Polychronous MoCC for open systems

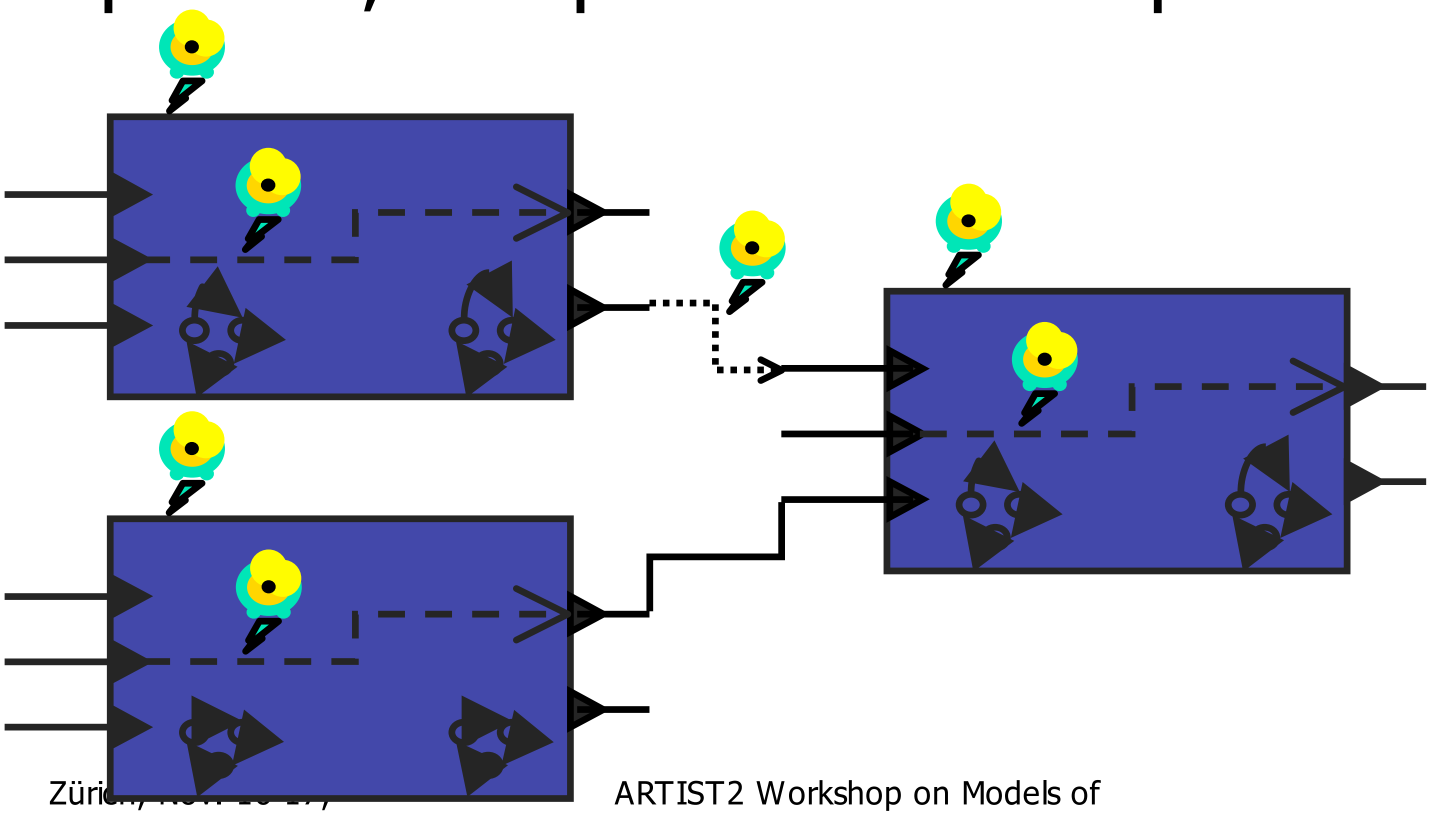
Thierry Gautier, Paul Le Guernic and Lionel Morel

IRISA/INRIA Rennes – France

http://www.irisa.fr/espresso/Polychrony

Introduction

- Component, composition of components



Introduction (cont'd)

- Checking consistency
  - Synchronization
  - Scheduling

- Properties
  - Abstractions (behavioral type)
  - Contracts A $\Rightarrow$ G

Trace on observables

$$T \quad : \mathcal{T} \rightarrow (V \rightarrow D)$$

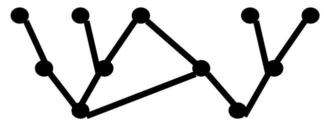
Outline

- Polychronous behaviors
- Assume/Guarantee specifications
- Using components

1.1 Abstraction of behavior

A component is abstracted by the definition of time relations on its observables.

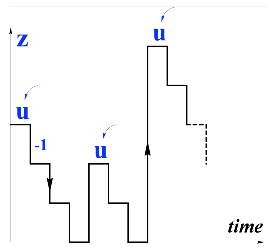
- Requirements on time domain



  - Ω partial order, inf-semilattice

$x = a \mid y = b$

  - Ω_x of a signal is a total order $\quad x_{t+1} \leftarrow x_t$



  - Ω is dense:
    context adaptation, implementation refinement
  - Any finite family of sets over Ω has an upper bound

$x = a \text{ default } b$

1.1 Abstraction of behavior (cont'd)

- Process: set of traces

  sub-process: set of traces on a subset of V

  - $P \subset \mathcal{T} \rightarrow (V \rightarrow D)$

counter of seconds

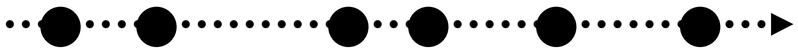
counter of minutes

1.2 Composition of time domains

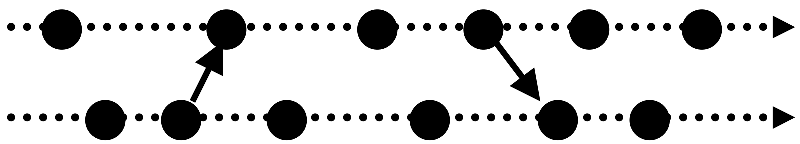
- Question
  - Processes $P_1 \subset \mathcal{T}_1 \rightarrow (V_1 \rightarrow D_1)$ and $P_2 \subset \mathcal{T}_2 \rightarrow (V_2 \rightarrow D_2)$
  - Which time domain for $P = P_1 \mid P_2$?

    $\mathcal{T} \rightarrow (V_1 \cup V_2 \rightarrow D_1 \cup D_2)$
- Different approaches
  - $\mathcal{T}_1 = \mathcal{T}_2$ is a universal time

SCCS

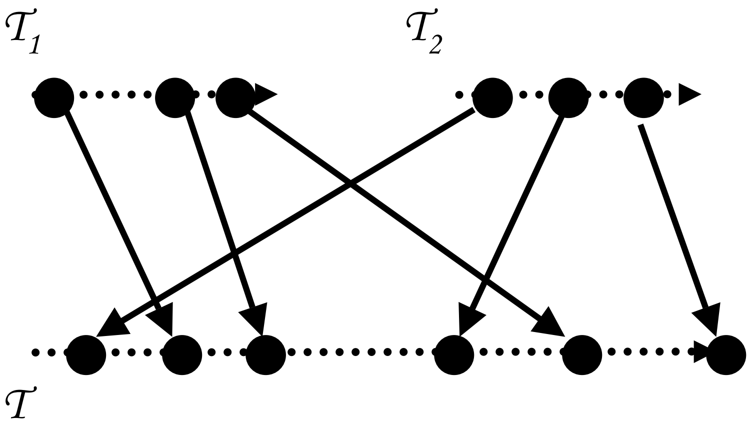


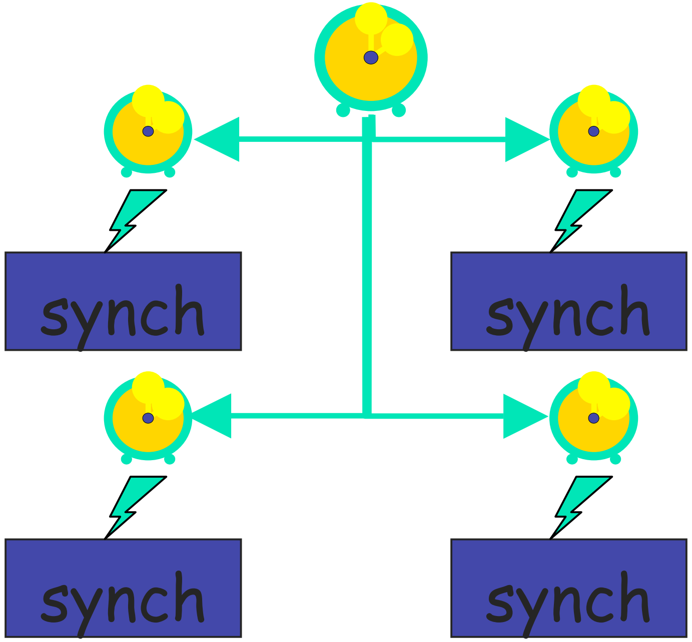
asynchronous: interleavings



1.2 Composition of time domains (cont'd)

- Endochronous time
  - $\mathcal{T}$ such that
  - $\mathcal{T}_1$ functional sampling of $\mathcal{T}$
  - $\mathcal{T}_2$ functional sampling of $\mathcal{T}$





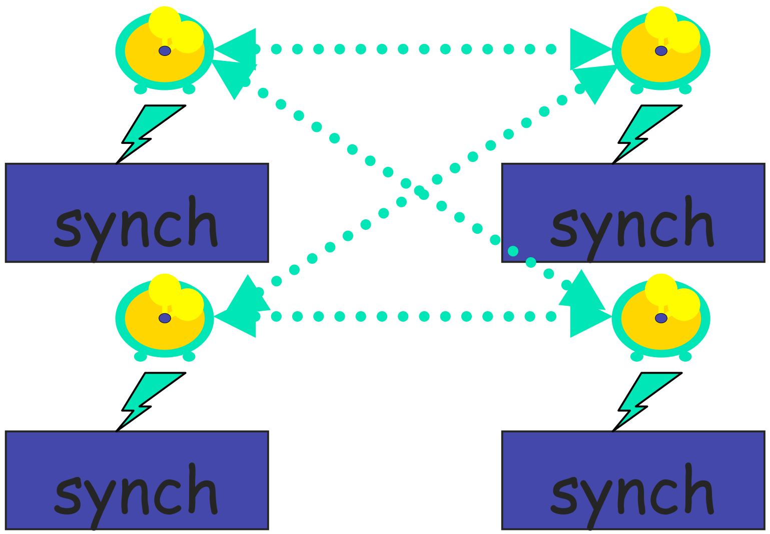
1.2 Composition of time domains (cont'd)

- Polychrony
  - $(\mathcal{T}_1 , \mathcal{T}_2)$ = set of properties that should be satisfied

    $\mathcal{R}(\mathcal{T}_1 , \mathcal{T}_2, \mathcal{T})$

    $\mathcal{T}_1 \subset \mathcal{T}$

    $\mathcal{T}_2 \subset \mathcal{T}$



process: $<\mathcal{R}(\mathcal{T}), \mathcal{T} \rightarrow (V \rightarrow D)>$

1.3 Computed time domain

- Esterel
  - Inputs: exochronous (every input may occur at any time)
  - Outputs: endochronous (determinism)

```
input A, B, R;
output O;
loop
  [await A || await B];
  emit O;
each R
```

1.3 Computed time domain (cont'd)

- Lustre
  - Inputs and outputs: endochronous

```
ck1 = true -> pre ck2;
ck2 = true -> not pre ck1;
c = a when ck1;
d = b when ck2;
e = current(c) + current(d);
```

1.3 Computed time domain (cont'd)

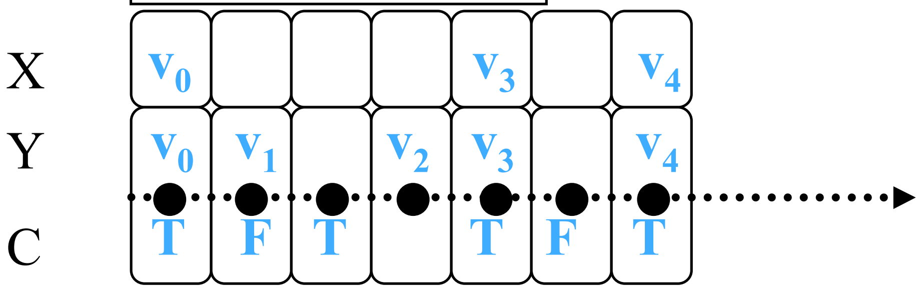
- Signal
  - Inputs and outputs: endochronous and/or exochronous
  - → mixed relation

x = a default ((x$ init 0) + 1)

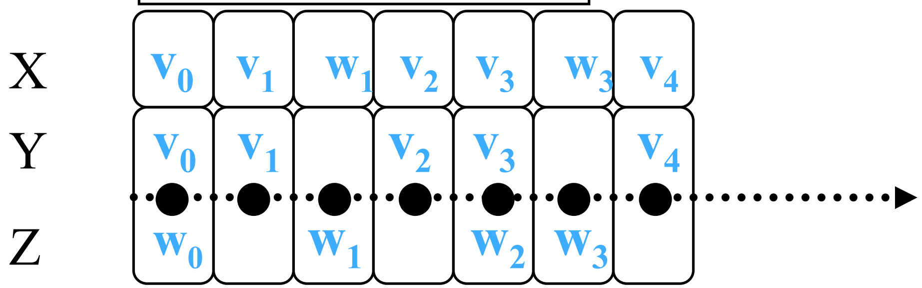
1.4 Signal: two categories of operators

- Combinational operators
  - $\mathcal{T}$ **discrete** (might be **continuous**)

$X = Y \text{ when } C$

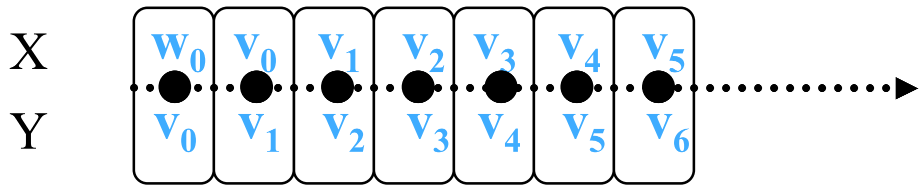


$X = Y \text{ default } Z$



- Discrete delay operator ($, pre)
  - $\mathcal{T}$ **discrete**

$X = Y\$ \text{ init } w_0$

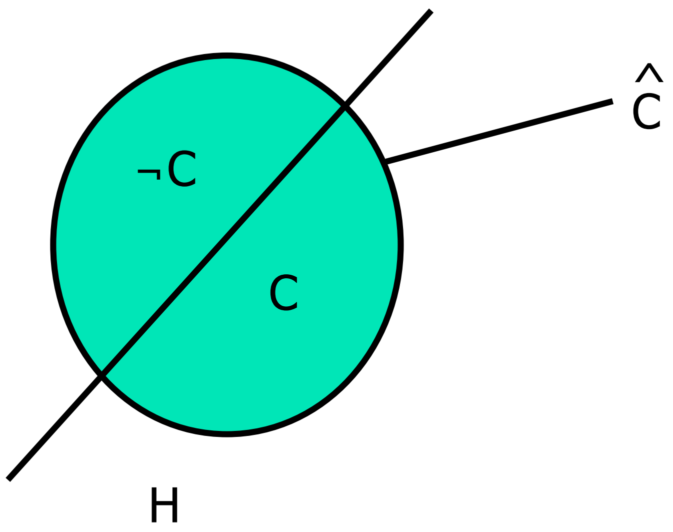


1.5 Clocks

- Basic relations

  C a boolean

  has denumerable occurrences

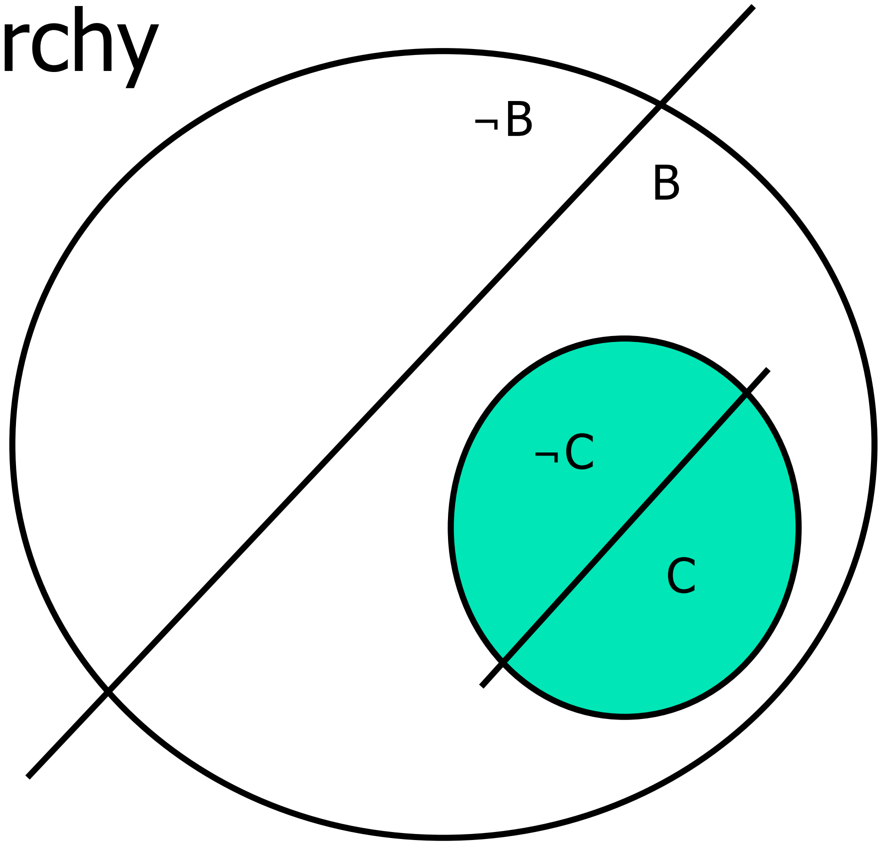


$H \setminus \hat{C}$: C is not present

$\overline{C}$ is $H \setminus \hat{C}$ default $\neg C$

1.5 Clocks (cont'd)

- Hierarchy



Outline

- Polychronous behaviors
- Assume/Guarantee specifications
- Using components

2.1 Definition

- Contract (A, G), interpreted as $A \Rightarrow G$
- How to go further than (true, G)?
- Which complement?

  - our answer: relations in including clocks

2.2 $A \Rightarrow G$ in polychronous model

- $P \vdash A \Rightarrow G$

  all traces in $P \cap A$ ($P \mid A$) are traces in $P \cap G$ ($P \mid G$)

Properties: $P \subset Q$ iff $P \mid Q = P$

$$P \mid P = P$$

$$\Rightarrow \; P \mid A \mid G = P \mid A$$

2.2 A $\Rightarrow$ G in polychronous model (cont'd)

- G, A descriptions may be:
  - Static relation

    x > 0
  - Dynamical relation

    x = not x $ init false
  - State relation (from automata translation)

    cs = s_i when R_1 when R_2(cs $)
  - Conjunction of such relations

2.2 A $\Rightarrow$ G in polychronous model (cont'd)

- Example

A: I always increasing

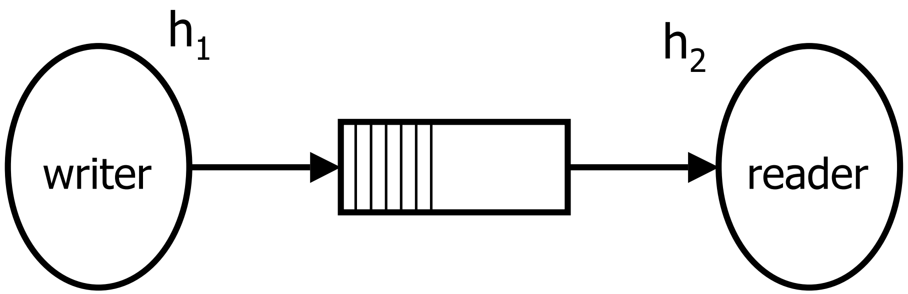
G: O never true at two consecutive instants

A: I >= I$

G: not (O and O$)

2.2 A $\Rightarrow$ G in polychronous model (cont'd)

- Example



A	G
true	$i \geq 0 \wedge i < n$
$i = 0 \wedge$ read	alarm
$i = n\text{-}1 \wedge$ write	alarm

2.2 $A \Rightarrow G$ in polychronous model (cont'd)

G, A are sets of relations $R_1, \ldots, R_n$

For each relation R_i in A (G), we define A_i (G_i), the boolean that is true iff R_i is satisfied

Let

$A = A_1$ when … when A_n default false

$G = G_1$ when … when G_m default false

(default false → "adjustable" clock)

2.2 $A \Rightarrow G$ in polychronous model (cont'd)

- Process $(A \Rightarrow G)$ as a H-observer:

  defined as:

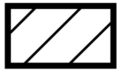
  $FAG = true \text{ when } \overline{G}^{H} \text{ when } A$

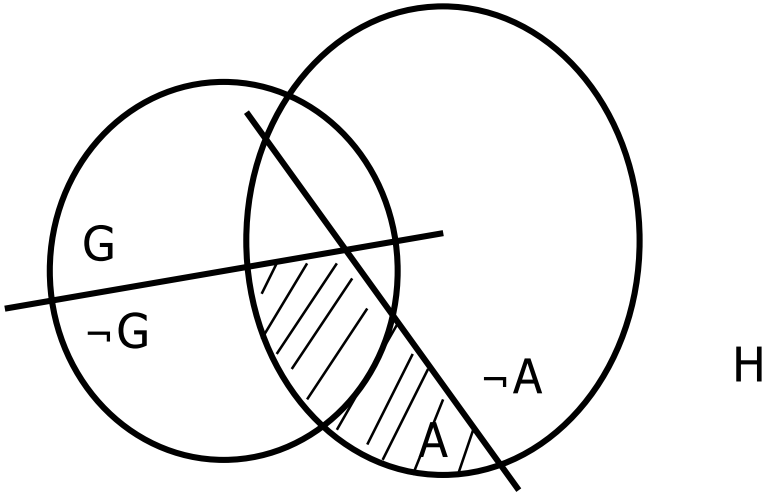
  $| \; (A = \ldots \; | \; (A_1 = \ldots \; | \; \ldots))$

  $| \; (G = \ldots \; | \; (G_1 = \ldots \; | \; \ldots))$

$$\text{when } \overline{G}^{H} : \; \text{when (not } G \text{ default } H)$$

2.2 A $\Rightarrow$ G in polychronous model (cont'd)

 true when (not G default H) when A



2.2 $A \Rightarrow G$ in polychronous model (cont'd)

- $P \models A \Rightarrow G$

  iff

$$\boxed{(P \mid (A \Rightarrow G) \mid \text{true when } FAG \mathbin{\hat{}=} 0) = P}$$

$\hat{}= 0$: null clock

2.3 Composition

- $P_1 \vdash A_1 \Rightarrow G_1$

  $\wedge$

  $P_2 \vdash A_2 \Rightarrow G_2$

$\Rightarrow$

$$P_1 \mid P_2 \vdash (A_1 \vee A_2) \Rightarrow (A_1 \Rightarrow G_1 \wedge A_2 \Rightarrow G_2)$$

2.3 Composition (cont'd)

- Example

$$x > -1 \;\Rightarrow\; \exists k,\ y = -3\,k\,x$$
$$x < 1 \;\Rightarrow\; \exists k,\ y = 2\,k\,x$$

$$\begin{aligned} true \Rightarrow\ & (x > -1 \;\Rightarrow\; \exists k,\ y = -3\,k\,x) \land \\ & (x < 1 \;\Rightarrow\; \exists k,\ y = 2\,k\,x) \end{aligned}$$

Outline

- Polychronous behaviors
- Assume/Guarantee specifications
- Using components

3.1 Basic schedulings

- x, y two signals, causality relation $x \rightarrow y$ if y_t uses x_t for its computation
- more exactly, $x \xrightarrow{h} y$ means that at h, x does not depend on y
- Example
  - if b then z = u else z = v

$$
\begin{tikzcd}
\mathbf{h_b} \arrow[r] & \mathbf{b} \arrow[d, "h_b \wedge (h_u \vee h_v)"] & \\
\mathbf{h_u} \arrow[r, "h_u \wedge b"] \arrow[d] & \mathbf{h_z} \arrow[d] & \mathbf{h_u} \arrow[l, "h_v \backslash (h_u \wedge b)"'] \arrow[d] \\
\mathbf{u} \arrow[r, "h_u \wedge b"] & \mathbf{z} & \mathbf{v} \arrow[l, "h_v \backslash (h_u \wedge b)"']
\end{tikzcd}
$$

3.2 Using the graph

- Path algebra
  - Sequence

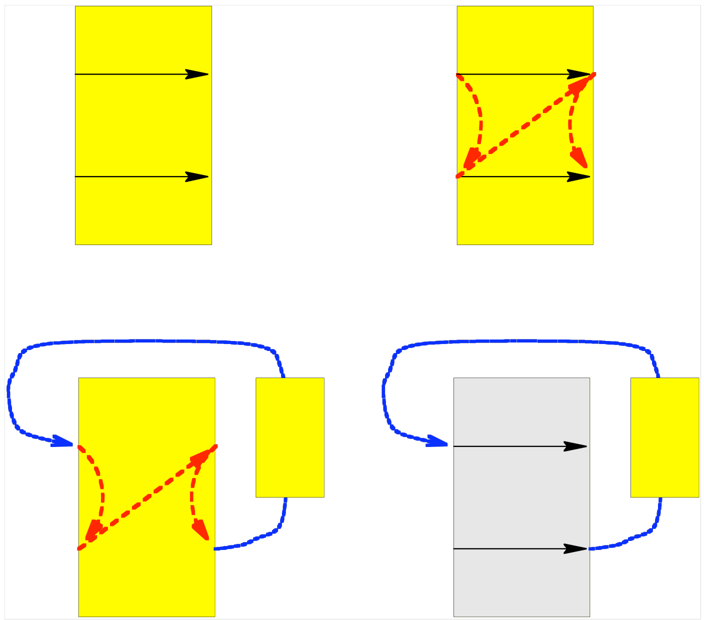
$$X_1 \xrightarrow[h_1]{} X_2 \xrightarrow[h_2]{} X_3 \qquad\qquad X_1 \xrightarrow[h_1 \text{ when } h_2]{} X_3$$

  - Parallel

$$X_1 \begin{array}{c} \xrightarrow{h_1} \\ \xrightarrow[h_2]{} \end{array} X_2 \qquad\qquad X_1 \xrightarrow[h_1 \text{ default } h_2]{} X_2$$

3.2 Using the graph (cont'd)

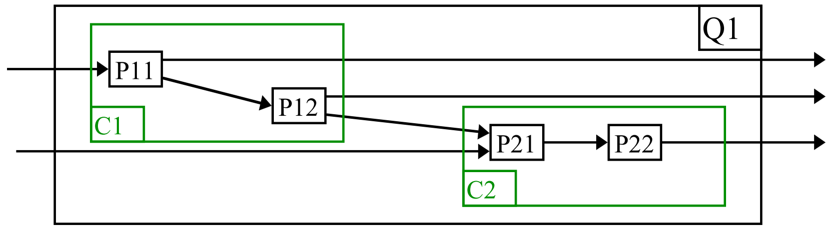
- Transitive closure
- Projection on inputs/outputs
- Compositional deadlock consistency



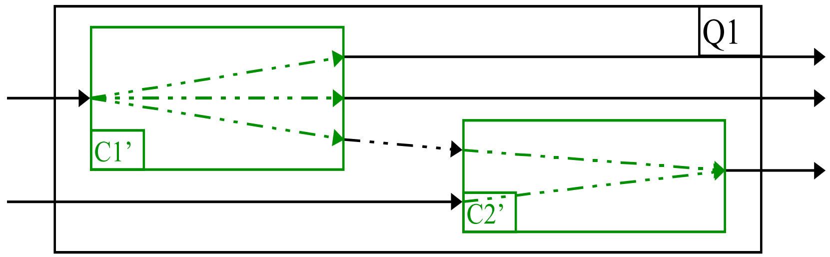
3.2 Using the graph (cont'd)

- Black box abstraction

  Each node of the same group depends on the same subset of inputs

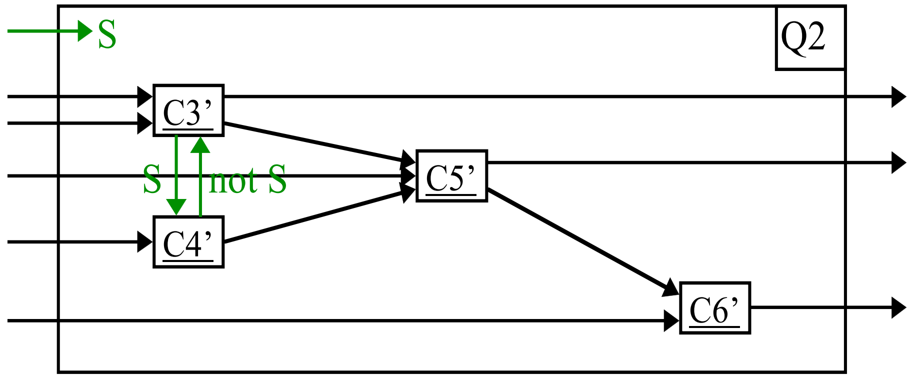


- Grey box abstraction



3.2 Using the graph (cont'd)

- Parameterized scheduling

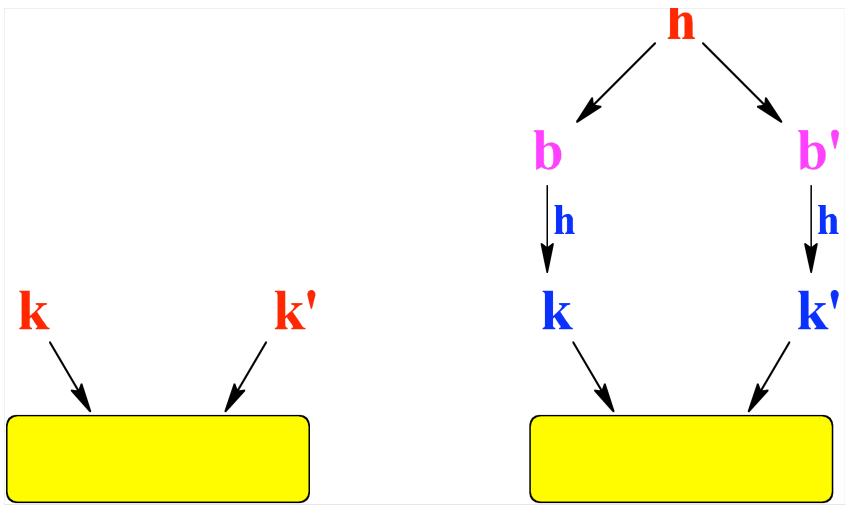


3.3 Taxonomy

- Reactive system
  - ability to react to each configuration in all states
- Endochronous system (function of flows)
  - any flow is entirely determined by the sequences of values of the “inputs”

3.4 Transformations

- Reactive → endochronous transformation



- Desynchronization
  - bufferized

Summary

- Time domain for polychrony
- Contracts in a polychronous context
- Polychronous components

- Extension to continuous time?