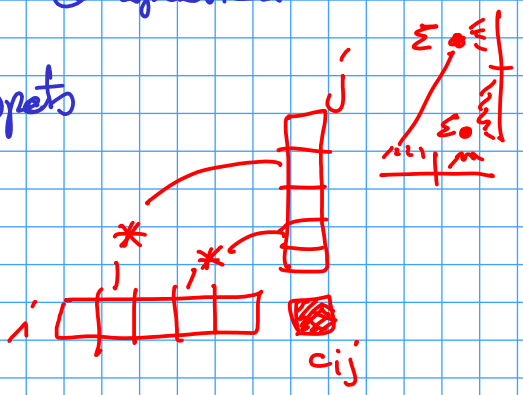
{techniques} for performance improvement

Methods for containing interaction overheads

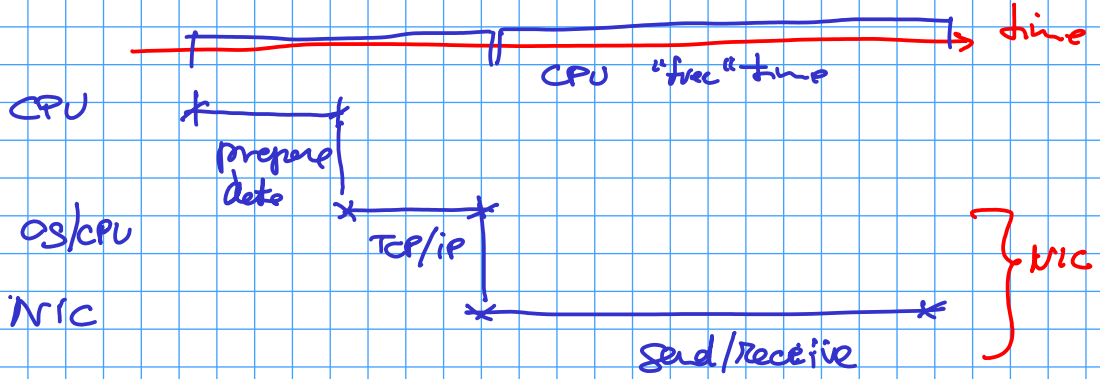
→ Maximize data locality ⟨ storage vs. computation

→ Minimize contention & hot spots

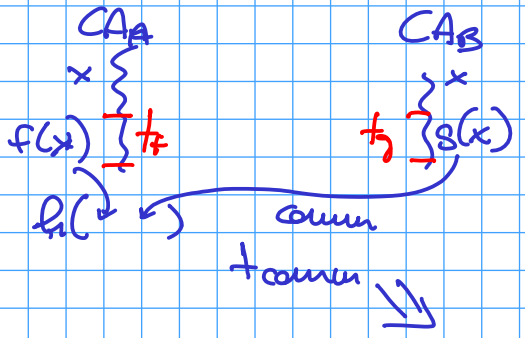
$c_{ij} \; (+=) \; \Sigma_k \, a_{ik} \cdot b_{kj}$



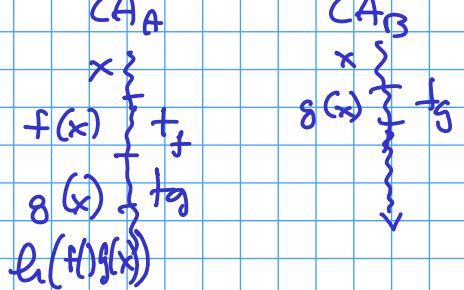
→ Overlap communication & computation



→ Replication of data and/or computation

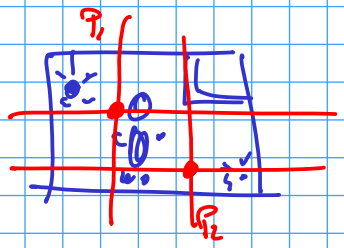


$t_f \approx t_g$



$t_f \approx t_{comm}$

Ray Tracing



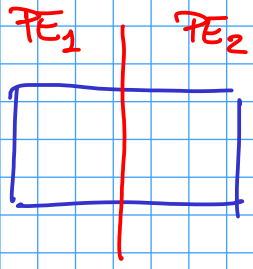
data { scene
light

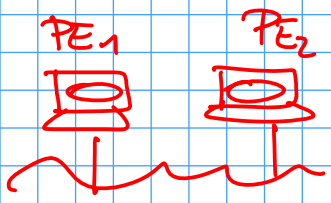
⇓

needed to compute

∀ pixel

with RO access





1st alternative

PE_1 loads scene/light data & computes left part

PE_2 computes right part of the picture

2nd alternative

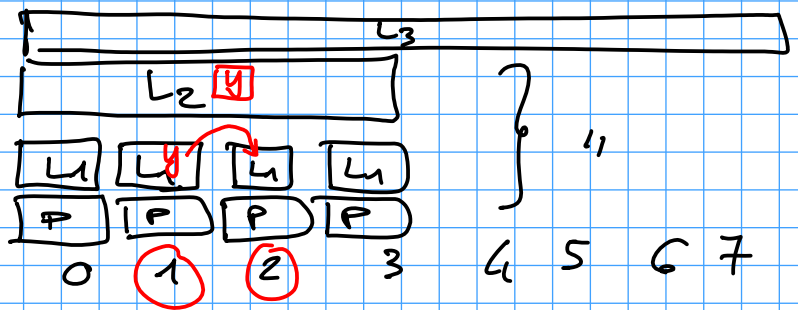
PE_1 sending to PE_2 all data { scene light

then compute left part

PE_2 receive scene data

then compute right part

Multicore

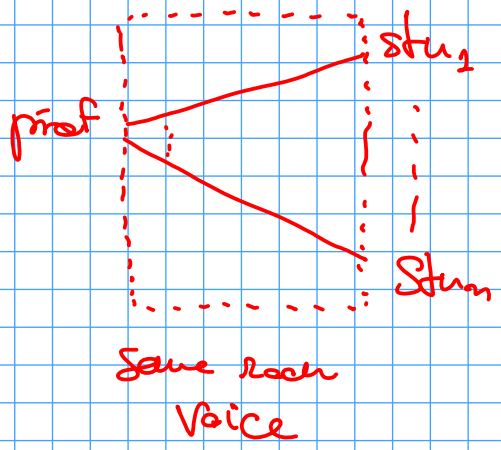


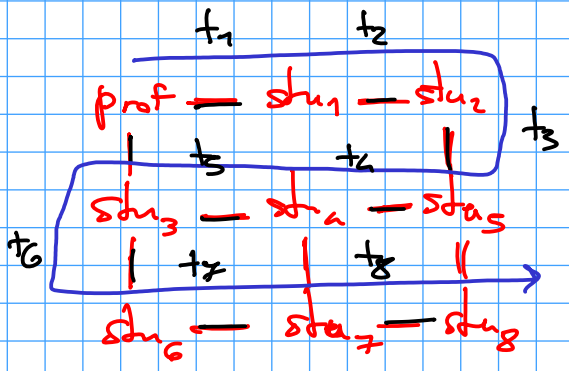
$y = f(x)$ ... $g(y)$

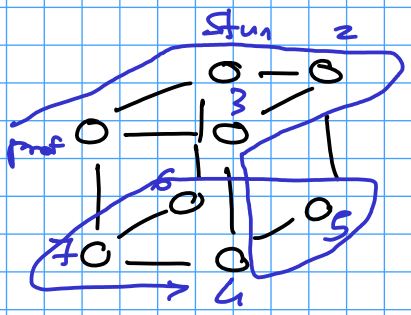
f_{e_1} f_{e_2}

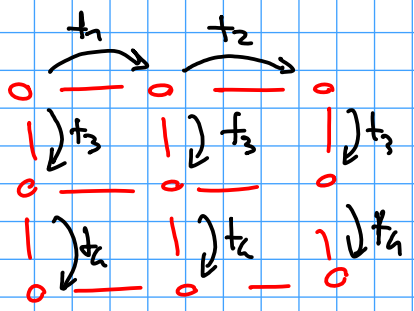
→ Use optimized collective interaction mechanism

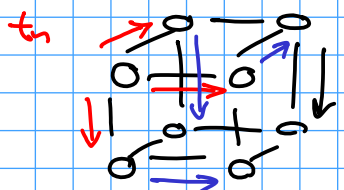
→ Use optimized collective interaction mechanism

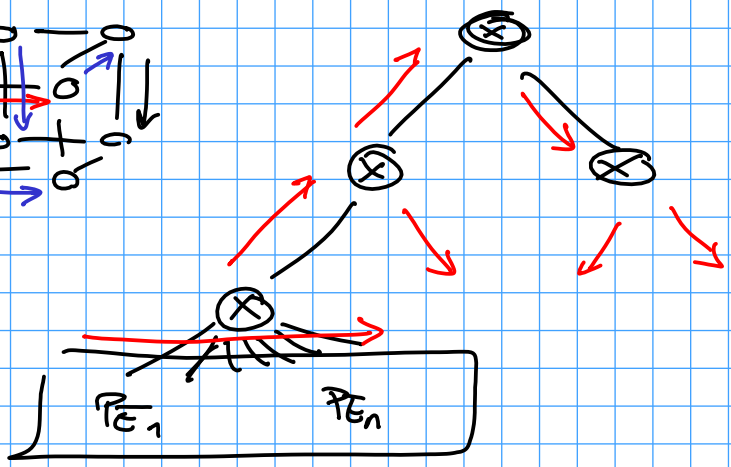












Models

data parallel

task graph

work pool

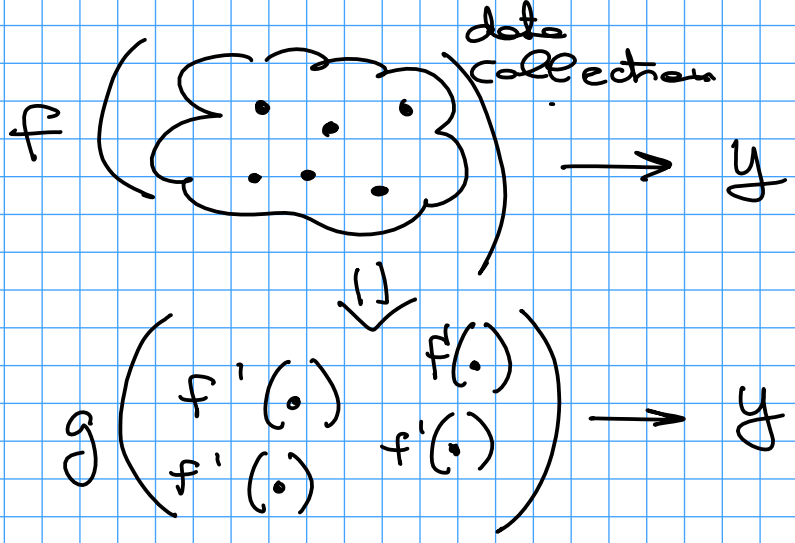
master slave

pipeline

producer consumer

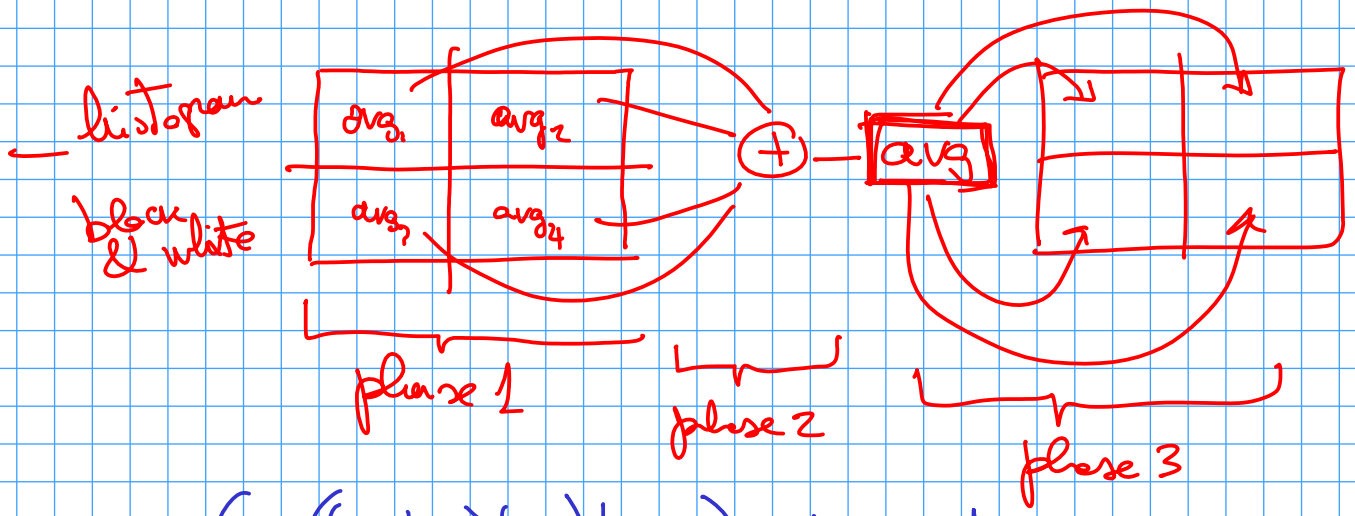
(hybrid)

DATA PARALLELISM



} seq implementation

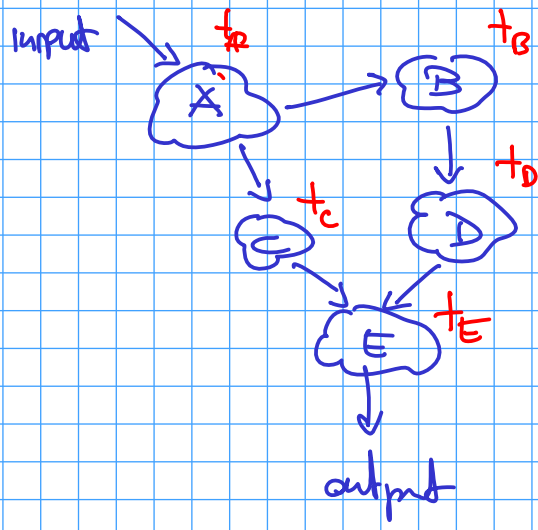
} par / distributed implementation



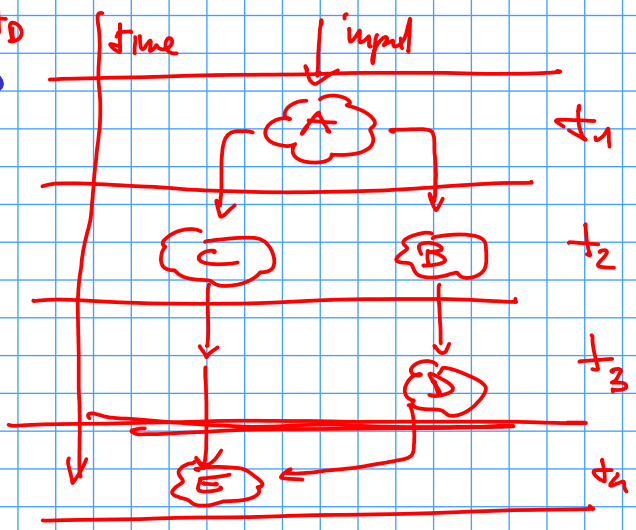
$$\Sigma \quad \left(\left((a_1 + a_2) + a_3 \right) + a_4 \right) \leftarrow 3 * t_{sum}$$

$$\left. \begin{array}{l} + \\ + \\ + \end{array} \right\} \leftarrow 2 * t_{sum}$$

TASK GRAPH

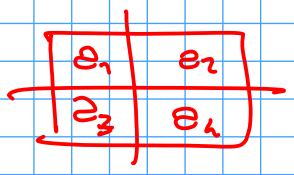


tasks + dependencies



$SEQ = t_A + t_B + t_C + t_D + t_E$

par $t_A + \max\{t_C, (t_B + t_D)\} + t_E$



$\overset{c}{\Sigma} a_i$