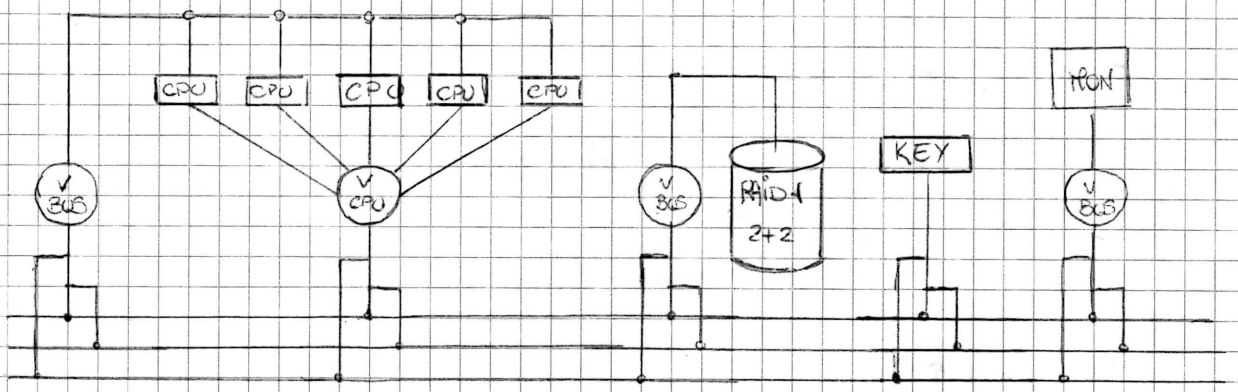
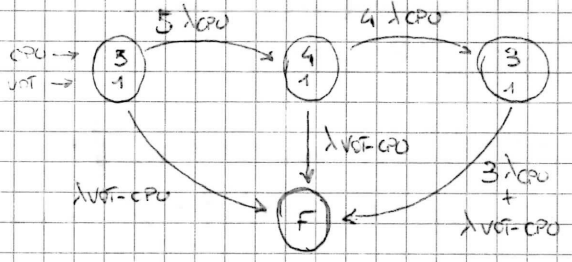
Exam 20.02.2014

Exercise 1 Hypothesis: S switch off the system when a component faults.



RELIABILITY

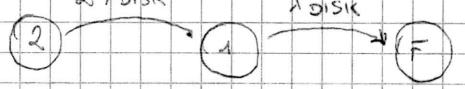
- CPU + VOTER CPU



$$\begin{cases} P'_{5,1}(t) = -P_{5,1}(t)\left[5\lambda_{CPU} + \lambda_{VOT\text{-}CPU}\right] \\ P'_{4,1}(t) = +P_{5,1}(t)\left[5\lambda_{CPU}\right] - P_{4,1}(t)\left[4\lambda_{CPU} + \lambda_{VOT\text{-}CPU}\right] \\ P'_{3,1}(t) = +P_{4,1}(t)\left[4\lambda_{CPU}\right] - P_{3,1}(t)\left[3\lambda_{CPU} + \lambda_{VOT\text{-}CPU}\right] \\ P'_{F}(t) = +\lambda_{VOT\text{-}CPU}\left[P_{5,1}(t) + P_{4,1}(t) + P_{3,1}(t)\,(3\lambda_{CPU})\right] \\ P_{5,1}(t) + P_{4,1}(t) + P_{3,1}(t) + P_F(t) = 1 \\ P_{5,1}(0) = 1 \end{cases}$$

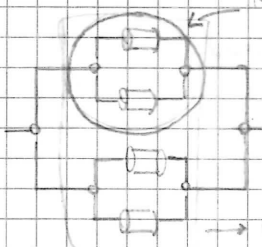
$$R_{CPU}(t) = 1 - P_F(t)$$

- RAID (single pair)



$$R_{pair} = 1 - P_F(t) \Longrightarrow R_{RAID} = (R_{pair})^2$$

$$R_{pair} = 1 - (1 - R_{disk})^2$$

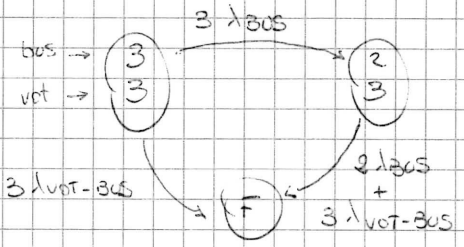


at least one disk has to work in the pair (no one of the two disks failed)

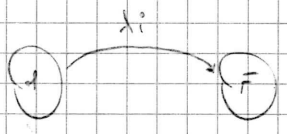
$R_{RAID} = [R_{pair}]^2$ it has to be valid for both of the pair

- BUS + VOTER BUS

$$R_{BUS}(t) = 1 - P_F(t)$$



- KEYBOARD / MONITOR



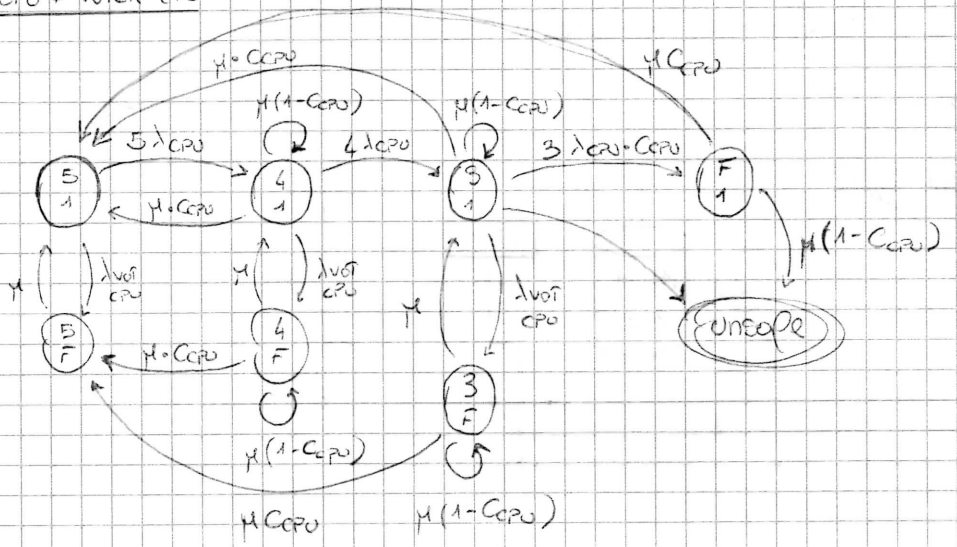
$$R_i(t) = 1 - P_F(t) \quad \text{where } i = \text{key / mon}$$

Finally:

$$R_{tot}(t) = R_{CPU}(t) \cdot R_{RAID}(t) \cdot R_{BUS}(t) \cdot R_{key}(t) \cdot R_{mon}(t)$$

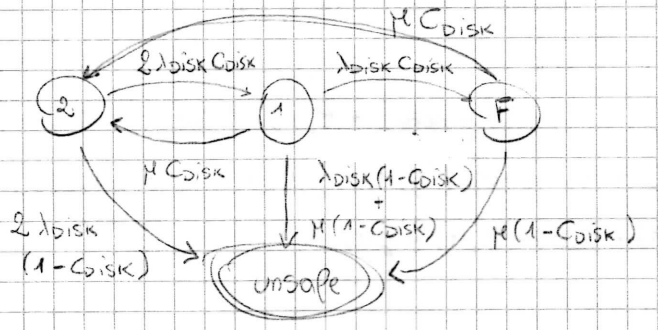
SAFETY Hypothesis: The coverage factor of the voters is 1.

- CPU + VOTER CPU



$$S_{CPU}(t) = 1 - P_{unsf}(t)$$

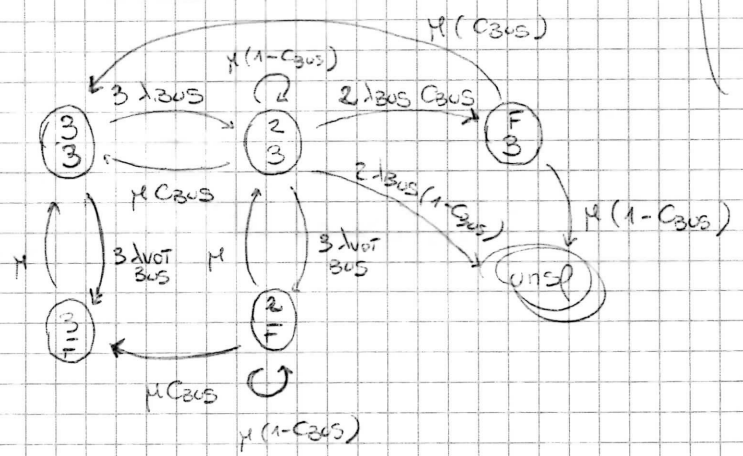
- RAID (single pair)



$$\begin{cases}
P_2'(t) = -P_2(t)\left[2\lambda_{DISK}\left(C_{DISK}+(1-C_{DISK})\right)\right] + \mu C_{DISK}\left(P_F(t)+P_1(t)\right) \\
P_1'(t) = -P_1(t)\left\{\lambda_{DISK}\left[C_{DISK}+(1-C_{DISK})\right] + \mu\left[C_{DISK}+(1-C_{DISK})\right]\right\} + P_2(t)\left(2\lambda_{DISK}C_{DISK}\right) \\
P_F'(t) = -P_F(t)\left[\mu\left(C_{DISK}+(1-C_{DISK})\right)\right] + P_1(t)\left(\lambda_{DISK}C_{DISK}\right) \\
P_{unsf}'(t) = +P_2(t)\left[2\lambda_{DISK}(1-C_{DISK})\right] + P_1(t)\left[\lambda_{DISK}(1-C_{DISK}) + \mu(1-C_{DISK})\right] + P_F(t)\left[\mu(1-C_{DISK})\right] \\
P_2(t)+P_1(t)+P_F(t)+P_{unsf}(t) = 1 \quad ; \quad P_2(0)=1
\end{cases}$$

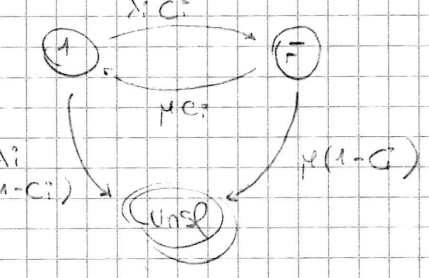
$$S_{PAIR}(t) = 1 - P_{unsf}(t) \Rightarrow S_{RAID}(t) = \left(S_{PAIR}\right)^2$$

- BUS + VOTER BUS



$$S_{BUS}(t) = 1 - P_{unsf}(t)$$

- KEYBOARD/MONITOR



$$S_i(t) = 1 - P_{unsf}(t) \quad \text{where } i = \text{key/mon}$$

Finally:

$$S_{tot}(t) = S_{CPU}(t) \cdot S_{RAID}(t) \cdot S_{BUS}(t) \cdot S_{KEY}(t) \cdot S_{MON}(t)$$

20.02.2014

Esercizio 2

L'MVA (Mean Value Analysis) è un metodo che ci permette di calcolare gli indici di performance (response time medio, throughput, lunghezza della coda, ecc...) per reti chiuse.

Per "rete chiusa" intendiamo una rete che è formata da un numero limitato e fisso di utenti quindi ho un limite sul n° di richieste che il sistema deve soddisfare.
(È composta da due fasi: richiesta del servizio e thinking time).

L'MVA sfrutta un metodo iterativo basato su tre equazioni e su un teorema.

Definizioni:

X_0 = throughput medio della rete di code

V_i = n° medio di visite per richiesta alla coda i

S_i = service time medio per richiesta al server i

$W_i(n)$ = waiting time alla coda i

= $n_i^a(n) \cdot S_i$ dove $n_i^a(n)$ = n° medio di richieste trovate nella coda i da una richiesta in arrivo

$R_i(n)$ = residence time medio per una richiesta alla coda i

= $S_i + W_i(n) = S_i + [n_i^a(n) \cdot S_i] = S_i [1 + n_i^a(n)]$

$R_i'(n)$ = media del residence time totale per una richiesta alla coda i considerando tutte le visite a tale coda

= $V_i \cdot R_i(n)$

D_i = media del service time totale per una richiesta alla coda i considerando tutte le visite alla coda

= $V_i \cdot S_i$

$R_0(n)$ = Response time medio della rete di code = $\sum R_i'(n)$

Arrival theorem:

Il n° medio di richieste che una nuova richiesta entrante vede nella coda i-esima quando già ci sono n richieste nella rete di code (cioè $n_i^a(n)$), è uguale al n° medio di richieste che ho nella coda i-esima quando ce ne erano $n-1$ di richieste (cioè $n_i(n-1)$).
nella rete di code

Quindi: $n_i^a(n) = n_i(n-1)$

Allora, da questo teorema, otteniamo:

$R_i = S_i [1 + n_i(n-1)]$

Moltiplicando entrambi i membri per V_i, otteniamo la 1ª equazione dell'MVA

$$R_i'(n) = D_i[1 + n_i(n-1)] \quad \text{per risorse in coda}$$
$$\text{or} \quad = D_i \quad \text{per risorse di ritardo}$$

} Residence time equation

Applicando la legge di Little alla intera rete di code ($n = X_0(n) R_0(n)$) otteniamo la 2ª:

$$X_0(n) = \frac{n}{R_0(n)} \quad \text{Throughput equation} \quad \text{where } R_0(n) = \sum_{i=1}^{M} R_i'(n) \quad M = \text{n° of the queue}$$

La 3ª equazione dell'MVA è ottenuta combinando la legge di Little e quella del flusso forzato:

$$n_i(n) = X_0(n) \cdot R_i'(n)$$

Ma come faccio a calcolarmi i valori di $R_i'(n)$ se questo dipende da $n_i(n-1)$?
A sua volta $n_i(n-1)$ dipende da $R_i'(n-1)$ che dipende da $n_i(n-2)$, ecc...
Dobbiamo partire da $n=0$ e calcolare fino al valore di n a cui siamo interessati (ie n° di richieste massimo che possono essere soddisfatte dalla rete).

Per fortuna abbiamo che:

$$n_i(\emptyset) = \emptyset \quad \forall i$$

Questo perché quando non ci sono richieste nel sistema, allora la lunghezza delle code è nulla per tutti i componenti!
Quindi, a questo punto cominciamo a calcolare:

$$n_i(\emptyset) \longrightarrow R_i'(1) \longrightarrow X_0(n) \longrightarrow n_i(1) \Longrightarrow R_i'(2) \longrightarrow X_0(2) \longrightarrow n_i(2) \Longrightarrow \dots$$

Calcolati tutti i valori, poi ci potremmo utilizzare per calcolare:

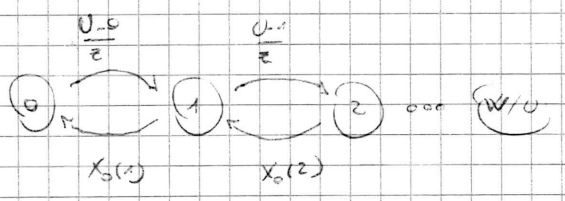
$$\bar{X} = \sum_{i=1}^{W/O} p_i \cdot X_0(i)$$ Average throughput of the network

W = lunghezza coda degli utenti (se c'è) oppure U = n° utenti del sistema

$$\bar{N} = \sum_{i=0}^{W} p_i \cdot i$$ average n° of users in the queue network

$$\bar{R} = \frac{\bar{N}}{\bar{X}}$$ average service time

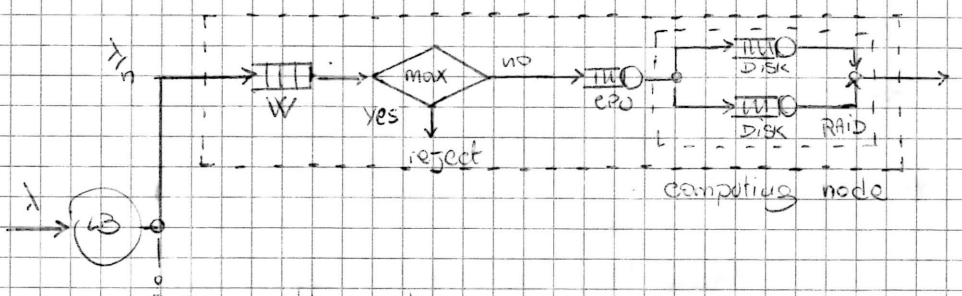
dove p_i rappresenta la probabilità di trovarsi nello stato in cui nel sistema ci sono i richieste.



Z = thinking time

20.02.2016

Exercise 3



$\lambda = 50$ (req/sec)

$W = 4$ (req)

$D_{CPU} = 50$ (msec)

$D_{disk} = 20$ (msec)

$D_{RAID} = \frac{D_{disk}}{2} = 10$ (msec)

Hypothesis: I hypothesize that there is a load balancer which equally distributes the load of the request to the n servers.

I have to calculate the n° of computing nodes that allows me to have the fraction of lost requests less than 10%.

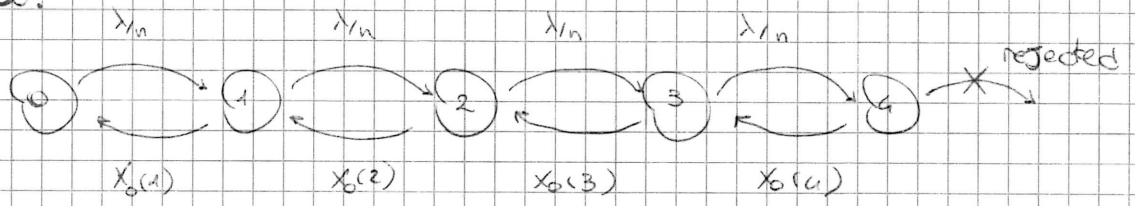
There aren't the MTTF, MTTR parameters so I don't have to consider the cases of faulty servers.

SINGLE SERVER SCENARIO

The arrival rate of the requests, due to the load balancer, is $\frac{\lambda}{n}$ and it's independent of the n° of requests already inside the server.

Instead, the throughput of the server is dependent on that.

So:



For the flow-in = flow-out principle:

$$\begin{cases} P_0 \cdot \frac{\lambda}{n} = P_1 \cdot X_0(1) \\ P_1 \cdot \frac{\lambda}{n} = P_2 \cdot X_0(2) \\ P_2 \cdot \frac{\lambda}{n} = P_3 \cdot X_0(3) \\ P_3 \cdot \frac{\lambda}{n} = P_4 \cdot X_0(4) \\ \sum_{i=0}^{4} P_i = 1 \end{cases}$$

$$P_1 = P_0 \left[\frac{\lambda}{n} \cdot \frac{1}{X_0(1)} \right]$$

$$P_2 = P_1 \left[\frac{\lambda}{n} \cdot \frac{1}{X_0(2)} \right] = P_0 \left[\left(\frac{\lambda}{n}\right)^2 \cdot \frac{1}{X_0(1)} \cdot \frac{1}{X_0(2)} \right]$$

$$P_3 = \dots = P_0 \left[\left(\frac{\lambda}{n}\right)^3 \cdot \frac{1}{X_0(1)\, X_0(2)\, X_0(3)} \right]$$

$$P_4 = \dots P_0 \left[\left(\frac{\lambda}{n}\right)^4 \cdot \frac{1}{X_0(1) \cdot X_0(2) \cdot X_0(3) \cdot X_0(4)} \right]$$

To calculate the throughput of the server (based on the n° of request inside it), I have to use the MVA algorithm.

The iterative procedure goes from $n=1$ to $n=4$.

We know that: $n_i(0) = 0 \quad \forall i$ {where i = CPU, disk}

So, using the 3 equations:

$$R_i(n) = D_i \left[1 + n_i(n-1)\right], \quad X_0^{(n)} = n / \sum R_i(n), \quad n_i(n) = X_0(n) \cdot R_i(n)$$

n = 1

$R'_{CPU}(1) = D_{CPU}[1 + n_{CPU}(\emptyset)] = 50$ (msec)

$R'_{DISK\ RAID}(1) = D^{RAID}_{DISK}[1 + n_{DISK}(\emptyset)] = \frac{1}{10}$ (msec) ⇝ Having a RAID 1 or 3 use a disk or 2 use the other one.

$R_0(1) = R'_{CPU}(1) + R'_{DISK\ RAID}(1) = \frac{6}{10}$ (msec)

$X_0(1) = 1 / R_0(1) = 1/[(\frac{6}{10} \cdot 10^{-3})\ (sec)] = \frac{16,67}{}$ (req/sec) (0,01429 (msec))

$n_{CPU}(1) = X_0(1) \cdot R'_{CPU}(1) = \frac{16,67}{}\ (\frac{req}{sec}) \cdot 0,05\ (sec) = \frac{83}{}$ (req)

$n_{DISK\ RAID}(1) = X_0(1) \cdot R'_{DISK\ RAID}(1) = \frac{16,67}{} \cdot \frac{0,01}{1} = \frac{0,17}{17}$ (req)

$\Big\}\ n_{CPU}(1) + n_{DISK}(1) = 1$ ok!

n = 2

$R'_{CPU}(2) = D_{CPU}[1 + n_{CPU}(1)] = 50\ (msec)\ [1, \frac{83}{}] = \frac{91,5}{}$ (msec) $= 0,0\frac{915}{}$ (sec)

$R'_{DISK\ RAID}(2) = D^{RAID}_{DISK}[1 + n_{DISK}(1)] = \frac{1}{10}\ (msec)\ [1, \frac{17}{17}] = \frac{11,8}{11,7}$ (msec) $= 0,0\frac{118}{117}$ (sec)

$R_0(2) = R'_{CPU}(2) + R'_{DISK\ RAID}(2) = \frac{103,2}{}$ (msec) $= 0,\frac{1032}{}$ (sec)

$X_0(2) = 2 / R_0(2) = 2 / 0,\frac{1032}{} = \frac{19,38}{}$ (req/sec)

$n_{CPU}(2) = X_0(2) \cdot R'_{CPU}(2) = \frac{19,38}{} \cdot \frac{0,0015}{} = \frac{77}{}$

$n_{DISK\ RAID}(2) = X_0(2) \cdot R'_{DISK\ RAID}(2) = \frac{19,38}{} \cdot \frac{0,0118}{0,0117} = \frac{23}{}$

$\Big\}\ \Sigma = 2$ ok

n = 3

$R'_{CPU}(3) = D_{CPU}[1 + n_{CPU}(2)] = 50\ (msec)\ (2, \frac{77}{}) = \frac{1385}{}$ (ms) $= 0,\frac{1385}{}$ (sec)

$R'_{DISK\ RAID}(3) = D^{RAID}_{DISK}[1 + n_{DISK}(2)] = \frac{1}{10}\ (msec)\ (1, \frac{23}{}) = \frac{32,2}{12,3}$ (ms) $= 0,0\frac{322}{123}$ (sec)

$R_0(3) = R'_{CPU}(3) + R'_{DISK\ RAID}(3) = 0,\frac{1508}{}$ (sec)

$X_0(3) = 3 / R_0(3) = 19,\frac{89}{}$ (req/sec)

$n_{CPU}(3) = X_0(3) \cdot R'_{CPU}(3) = \frac{19,89}{} \cdot \frac{0,1385}{} = \frac{2,75}{}$

$n_{DISK\ RAID}(3) = X_0(3) \cdot R'_{DISK\ RAID}(3) = \frac{19,89}{} \cdot \frac{0,0123}{} = \frac{0,516}{25}$

$\Big\}\ \Sigma = 3$ ok!

n = 4

$R'_{CPU}(4) = D_{CPU}[1 + n_{CPU}(3)] = 50\ (msec) \cdot 3,\frac{75}{} = \frac{187,5}{}$ (msec) $= 0,\frac{1875}{}$ (sec)

$R'_{DISK\ RAID}(4) = D^{RAID}_{DISK}[1 + n_{DISK}(3)] = \frac{1}{10}\ (msec) \cdot 1,\frac{516}{25} = \frac{12,9}{12,5}$ (msec) $= 0,0\frac{129}{125}$ (sec)

$R_0(4) = R'_{CPU}(4) + R'_{DISK\ RAID}(4) = 0,2\frac{003}{}$ (sec)

$X_0(4) = 4 / R_0(4) = 19,\frac{80}{20}$ (req/sec)

20.02.2014

Now, I can calculate:

$$X(0) = \sum_{i=1}^{4} p_i \cdot X_0(i) = [p_1 X_0(1)] + [p_2 \cdot X_0(2)] + [p_3 \cdot X_0(3)] + [p_4 \cdot X_0(4)]$$

↑ average throughput of a single server / computing node

$$N(0) = \sum_{i=0}^{4} p_i \cdot i = 0 + p_1 + 2p_2 + 3p_3 + 4p_4$$

↑ average n° of requests into the single server

$$R(0) = \frac{N(0)}{X(0)} \quad \text{(from the Little's law)}$$

↑ response time of the single server.

And, for the whole system (composed of n servers):

$\bar{X} = n \cdot X(0)$ ← web site throughput

$\bar{R} = R(0)$ ← web site response time

Finally, the fraction of rejected requests is given by:

$$\frac{\lambda - \bar{X}}{\lambda} \overset{!}{<} 10\% \quad \leftarrow \text{This concerns the whole web server!}$$

If instead, the professor meant about the single computing node, then the fraction of lost requests < 10% is given by:

$$p_4 \overset{!}{<} 0{,}01$$

(Where p_4 is the probability to stay in the case in which the computing node has 4 request in queue, that is the maximum supported. Beyond 4, the request is reject.)