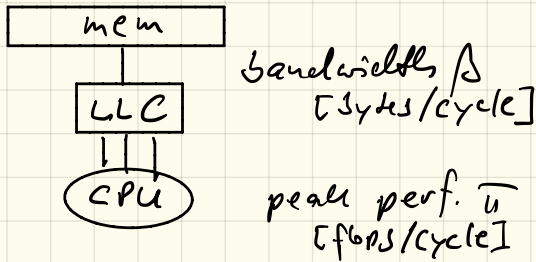


Design of Parallel & High Performance Computing
Reasoning about Performance II

Roofline Model - Balance Principles - Analysis Scheduling

Roofline Model (Williams et al. 2008)

Computer:



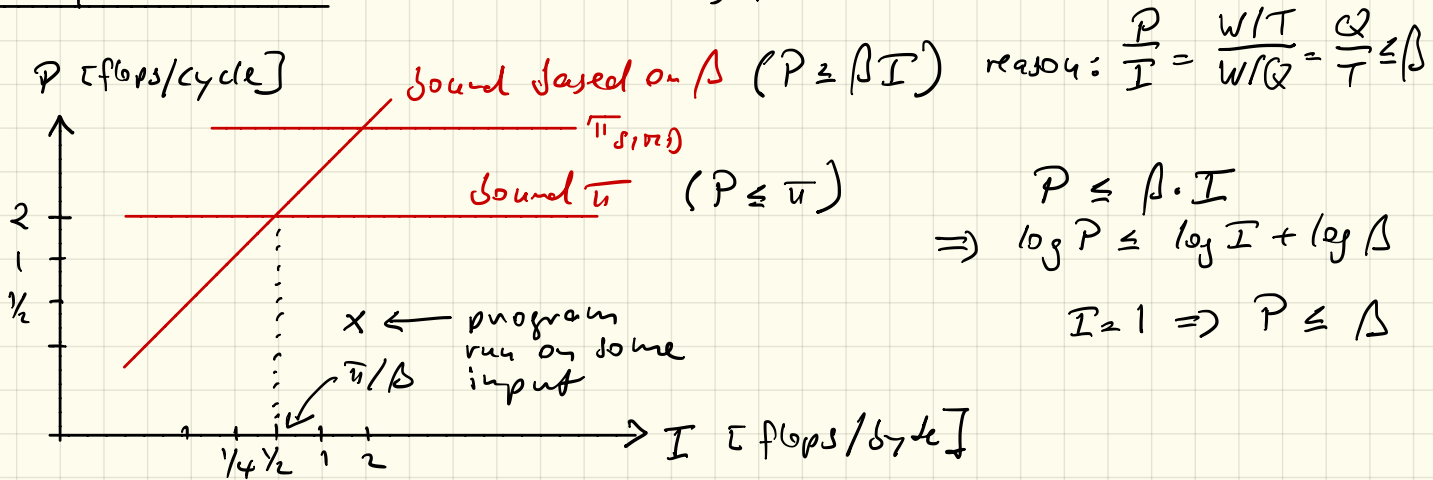
Program:

$$I = W/Q \quad [\text{flops/byte}]$$

$$T = \text{runtime} \quad [\text{cycles}]$$

$$P = W/T \quad (\text{performance}) \quad [\text{flops/cycle}]$$

Roofline plot: (example $\bar{u} = 2$, $\beta = 4$)



Operational Intensity: Upper bounds

$$I(u) = W(u) / Q(u)$$

#flops / #bytes
mem \leftrightarrow cache

x, y : length n , $A, B, C = n \times n$, α : scalar

		asymptotic	best	exact
$daxpy$	$y = \alpha x + y$	$O(1)$	$O(1)$	$\leq 1/12$
$dgemv$	$y = Ax + y$	$O(1)$	$O(1)$	$\leq 1/4$
fft		$O(\log n)$	$\Theta(\log p)$	—
$dgemm$	$C = AB + C$	$O(n)$	$\Theta(\sqrt{p})$	$\leq \frac{n+1}{16}$

$p = \text{cache size}$

should be sharp if $3n^2 \cdot 8 \leq p$

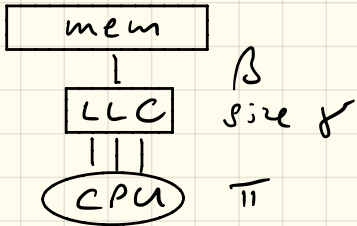
Example: $p = 12773$

$\Rightarrow n \leq 700$

back to slides \rightarrow measured roofline plots

B. Balance Principles I (Kung 86)

Computer:



Program:

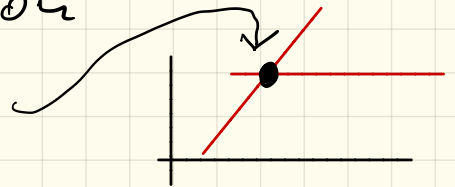
Q_γ : data transfer $\text{mem} \leftrightarrow \text{LLC}$

W : work = # ops

The computer is called **balanced** if
compute time = data transfer time

i.e., assuming perfect utilization

$$\frac{W}{\pi} = \frac{Q_\gamma}{\beta} \Leftrightarrow \frac{\pi}{\beta} = \frac{W}{Q_\gamma}$$



assume $\pi \rightarrow a\pi$ increases; how to rebalance?

a.) $\beta \rightarrow a \cdot \beta$ ✓

b.) $\gamma \rightarrow x \cdot \gamma$

but β grows slower than π !
what is x ?

Example 1: Matrix multiplication $C = AB + C$
algorithm with optimal $I = \frac{W}{Q_{\gamma}} = \Theta(\sqrt{\gamma})$

$$\frac{\overline{W}}{\overline{B}} = \frac{W}{Q_{\gamma}} = \Theta(\sqrt{\gamma}) \quad \overline{w} \rightarrow a\overline{w}, \gamma \rightarrow a^2 \cdot \gamma$$

Example 2: FFT / Sorting
algorithm with optimal $I = \Theta(\log \gamma)$

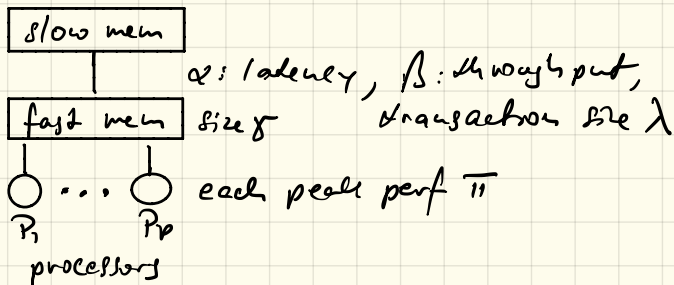
$$\frac{\overline{W}}{\overline{B}} = \frac{W}{Q_{\gamma}} = \Theta(\log \gamma) \quad \overline{w} \rightarrow a\overline{w}, \gamma \rightarrow \gamma^a$$

Both are unrealistic.

7. Balance principles II (Czechowski et al. 2011)

Goal: more detailed principles for multicores,
algorithm/architecture co-design, assessment of HW trends

Computer:



Algorithm:

PRAM: W : work, D : depth

$Q_{p,\lambda}$: mem. transfers of size λ

assumptions: optimal W , W/Q

How to get $Q_{p,\gamma,\lambda}$ from $Q_{\gamma,\lambda}$?

- there are general bounds and some direct results

Processor is balanced if

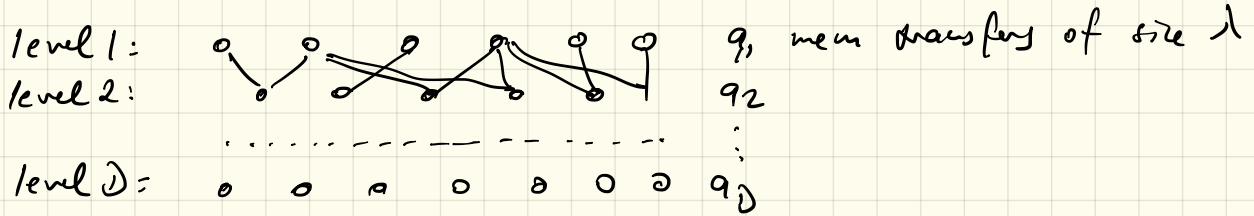
$T_{mem} \leq T_{comp}$ ("compute-bound")

Using:

$$\frac{Q \cdot \lambda}{\beta} \leq \frac{W}{p \cdot \pi} \Leftrightarrow \frac{p \pi}{\beta} \leq \frac{W}{Q \lambda}$$

Derivation principles:

1.) Estimate T_{mem} : Idea divide DAG in levels



$$\Sigma = Q_{\lambda, \lambda}$$

In each level α - β model: $\alpha + \frac{q_{i-1} \cdot \lambda}{\beta}$

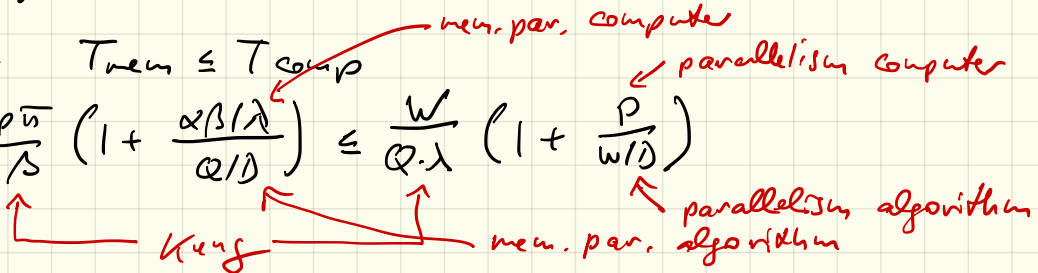
$$\Rightarrow T_{mem} \approx \sum_{i=1}^D \left(\alpha + \frac{q_{i-1} \cdot \lambda}{\beta} \right) = \alpha \cdot D + \frac{Q \cdot \lambda}{\beta}, \quad Q = Q_{\lambda, \lambda}$$

2.) Estimating T_{comp} : Brent's lemma

$$T_{comp} \approx \left(D + \frac{W}{P} \right) \cdot \frac{1}{\pi}$$

Balance: $T_{mem} \leq T_{comp}$

$$\Leftrightarrow \frac{p \bar{u}}{\beta} \left(1 + \frac{\alpha \beta \lambda}{Q \lambda} \right) \leq \frac{W}{Q \cdot \lambda} \left(1 + \frac{P}{W/D} \right)$$



Example 1: matrix multiplication

use $Q \geq \frac{W}{\sqrt{2} \cdot \lambda \sqrt{r/p}}$ (Irony et al. 2004) $\Rightarrow I(u) = O(\sqrt{\frac{r}{p}})$

and $\delta \ll W, \delta \ll Q$ (\ll means much smaller)

\Rightarrow balance principle $\frac{p^u}{\lambda} \leq O(\sqrt{\frac{r}{p}})$

$\pi \rightarrow a \cdot \pi$ rebalance: $\lambda \rightarrow a\lambda$ or $r \rightarrow a^2 r$
 $p \rightarrow a \cdot p$ " : $(\lambda \rightarrow a\lambda \text{ and } r \rightarrow ar)$ or $(r \rightarrow a^3 r)$

Example 2: bounding $\sqrt{r/p}$

$\Rightarrow \frac{p^u}{\lambda} \leq O(\log \frac{r}{p})$

Back to roles \rightarrow Evolution of balance

Greedy scheduling: Analysis

Reminder: $T_1 = W$, $T_{\infty} = D$ $T_p \geq T_1/p, T_{\infty}$ $T_p \leq D + W/p = T_{\infty} + T_1/p$

Greedy scheduler:

complete steps $\leq T_1/p$

incomplete steps $\leq T_{\infty}$ (every incomplete step reduces critical path by 1)

$$\Rightarrow T_p \leq T_1/p + T_{\infty}$$

How far from optimal?

Assume T_p^* is optimal:

$$T_p^* \geq \max\{T_1/p, T_{\infty}\}$$

$$T_p \leq T_1/p + T_{\infty} \leq 2 \max\{T_1/p, T_{\infty}\} \leq 2T_p^* \quad \text{at most factor 2 away}$$

enough parallelism: $W/D = T_1/T_{\infty} \gg p \Leftrightarrow T_{\infty} \ll T_1/p$

$$\Rightarrow T_p \leq T_1/p + T_{\infty} \approx T_1/p$$

Work stealing scheduling: Analysis

Theorem: The scheduler achieves $T_p \leq T_1/p + O(T_{\infty})$

Proof sketch: Every processor is either working or stealing

Total time working: T_1

Total time stealing: Every steal has $1/p$ chance to reduce the critical path; hence $O(p T_{\infty})$

\Rightarrow total time: $T_1 + O(p T_{\infty})$

\Rightarrow time per processor: total time $/ p = T_1/p + O(T_{\infty})$

Note: Space requirement $S_p \leq p S_1$.