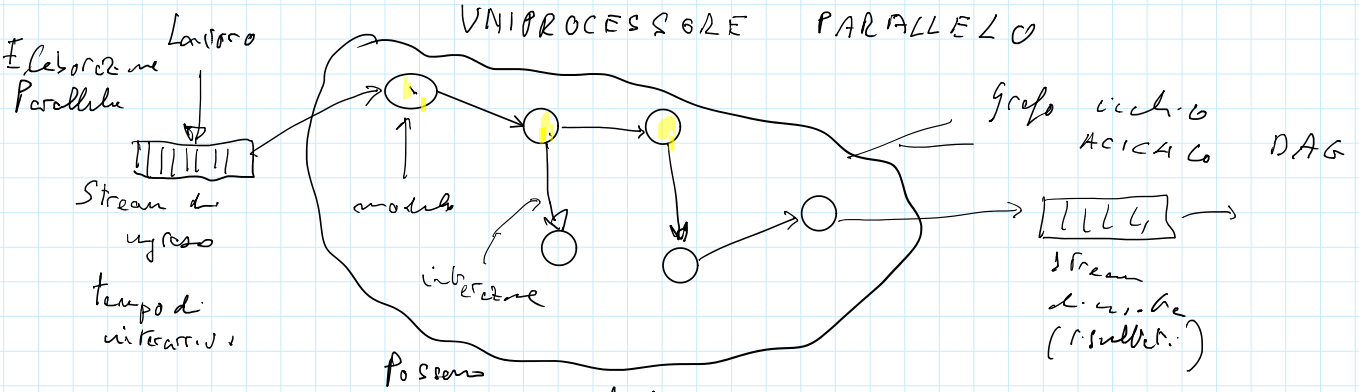
ARCHITETTURE PARALLELE

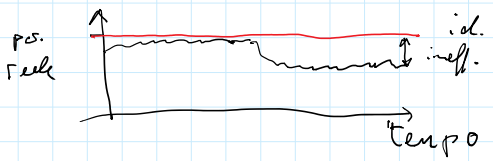
UNIPROCESSORE PARALLELO



Possono operare in parallelo su richieste ≠ dello stream in ingresso

Grado di parallelismo
- ideale = n° di moduli
- reale ≡ n° di moduli che operano contemp.

p. reale < p. ideale ⇒ inefficienza

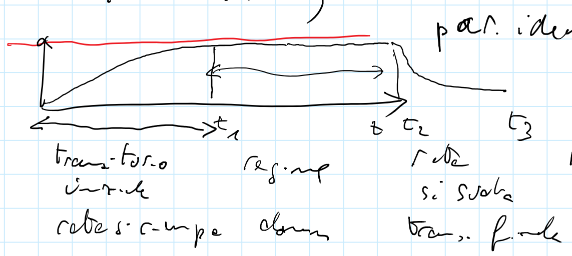


Eq. funz. = Ret. diverse producono lo stesso risultato (stesso output dallo stesso input)

martedì 22 novembre 2016 11:53

- Latenza ≡ tempo per elab. una richiesta nello stream in ingresso → min / max / medio $L^{(n)}$
- Tempo di servizio ≡ tempo tra due prelievi nello stream di ingresso = $T^{(n)}$ n = par. id.
- Banda di elab = $1/T^{(n)}$
- Tempo di compl. per m richieste, $m >> n$ $T_c^{m} = m * T^{(n)}$

(possiamo avere dei transitori)

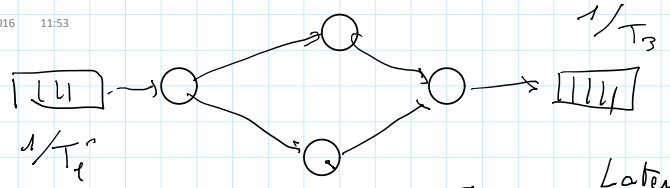


$(t_3 - t_2) + (t_1 - 0)$ tras. risp. $(t_2 - t_1)$

EFFICIENZA ≡ $\frac{B^{(n)}}{B^{(1)}} = \frac{T^{1}}{T^{(n)}} \equiv \varepsilon$

SCALABILITÀ ≡ $\frac{B^{(n)}}{B^{(1)}}$ = $S = \varepsilon * n$

martedì 22 novembre 2016 11:53



Latenza $= T_1 + T_2 + T_3$ $(T_1 \geq T_2 \geq T_3)$

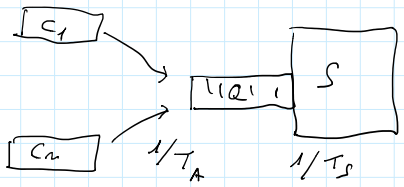
Tempo di servizio $= T_1$ $T^{(4)} = T_1$

Banda $= \frac{1}{T_1}$

$B^{(4)} = T_1 + 2T_2 + T_3$ (pr hyp)

$\varepsilon = \frac{1}{T_1} / \frac{1}{T_1 + T_2 + 2T_3}$

Teoria delle Code



$C_1, \dots, C_n$ = clienti → frequenza di int. ≡ somma delle freq.

FIFO ≡ S serve richieste nell'ordine di arrivo

tempo di servizio = T_S ≡ tempo medio
σ_S ≡ varianza
$N = 1/T_S$ = frequenza del servizio

tempo di interarrivo = T_A
$1/T_A$ = frequenza di interarr. = λ

ρ ≡ grado di congestione

$$\equiv \frac{T_S}{T_A} \simeq \frac{\lambda}{N}$$

Tempo di uscita
T_A se $\rho < 1$
T_S se $\rho \geq 1$

Banda richiesta = $1/T_A$

Banda fornita = $1/T_S$

Banda fornita ≥ Banda richiesta ⇒ $\rho \leq 1$

lunghezza coda media cresce con ρ

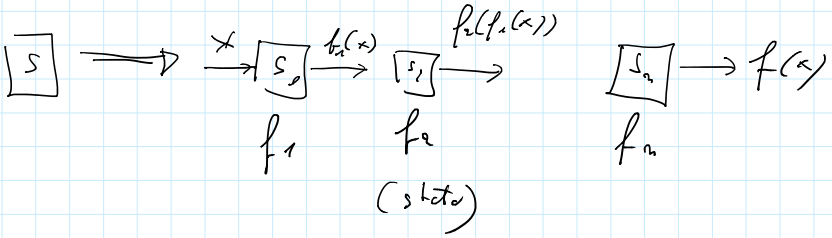
martedì 22 novembre 2016 11:53

Forme di parallelismo

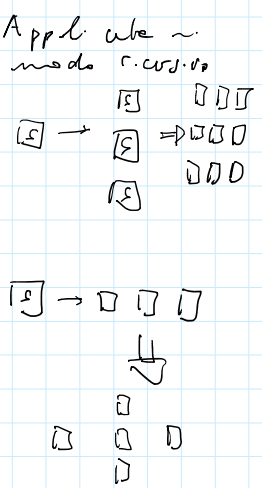
1) replicazione

~ copie dello stesso servizio (stesso ??)

2) pipeline $S \equiv f(x) \longrightarrow f = f_n(\dots f_1(x)\dots)$



Applicazione ~ modo ricorsivo



Pipeline

Nessuno è più lento degli altri

Bilanciamento = $f_1 \to T_1$, $f_2 \to T_2$ ... $f_n \to T_n$

Bilanciato se $T_1 = T_2 = \dots = T_n = T/n$ $T \equiv$ tempo di f

$T_i >> T_1 = T_2 = T_{i-1} = T_{i+1} = \dots = T_n$

T_{i-1} T_i $T_{i-1} < T_i \Rightarrow \frac{1}{T_{i-1}} > \frac{1}{T_i}$

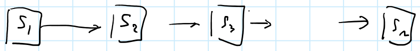
$\equiv$ Pipeline funziona alla velocità del più lento

Pipeline bilanciata $\equiv$ Banda $= \frac{1}{(T/n)} = \frac{n}{T}$

Latenza pipeline $= T$

Non bilanciato $T_p = \max(T_1, \ldots T_n)$

Stadi del pipeline



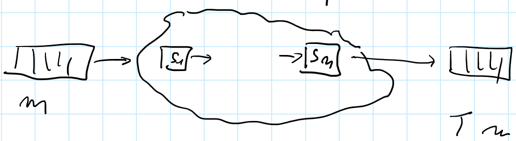
Bilanciato $= T/n = T_p$

Tempo di riempimento del pipeline $(n-1)\left(\frac{T}{n}\right)$

Dopo Transitorio produce valori in uscita

freq. $= \frac{n}{T}$, Tempo $\frac{T}{n}$ fra due elem.

m elementi nella coda in ingresso

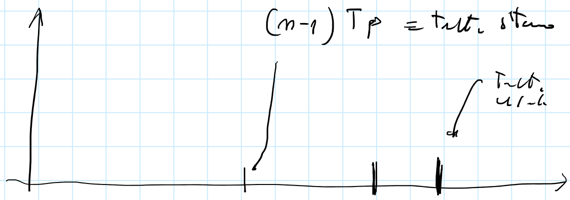


Tempo di completamento per m element.

T m?

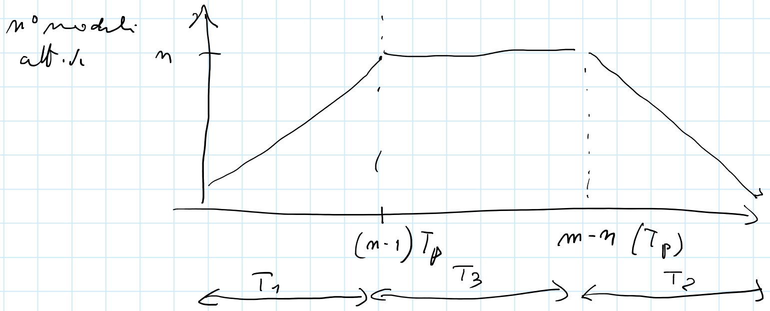
$(m >> n-1)$

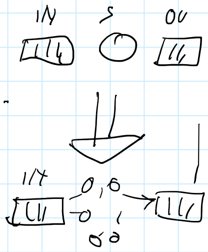
$(n-1)\,T_p$ = tutti stanno lavorando



$T_p(n-1) \quad m * T_p$

$\underbrace{m * T_p}_{\text{regime + svuota}} + \underbrace{(n-1)\,T_p}_{\text{Trans. iniziale}} \simeq m\,T_p$





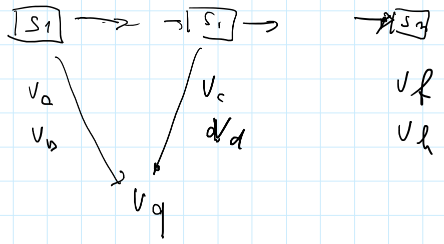
$$T_1 + T_2 + T_3 \simeq T_3$$

$$\Rightarrow \frac{T_1 + T_2}{T_3} \simeq 0$$

$S \longleftrightarrow V_1, \dots, V_n$

Dipendenze

Si può operare su una richiesta solo se una richiesta precedente è stata elaborata da uno stadio successivo



Dipendenze tra stadi del pipeline
⟹ due stadi devono leggere/scrivere le stesse variabili
stato condiviso tra gli stadi

V_q condivise ≡ vincoli sulle operazioni.
Si può lavorare su V_q solo se una certa richiesta è stata trattata da S_{i+k}

- w - w write after write
- r - w read after write
- w - r write after read