DATA SHARING

PROBLEM : decomposition in tasks → how can I share information in efficient way?

identify the shared info → shared data
& ↳ data that cannot be made private to a task
↳ are accessed R/W by multiple tasks

→ to avoid race conditions
→ to use copies for read only sharing
→ in distributed computation
to overlap computation & communications

Special cases :

① – read only sharing

② – effectively local (fully partitioned)
– read write access } accumulator ③ $x \equiv$
multiple readers / single writer

$x_1 \leftarrow CA_1$
⋮
$x_n \leftarrow CA_n$

$x = \Sigma x_i$

FINDING CONCURRENCY

DECOMPOS. DEPENDENCY

task
data

group
ordering
sharing

design evaluation pattern

PROBLEM SOLVED :

evaluation of what we picked up in decomposition and dependency phases

x y z
MULADD

$z += x * y$

targeting the hw

simple
efficient
flexible
} design quality

good for the next steps

Finding concurrency space → **Algorithm structure design space** ↔ Supporting structure space

Organize by Task

- Task parallelism
- Divide & Conquer

Organize by data decomposition

- Geometric decomposition
- Recursive data

Organize by flow of data

- Pipeline
- Event driven coordinatio

TASK PARALLELISM

{ concurrent! tasks }

? how can I orchestrate them concurrently

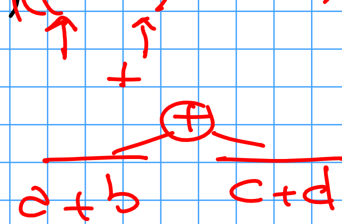
↳ #tasks ⇒ # PE/thread/cores

↳ ensure dependencies < constrains / data

removable (bad coding) $(((a+b)+c)+d)$

separatable

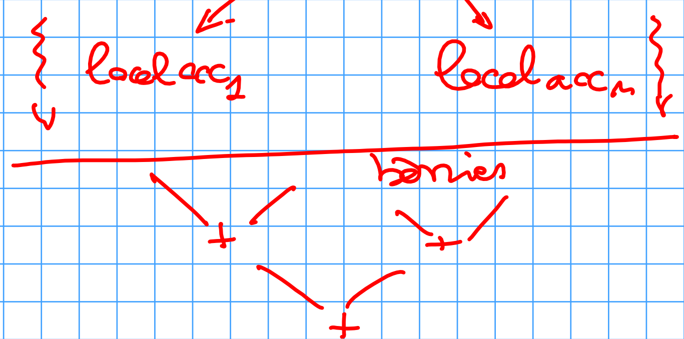
↳ scheduling — load bal. / fairness



program structure

- loop parallelism
- master/worker
- task queue
- SPMD

accumulator

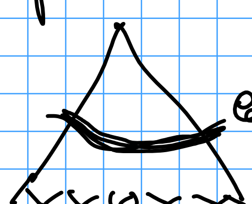


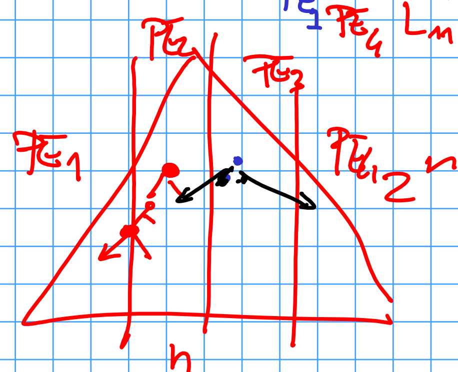
DIVIDE & CONQUER

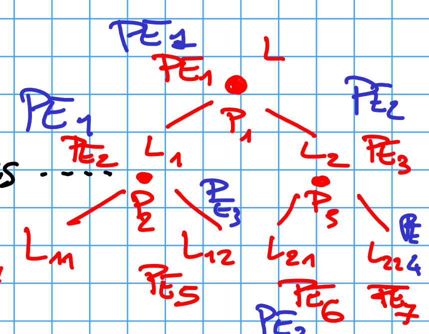
↳ mapping of tasks to PE/threads/cores ...

↳ cost communications

↳ optimizations





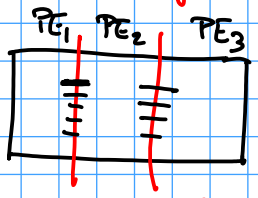


ORGANIZE BY DATA DECOMP.

B: send right receive right

W: receive send

GEOMETRIC decomp



update of clusters happens concurrently

↓ exchange

↓ update

RECURSIVE DATA