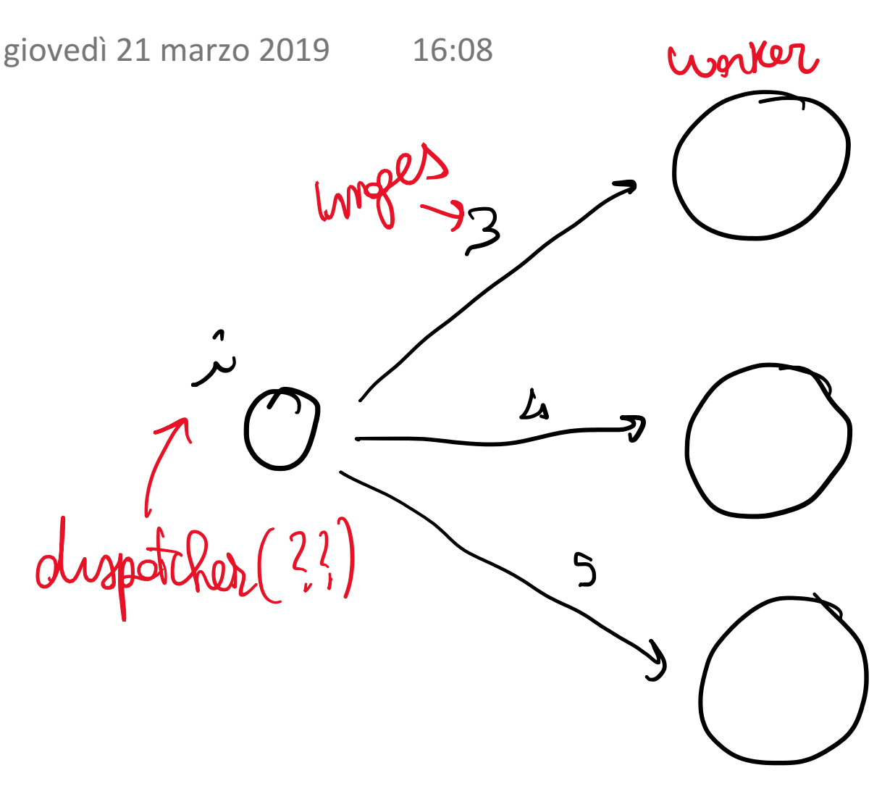
15 Lezione

giovedì 21 marzo 2019 16:08



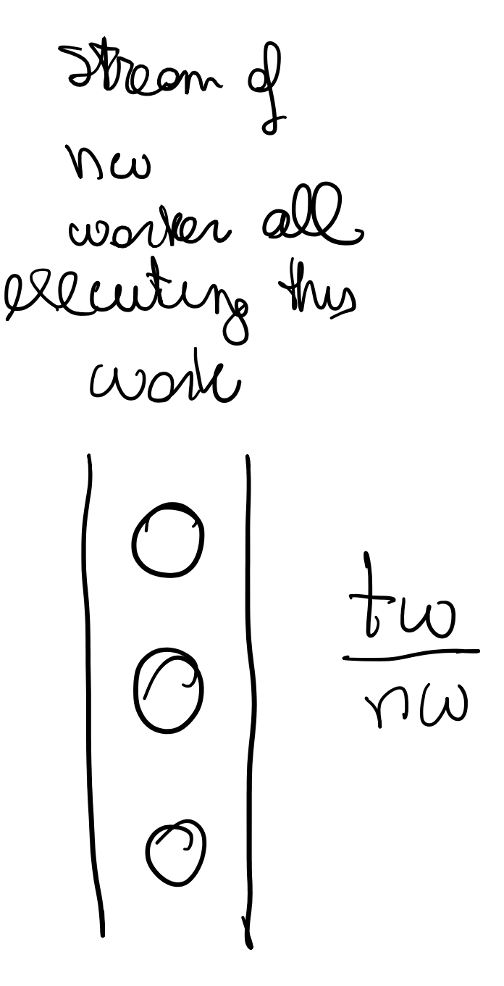
each of this loads the images in 0.6 seconds
Process the image and then write the images on disk

LOAD → 0.6
PROC → 0.4
WRITE → 1.45
} t_w 2.45 · image

with 300 MB/S bandwidth we can observe that:

$$t_w = t_{w_{parallel}} + t_{w_{serial}}$$

↓ time to access disk



Our performance of the stream is $\frac{t_{w_p}}{nw} + t_{w_s} \cdot nw$

↓

questo non può essere diviso ma lo devo moltiplicare perché ognuno lo fa e non in parallelo perché sono risorse condivise

Another Version of the algorithm to solve this problem

→ we could use open mp

①

```
pragma section {
    pragma section
        process image 1
    pragma section
        process image 2
}
```

② Parallel for is the easiest way

③ Pragma omp parallel . . .

{ } to create the thread pool

```
do_img (int i)
for ( 3  i  argc )
    { #pragma omp tasks
    } doimg(i)
```

for vs task

I have to do:

```
pragma omp parallel
    pragma omp single
        for (      ) {
            pragma omp task
                do_img
```

Se Vogliamo usare i pattern?

Seq.cpp → sequential version

```
for (i = 3 → argc) {
    do_img (i) {
        read
        process
        write result
    }
}
```

Trasformare con pattern usando GRPPI.

Per eseguirlo con GRPPI:

```
pipeline ( ________
    stage 1
    stage 2  ← farm
    stage 3  ← possiamo fare in modo
               che restituisca il
               numero di elementi
               già processati.
```

[GRPPI pipeline and the second stage is a farm]

BYPASS HDD BOTTLENECK

→ stamp

→ a.jpg

```
for ( ——→ ) { n = - - - -
    current = a.jpg  → copy of the image, do all the process and eventually discard the result
    }
    —
```

Computation:

load | process | save

farm(pipe(load, process, save))

⇓

pipe(L, P, S)

oppure:

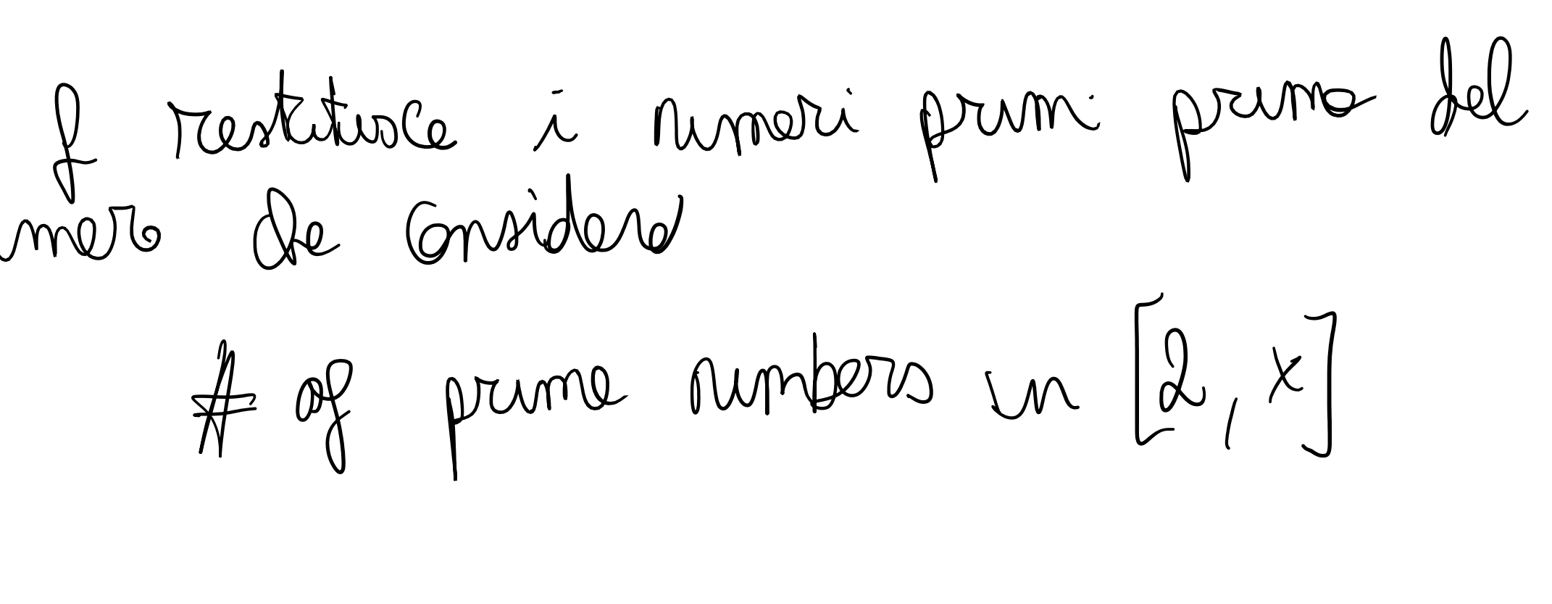
farm(Com(L, P, S))

farm(pipe(load, farm(proc), save))

Key point to evaluate this stuffs:

→ You have to evaluate which is the good solution → timer(??)

ASSIGNMENT 2



Integers in the shared data structure

Le f restituisce i numeri primi prima del numero da considerare

\# of prime numbers in [2, x]

Deal with the task as if they were independent.

parameter:

→ Number of items in shared data structure

Use
- Native thread
- Grppi / openMP (dove avete fatto con Native thread).