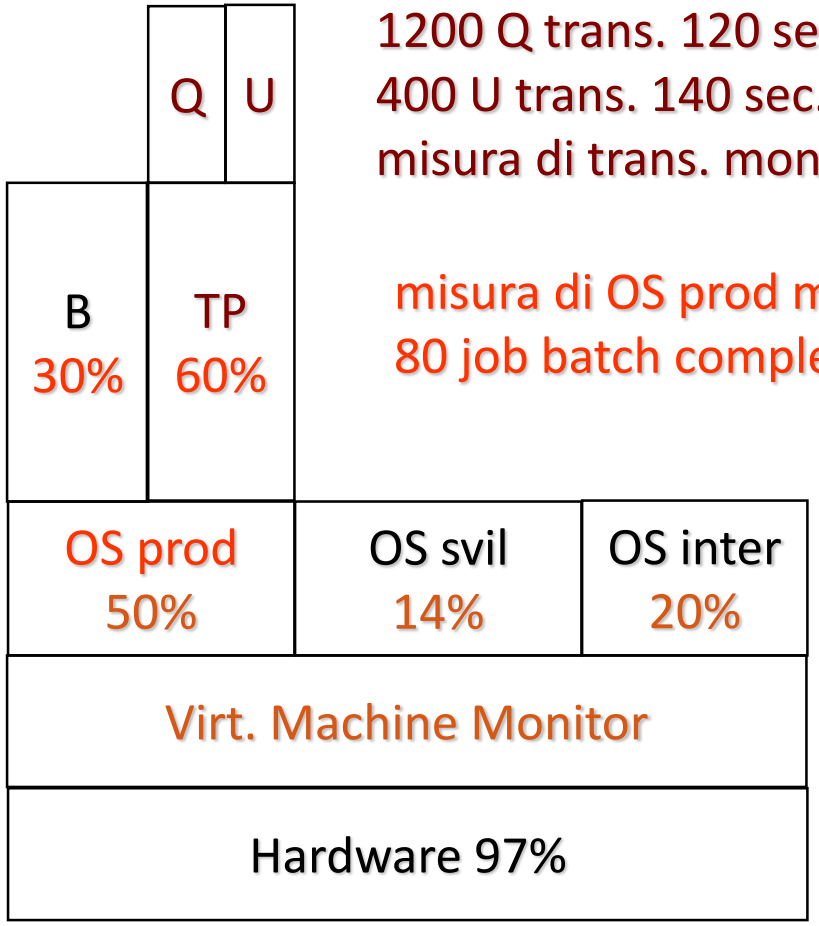
Dischi e CPU

Alcuni esercizi sulle prestazioni (seconda parte)

Calcolo della «domanda» di servizio alla CPU

- $D_{C,CPU} = U_C / X_C$ tempo medio di CPU consumato per transazione
  - U_C = utilizzo della classe C;
  - X_C = numero di transazioni C eseguite nell'unità di tempo
- gli utilizzi della classe C non sono noti esattamente ma si suppone che:
  I. l'utilizzo totale misurato dal monitor di sistema sia corretto;
  II. gli utilizzi misurati negli strati superiori siano corretti per quanto riguarda la loro ripartizione (ma non in termini assoluti).
- (se il numero di processori fosse > 1 allora: U_C contiene la somma degli utilizzi della classe C sui processori usati)

Calcolo della «domanda» di servizio alla CPU



1200 Q trans. 120 sec. CPU
400 U trans. 140 sec.CPU
misura di trans. monitor

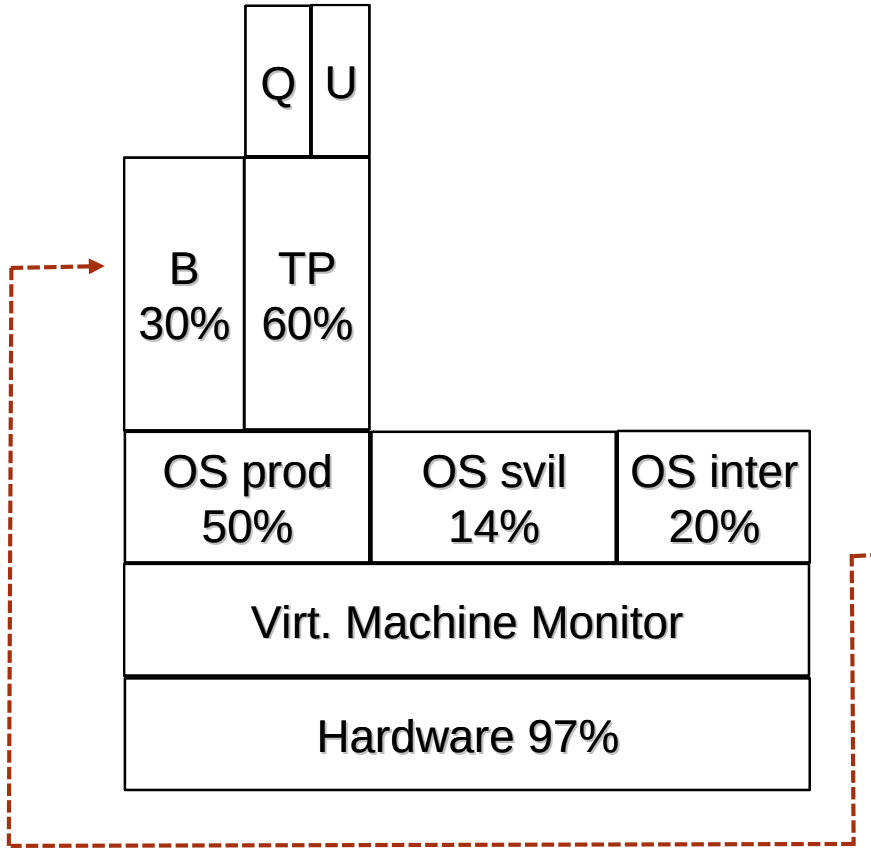
misura di OS prod monitor
80 job batch completati

durata misura:
1800 secondi

misura di VMM software monitor

misura di system monitor

Calcolo della «domanda» di servizio alla CPU



$$D_{U,CPU} = \frac{0.97 \times \frac{50}{50+14+20} \times \frac{60}{30+60} \times \frac{140}{120+140}}{400/1800} = 0.933$$

$$D_{Q,CPU} = \frac{0.97 \times \frac{50}{50+14+20} \times \frac{60}{30+60} \times \frac{120}{120+140}}{1200/1800} = 0.266$$

$$D_{B,CPU} = \frac{0.97 \times \frac{50}{50+14+20} \times \frac{30}{30+60}}{80/1800} = 4.33$$

utilizzo B

utilizzo OS prod

job/sec

Calcolo delle prestazioni di una macchina virtuale

- X_i: throughput della macchina (i) (i= 1,...) = lavoro fatto dalla macchina (i)
  è la variabile fondamentale delle prestazioni = numero di esecuzioni nell'unità di tempo
- si considerano due casi limite:
  1. ad ogni macchina (i) è associata rigidamente la frazione f_i di cicli di CPU; ($\Sigma_i f_i = 1$)
  2. la macchina i può usare tutti i cicli richiesti (se disponibili);
     CPU completamente condivisa
- U_i: utilizzo reale visto dal monitor
- V_i: utilizzo virtuale (cioè misurato dalla macchina (i) che vede solo se stessa)
- S_i: tempo di servizio reale nel sistema nativo ($U_i = X_i S_i$)
- s_i: tempo di servizio virtuale ($V_i = X_i s_i$)

Macchina virtuale caso 1

- la macchina i vede solo la frazione f_i del sistema reale perciò:
  - $V_i = U_i / f_i$
  - $s_i = V_i / X_i = V_i / (U_i / S_i) = S_i / f_i$

- $Ri = s_i /(1 – V_i)$ (tempo di risposta del modello aperto)

$$R_i = \frac{S_i}{f_i - U_i}$$

- Attenzione!
  - solo X_i ha significato di lavoro fatto dal sistema

    s_i e V_i sono grandezze addizionali che non indicano le prestazioni

Macchina virtuale caso 2

- la macchina i vede solo una frazione del sistema reale variabile da istante a istante in funzione dei consumi delle altre macchine e in particolare vede il sistema reale occupato con una probabilità:
  - $U - U_i$ dove U è l'utilizzo totale del sistema reale
- la macchina i è allora rallentata (rispetto alla macchina nativa) del fattore:
  - $(1- (U - U_i))$
- e vede rispettivamente un utilizzo virtuale e un tempo di servizio:
  - $Vi = Ui / (1- (U - U_i))$
  - $s_i = V_i / X_i = V_i / (U_i / S_i) = S_i / (1- (U - U_i))$
- a questo punto possiamo ancora utilizzare la formula della pagina precedente per ottenere R_i

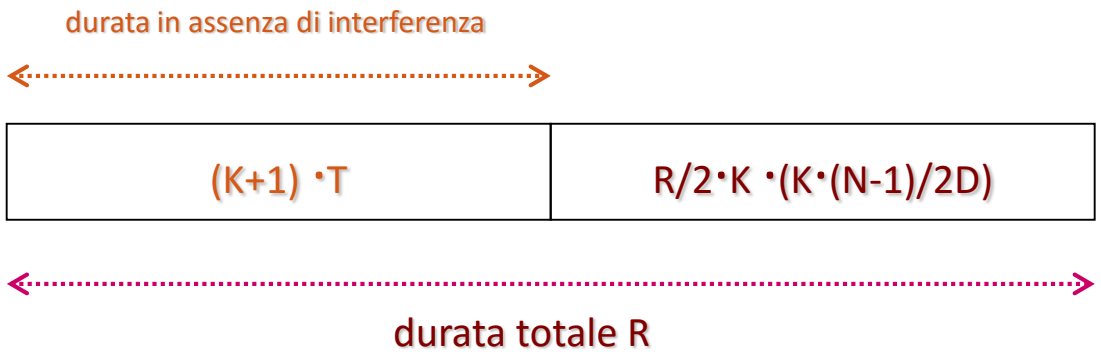
$$R_i = \frac{S_i / (1 - (U - U_i))}{1 - U_i / (1 - (U - U_i))}$$

Transazioni che condividono i dati

- si suole indicare con *lock* lo stato di possesso (esclusivo o condiviso) di una risorsa da parte di un processo.
  - per esempio una *transazione* (unità atomica di carico) acquisisce un lock di tipo esclusivo su un *granulo* di data base del quale intende modificare un dato, per assicurarne la consistenza e integrità. Il lock è rilasciato al termine della transazione quando vengono consolidati gli aggiornamenti (*commit*).
- facendo una serie di ipotesi semplificatrici possiamo calcolare gli effetti quantitativi della condivisione dei dati sulle prestazioni:
  - N transazioni sono mediamente attive, ognuna perciò compete con altre N-1
  - K lock sono mediamente acquisiti da una transazione
  - T è l'intervallo medio fra due successivi lock quindi:
  - $(K+1)\cdot T$ durata media di una transazione in assenza di contesa
  - D numero di *granuli* di cui è costituito il data base - gli accessi sono uniformemente distribuiti su di esso

Transazioni che condividono i dati

- R tempo medio di risposta di una transazione allora:
- R/2 durata media di un lock
- K/2 numero medio di lock attivi nella vita della transazione
- K/2 · (N-1) numero totale di conflitti che la transazione potenzialmente incontra
- K · (N-1)/2D probabilità di conflitto
- K · (K·(N-1)/2D) numero medio di conflitti per transazione
- R/2 · K · (K·(N-1)/2D) ritardo medio dovuto ai lock



Transazioni che condividono i dati

- il procedimento calcola, in modo semplificato, l'effetto sul tempo di risposta di una transazione della presenza contemporanea di altre che vogliano accedere alle stesse risorse (dati condivisi). In particolare si suppone che:
- *N = costante*: questa approssimazione vale se si pone un blocco esterno alle transazioni, in caso contrario a parità di flusso entrante λ, N = λ R. Perciò si avrebbe un effetto aggiuntivo dovuto alla crescita di N
- *T = costante*: cioè l'effetto di contesa su altre risorse (per esempio hardware) è trascurabile;
- gli accessi sono distribuiti nel modo migliore cioè *uniformemente*.

Transazioni che condividono i dati

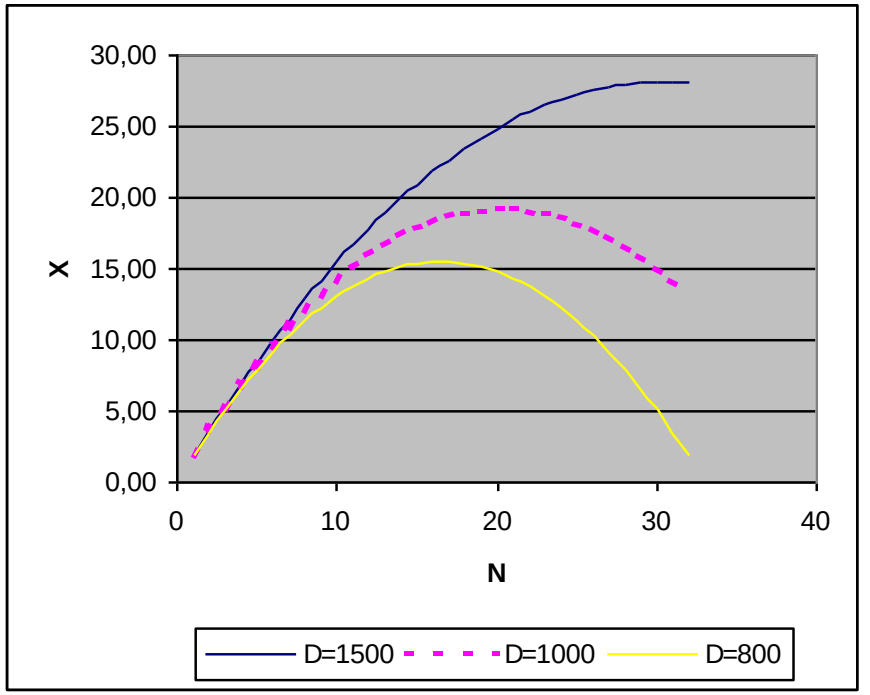
- $R = (K+1)\cdot T + R/2 \cdot K \cdot (K\cdot(N-1)/2D)$ ⇐ tempo medio di risposta
- $R = (K+1) \cdot T / (1-(K^2 \cdot(N-1)/4D))$
- $N = R \cdot X$ - (legge di Little)
- da cui :

$$X = \frac{N\left(1 - K^2 \frac{N-1}{4D}\right)}{(K+1)T}$$

la dipendenza di X da N è quadratica

per $N > 2D/K^2 + 1/2$ il sistema diviene instabile (al crescere di N il throughput X diminuisce)

il grafico mostra alcuni casi per:
K=10 , T = 0.05



Prestazioni di un server

- Il sistema è studiato a livello «esterno»: il suo stato è caratterizzato dal numero k di richieste presenti
  - il web server riceve 10 richieste/sec.
  - il massimo numero di richieste presenti è 3 cioè:
  - le richieste che arrivano, se ne trovano altre tre in esecuzione, vengono rifiutate
  - il throughput misurato in funzione delle richieste è il seguente:

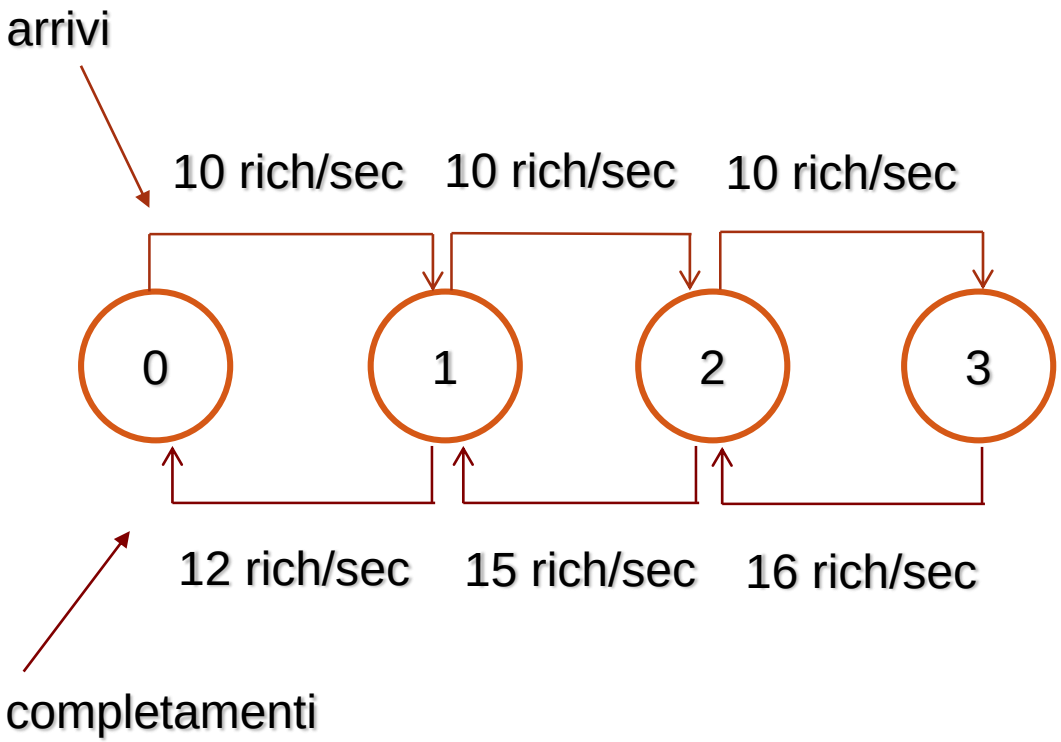
numero richieste k	throughput (richieste/sec)
0	0
1	12
2	15
3	16

Prestazioni di un server

- *Ipotesi*
- omogeneità del carico: tutte le richieste sono equivalenti
- memoryless: non ha importanza alcuna di come il sistema arriva allo stato k
- equilibrio operazionale: lo stato (numero delle richieste presenti) è lo stesso all'inizio e alla fine dell'intervallo in esame
- *Domande*

1. Quale è la probabilità che una richiesta che arriva sia rifiutata?
2. Quale è il numero medio di richieste in esecuzione?
3. Quale è il throughput medio del Web server?
4. Quale è il tempo medio speso nel Web server da una richiesta HTTP?

Schema di funzionamento



Come rispondere alle domande

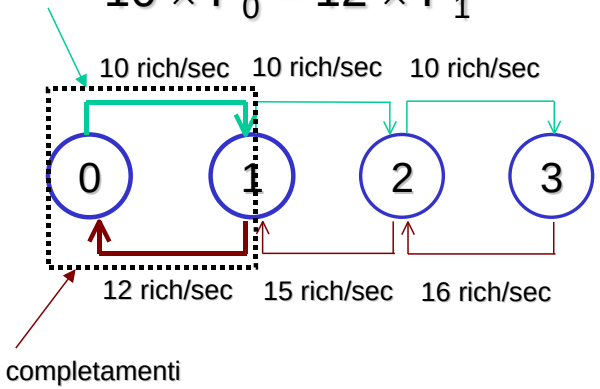
- se conoscessimo le probabilità P_0, P_1, P_2, P_3 che il sistema si trovi negli stati rispettivamente 0, 1, 2, 3 allora:
  1. una richiesta viene rifiutata se tre richieste sono in corso, perciò con probabilità P_3
  2. il numero medio di richieste in esecuzione vale:

     $N = 0 \times P_0 + 1 \times P_1 + 2 \times P_2 + 3 \times P_3$
  3. analogamente il throughput medio si calcola da:

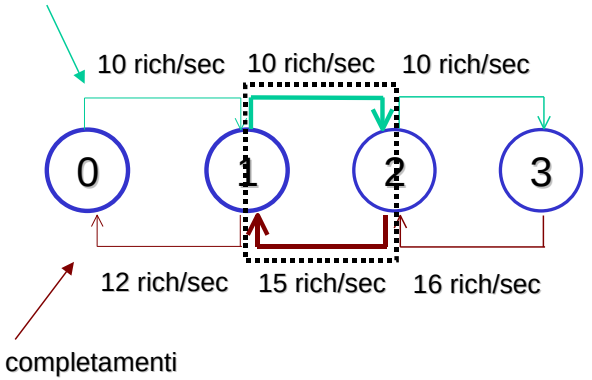
     $X = 0 \times P_0 + 12 \times P_1 + 15 \times P_2 + 16 \times P_3$
  4. il tempo medio R di permanenza nel Web server si ottiene infine dalla legge di Little: $N = X \times R$

Calcolo di P_k – flusso entrante = flusso uscente

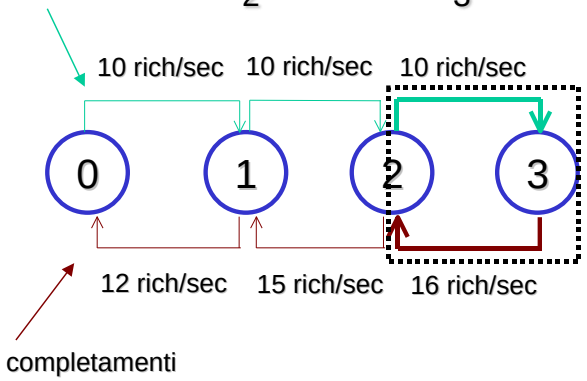
$10 \times P_0 = 12 \times P_1$



$10 \times P_1 = 15 \times P_2$



$10 \times P_2 = 16 \times P_3$



Calcolo di P_k

- $P_1 = 10/12\ P_0$
- $P_2 = 10/15\ P_1 = 10/15 \times 10/12\ P_0$
- $P_3 = 10/16\ P_2 = 10/16 \times 10/15 \times 10/12\ P_0$

  ma il Web server può essere in uno dei quattro possibili stati in ogni istante perciò:

- $P_0 + P_1 + P_2 + P_3 = 1$

K	P_k
0	0,365
1	0,305
2	0,203
3	0,127

risultati

domanda		
1	P_3	0,127
2	N	1,091
3	X	8,731
4	R	0,125