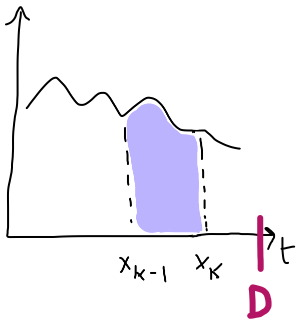
# Introduzione al SIZE-BASED

02/05/23

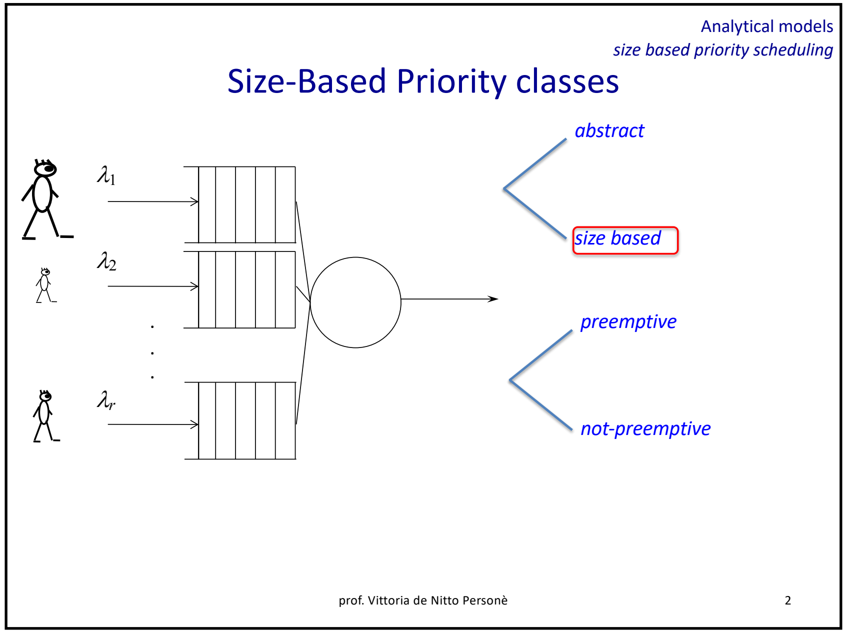
Se riprendiamo, per il caso astratto, la formula del tempo medio in coda, abbiamo: $E[T_q] = \frac{E[S_{rim}]}{(1-\sum_{i=1}^{k}\rho_i)(1-\sum_{i=1}^{k-1}\rho_i)}$

Nel caso size-based, l' "unico" cambiamento lo si ha in come calcoliamo $\rho_i = \lambda_i E[S_i]$

Nel caso astratto E[Si] è uguale per ogni classe, qui varia.
Nel caso astratto il tasso di arrivo della singola classe dipende da fattori esterni alla size, ora dipende dalla size.
Possiamo dunque dire che nel size-based entrambi questi fattori dipendono dalla distribuzione del tempo di servizio.




la sottoarea viola mi dice quanti job sono di classe k. La funzione in figura è la densità del tempo di servizio f(t), che è continua.
Ciò che si fa è fissare un upper bound "D", grande abbastanza per far si che la probabilità di servizi più grandi di D tenda a 0.




2

Un utente chiede servizio "s", a quale classe appartiene?
Appartiene alla classe "k" se s $\in (X_{k-1}, X_k]$

Poichè si tratta di un'area, dobbiamo integrare.
L'area viola è data da : F(Xk) - F($X_{k-1}$), dove F è la cumulativa.
Se il tasso è $\lambda$ (generale), a me arriva una sottoporzione data da:

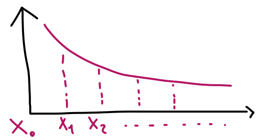
$\lambda_k = \lambda(F_{(X_k)} - F_{(X_{k-1})})$ Quindi quell'area è come se fosse la "Probabilità di cadere li"

Passiamo adesso al tempo di servizio medio $E[S_k] = \int_{X_{k-1}}^{X_k} t \cdot f^n(t)\,dt$ dove f(t) viene "Normalizzata", ovvero la densità viene normalizzata per poi avere somma 1.

$f^n(t) = \frac{f(t)}{F(X_k)-F(X_{k-1})}$

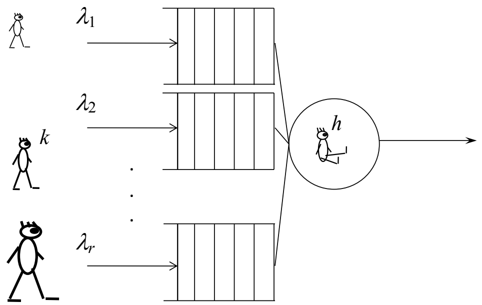
Nell'integrale abbiamo un'impostazione di tipo "peso" * "valore assunto" (per quanto tempo assumo un comportamento * il comportamento).
Possiamo notare che, nel calcolo di $\rho_k = \lambda_k \cdot E[S_k]$ , $F_{(X_k)} - F_{(X_{k-1})}$ si semplifica (appare sia in lambda che nella normalizzazione dentro E[Sk].

Osservazione: prendiamo un'esponenziale (ma vale anche per hyper, etc con forma simile).
Si osserva che $\lambda_1 \geq \lambda_2 ... \geq \lambda_R$ , ovvero l'area è monotona decrescente. Se pensiamo ai tempi di servizio associati, possiamo invece dire: $E[S_1] < E[S_2] < ... < E[S_R]$
ovvero è monotona crescente. Esiste una proprietà che ci dice che: la somma di una serie crescente ed una serie decrescente forma una somma minima
Per questo la size based di classe k è migliore di una astratta di classe k, perchè è sempre una somma minima, sotto non scendo!



## Size-Based Priority classes

"minimum size"




"maximum size"

$h < k$

$$\downarrow$$

$$E(S_h) < E(S_k)$$

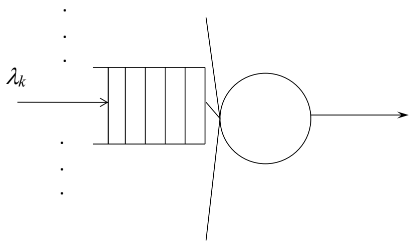






## Size-based priority without preemption

$u$ belongs to class $k$ if $\quad S \in (x_{k-1}, x_k]$




$S_k$

$$E(S_k) = \frac{1}{\mu_k} \qquad \sigma^2(S_k)$$

$$\rho_k = \lambda_k E(S_k) \qquad \rho = \sum_{i=1}^{r} \rho_i$$

$$\lambda = \sum_{i=1}^{r} \lambda_i$$

$\lambda_k$ , $E(S_k)$ depend on the distribution "shape"

$$E(S_k) = \int_{x_{k-1}}^{x_k} t f^n(t)\,dt$$

$$f^n(t) = \frac{f(t)}{F(x_k) - F(x_{k-1})}$$ densità normalizzata, così le medie sono 'corrette' in modo che la somma sia 1.

$$\lambda_k = \lambda\big(F(x_k) - F(x_{k-1})\big), \quad p_k = \frac{\lambda_k}{\lambda} = F(x_k) - F(x_{k-1})$$ = probabilità di essere di classe k

$$\rho_k = \lambda_k E(S_k) = \lambda\big(F(x_k) - F(x_{k-1})\big) \int_{x_{k-1}}^{x_k} t f^n(t)\,dt$$

$$= \lambda\cancel{\big(F(x_k) - F(x_{k-1})\big)} \int_{x_{k-1}}^{x_k} t \frac{f(t)}{\cancel{F(x_k) - F(x_{k-1})}}\,dt$$

$$= \lambda \int_{x_{k-1}}^{x_k} t f(t)\,dt$$ comunque lambda è sempre dipendente dalla classe k, anche se qui si semplifica.

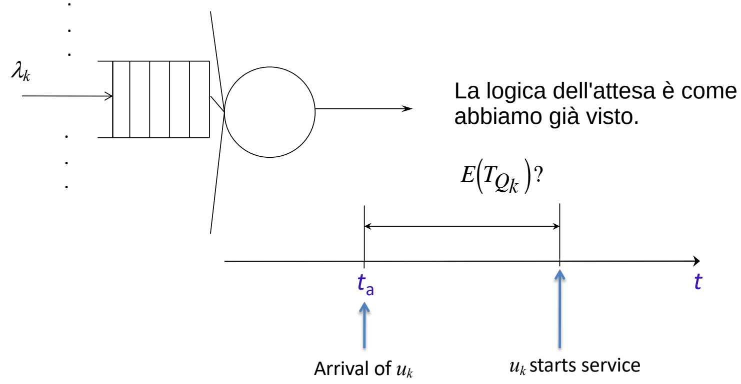






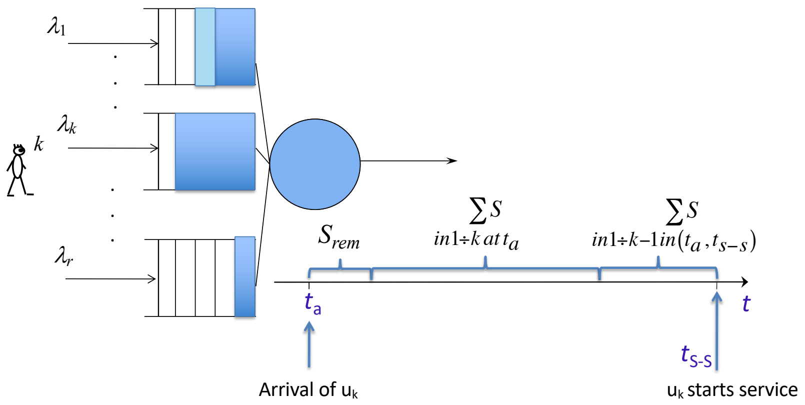
## Size-based priority without preemption

### local performance measures



# Size-based priority without preemption

## local performance measures



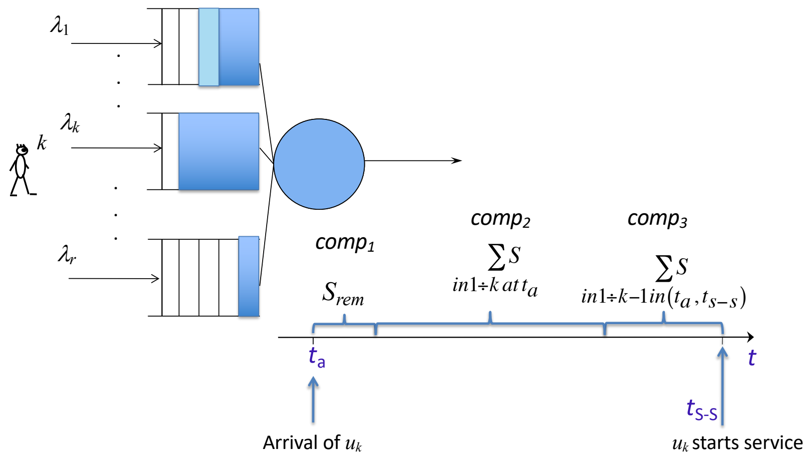



7

Analytical models
*size based priority scheduling*

# Size based priority without preemption

## local performance measures





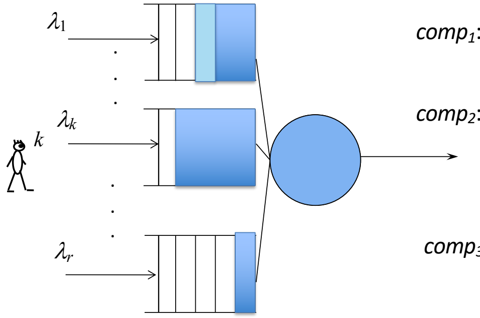

8

ciò che cambia è come scriviamo "rho", le formule sono uguali.



# Size based priority without preemption

## local performance measures




*comp₁*: $E(S_{rem}) = \frac{\lambda}{2} E(S^2)$

*comp₂*: proportional to the load of queues 1:$k$ $= \dfrac{1}{1 - \sum_{i=1}^{k} \rho_i}$

*comp₃*: proportional to the load of queues 1:$k$-1 $= \dfrac{1}{1 - \sum_{i=1}^{k-1} \rho_i}$

$$E(T_{Q_k})^{SB\_NP\_priority} = \frac{\frac{\lambda}{2} E(S^2)}{\left(1 - \sum_{i=1}^{k} \rho_i\right)\left(1 - \sum_{i=1}^{k-1} \rho_i\right)}$$



9



$$\rho_k = \lambda \int_{x_{k-1}}^{x_k} t f(t) dt$$

$$\sum_{i=1}^{k} \rho_i = \sum_{i=1}^{k} \lambda \int_{x_{i-1}}^{x_i} t f(t) dt$$

$$= \lambda \int_{0}^{x_k} t f(t) dt$$

$$E(T_{Q_k})^{SB\_NP\_priority} = \frac{\frac{\lambda}{2} E(S^2)}{\left(1 - \lambda \int_0^{x_k} t f(t) dt\right)\left(1 - \lambda \int_0^{x_{k-1}} t f(t) dt\right)}$$



10

## Size based priority without preemption

global performance measures

And the "global" performance?

somma pesata delle attese delle varie classi.

$$E(T_Q)^{SB\_NP\_priority} = E\left(E\left(T_{Q_k}\right)\right) = \sum_{k=1}^{r} p_k E\left(T_{Q_k}\right)$$

$$p_k = \frac{\lambda_k}{\lambda} = \frac{\lambda(F(x_k) - F(x_{k-1}))}{\lambda} = F(x_k) - F(x_{k-1})$$

probabilità per ogni "k"

$$E(T_Q)^{SB\_NP} = \frac{\lambda}{2} E\left(S^2\right) \sum_{k=1}^{r} \frac{F(x_k) - F(x_{k-1})}{\left(1 - \lambda \int_0^{x_k} tf(t)dt\right)\left(1 - \lambda \int_0^{x_{k-1}} tf(t)dt\right)}$$



11



## Size-based vs abstract priority

local performance measures

$$E\left(T_{Q_k}\right)^{NP\_priority} = \frac{\frac{\lambda}{2} E\left(S^2\right)}{\left(1 - \sum_{i=1}^{k} \rho_i\right)\left(1 - \sum_{i=1}^{k-1} \rho_i\right)}$$

$$\boxed{E\left(T_{Q_k}\right)^{SB\_NP} \le E\left(T_{Q_k}\right)^{abstract\_NP}}$$

$$\left[\left(1 - \sum_{i=1}^{k} \rho_i\right)\left(1 - \sum_{i=1}^{k-1} \rho_i\right)\right]^{SB\_NP} \ge \left[\left(1 - \sum_{i=1}^{k} \rho_i\right)\left(1 - \sum_{i=1}^{k-1} \rho_i\right)\right]^{abstract\_NP}$$

$$\left[\sum_{i=1}^{h} \rho_i\right]^{SB\_NP} \le \left[\sum_{i=1}^{h} \rho_i\right]^{abstract\_NP}$$ for each $h$

$$E\left(T_{S_k}\right)^{SB\_NP} \, ? \, E\left(T_{S_k}\right)^{abstract\_NP}$$



12

Sulla risposta classe per classe non posso dire nulla, il tempo di servizio SIZEBASED sulle classi meno importanti, possono eccedere la media del tempo. Non posso dirlo a priori.

## Size-based vs abstract priority

global performance measures

(come indice globale, non in funzione della classe)

$$\boxed{E(T_Q)^{SB\_NP} \le E(T_Q)^{abstract\_NP}}$$

$$\boxed{E(T_S)^{SB\_NP} \le E(T_S)^{abstract\_NP}}$$

$$E(T_S)^{x\_NP} = E(T_Q)^{x\_NP} + E(S)^{x\_NP}$$

(sia size based che astratto)

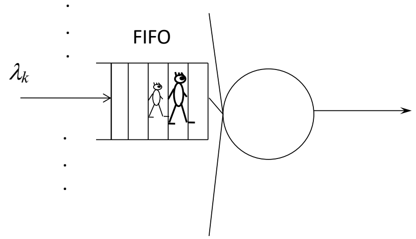
$$E(S)^{SB\_NP} = E(S)^{abstract\_NP} = E(S)$$





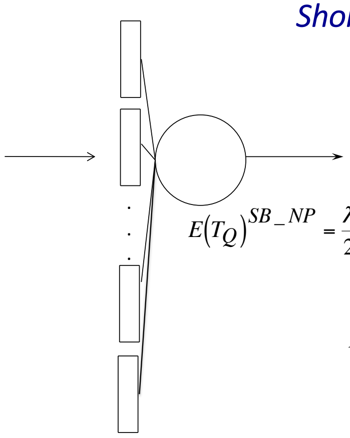


## Size based priority without preemption




Nelle singole classi, abbiamo ordine FIFO. Senza prelazione, nella coda posso avere disordine in base alla size. Per ordinare, perchè non ordino tra le size presenti?
cioè invece di raggruppare per classi, raggruppo per singolo tempo di servizio richiesto.

## Size based priority without preemption
## *Shortest Job First*



$r \to \infty$

faccio il limite di E[Tq] SB_NP

$$E(T_Q)^{SB\_NP} = \frac{\lambda}{2} E(S^2) \sum_{k=1}^{r} \frac{F(x_k) - F(x_{k-1})}{\left(1 - \lambda \int_0^{x_k} tf(t)dt\right)\left(1 - \lambda \int_0^{x_{k-1}} tf(t)dt\right)}$$

$$E(T_Q)^{SJF} = \frac{\lambda}{2} E(S^2) \int_0^{\infty} \frac{dF(x)}{\left(1 - \lambda \int_0^{x} t f(t)dt\right)^2}$$

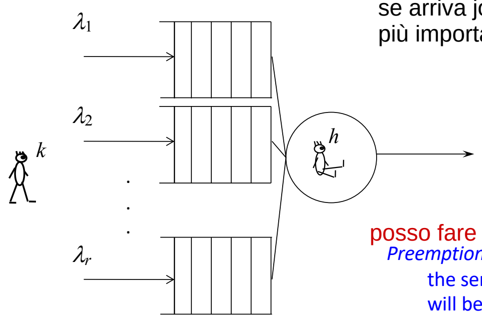


15

eseguo sempre job con size più piccola (ovviamente sapendo la size).
Per r->infinito, non calcolo più prestazioni singola classe (questo "concetto" sparisce) ma solo prestazioni globali, in quanto "ordino il singolo".



## SB preemptive priority




se arriva job $k < h$
più importante
$E(S_k) < E(S_h)$

ma se il job h stava proprio per finire, mi conviene cambiarli?

posso fare prelazione solo se:

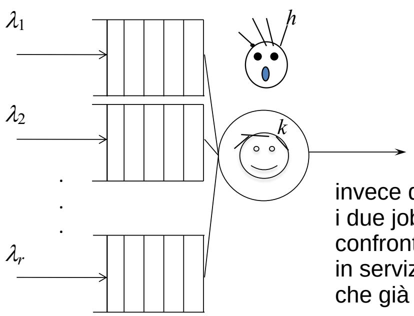
*Preemption without loss*:

the service of the interrupted job will be resumed from the interruption point



16

## SB preemptive priority




$k < h$

$E(S_k) < E(S_h)$

$E(S_k) < E(S_{h\_rem})$

invece di confrontare quanto i due job starebbero in servizio, confronto quello che vorrebbe entrare in servizio con quanto rimane a quello che già sta in esecuzione.

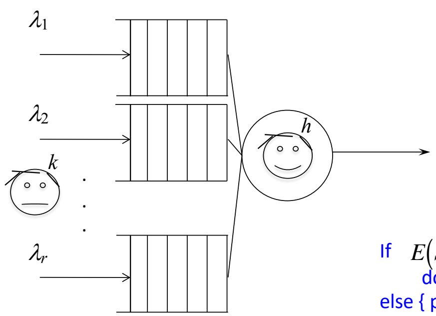


17

Se invece faccio lo scambio, il job tolto lo metto non nella sua classe di origine, bensì nella classe in funzione a quanto tempo di servizio gli rimane! (es: classe 0: esecuzione 1 min, classe 1: esecuzione 2 min)
Se job classe 1 esegue, e gli rimangono 58 secondi, ma viene sostituito, viene messo in classe 0, non in classe 1.



## SB preemptive priority




$k < h$

$E(S_k) < E(S_h)$

$E(S_{h\_rem}) < E(S_k)$

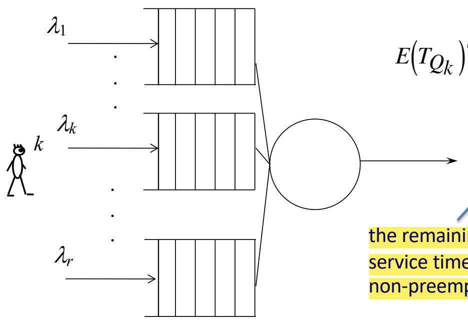
If $E(S_{h\_rem}) < E(S_k)$
don't preempt!
else { preemtion!
put the preempted
job in class $h'$
s.t. $x_{h'-1} < E(S_{h\_rem}) \le x_{h'}$ }



18

# SB preemptive priority

## local performance measures




$$E\left(T_{Q_k}\right)^{SB\_P} = \frac{\boxed{??}}{\left(1-\sum_{i=1}^{k}\rho_i\right)\left(1-\sum_{i=1}^{k-1}\rho_i\right)}$$

the remaining service time of the non-preemptible jobs + the remaining service time of the preemptible jobs that immediately restart service

(non interrompibili)

(interrompibili, ma nel valutarlo
vedo che tempo rimanente è <k,
e quindi riprendono servizio subito)







# SB preemptive priority

## local performance measures

the remaining service time of the non-preemptible jobs

$$\frac{\lambda}{2}\int_0^{x_k} t^2 dF(t)$$

abstract: $\frac{1}{2}\sum_{i=1}^{k}\lambda_i E\left(S^2\right)$ tempo servizio rimanente con prelazione

size based: $E\left(S_k^2\right) = \int_{x_{k-1}}^{x_k} t^2 f^n(t)\,dt$

# SB preemptive priority

## local performance measures

the remaining service time of the non-preemptible jobs

$$\frac{\lambda}{2}\int_0^{x_k} t^2 dF(t)$$

the remaining service time of the preemptible jobs that immediately restart service

$$\frac{\lambda}{2}x_k^2$$

Prob{preemptible job}= Prob{$S>x_k$}=1-Prob{$S\le x_k$}







# SB preemptive priority

## local performance measures

$$E(T_{Q_k})^{SB\_P} = \frac{\frac{\lambda}{2}\left[\int_0^{x_k} t^2 dF(t) + (1-F(x_k))x_k^2\right]}{\left(1-\sum_{i=1}^{k}\rho_i\right)\left(1-\sum_{i=1}^{k-1}\rho_i\right)}$$

$$E(T_{Q_k}) \le E(T_{Q_{k+1}})$$

$$E(T_{Q_k})^{SB\_P} \le E(T_{Q_k})^{SB\_NP}$$

$$E(T_{S_k}) = E(T_{Q_k}) + E(S_{virt-k})$$

dove conto tutte le "interruzioni"

$$E(S_{virt-k}) = \frac{E(S_k)}{1-\sum_{i=1}^{k-1}\rho_i}$$

allungamento tempo di servizio proporzionale al carico delle classi che hanno diritto di interrompere.

# SB preemptive priority
## global performance measures

$$E(T_Q)^{SB\_P} = E\left(E(T_{Q_k})^{SB\_P}\right) = \sum_{k=1}^{r} p_k E(T_{Q_k})^{SB\_P}$$

$$p_k = \frac{\lambda_k}{\lambda} = F(x_k) - F(x_{k-1})$$

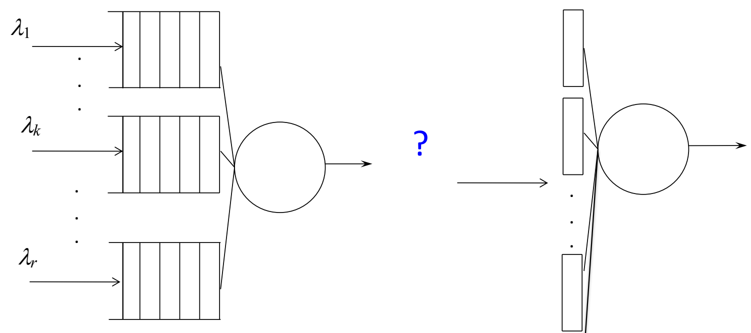
$$E(T_Q)^{SB\_P} = \frac{\lambda}{2}\sum_{k=1}^{r} \frac{[F(x_k) - F(x_{k-1})]\left[\int_0^{x_k} t^2 dF(t) + (1 - F(x_k))x_k^2\right]}{\left(1 - \sum_{i=1}^{k}\rho_i\right)\left(1 - \sum_{i=1}^{k-1}\rho_i\right)}$$





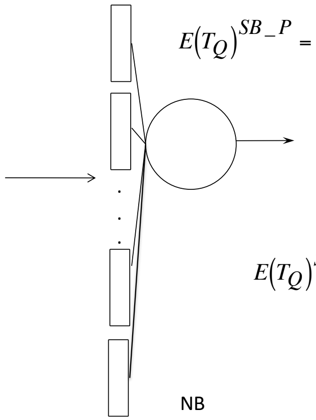


# *SB_P* vs *SJF*




Non esiste risposta generale, dipende dal numero di classi che ho,
se ne ho troppe il caso sinistro tende a quello a destra.
Meno classi sono più la singola classe "pesa", mantiene più cose.
Molto dipende anche dalla variabilità, se lo è poco il peso della
singola classe è poco, e conviene prendere i vantaggi della prelazione.
Perchè non mettere la prelazione nel caso SJF? E infatti...

## *Shortest Remaining Processing Time*



$$E(T_Q)^{SB\_P} = \frac{\lambda}{2}\sum_{k=1}^{r}\frac{[F(x_k)-F(x_{k-1})]\left[\int_0^{x_k} t^2 dF(t) + (1-F(x_k))x_k^2\right]}{\left(1-\sum_{i=1}^{k}\rho_i\right)\left(1-\sum_{i=1}^{k-1}\rho_i\right)}$$

$r \rightarrow \infty$

$$E(T_Q)^{SRJF} = \frac{\lambda}{2}\int_0^{\infty}\frac{\left[\int_0^{x} t^2 dF(t) + (1-F(x))x^2\right]}{\left(1-\lambda\int_0^x t f(t)dt\right)^2}dF(x)$$

web servers under overload

NB
*SRJF=SRPT*

prelazione valutata su tempo di servizio rimanente, sia per sostituzione che per posizionamento.





## *Shortest Remaining Processing Time*

non interrompibili

interrompibili (sarebbe F>x) ma che riprendono servizio subito

$$E(T_Q(x)) = \frac{\frac{\lambda}{2}\int_{t=0}^{x} t^2 f(t)dt + \frac{\lambda}{2}x^2(1-F(x))}{(1-\rho_x)^2}$$

In questa prima formula, se volessi sapere tempo attesa di tutti i job che hanno una certa size "x", non ho il problema dell'integrale doppio, è un caso "più semplice".

$$E(T_S(x)) = E(T_Q(x)) + \underbrace{\int_{t=0}^{x}\frac{dt}{1-\rho_t}}$$

tempo di servizio virtuale, varia nel tempo. Il job diventa più piccolo quando tempo avanza, e sale di priorità. (anche rho cambia, è in funzione di t, perchè sono sempre meno i job che possono interromperlo).

$$\rho_x = \lambda\int_0^x tf(t)dt$$

Queste formule avanzate, se ci fossero, verranno fornite all'esame. Quelle base no.