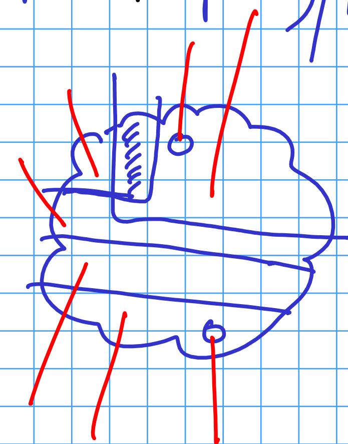
SEQ

loop
conditional
call procedures
methods

(PAR)

$(\#points * T_{point}) / \#cores$



decrease latency

DATA PARALLELISM

large par degrees

PARALLELISM

STREAM PARALLELISM

small par degree

CONTROL PARALLELISM

parallelism from the control structure of your computation

$X_m \ldots X_2\, X_1 \rightarrow$

t_a

$f(X_m) \ldots f(X_2)\, f(X_1) \rightarrow$

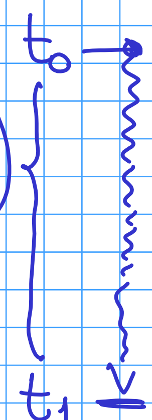
decrease service time

increase throughput

$X \longrightarrow f(x)$

par degree n

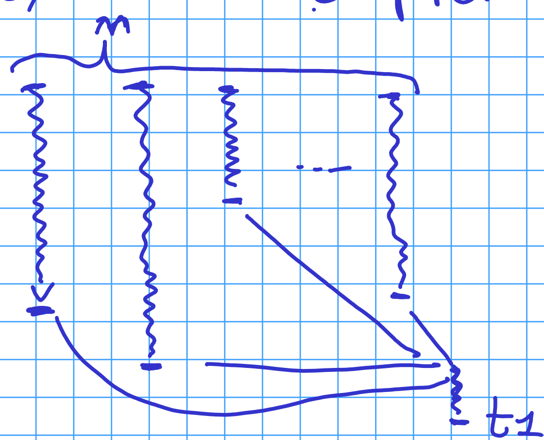
$T_{seq} = (t_1 - t_0)$



$T(n) = (t_1 - t_0)$

Latency (for 1 task)

Completion time → m (tasks results)



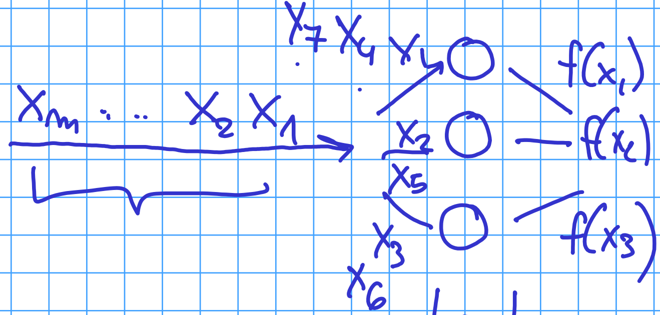
Service time

↓

time between the delivery of 2 consecutive results

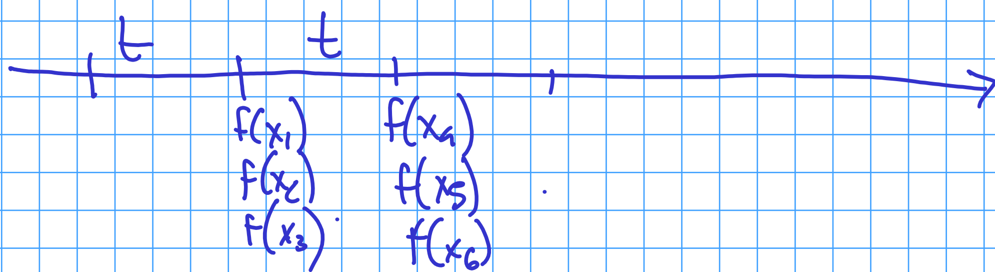
in seq computation

Service time = latency..

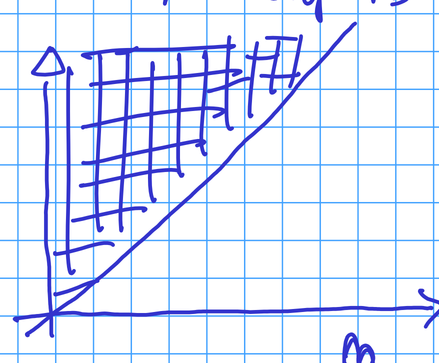


$$T_S(3) = \frac{t}{m}$$

$L = t$



Completion time for m inputs is the latency for computing all the m inputs

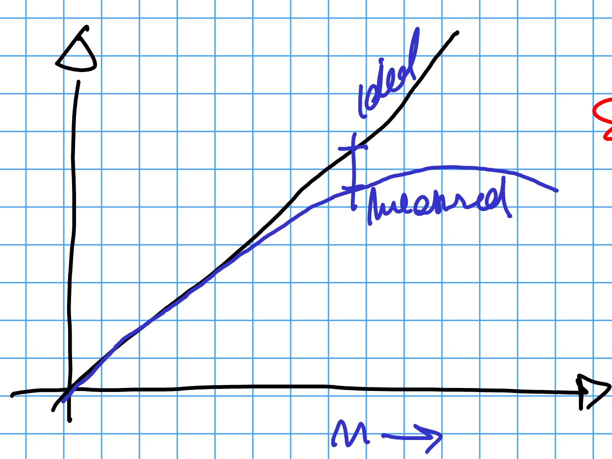


$$\text{Speedup}(m) = \frac{T_{seq}}{T(m)}$$

↓

par degree

~~$sp(10) = 12$~~

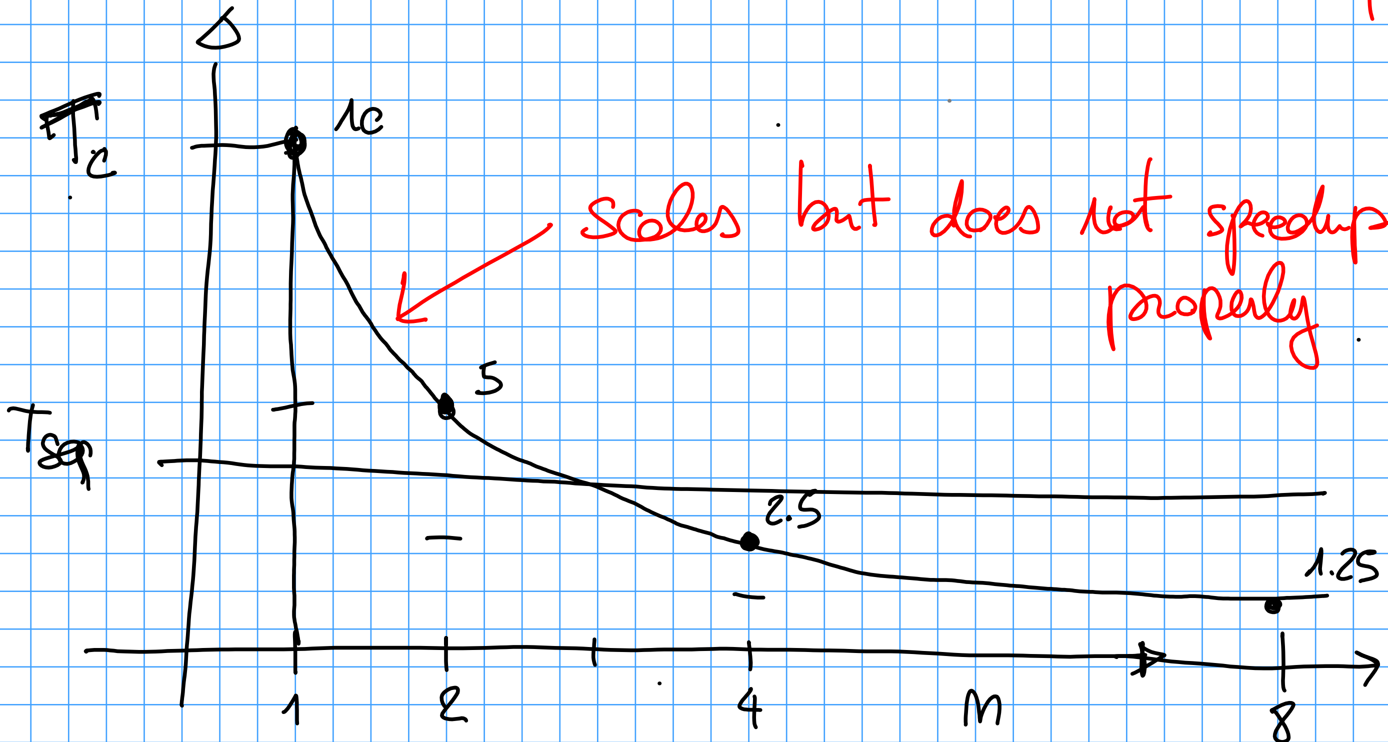


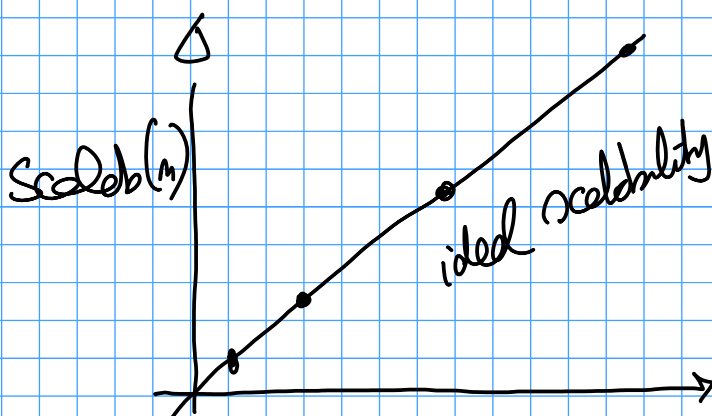
$$\text{speedup}(m) \leq m$$

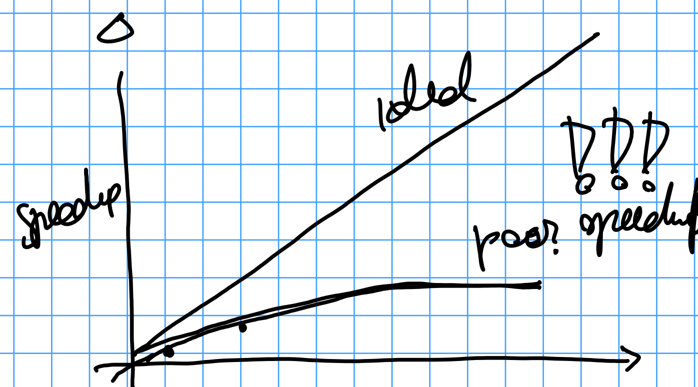
$$\text{scalability}(n) = \frac{T(1)}{T(n)}$$

$T(1) \neq T_{seq}$

$T(1) > T_{seq}$

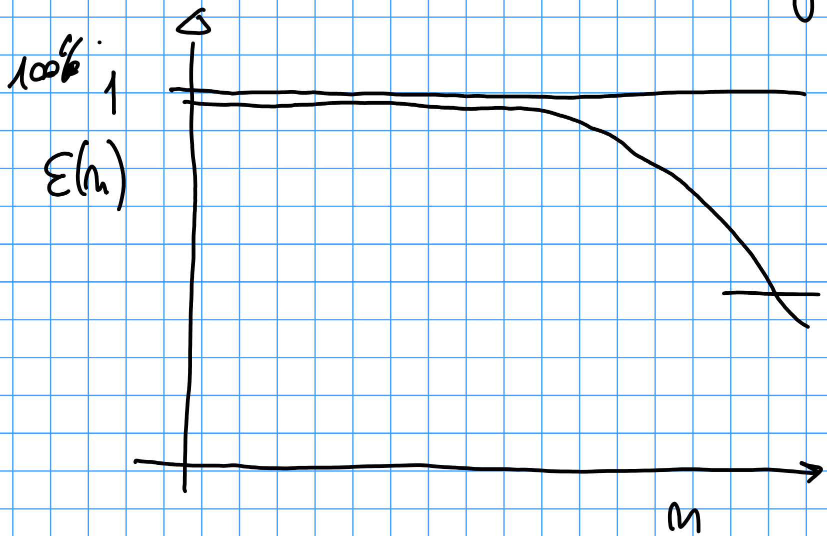




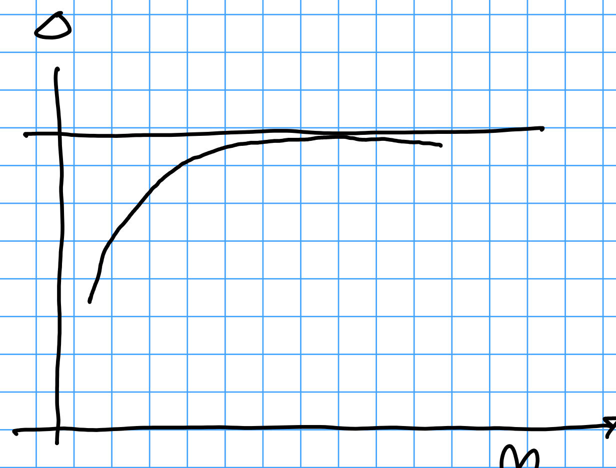


$$\text{efficiency}(n) = \frac{T_{seq}}{n \times T(n)} = \frac{\text{speedup}(n)}{n}$$

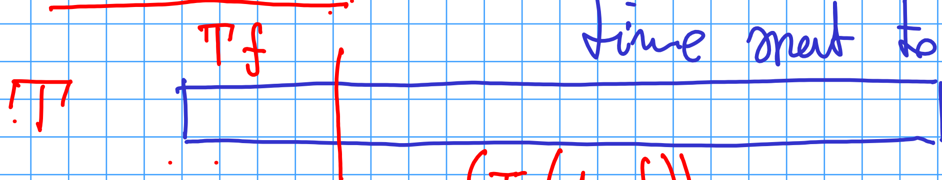
$$\mathcal{E}(n) \leq 1$$



good for program has $\mathcal{E}(n)$ close to 95 ÷ 99%



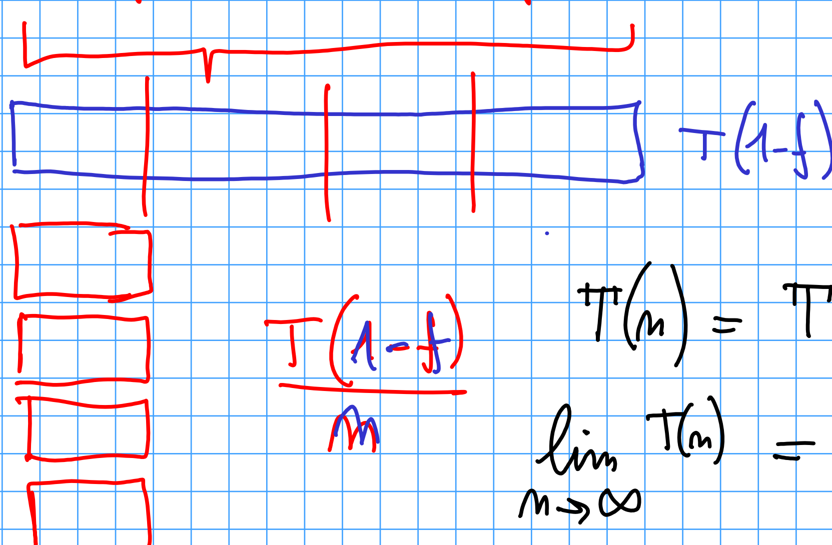
Amdahl



time spent to compute (SEQ) your application

f. serial fraction

part that can be parallelized



$$\frac{T(1-f)}{n}$$

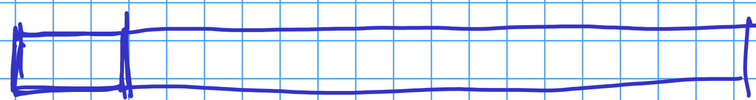
$$T(n) = Tf + \frac{T(1-f)}{n}$$

$$\lim_{n \to \infty} T(n) = Tf$$

$$\text{Speedup}(n \to \infty) = \frac{\not{T}}{\not{T} f} = \frac{1}{f}$$

Gustafsson

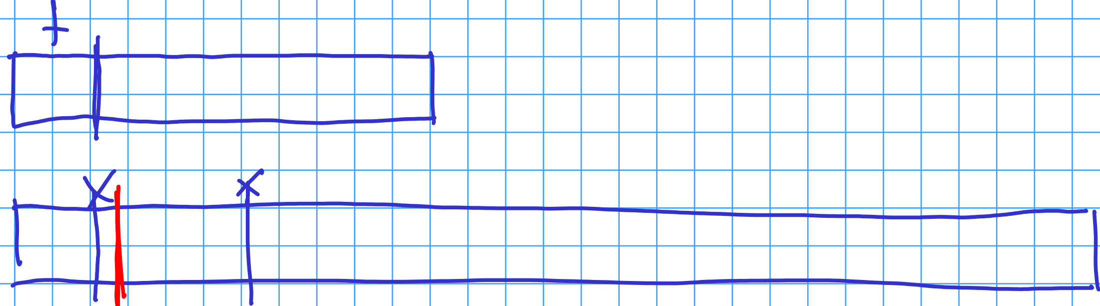
size of the data

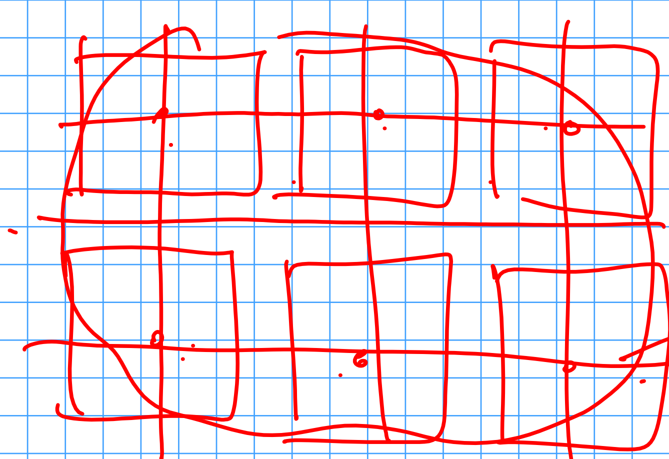


$f = x$

Appl ($dataset_1$)   # $dataset_1$ = K

Appl ($dataset_2$)   # $dataset_2$ = $1M_K$





3 km

t^c premier und direct