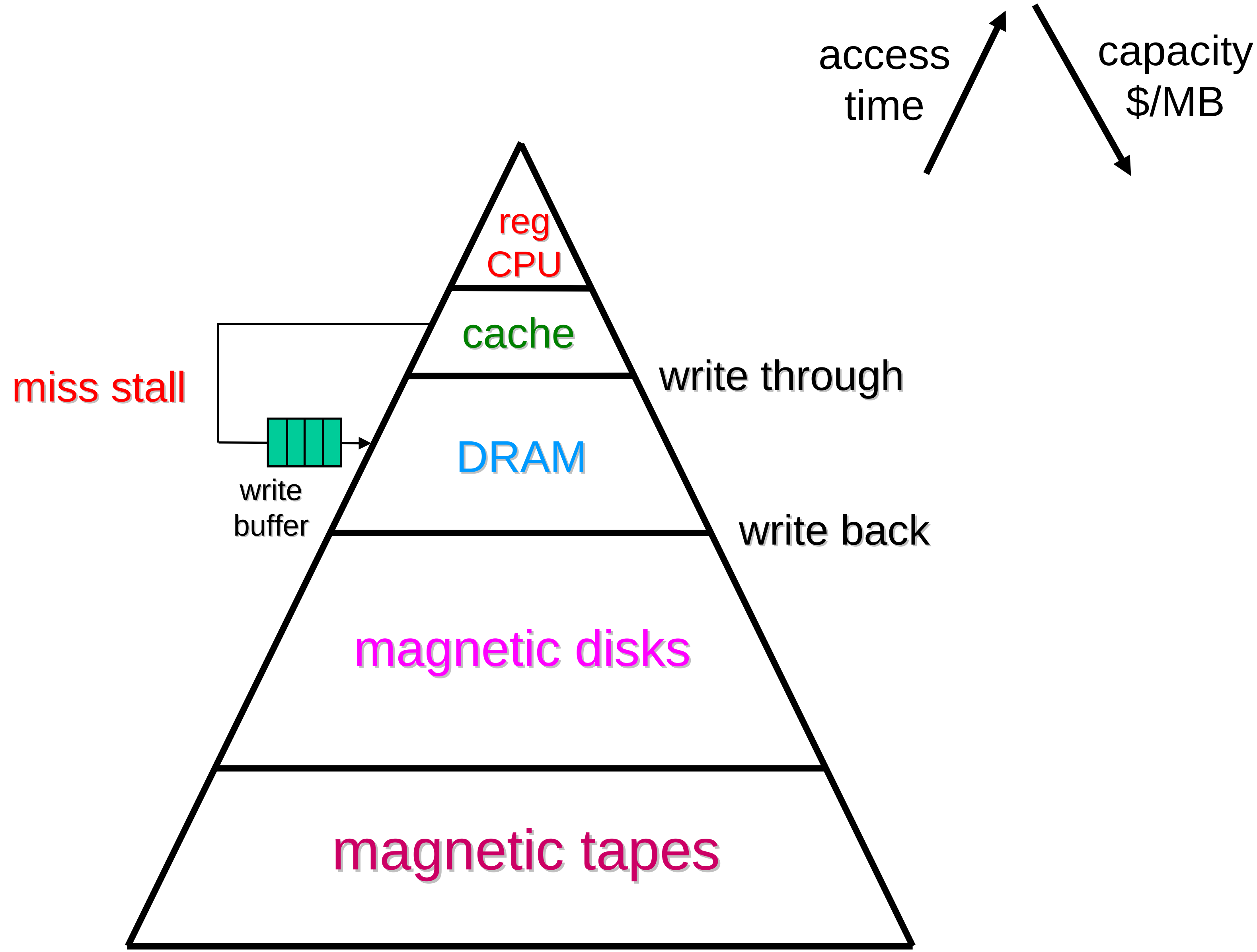
V_i visits to resource i $\quad$ D_i service demand for resource i

$$D_i = V_i\, S_i \qquad \lambda_i = \lambda V_i \qquad U_i = \lambda_i S_i = \lambda D_i$$

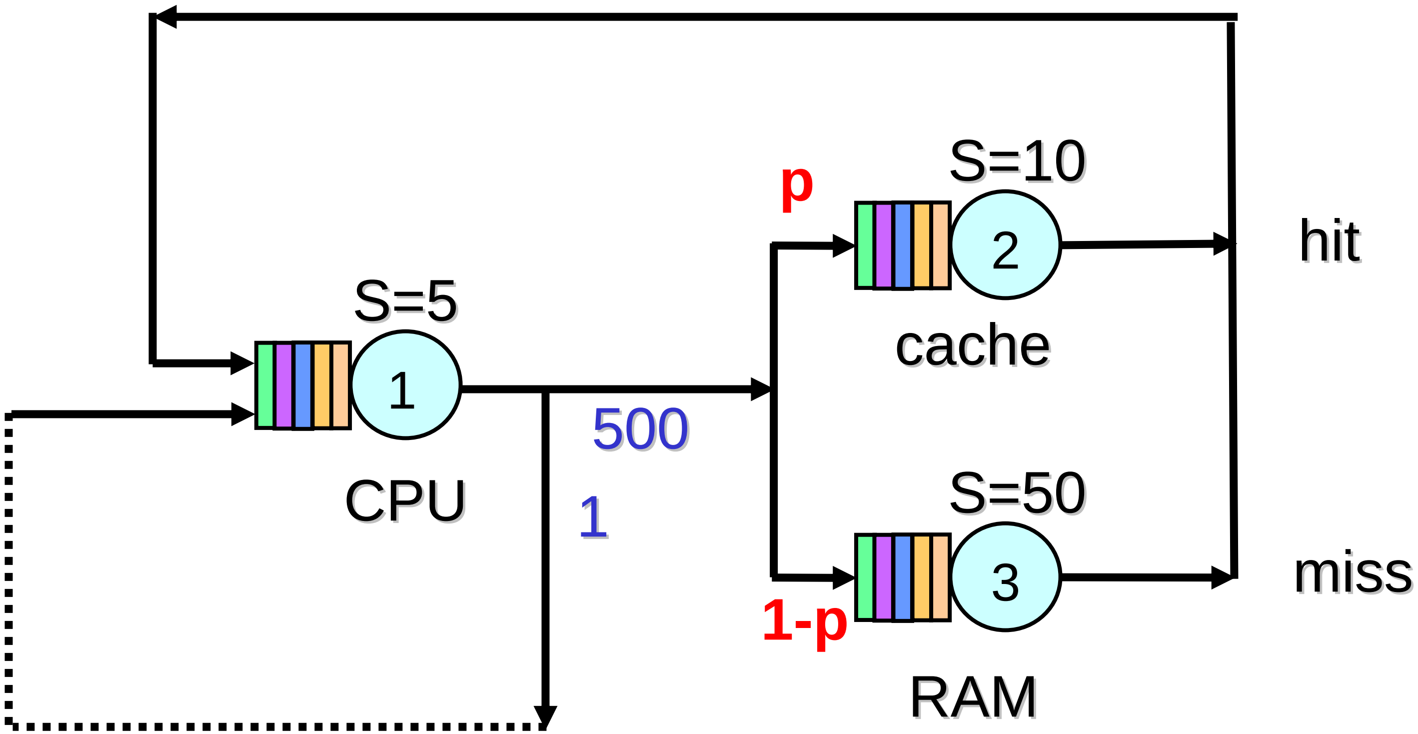
case study: model of cache and DRAM accesses

(write through algorithm)

memory hierarchies (1)



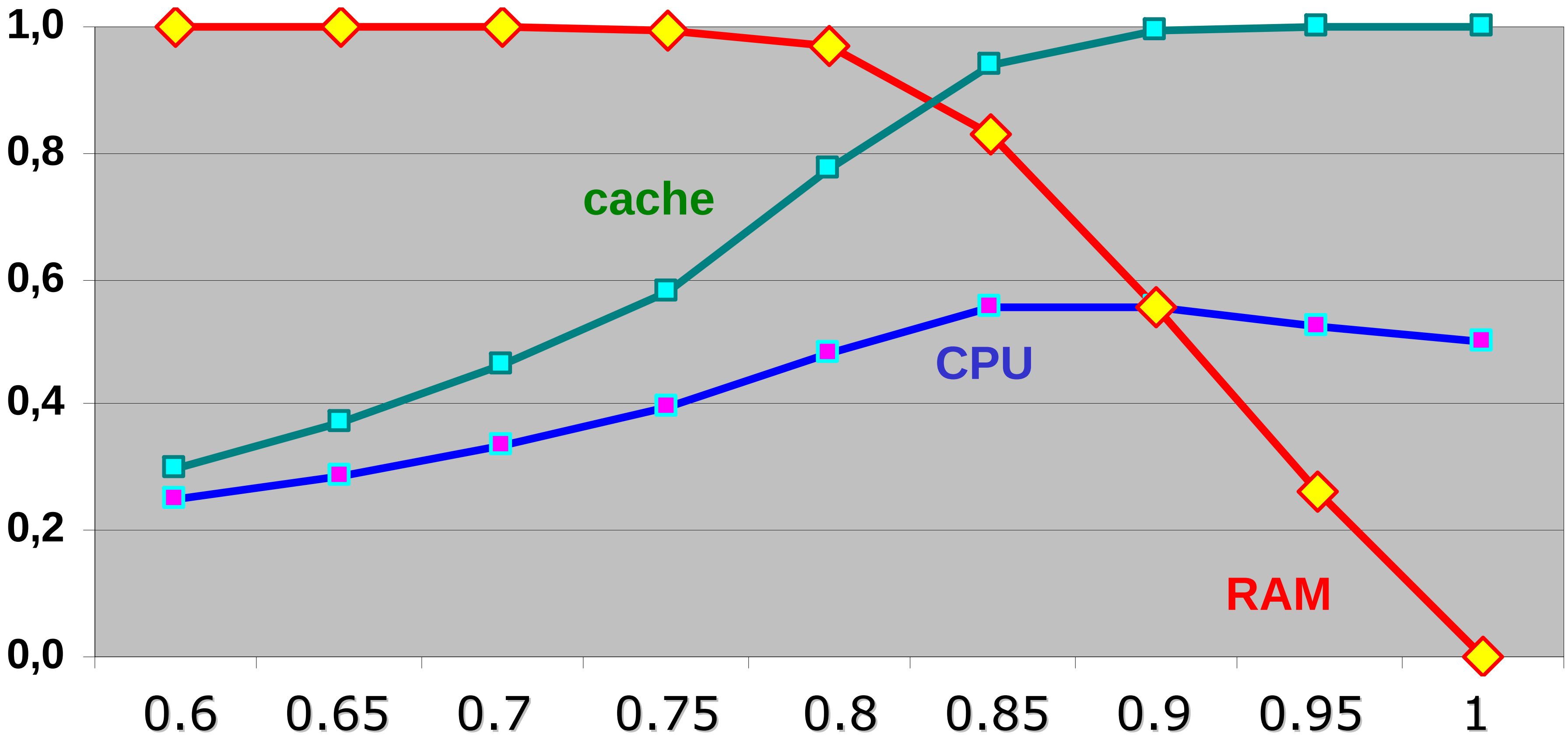
case study: CPU, cache, RAM, write through (2)



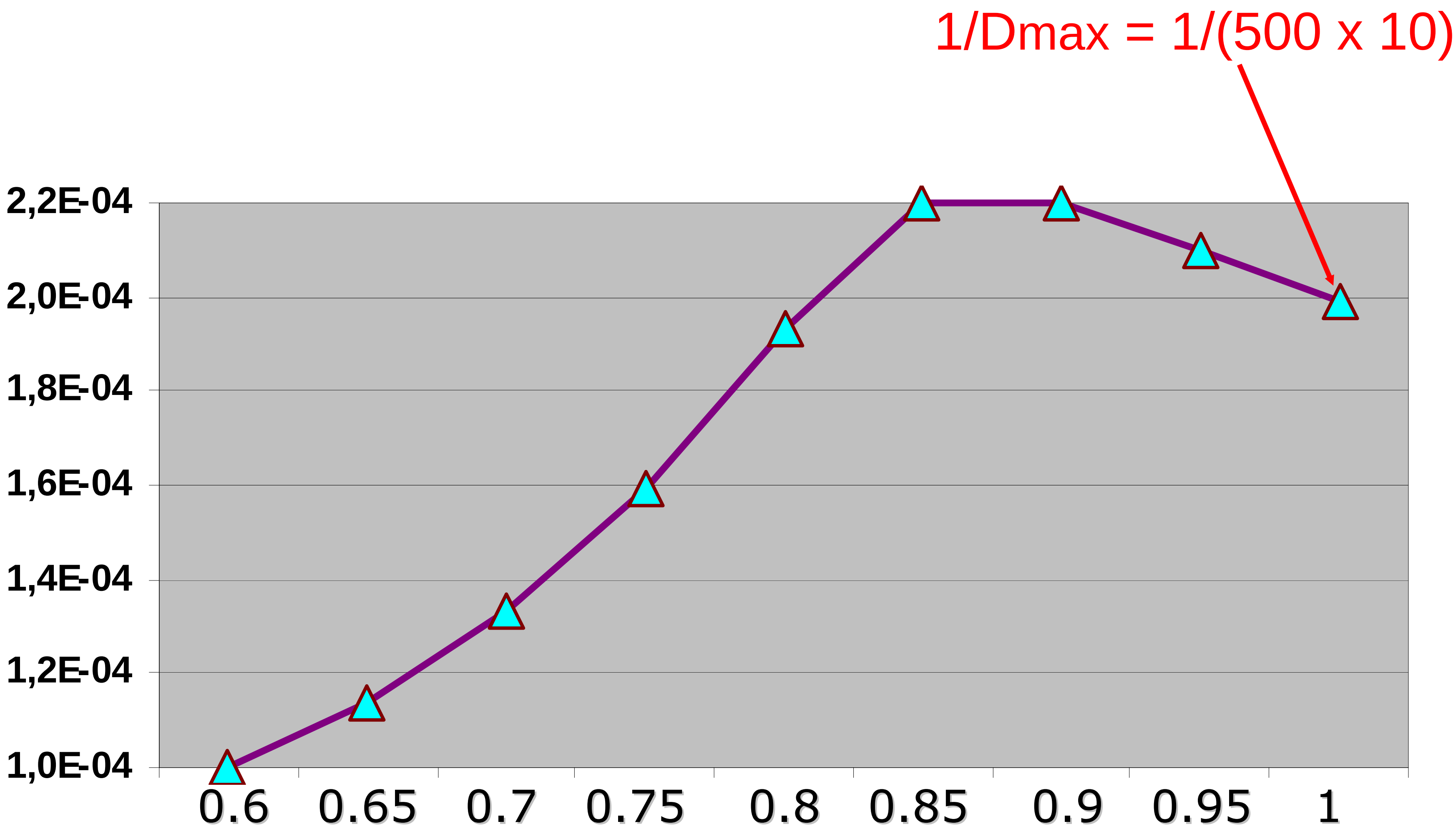
p hit probability = 0.6 ÷ 1

N = 10 threads

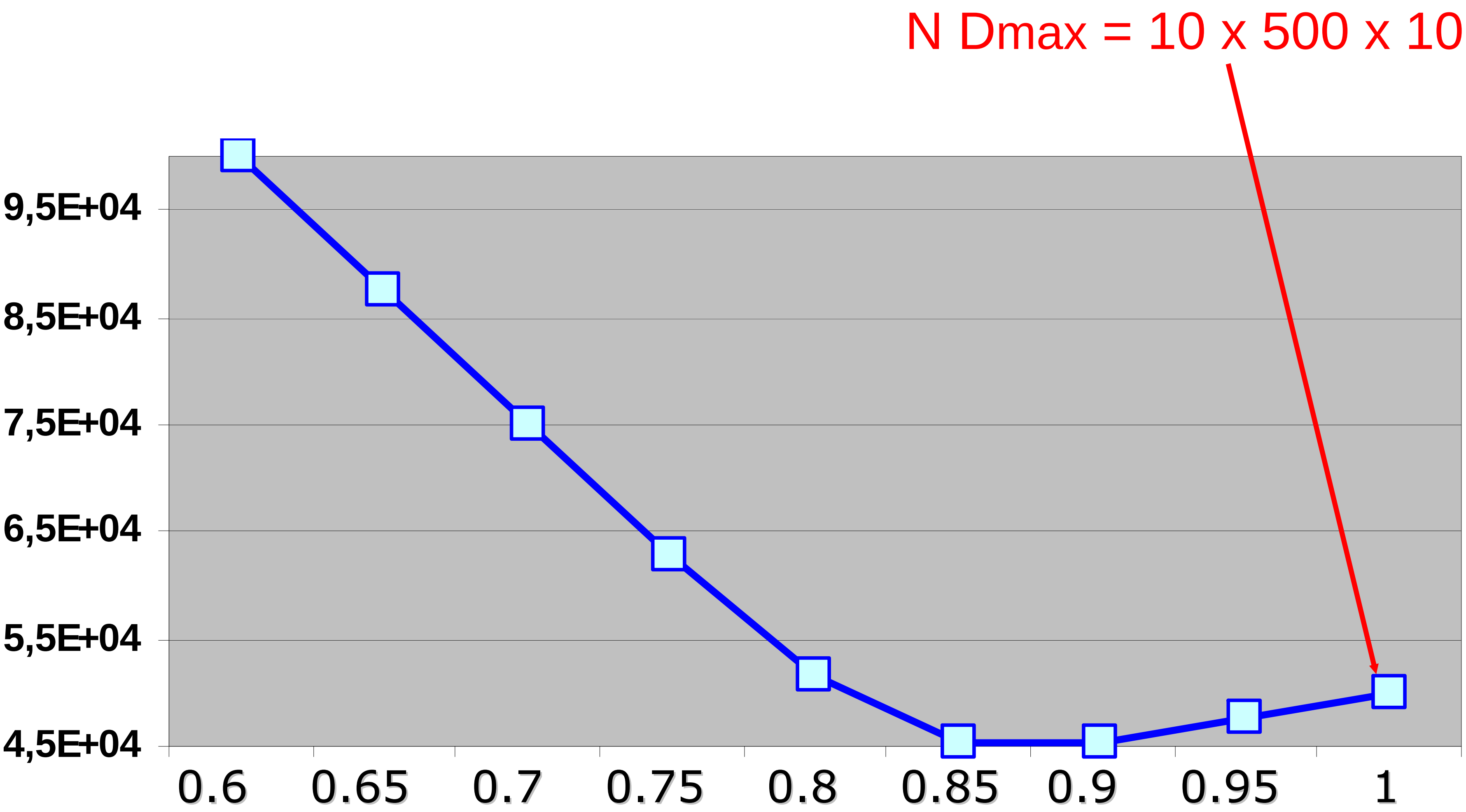
case study: utilization vs hit prob. (3)



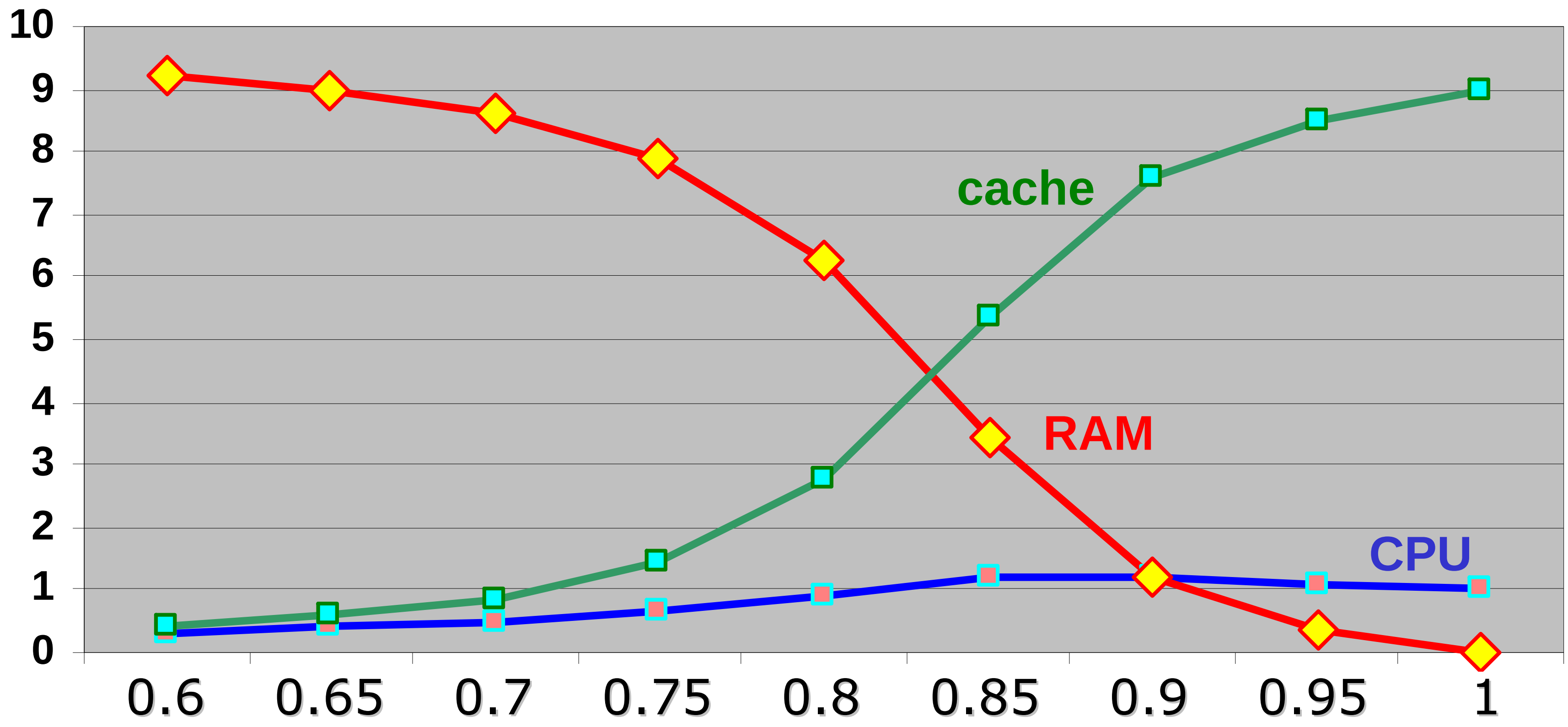
case study: throughput vs hit prob. (4)



case study: response time vs hit prob. (5)



case study: number of jobs vs hit prob. (6)



case study: paging and disk I/O model

(write back algorithm)

principle of locality

programs access a relatively small portion of their address space at any instant of time

- temporal locality (in time): an object (data, instruction) that is referenced will tend to be referenced again soon
- spatial locality (in space): when an object is referenced, objects whose addresses are close by will tend to be referenced soon

locality arises from simple and natural concepts (sequentiality in instructions execution, presence of loops, arrays and other basics data structures accesses)

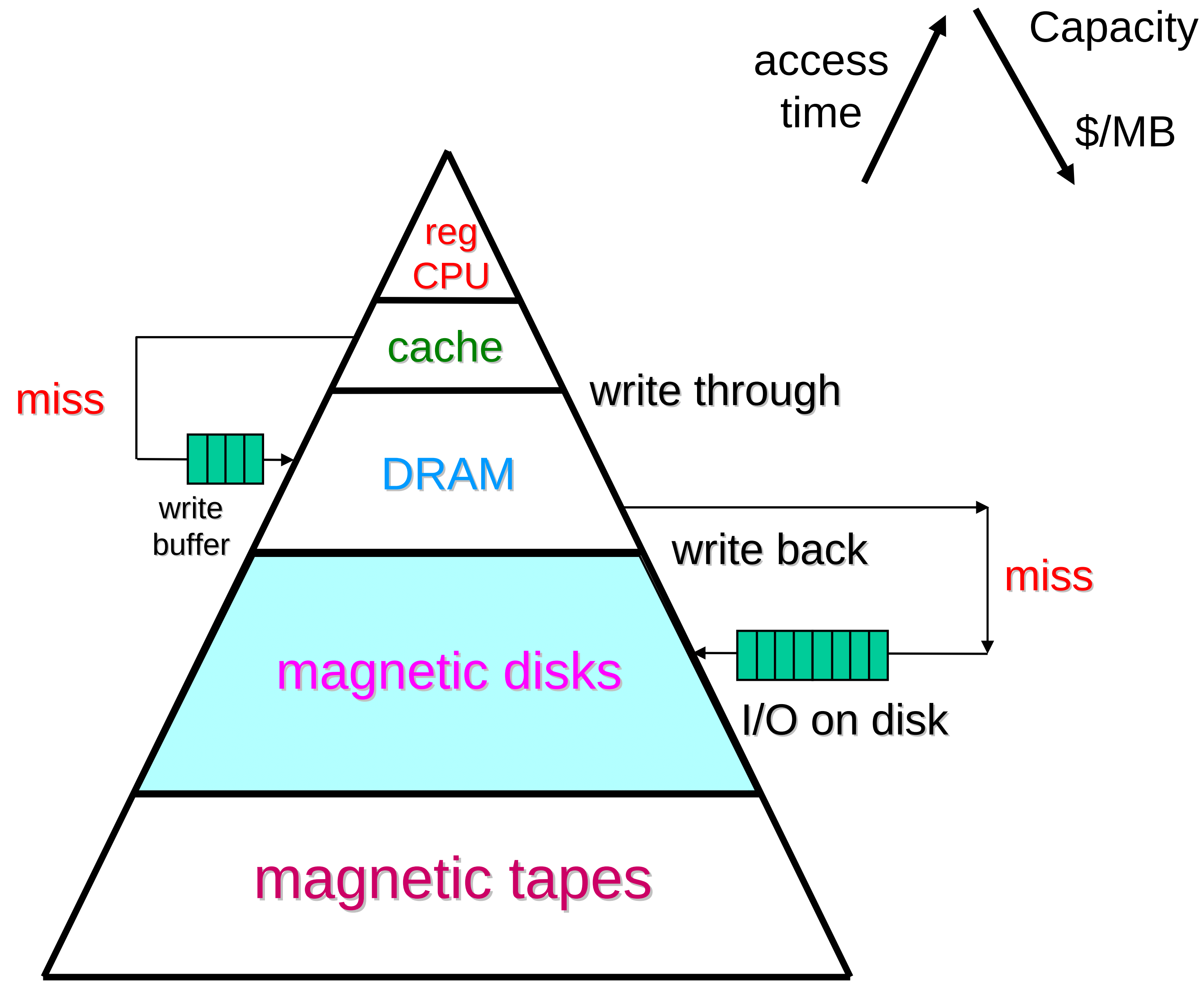
sources of misses in a memory hierarchy

the same ideas carry over directly to any memory level in the hierarchy

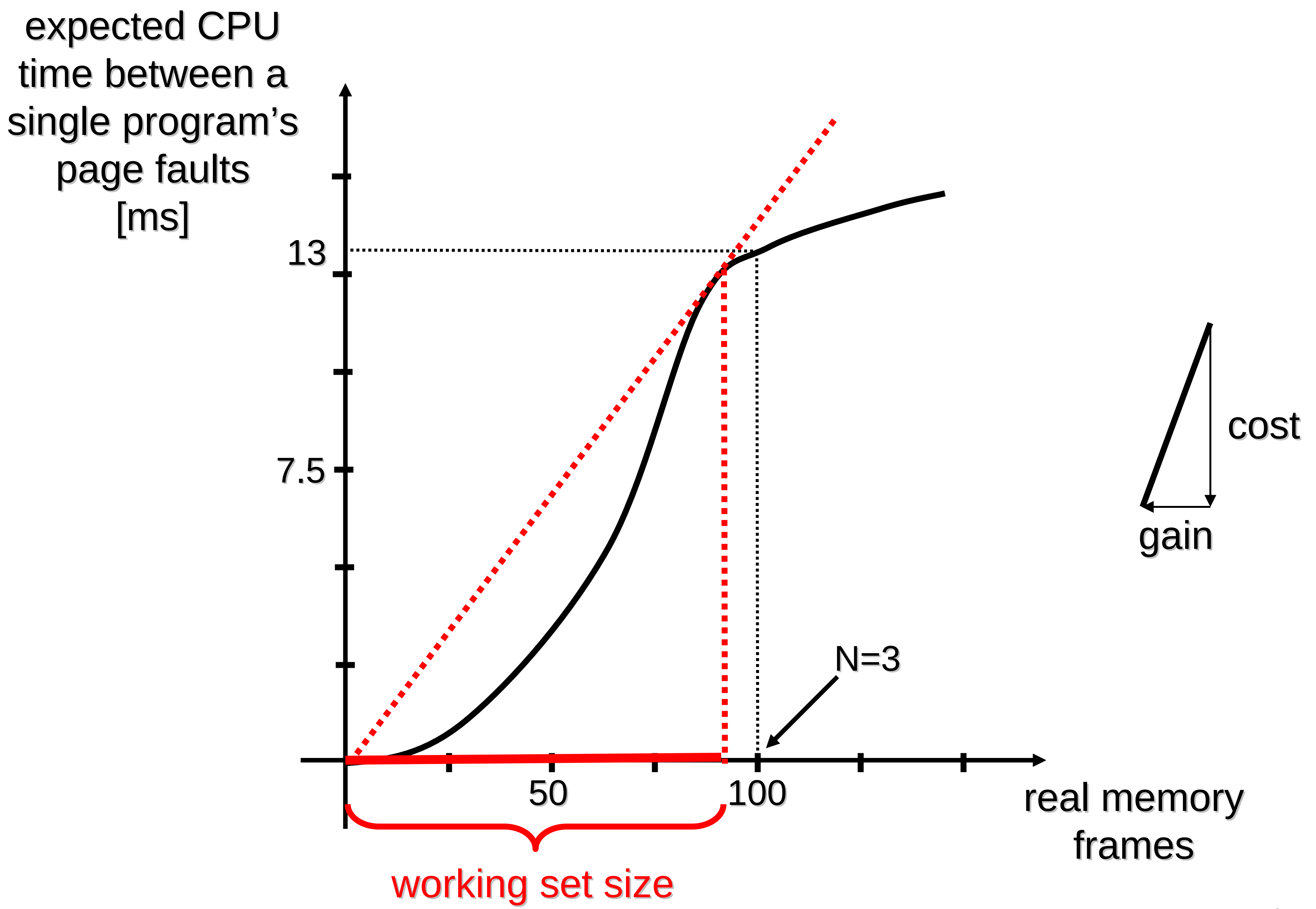
capacity misses: will occur due to blocks being discarded and later retrieved, when cache cannot contain all the blocks of the working set of the program (or at least a significative fraction of it)

the capacity misses represent the largest portion of all the misses (compulsory and conflict misses)

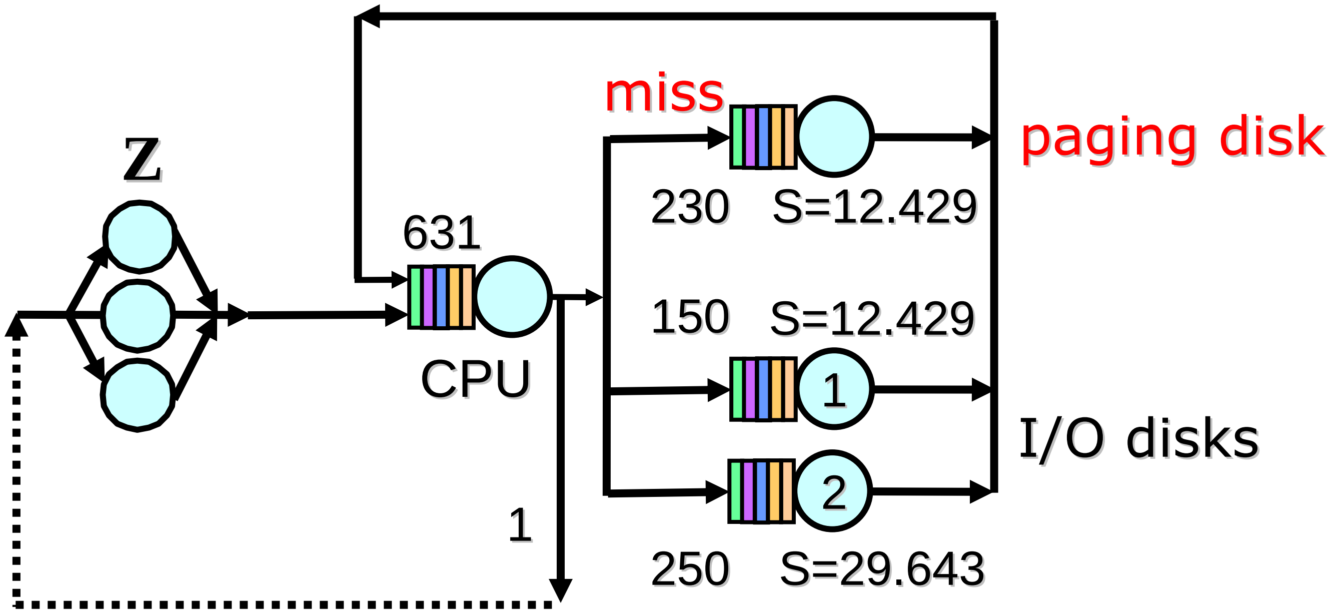
memory hierarchy



lifetime curve - working set size

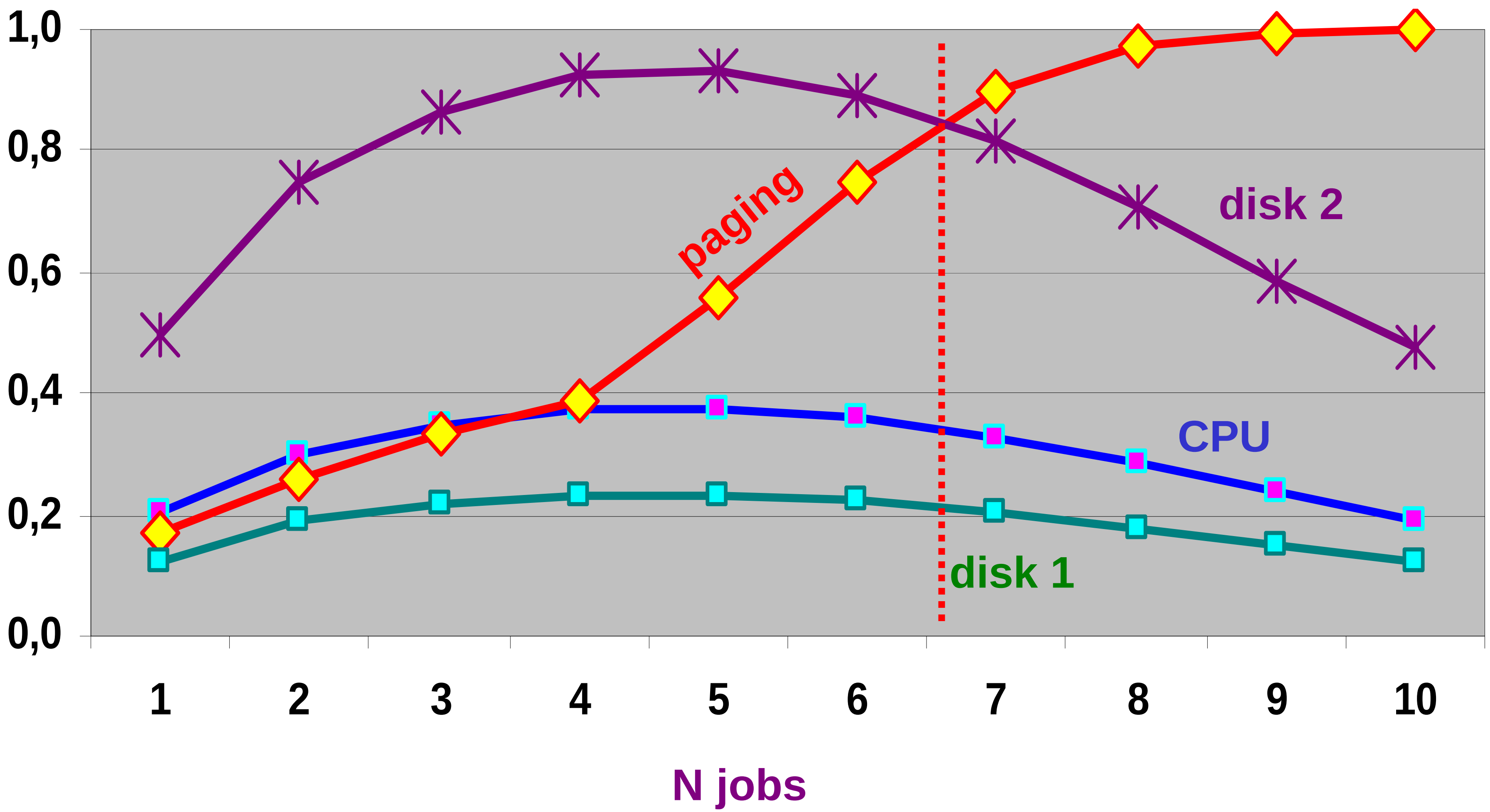


case study: CPU, paging, I/O, write back (1)

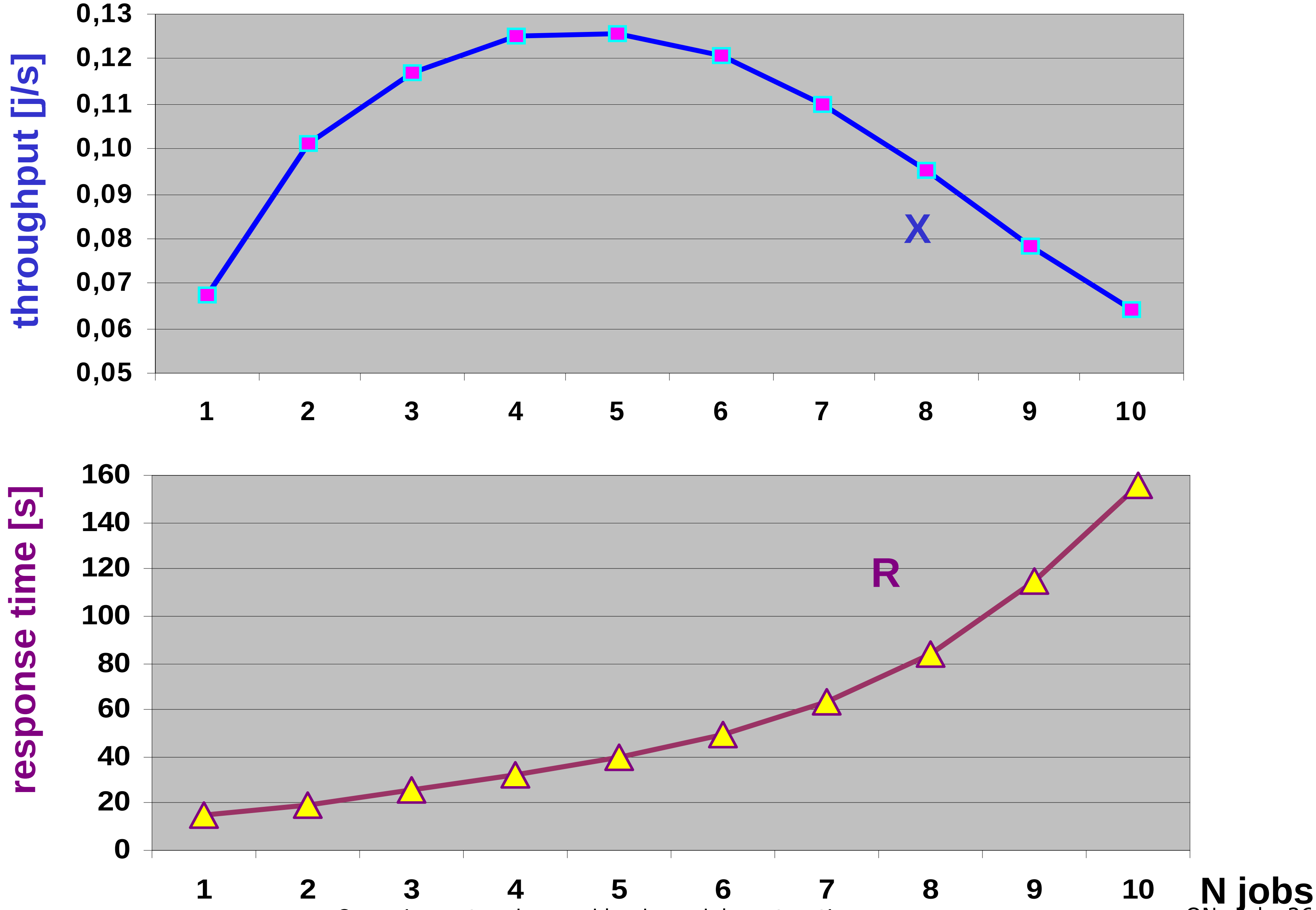


N = 3 jobs

case study: utilization, bottleneck (2)

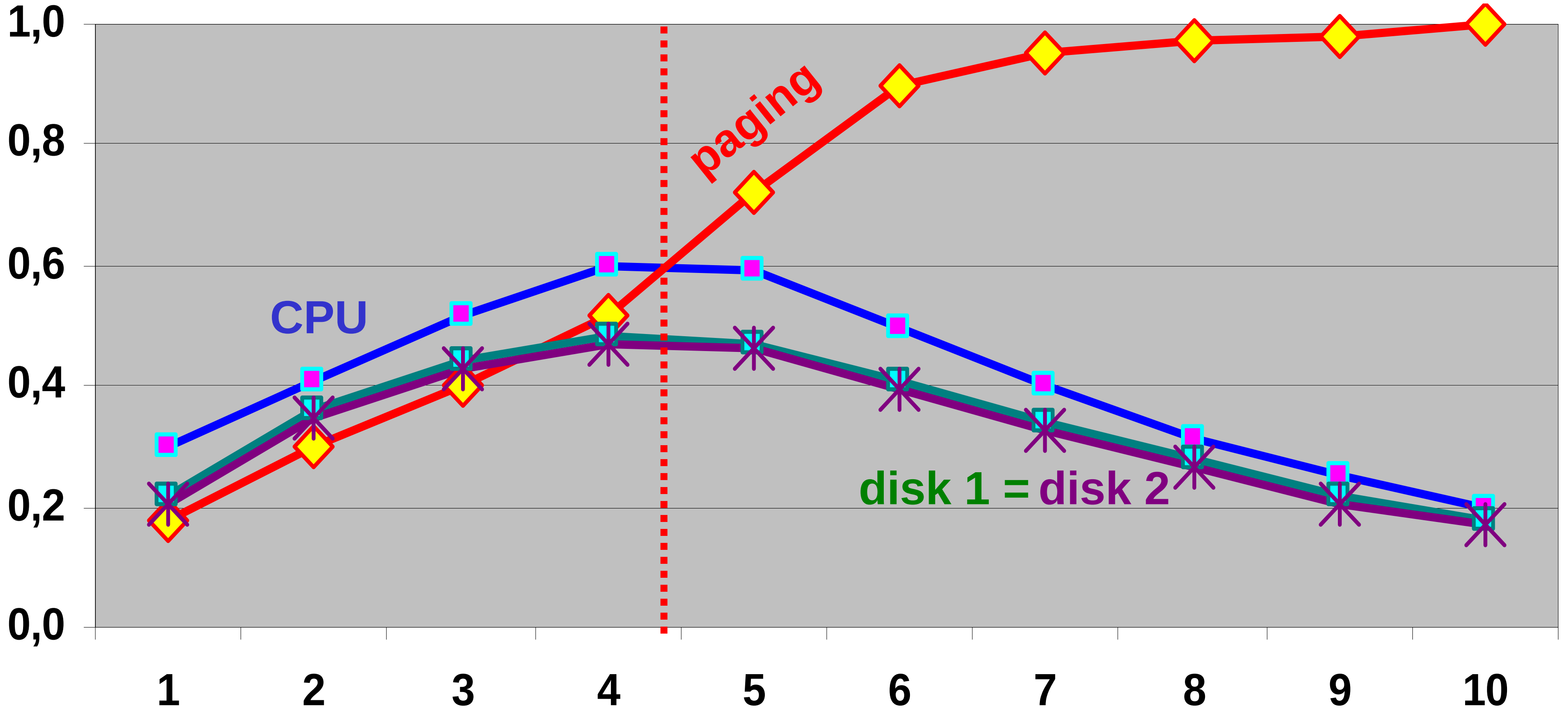


case study: throughput, response time (3)



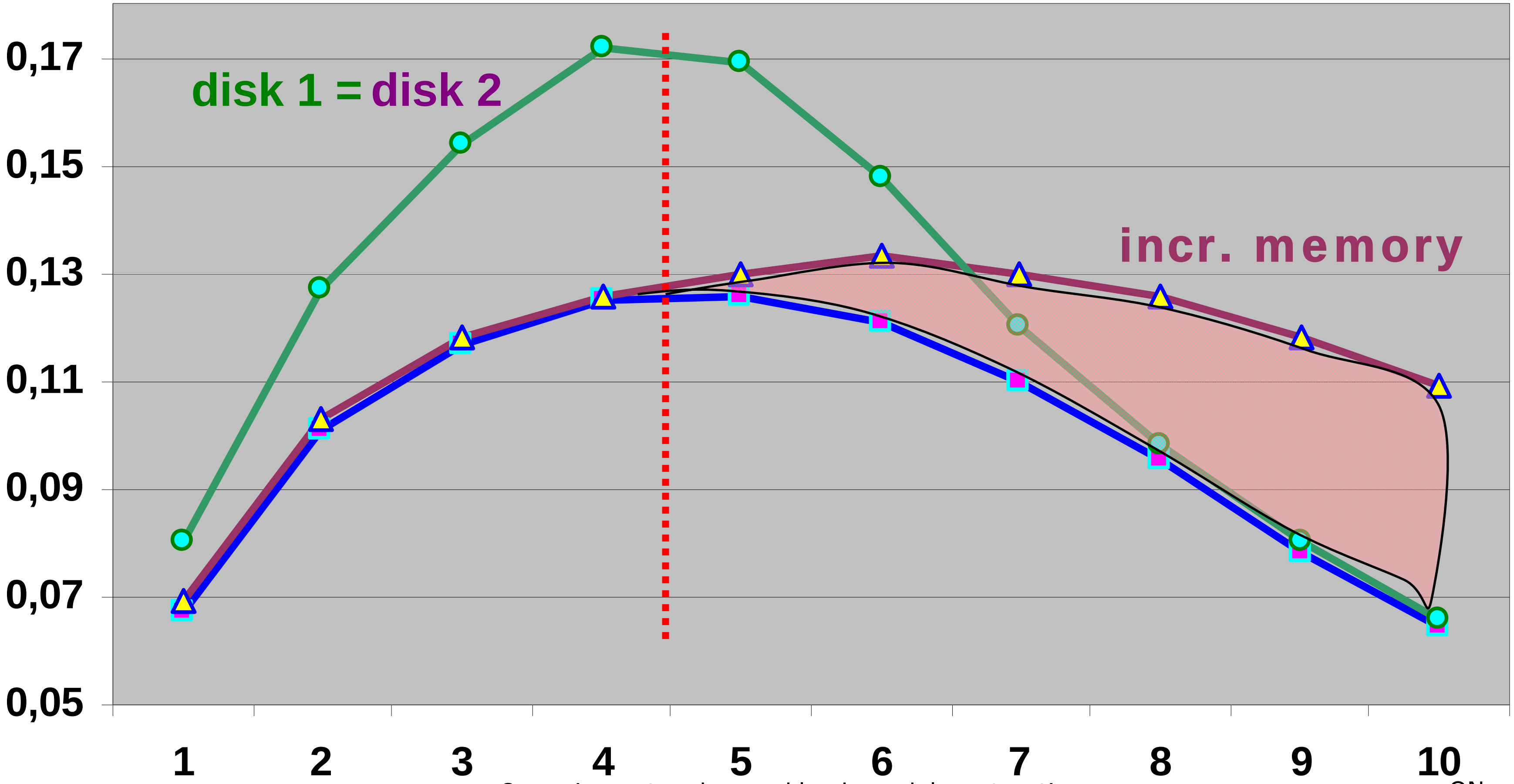
case study 4: utilizations (4)

load balancing of the two I/O disks
visits disk 1 from 150 to 282 - visits disk 2 from 250 to 118



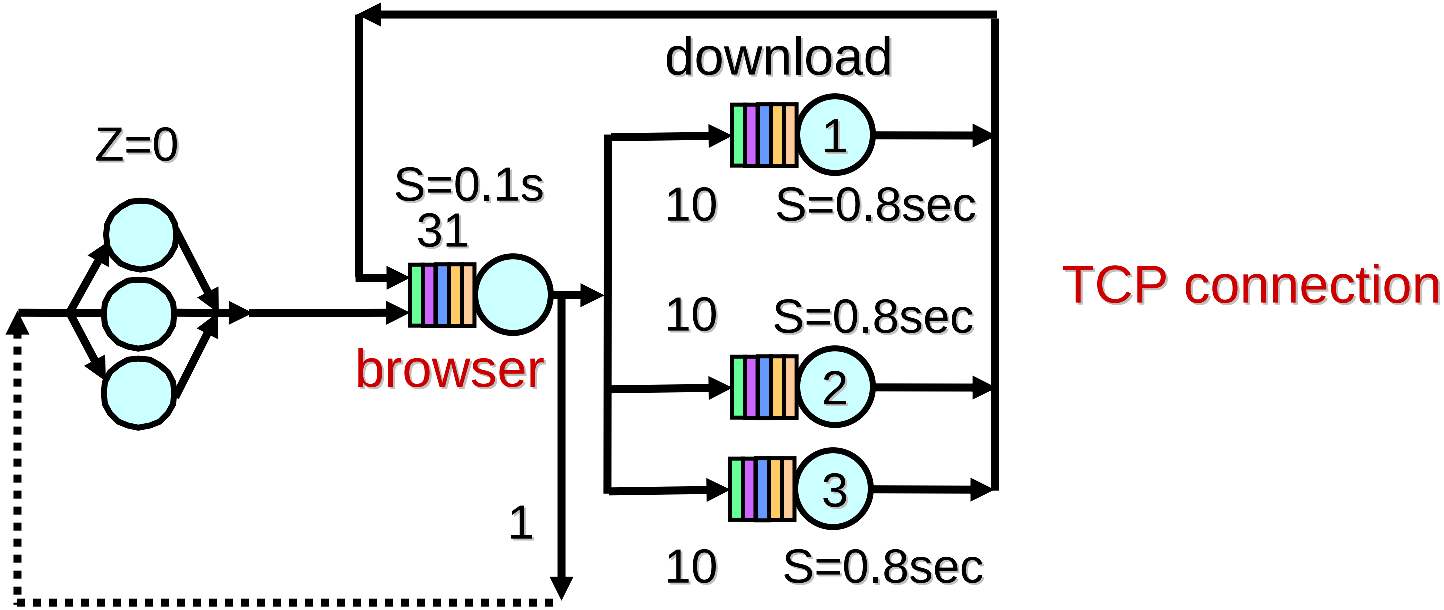
case study: throughput (5)

memory space for user programs from 300 to 400 frames



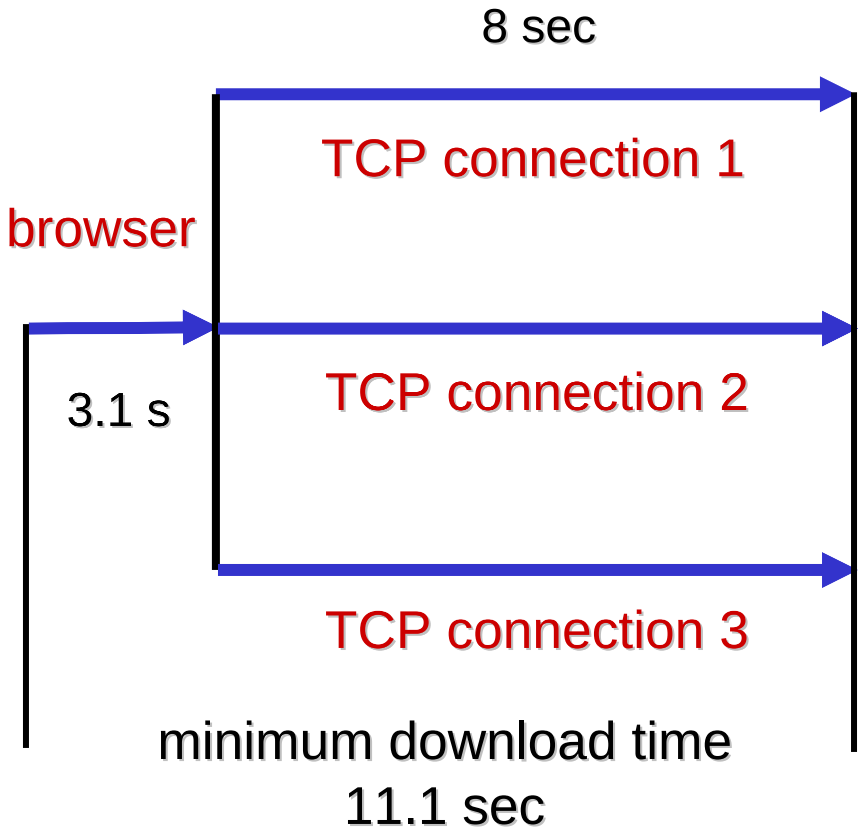
case study:
http download
and
parallel connections

page download, HTTP



N = 3 jobs, 3 parallel connections, 30 objects per page
5 KB avg object size, download bandwidth 50 Kbps

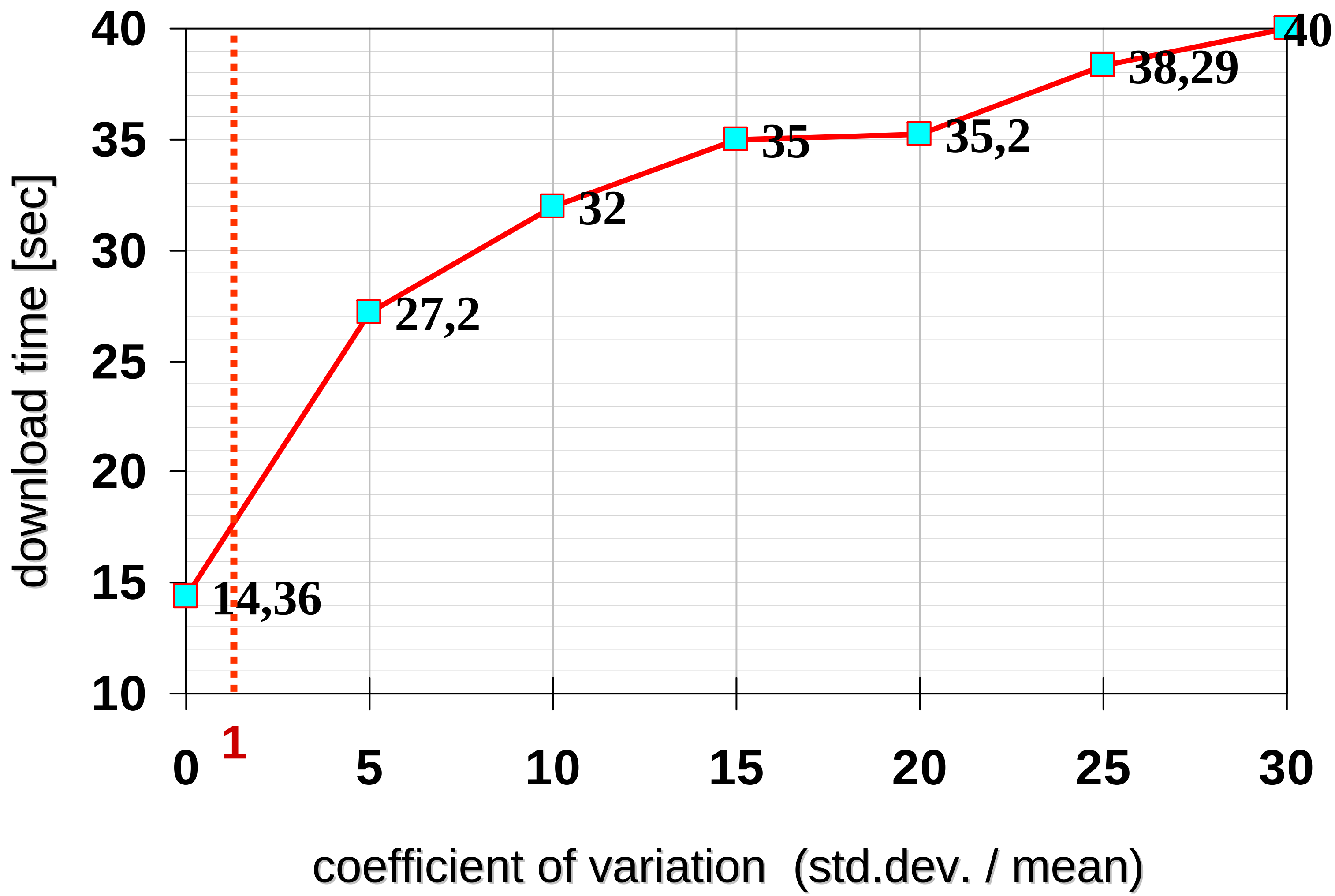
minimum theoretical time



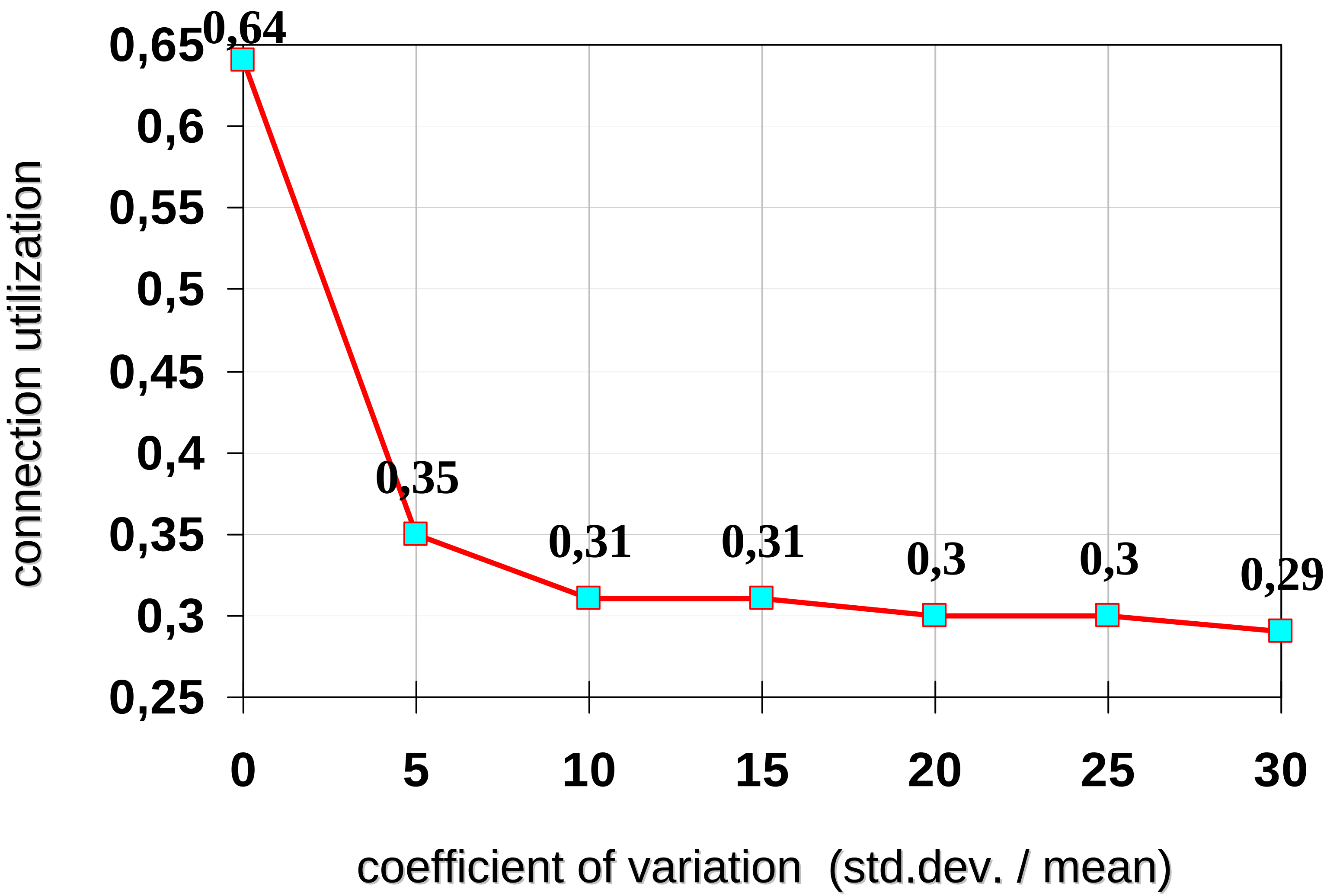
$$R_{min\ \ seq} = (0.8 \times 10) \times 3 + 0.1 \times 31 = 27.1 \text{ sec}$$

$$R_{min\ \ par} = (0.8 \times 10) + 0.1 \times 31 = 11.1 \text{ sec}$$

page download time



connection utilization

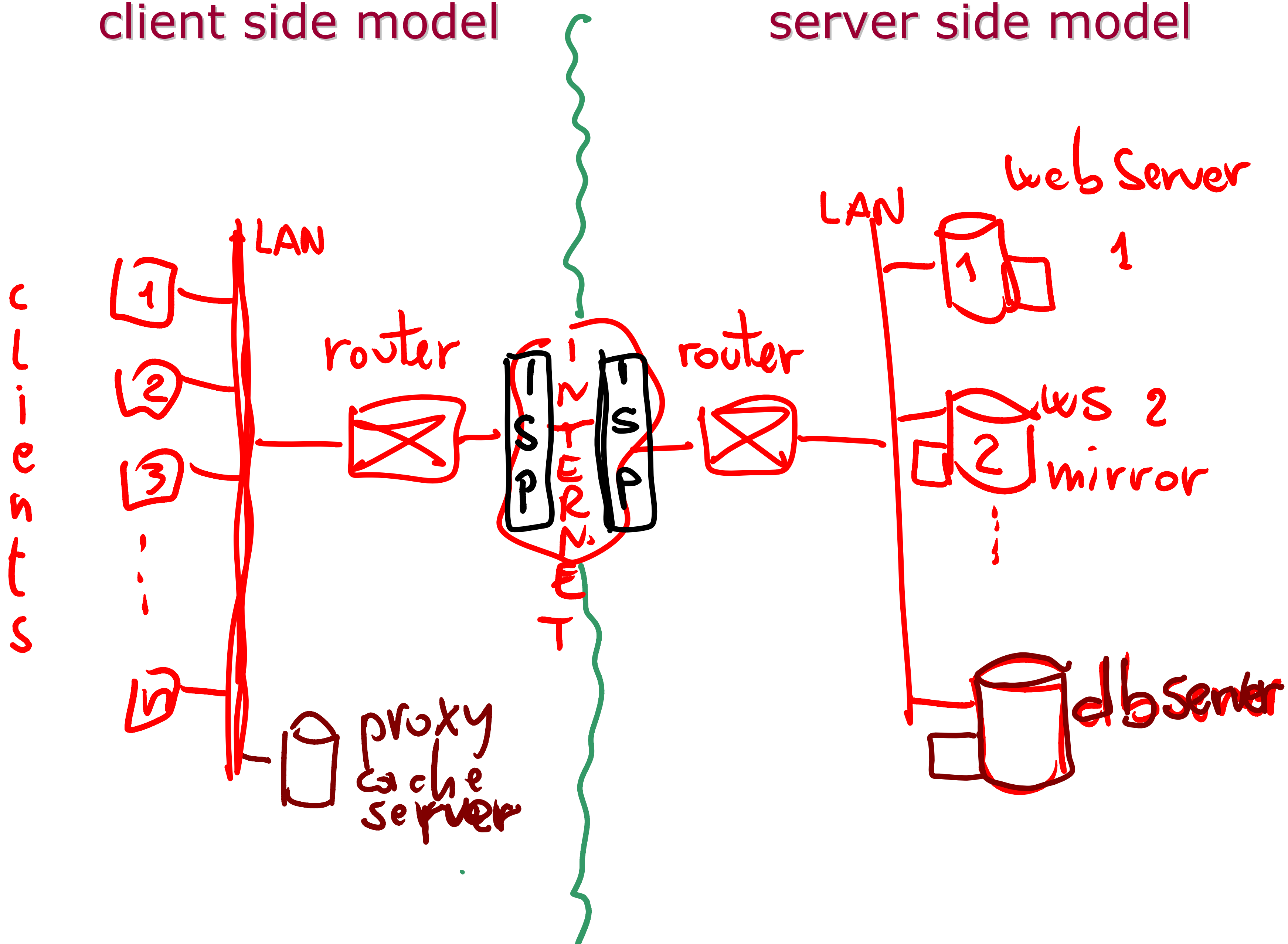


client side and web server side models

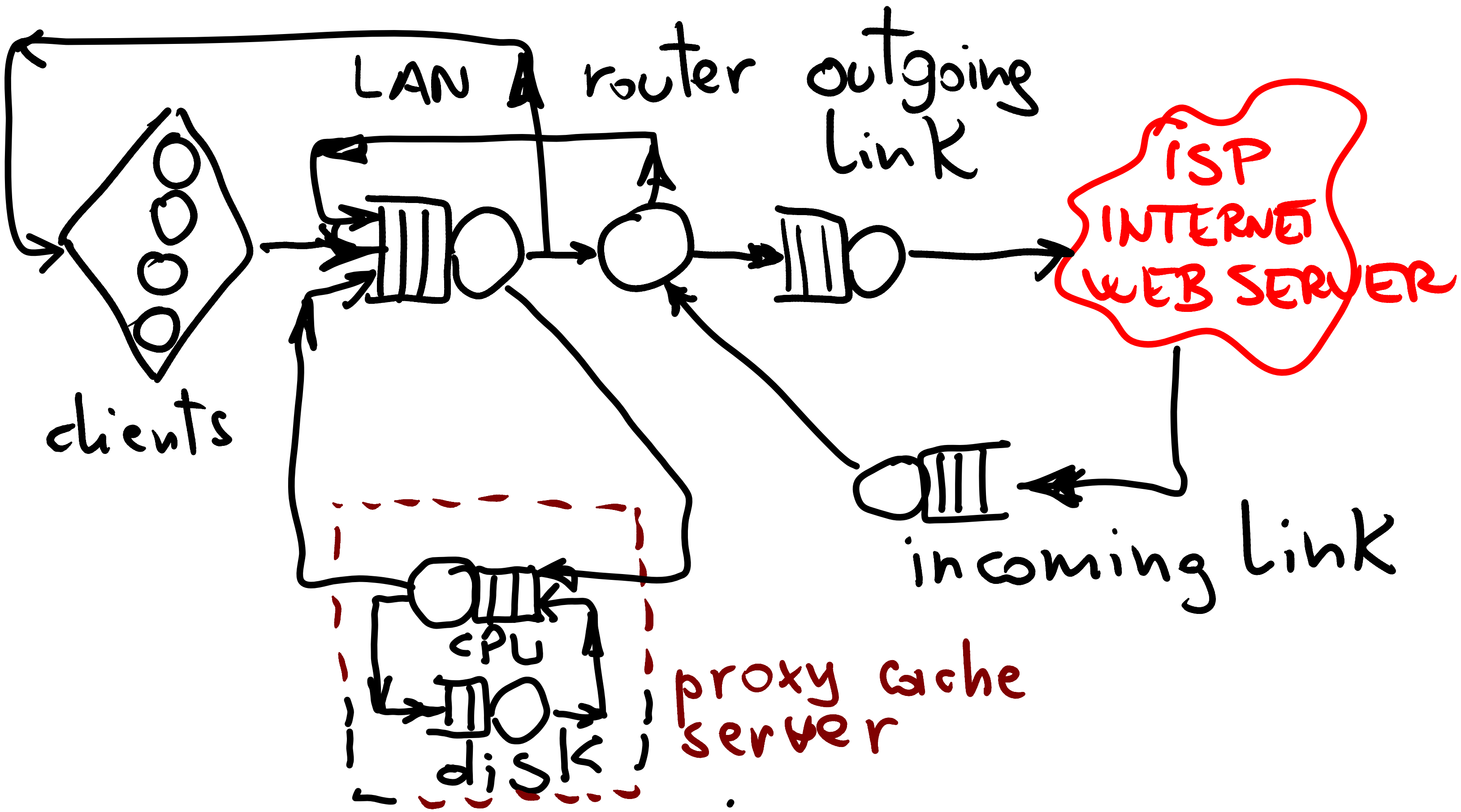
from: D.Menasce, V.Almeida, Capacity Planning for Web Performance, Prentice Hall, 1998

client side model

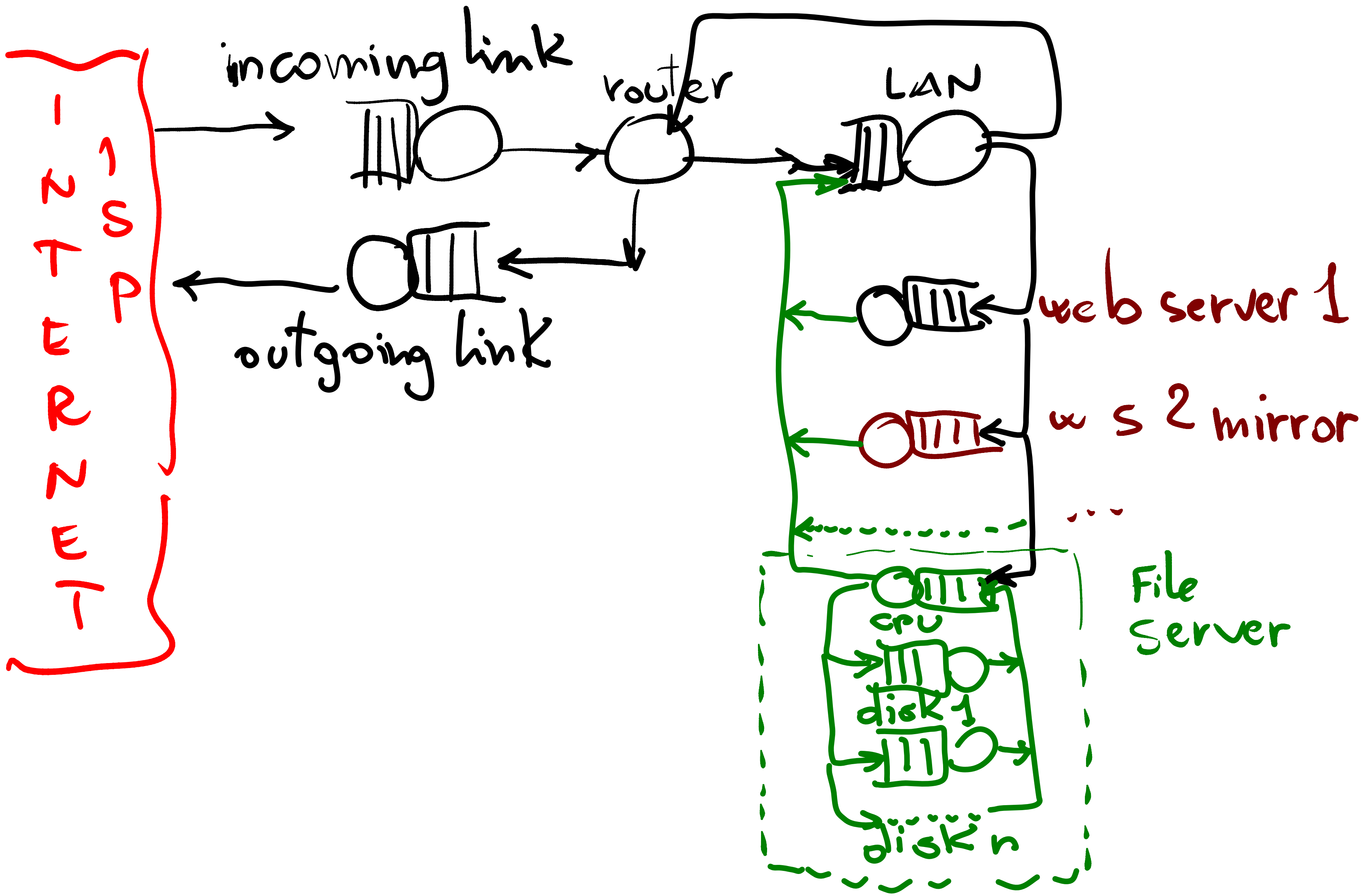
server side model



client side model



server side model



parameters (no proxy server)

- LAN bandwidth = 10 Mbps
- Frame Overhead = 18 Bytes (LAN's link layer protocol)
- Router latency = 50 μsec/packet
- Link bandwidth = 56 Kbps (full duplex, from the router to ISP)
- Internet Delay RTT avg=100ms (89-161 using ping command)
- Internet Data Rate = 20 KB/s (Int. data transfer rate observed from the same remote servers)
- Workload:
  - Browser Rate = 0.3 req/sec (rate of HTTP op/s, inverse of think time Z)
  - Number Clients = 150
  - Percent Active = 0.1 (% of clients actively browsing the web)
  - AvgSizeHTTPRequest = 100 Bytes (req. sent by the browser to the server)

workload: documents requested

- the documents requested from clients are classified into four categories

freq. occurrence %	0.35	0.50	0.14	0.01
avg. size [KB]	0.8	5.5	80	800

- avg. document size

DocSize=0.8x0.35+5.5x0.5+0.14x80+0.01x800=22.23KB

service times in networks

- a message from a client to a server has to go through several protocol layers
- protocol entities at each layer communicate with each other by exchanging PDUs Protocol Data Unit (packet) composed of a header and a data area
- PDUs receive different names for different protocols and usually have a maximum size for the data area (MTU Maximum Transmission Unit)
- routers have to be able to fragment datagrams as they go through networks of decreasing MTUs

frame header	IP header	TCP header	client request	frame trailer

interaction over TCP connection

characteristics of network protocols

Protocol	PDU name	max PDU size (B)	Overhead (B)	Maximum Trasmission Unit max data (B)
TCP	segment	65,535	20	65,515
IP v4	datagram	65,535	20	65,515
IP v6	datagram	65,535	40	65,495
Ethernet IEEE 802.3	frame	1,518	18 21	1,500 1,497
FDDI (RFC 1390)	frame	4,500	28	4,472

service times in networks

- number of datagrams needed to send an m-Bytes long message

$$N\,datagrams\,(m) = \left\lceil \frac{m + TCP\,ovhd}{\min MTU - IP\,ovhd} \right\rceil = \left\lceil \frac{m + 20}{\min MTU - 20} \right\rceil$$

- the total protocol TCP+IP overhead for a m-Bytes message is

$$\begin{aligned} Overhead\,(m) &= TCP\,ov + N\,datagrams\,(m)\ x\ (IP\,ov + Frame\,ov) \\ &= 20 + N\,datagrams\,(m)\ x\ (20 + Frame\,ov) \end{aligned}$$

- the service time of network for a m-Bytes message

$$Network\ Service\ Time\,(m) = \frac{8bits\ x\,[m + Overhead\,(m)]}{10^6\ x\ LAN\,bandwidth}$$

service times at routers

- datagrams incoming are queued up until the router processor is available to inspect the packet
- the datagram’s destination address is used to determine the next best outgoing link, based on routing tables at the router
- time taken by a router to process a datagram: router latency (e.g., 50-130 microseconds per packet)

$$Router\,Service\,Time = Ndatagrams \;\; x \;\; router\,latency$$

service demands client side

$$D_{cl} = \frac{1}{Browser\,rate} = \frac{1}{0.3} = 3.333\text{sec}$$

$$D_{LAN} = browser\ to\ server\ +\ server\ to\ browser\ =$$

$$= S_{Netw}(AvgHTTP\,req) + S_{Netw}(1{,}024\,x\,DocSize) =$$

$$= S_{Netw}(100) + S_{Netw}(1{,}024\,x\,22.23) = 0.01884\text{sec}$$

$$D_{router} = [N\,datagrams\ (22.23x1{,}024) + 7]\,x\,50\,(micros)\,x10^{-6} =$$

$$= 0.00115\text{sec}$$

$$7 = 3\,open\ TCP\,connect + 1\,HTTP\ request + 3\,close\ connect$$

service demands client side

$$D_{outLink} = \frac{8\,[avg\ HTTP\ req + 5\ (TCP\ ov + IP\ ov)]}{1{,}024\ link\ bandwidth} =$$

$$= \frac{8\,[100 + 5\,x\,(20 + 20)]}{1{,}024\,x\,56} = 0.04185\,\text{sec}$$

$$5\ synchronization\ segments\ through\ outgoing\ link =$$
$$= 2\ open\ TCPconnection + 1\ HTTP\ request + 2\ close\ connection$$

$$D_{Internet} = 1.5\ (Int.RTT/1000) + Doc.Size/InternetDataRate =$$

$$= 1.5\ \ (100/1000) + 22.23/20 = 1.2615\,\text{sec}$$

$$1.5 = 1\ RTT\ for\ TCP\ connect + 0.5\ RTT\ for\ HTTP\ req$$

service demands client side

$$D_{incLink} = \frac{8\,x\,(1{,}024\,x\,Doc\,Size + Link\,ov)}{1{,}024\,x\,link\,bandwidth}$$

$$Link\,overhead = \left\lceil \frac{1{,}024\,x\,DocSize}{65{,}535} \right\rceil x\,(TCP\,ov + IP\,ov)$$

$$65{,}535 = \max PDU\ \ size\ in\ Bytes\ \ for\ TCP$$

$$D_{incLink} = \frac{8\,x\,(22.23\,x\,1{,}024 + \lceil (22.23\,x\,1{,}024)/65{,}535 \rceil x\,40)}{56\,x\,1{,}024} =$$

$$= 3.18129\,\sec$$

results for client side model

- effective number of requests in the sytem = 150 x 0.1 (% active) = 15
- the bottleneck is the link to the Internet
- R=44.7sec   X=0.3121req/sec (55.5 Kbps)
- 97% of the response time (43.4sec) is spent in the incoming link, that is utilized at 99.29%

- use a faster connection to the Internet
- decrease the demand on the incoming link by using a cache proxy server

resource saturation (bottleneck)

bottleneck

the resource (hw or sw) that determine (and limit) the performance of the system

⇓

very long queue of requests waiting to be serviced in front of it, shorter queue on the other resources

low performances
low throughput, high response time

arrival rate of requests >> max output rate

sources of contention

resource contention is due to two factors:

- workload characteristics
  - routing of requests among resources, visits **V**
  - type of operations executed at each visit
  - traffic fluctuations (arriving requests)
- resource characteristic
  - average resource service time **S**

$$D_{\max} = \max_i V_i S_i = \max_i D_i$$

bottleneck

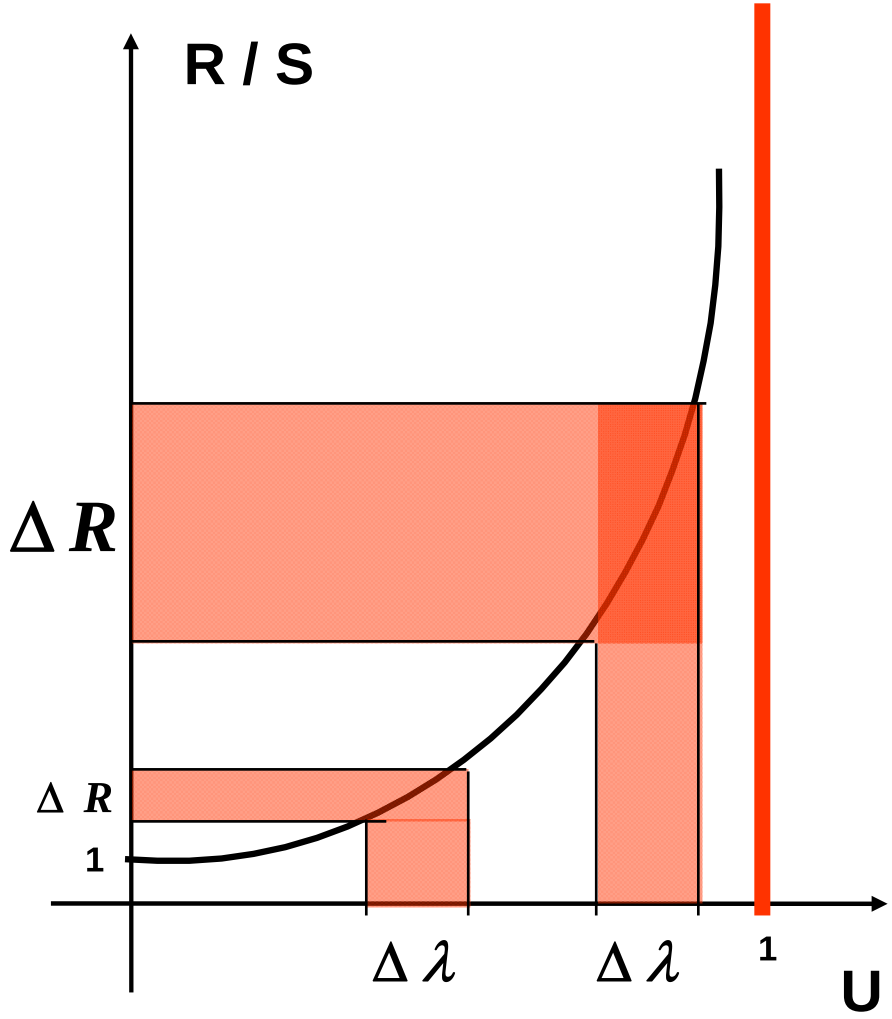
saturation asymptote

$$\Delta\lambda \Rightarrow \Delta R = \frac{S^2}{(1-U)^2}\Delta\lambda$$

$$U = 0.2 \Rightarrow \Delta R = 1.56\ S^2\Delta\lambda$$

$$U = 0.5 \Rightarrow \Delta R = 4\ S^2\Delta\lambda$$

$$U = 0.9 \Rightarrow \Delta R = 100\ S^2\Delta\lambda$$



throughput asymptotes

$$\lambda_{sat} \leftrightarrow U_{\max} = 1$$

$$U_i = X\, V_i\, S_i = X\, D_i$$

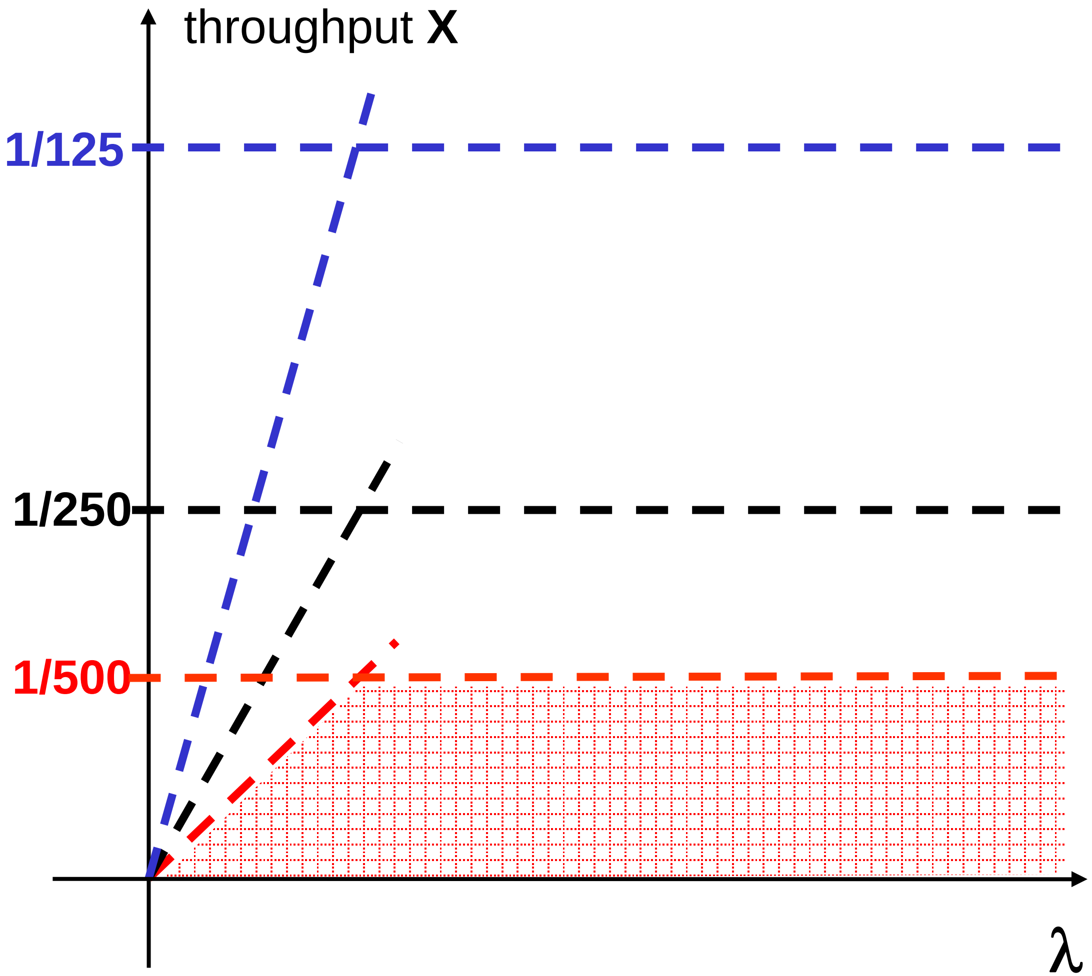
$$U_{CPU} = X \;\; 125\, ms$$

$$U_{IO1} = X \;\; 250\, ms$$

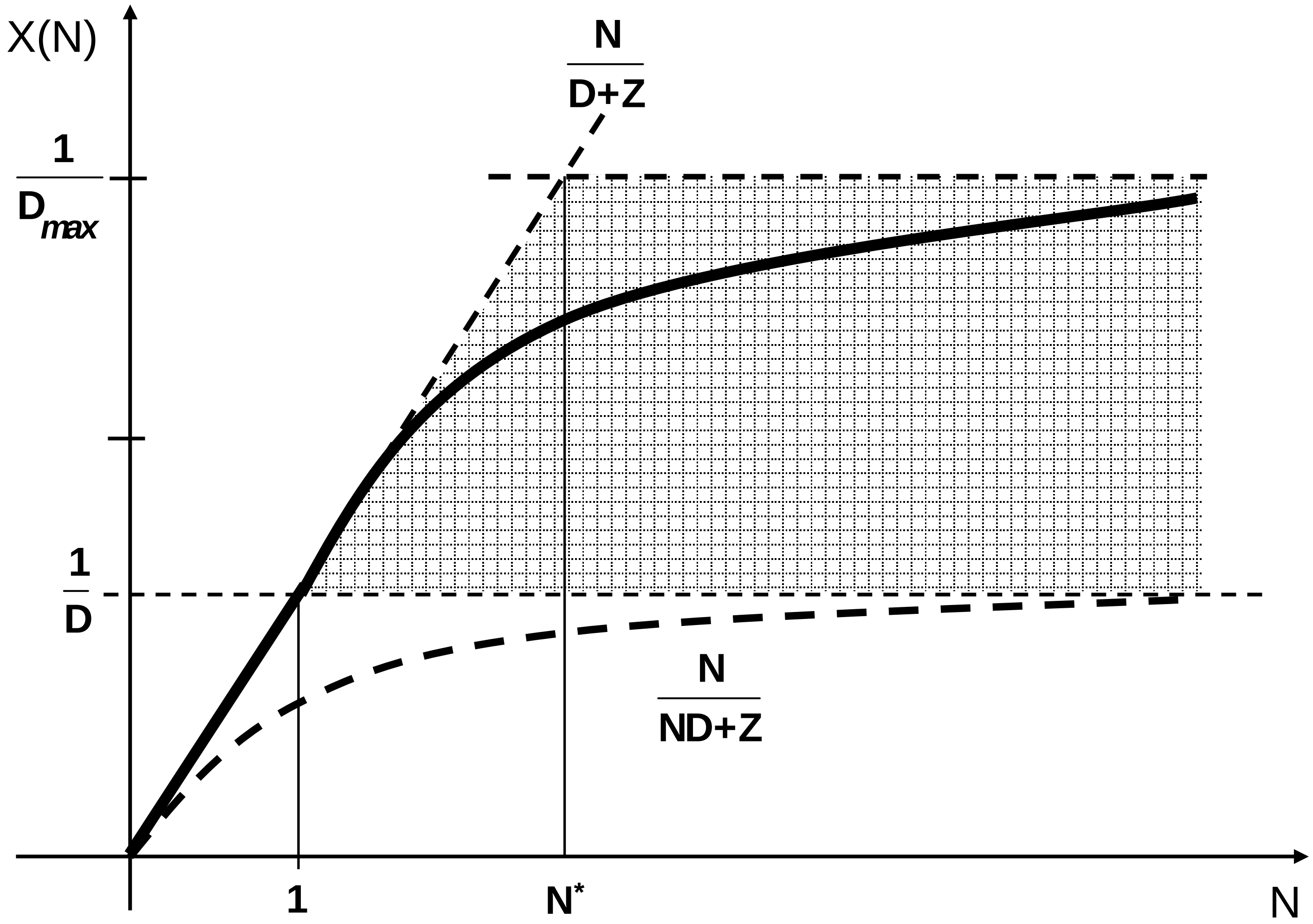
$$U_{IO2} = X \;\; 500\, ms$$

$$U_{\max} = X\, D_{\max} \qquad X \le \frac{1}{D_{\max}}$$

$$X_{\max} = \lambda_{sat} = \frac{1}{D_{\max}}$$



bound of throughput



response time asymptotes

$$N = X\ R \leftrightarrow R = \frac{N}{X}$$

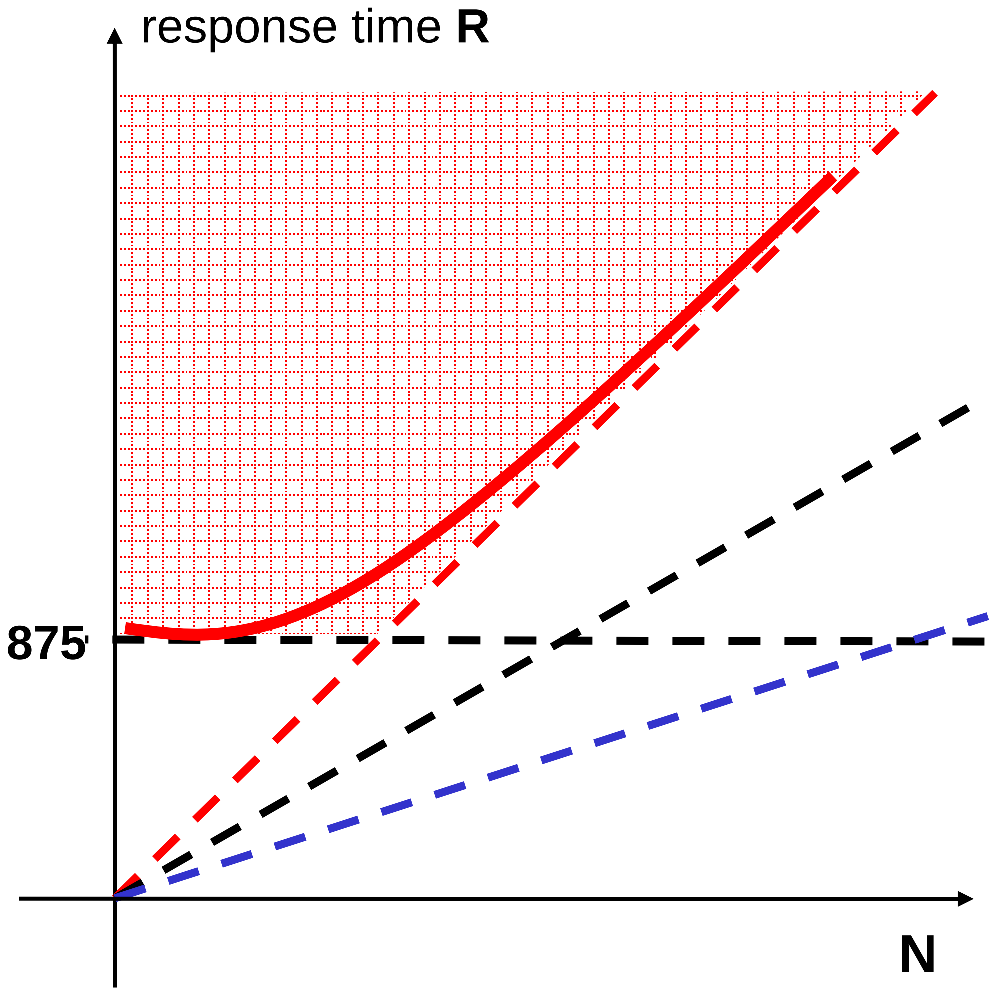
$$U_{\max} = X\ \ D_{\max}$$

$$X \le \frac{1}{D_{\max}} \Rightarrow R \ge N\ D_{\max}$$

$$R \ge N\ 500\,ms$$

$$D = \sum_i D_i = 875\,ms$$

$$R \ge D$$



bound of response time