CNP I

Definizione del Problema (i.e. voglio X ed R)

Le grandezze che sono interessato a mettere in relazione sono

- il ***throughput*** X, i.e. quanti job processa il sistema complessivo in un periodo di osservazione unitario;
- il ***response time*** R (o T), i.e. quanto ci mette il sistema complessivo a processare il job

Se N è il numero di job nel sistema, la legge più generale possibile che lega queste grandezze è la ***Little's Law***. È praticamente una legge di conservazione, quindi per applicarla devo definire un volume chiuso che racchiude il sistema di cui voglio studiare le grandezze. Segue che X sarà lo stesso sia all'ingresso che all'uscita della box.

Distinguiamo due casi:

- il sistema dentro la box è ***aperto*** - N è un valore atteso.

$$\langle N \rangle = X \langle R \rangle$$

- Il sistema dentro la box è ***chiuso*** - N è una costante, il ***think time*** Z è il response time degli utenti (i.e. posso vedere il lato client come un server avente response time Z).

$$N = X\Big(\langle R \rangle + \langle Z \rangle\Big)$$

  - Nota che dentro una box in cui applico la Little's Law per sistemi chiusi posso ancora applicare la Little's Law. Una cosa che mi sembra interessante è che se isolo solo il client e solo il server trovo

$$X = \frac{N_{client}}{Z} = \frac{N_{server}}{R} \Rightarrow \frac{N_{client}}{N_{server}} = \frac{Z}{R}$$

    Boh così, era per dire.

Strumenti utili a trovare X ed R

Se ci sono dei loop, è possibile che lo stesso processo visiti più di una volta lo stesso server. In questo caso parliamo di ***numero atteso di visite*** V. Posso esprimere il numero di visite di un singolo processo al server i come rapporto tra i processi completati dal server i e i processi completati dal sistema complessivo:

$$V_i = \frac{C_i}{C}$$

Il ***service time*** è l'inverso della velocità della CPU.

$$S = \frac{1}{\mu}$$

Se il processo ha $V = 1$ è banalmente il tempo che ci mette quel processo ad essere processato una volta (quindi escludendo la coda), altrimenti si estende immediatamente dicendo che la ***demand*** di un server è il service time per il numero di visite:

$$\langle D_i \rangle = \langle V_i \rangle \langle S_i \rangle$$

Se osservo C job completati ognuno dei quali richiede mediamente un tempo di esecuzione pari alla demand D, ricaviamo il ***busy time*** del server come

$$B_i = D_i \cdot C = C_i\, S_i$$

- C è il numero di job completati dall'intero sistema, ***NON*** quello del singolo server C_i. Nella prima uguaglianza, l'informazione del numero di visite al server i è inglobata in D_i.

B_i sta in qualche modo provando a dirci quanto tempo è occupato il server i, ma dipende dal tempo di osservazione. Per avere una grandezza normalizzata introduciamo la ***utilization*** $\rho \in [0, 1]$, che è il busy time diviso il tempo di osservazione

$$\rho_i = \frac{B_i}{\tau} = \frac{C_i}{\tau} S_i = S_i X_i$$

Questa roba si chiama ***utilization law***. Se invece sostituisco l'altra definizione di B_i ottengo

$$\rho_i = \frac{B_i}{\tau} = \frac{C}{\tau} D_i = D_i X$$

che in Cina chiamano ***bottleneck law***. In pratica tutte queste leggi sono definizioni di cose, ma tornano comodo perché alcune grandezze sono più facili da misurare rispetto ad altre.

Bounds su X ed R (i.e. Come ottimizzare il sistema?)

Che ci faccio con tutta questa roba? La uso per capire quali sono gli ***upper bound sul throughput*** e i ***lower bound sul response time*** (e con questo intendo quelli complessivi del sistema).

Essendo normalizzata, $\rho \leq 1$. Dalla ***bottleneck law*** ricavo che

$$D_i X \leq 1 \Rightarrow X \leq \frac{1}{D_i} \Rightarrow X \leq \frac{1}{D_{max}} \qquad \left[\text{Sarebbe } X \leq \frac{1}{D_{max}} \leq \frac{1}{D_i}\right]$$

Il server che all'interno del sistema complessivo ha la massima demand è il ***bottleneck*** per X;

- Segue dalla ***Little's Law*** che

$$N = X(R+Z) \Rightarrow R = \frac{N}{X} - Z \geq \frac{N}{X_{max}} - Z = ND_{max} - Z$$

È chiaro che se $N=1$ avrò sempre un response time minore rispetto al caso $N>1$. Quindi

$$R(N) \geq R(1)$$

Ma il tempo di processamento di un singolo job $R(1)$ si può ottenere facilmente come somma di tutte le demand dei singoli server i (posso farlo perché con un solo processo non c'è ritardo in coda). Quindi

$$R \geq \sum_i D_i = D_{tot}$$

- Segue dalla ***Little's Law*** che

$$X = \frac{N}{(R+Z)} \leq \frac{N}{R_{min}+Z} = \frac{N}{D_{tot}+Z}$$

In definitiva abbiamo che

$$X \leq \min\left\{\frac{N}{D_{tot}+Z}, \frac{1}{D_{max}}\right\}$$

$$R \geq \max\left\{D_{tot},\, ND_{max} - Z\right\}$$

In entrambi i casi, l'intersezione tra le due curve è data da

$$N^* = \frac{D_{tot}+Z}{D_{max}} = \frac{D_{max} + \sum_{i \neq max} D_i Z}{D_{max}} = 1 + \frac{\sum_{i \neq max} D_i + Z}{D_{max}}$$

L'ultima è giusto per sapere che sicuramente l'incrocio avviene dopo 1. N^* separa infatti i regimi di

- ***low population*** ($N < N^*$), in cui per migliorare le performance è sufficiente ridurre D_{tot};
- ***high population*** ($N < N^*$), in cui per migliorare le performance bisogna ridurre D_{max}.