RR : Patterns → Patterns

parallel splitter & merger ?s possibility

Map (f) ≡ Pipeline (Splitter, farm (f), Merger)

↑ fun or a skeleton

Seq code taking a collection $\{x_1 \dots x_n\}$ producing a stream of values $x_1 \dots x_n$

Seq code getting a stream of $y_1 \dots y_n$ and producing the collection $\{y_1 \dots y_n\}$

⇒ isomorphic w.r.t. the input collection

→ improve load balancing

Pipe (Pipe (S_1, S_2), S_3) ≡ Pipe (S_1, Pipe (S_2, S_3))

≡ ≡

Pipe (Comp (S_1, S_2), S_3) ≡ Pipe (S_1, Comp (S_2, S_3))

↓↓

Pipe (S_1 S_2 S_3 ... S_k ... S_n)

Comp $S'_{2 \text{ to } k}$

t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
8	9	2	1	3	1	7	8

7

Farm (f) with (Pd (10))

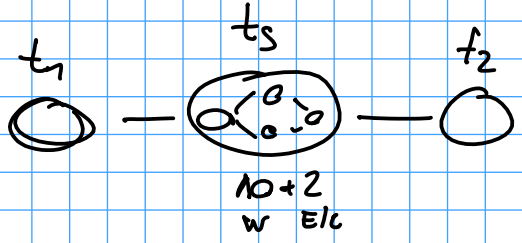
≡

Farm (f) with (Pd (11))

≡

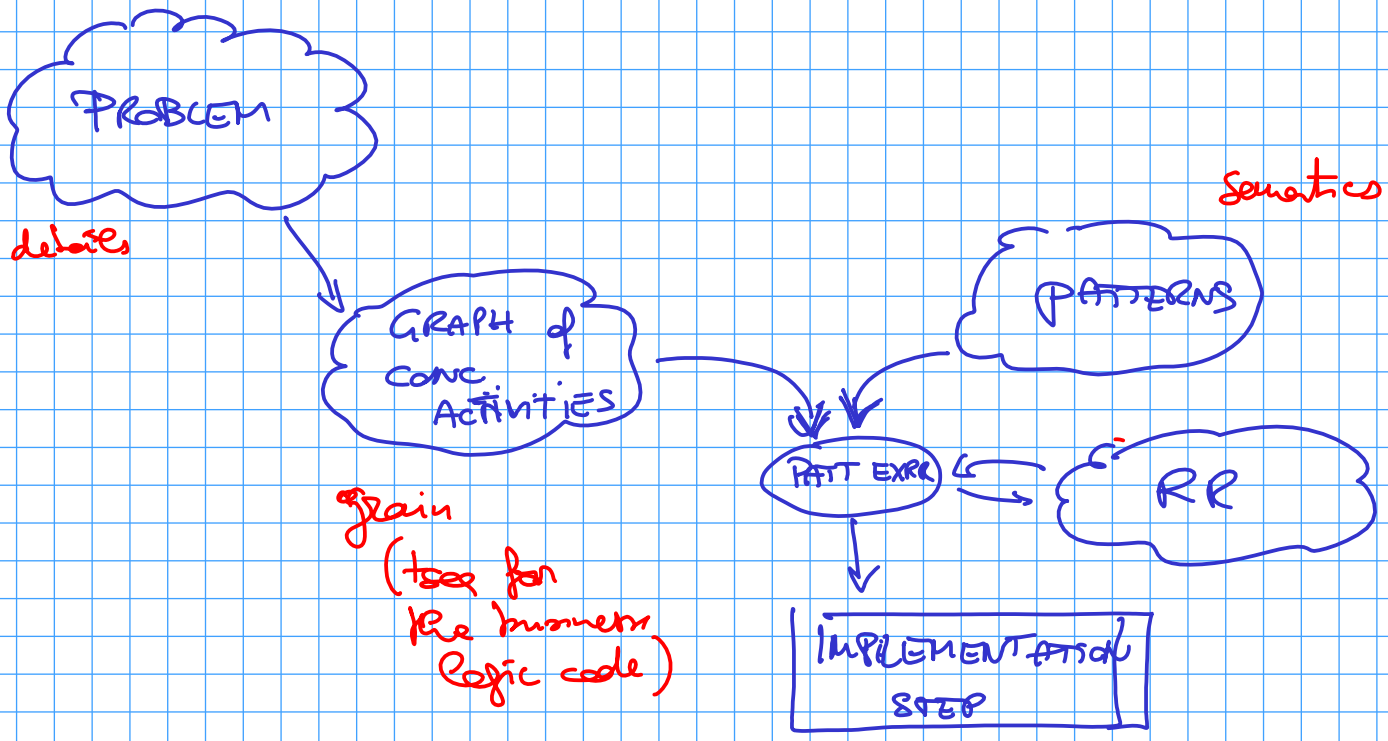
Farm (f) with (Pd (9))

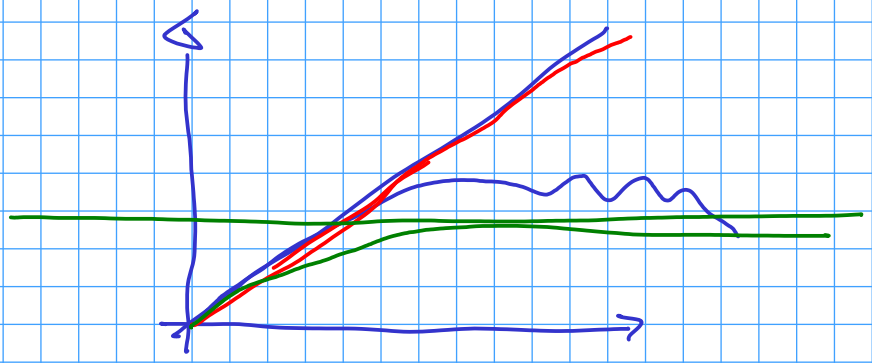
Farm (f) with Pd (1) ≡ f

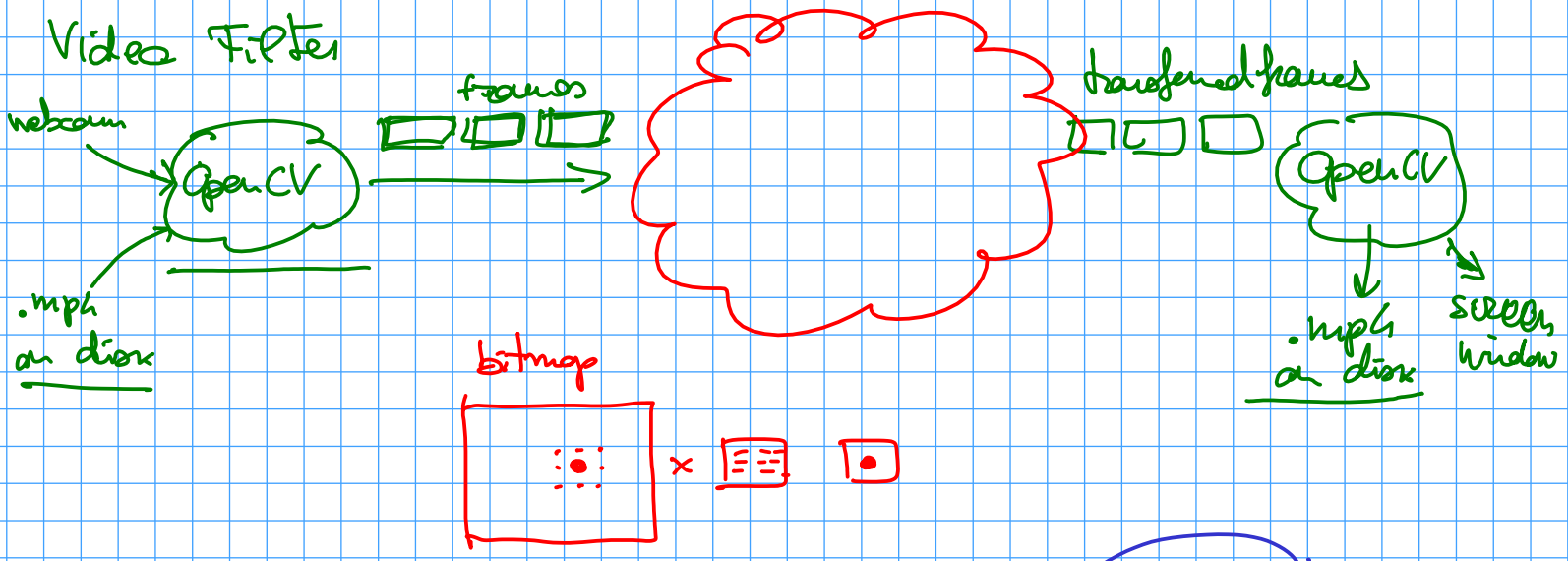


$t_1 \approx t_2 > t_S$

Pd (10) → P(9) for the Farm


PROBLEM
details
GRAPH of CONC ACTIVITIES
PATTERNS
Semantics
PATT EXPR
RR
grain
(test for RR business logic code)
IMPLEMENTATION STEP





Jacobi

$$A x = b$$

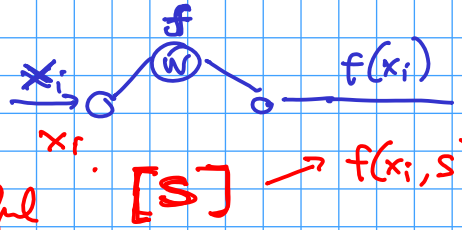
$$\begin{cases} a_{11} x_1 + a_{12} x_2 = b_1 \\ a_{21} x_1 + a_{22} x_2 = b_2 \end{cases}$$

$x_1 = b_1$

$x_2 = b_2$

$$\begin{cases} x_1 = \dfrac{b_1 - a_{12} x_2}{a_{11}} & x_1' \\ x_2 = \dfrac{b_2 - a_{21} x_1}{a_{22}} & x_2' \end{cases}$$

RR → stateless patterns



stateful [S] → $f(x_i, s)$

{ "resource" state

```
f(x, s) {
    lock(s)
    ⋮
    unlock(s)
}
```

}

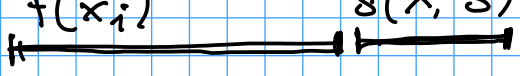
{ accumulator state

s : count of processed items }

{ separate functions to compute the output result out of the input task and to update the state out of the previous state value and of the input task

x_i s

$f(x_i)$ $s'(x_i, s)$



Successive approx pattern

$x_n \cdots x_3 \; x_2 \; x_1 \rightarrow$

S

$f(x_1, S'_0) \rightarrow$

$S'(x_1, S'_0)$

$\uparrow$