Wrappers for Mapping Synchronous Systems to GALS Architectures

Benoît Caillaud

IRISA / INRIA, Rennes, France

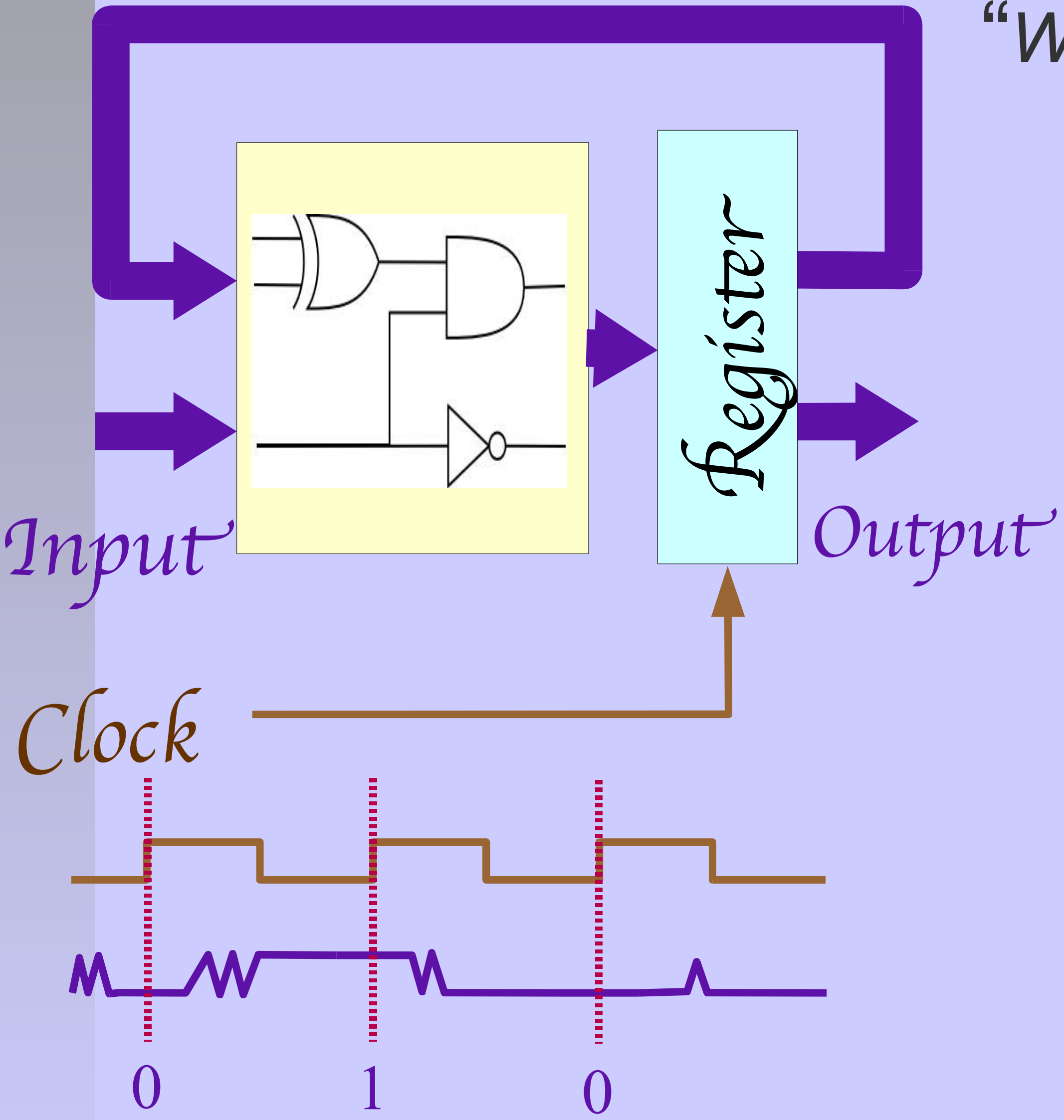
Dumitru Potop-Butucaru

VERIMAG, Grenoble, France

Benoit.Caillaud@irisa.fr

http://www.irisa.fr/prive/Benoit.Caillaud/date2005.pdf

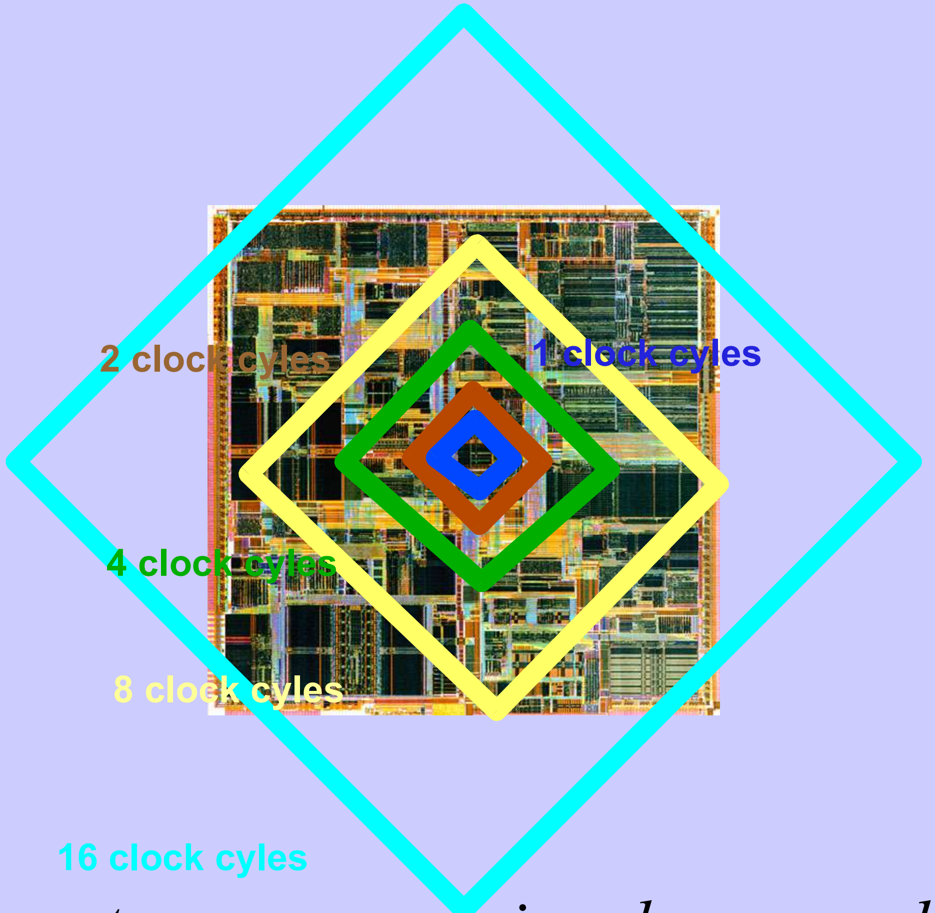
Synchrony in hardware



"*We might say that the clock enables us to introduce a discreteness into time, so that time for some purposes can be regarded as a succession of instants instead of a continuous flow. A digital machine must essentially deal with discrete objects, ... All other computing machines except for human and other brains that I know of do the same. One can think up ways of avoiding it, but they are very awkward.*" A. Turing

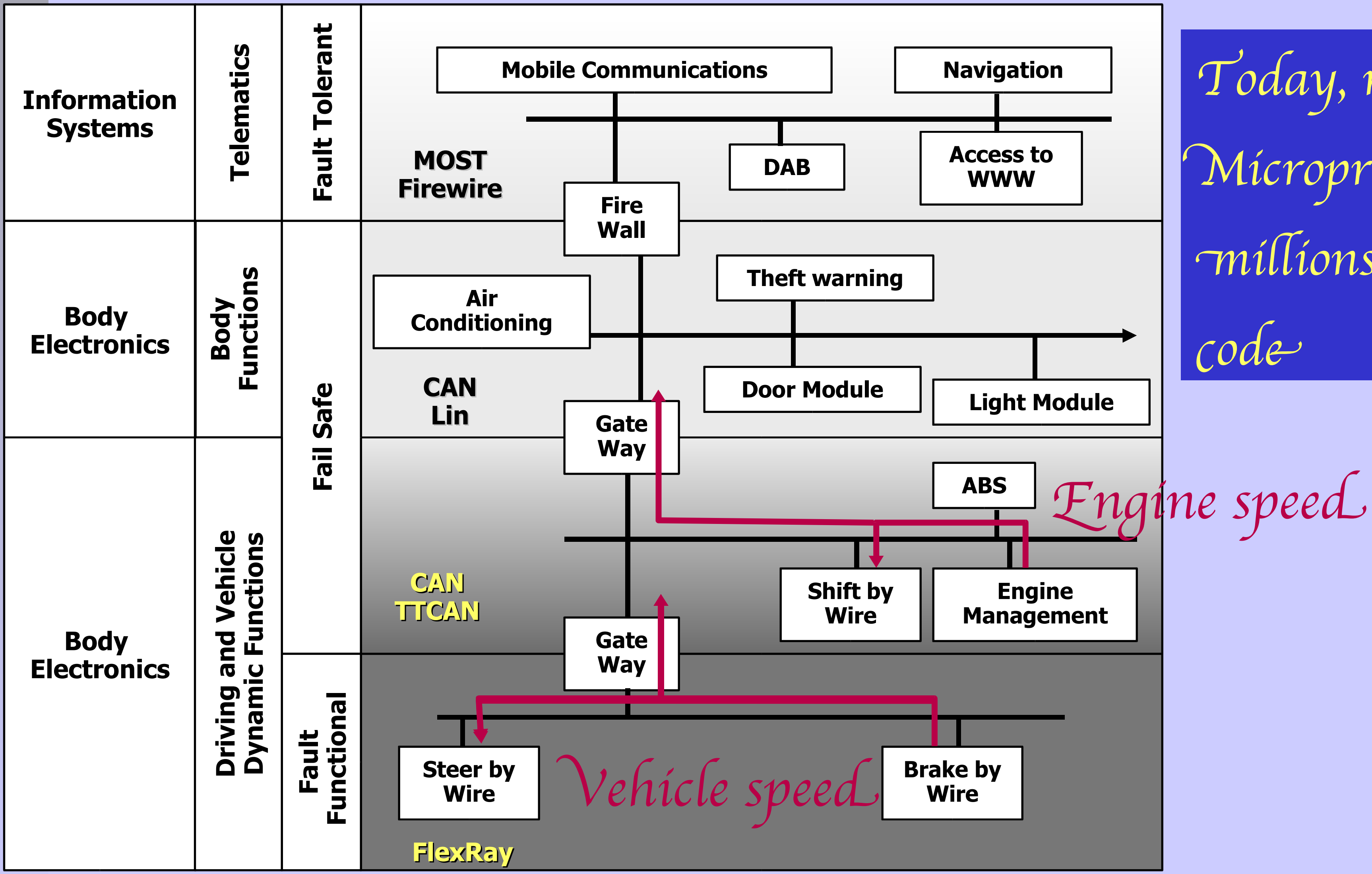
State of the art hardware circuit can not be synchronous

- Asynchronous hardware circuits: *Cortadella et al. Logic Synthesis of Asynchronous Controllers and Interfaces. Springer, 2001.*
- GALS : *Globally asynchronous, Locally Synchronous circuits.*
- Latency insensitive circuits: *Carloni et al. A Methodology for Correct-by-Construction Latency-Insensitive Design. ICCAD'99.*



"For a 60 nanometer process a signal can reach only 5% of the die's length in a clock cycle" [D. Matzke,1997]

What about embedded software ?



Information Systems

Telematics

Fault Tolerant

MOST Firewire

Mobile Communications

Navigation

DAB

Access to WWW

Fire Wall

Body Electronics

Body Functions

Fail Safe

Air Conditioning

Theft warning

CAN Lin

Door Module

Light Module

Gate Way

Body Electronics

Driving and Vehicle Dynamic Functions

ABS

Engine speed

CAN TTCAN

Shift by Wire

Engine Management

Gate Way

Fault Functional

Steer by Wire

Vehicle speed

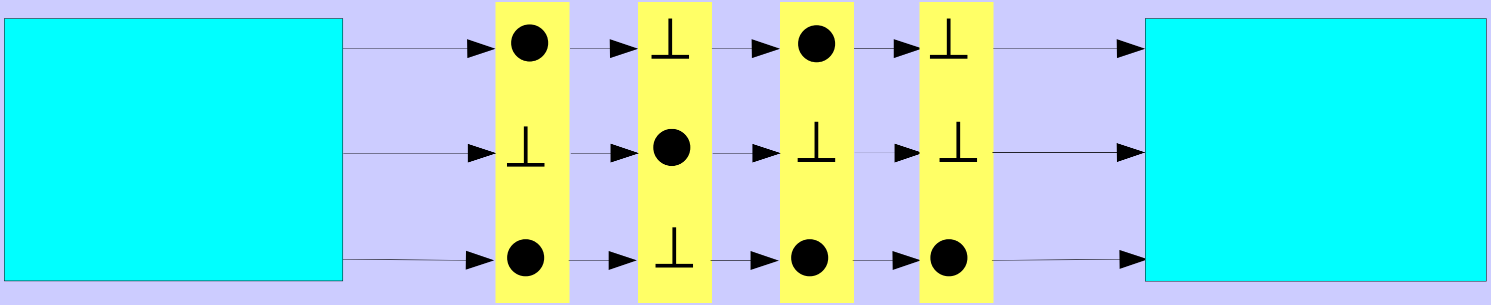
Brake by Wire

FlexRay

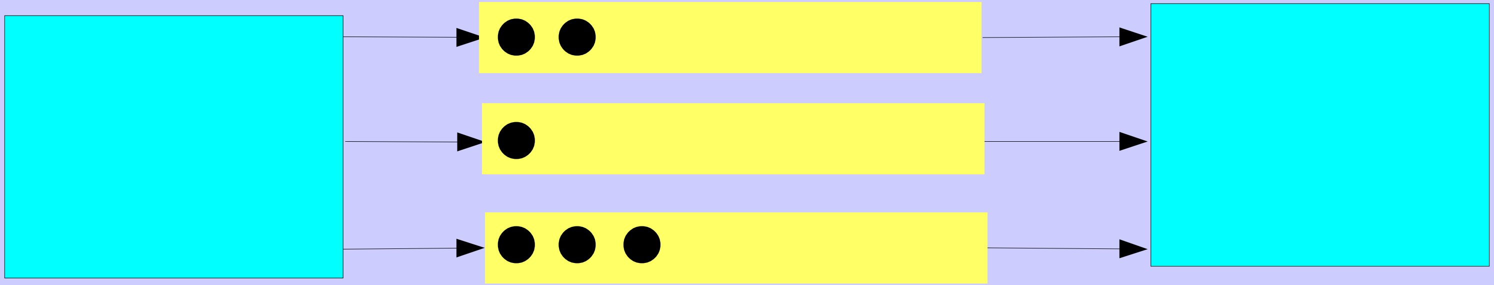
Today, more than 80 Microprocessors and millions of lines of code

Synchrony vs. Asynchrony

- Synchrony = ease of modeling, reasoning and verification.

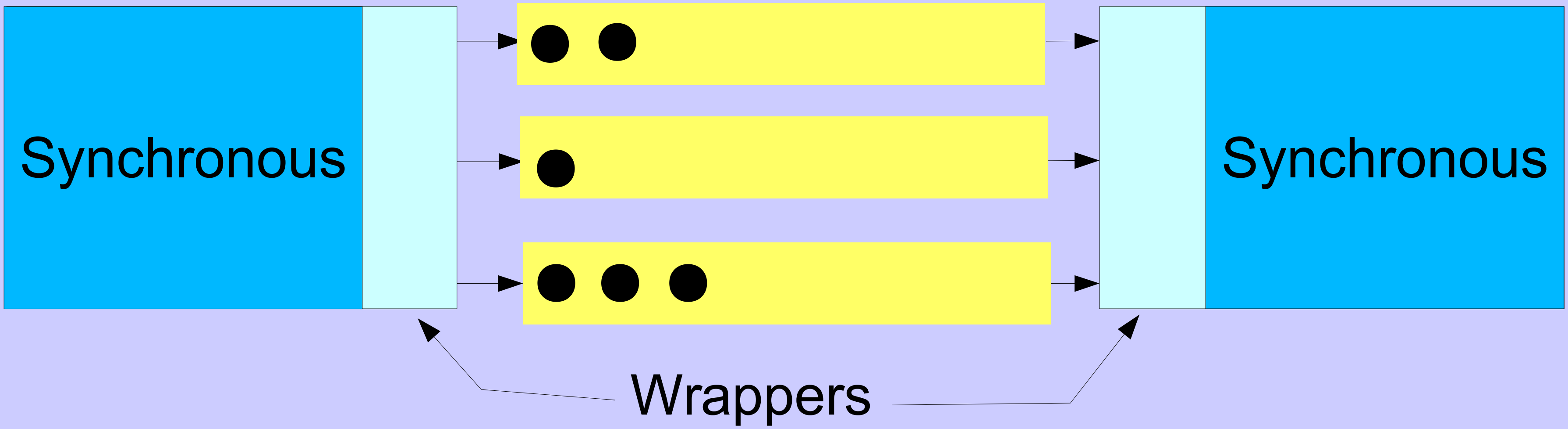


- Asynchrony = efficiency, scalability.
  Verification is difficult

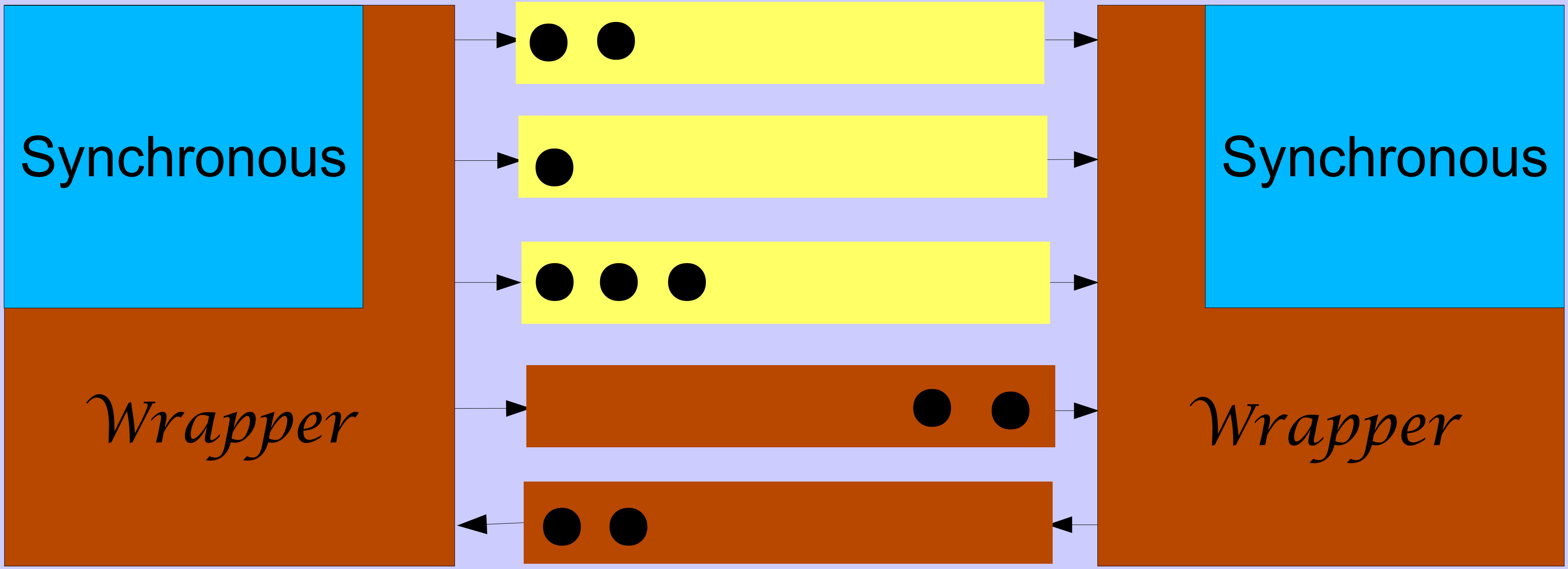


Globaly Asynchronous, Localy Synchronous

GALS Architectures = trade-off btw Synchrony / Asynchrony



Wrappers in GALS Architectures



Wrapper = *Scheduler* + *Handshake protocol*

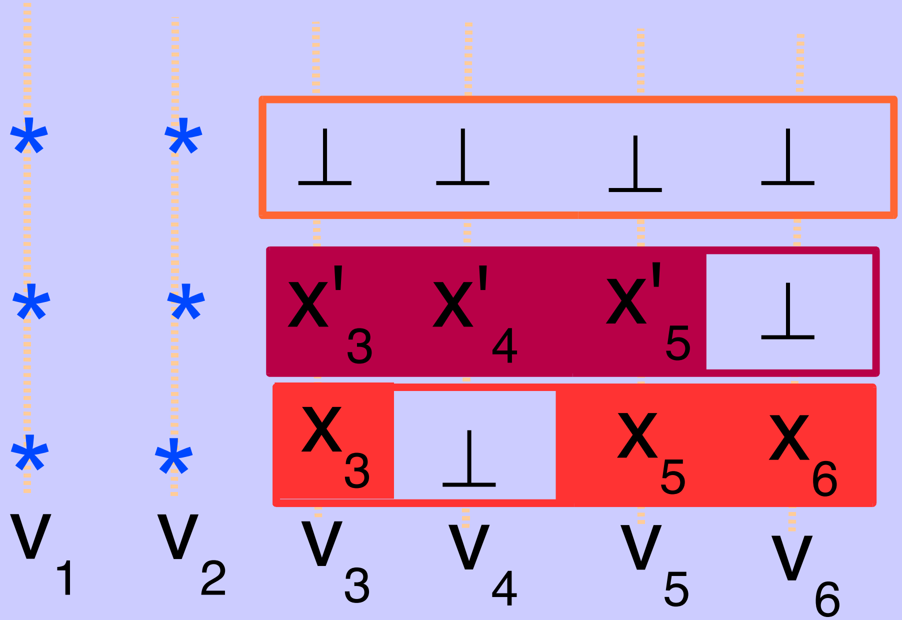
Modeling Synchrony

- $\mathcal{V}$: variables, $\mathcal{D}$: values
- Reaction $r : \mathcal{V} \rightarrow \mathcal{D} \cup \{\perp, *\}$,
- Behaviour

Absence

Undefined

$$\text{Traces} = \mathbb{N} \rightarrow \mathcal{V} \rightarrow \mathcal{D} \cup \{\perp, *\}$$

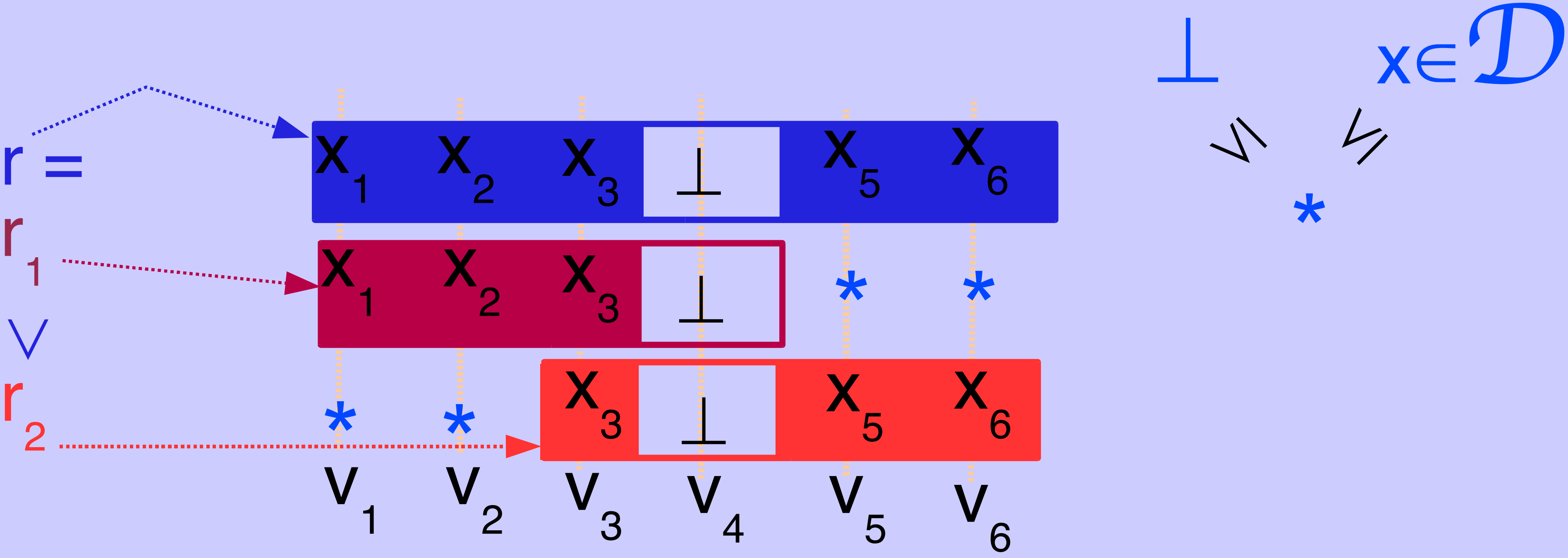


$*$ $*$ $\perp$ $\perp$ $\perp$ $\perp$
$*$ $*$ x'_3 x'_4 x'_5 $\perp$
$*$ $*$ x_3 $\perp$ x_5 x_6
v_1 v_2 v_3 v_4 v_5 v_6

Silent reaction

Strong Synchronization

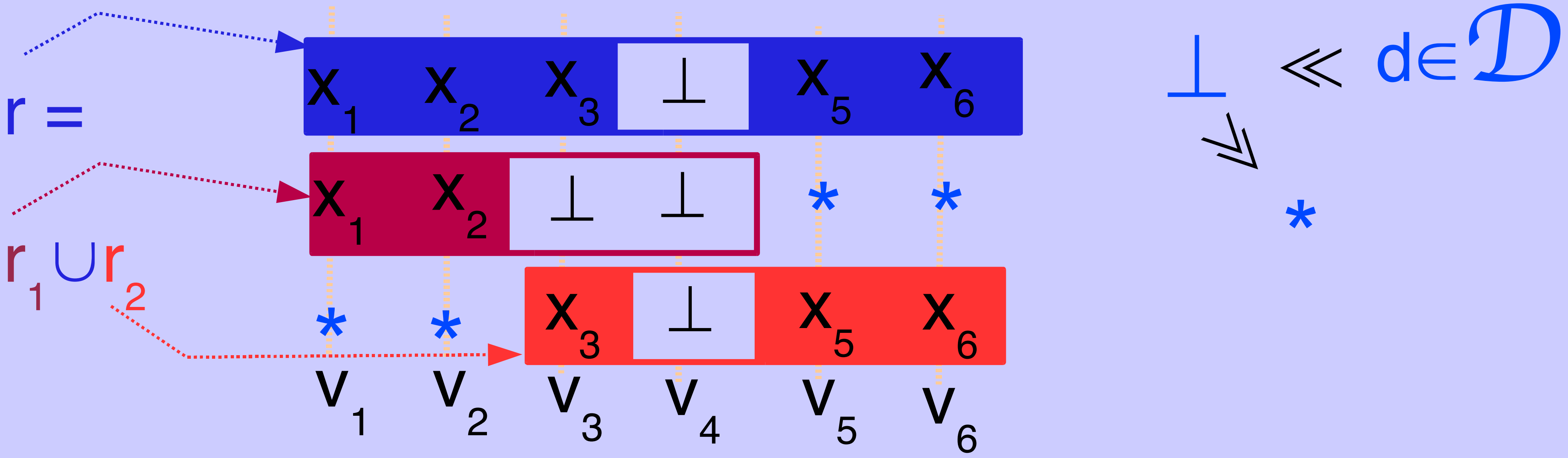
Synchronization = *partial lub*



	v_1	v_2	v_3	v_4	v_5	v_6
$r =$	x_1	x_2	x_3	$\bot$	x_5	x_6
r_1	x_1	x_2	x_3	$\bot$	*	*
$\vee$						
r_2	*	*	x_3	$\bot$	x_5	x_6

r_1 and r_2 are *synchronizable*

Weak Synchronization


r =
$r_1 \cup r_2$
X_1 X_2 X_3 $\perp$ X_5 X_6
X_1 X_2 $\perp$ $\perp$ * *
* * X_3 $\perp$ X_5 X_6
V_1 V_2 V_3 V_4 V_5 V_6
$\perp \ll d \in \mathcal{D}$
$\gg$
*

r_1 and r_2 are *non-contradictory*

Modeling Asynchrony

- Asynchronous behaviour

Histories = $\mathcal{V} \rightarrow \mathcal{D}^{\infty} \cup \{*\}$

- Desynchronization

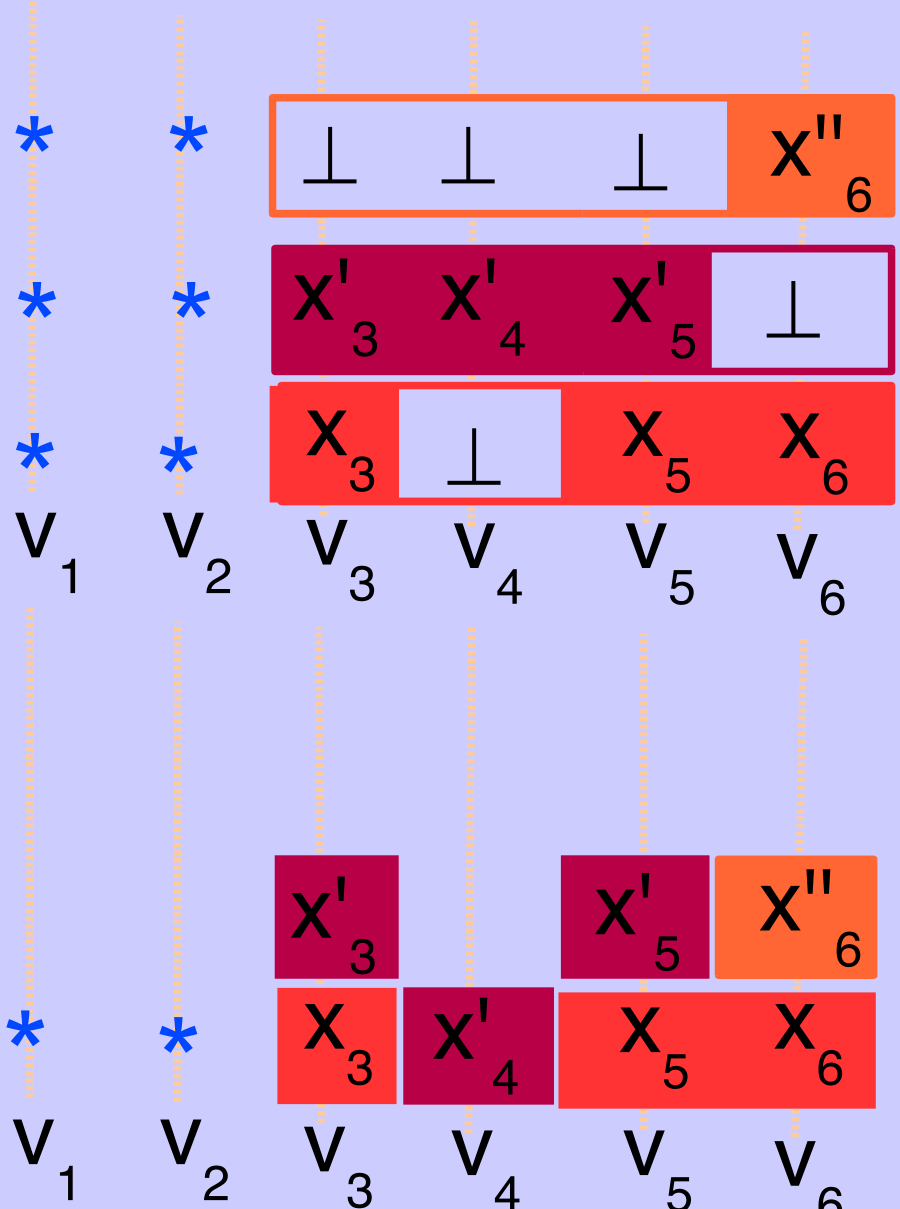
δ : Traces(U) $\longrightarrow$ Histories(U)

$\forall v \in U$, $\delta(r.t)(v) =$

$\delta(r(v)).\delta(t)(v)$

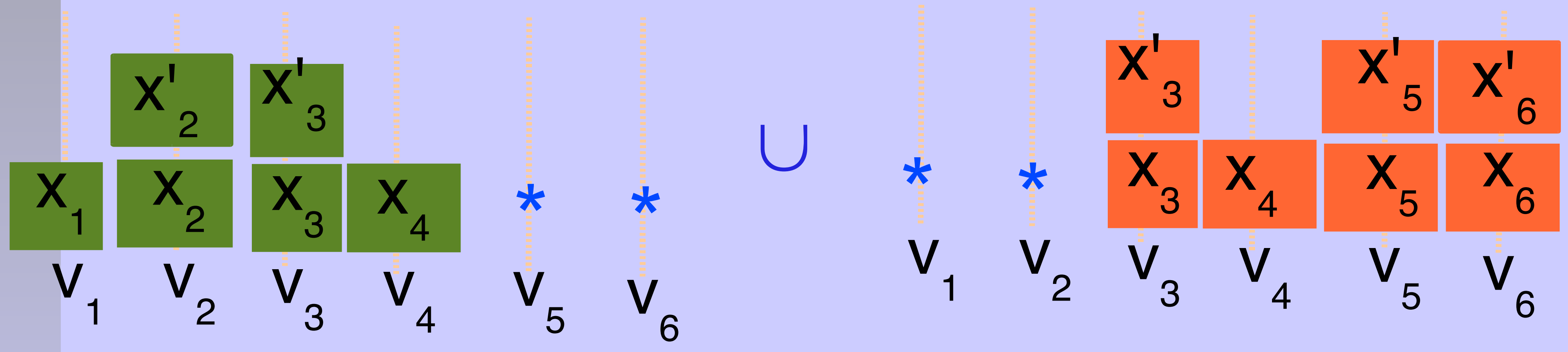
$\forall v \notin U$, $\delta(t)(v) = *$

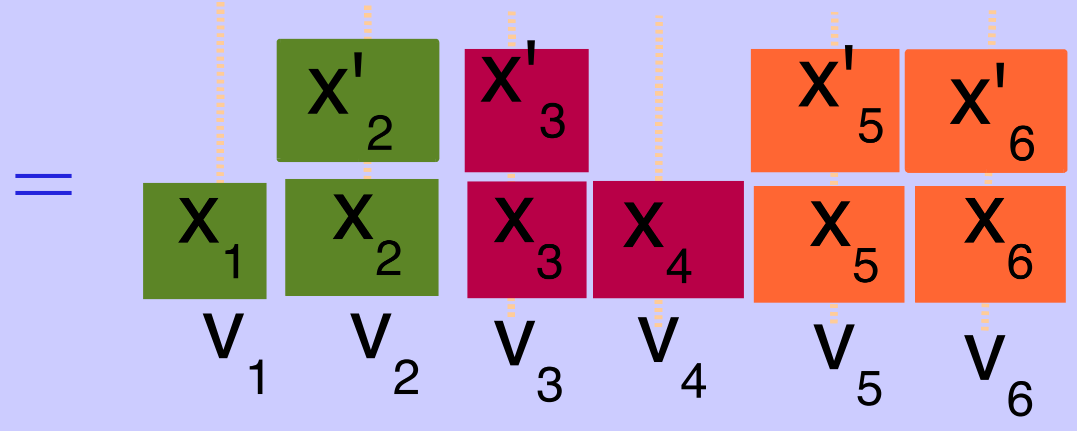
$\delta(\perp) = \epsilon$, $\delta(x) = x$



Asynchronous composition

$h_1,h_2 \in$ Histories, $h_1 \subseteq h_2$ iff $\forall u \in \mathcal{V}$, $h_1(u) \neq * \Rightarrow h_1(u) = h_2(u)$





h_1,h_2 non-contradictory

iff $h_1 \cup h_2$ defined

Synchronous Transition Systems

$U \subseteq \mathcal{V}$ *finite set of variables*

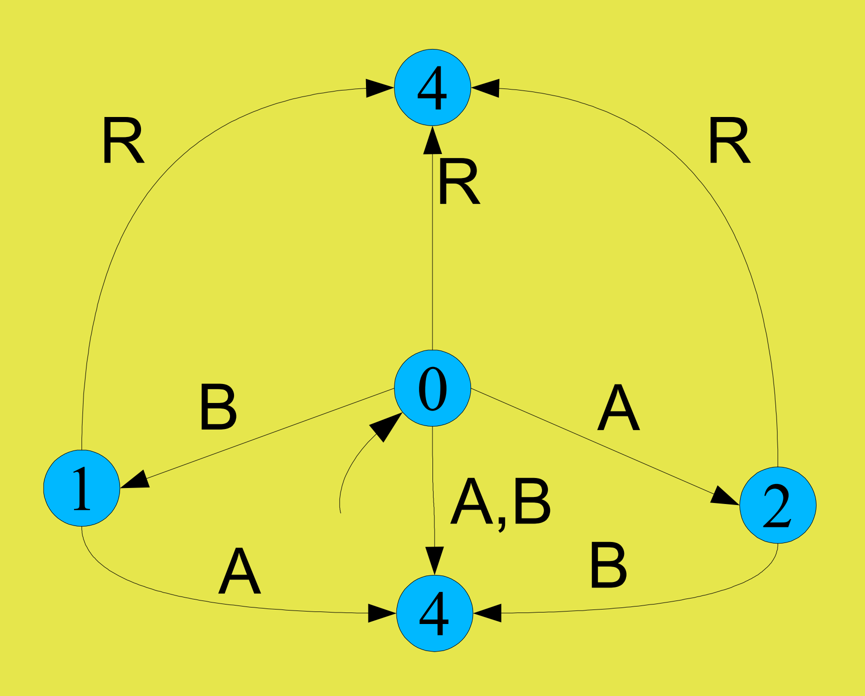
S set of states

$$\Sigma = (U , S , \rightarrow , \hat{s})$$

$\rightarrow \; \subseteq \; S \times \mathrm{Reactions}(U) \times S$

stuttering invariant transition relation

$\hat{s} \in S$ *Initial state*

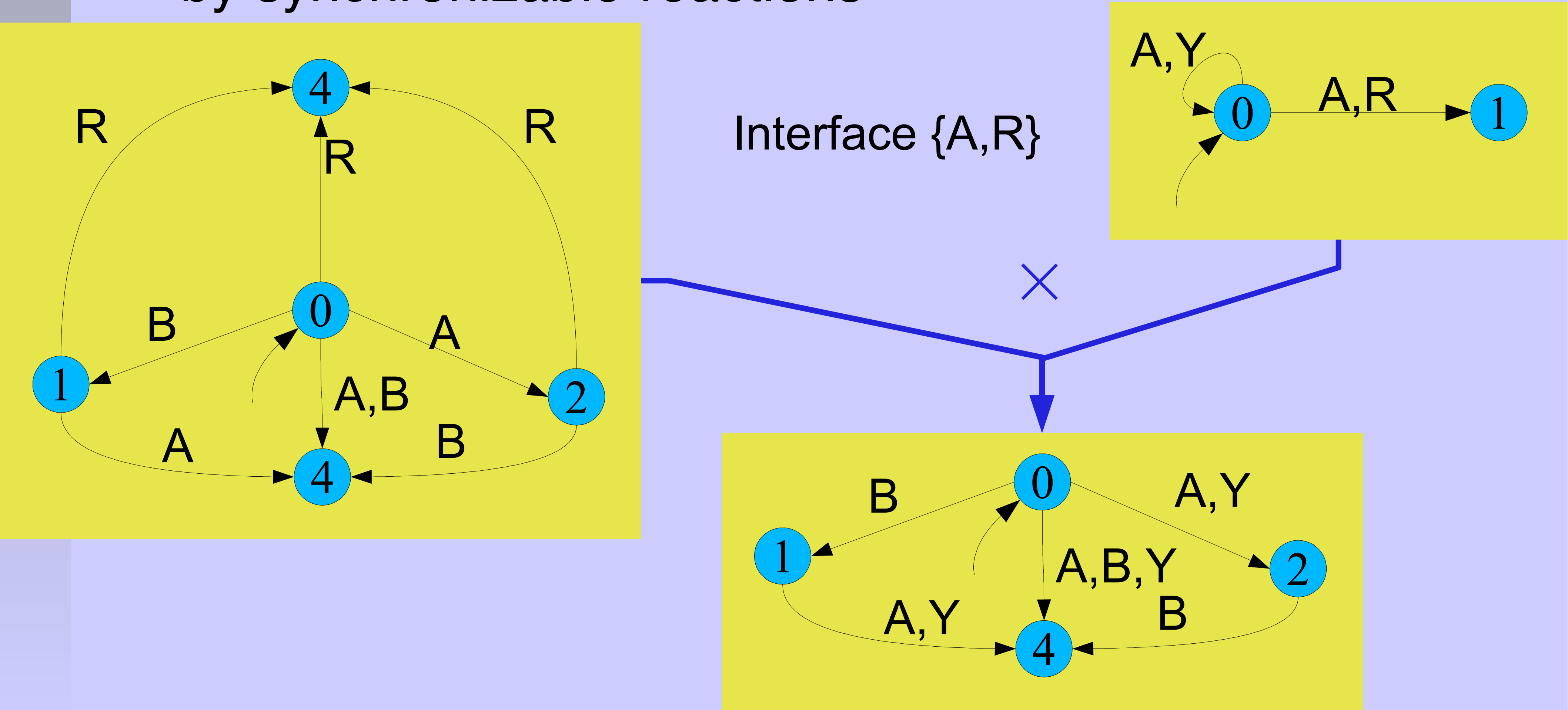


Product of STS

$\Sigma_i = (U_i , S_i , \rightarrow_i , \hat{s}_i)$

$\Sigma_1 \times \Sigma_2 = (U_1 \cup U_2 , S_1 \times S_2 , \rightarrow , (\hat{s}_1,\hat{s}_2))$

$\rightarrow \approx$ restriction of $\rightarrow_1 \times \rightarrow_2$ to pairs of transitions labeled by synchronizable reactions


Interface {A,R}

Two desynchronization problems

1. Asynchronous determinacy of a finite STS

Given a finite STS Σ, decide whether:

$\forall t,t' \in \mathrm{Traces}(\Sigma)$,
$\delta(t) = \delta(t') \Rightarrow t = t'$, up to stuttering (1)

2. Synchronous/asynchronous equivalence of two finite STS

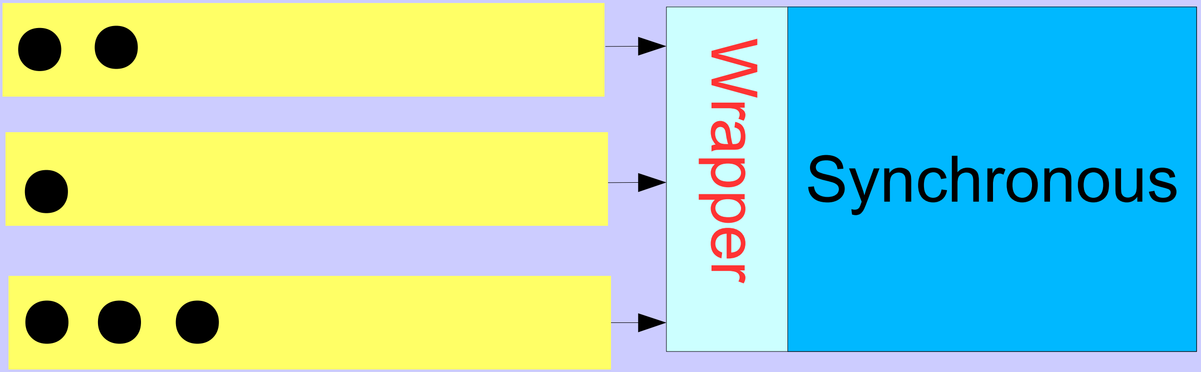
Given two finite STS Σ_1, Σ_2, decide whether:

$\mathrm{Histories}(\Sigma_1 \times \Sigma_2) = \mathrm{Histories}(\Sigma_1) \| \mathrm{Histories}(\Sigma_2)$ (2)

What do these problems mean in practice ?

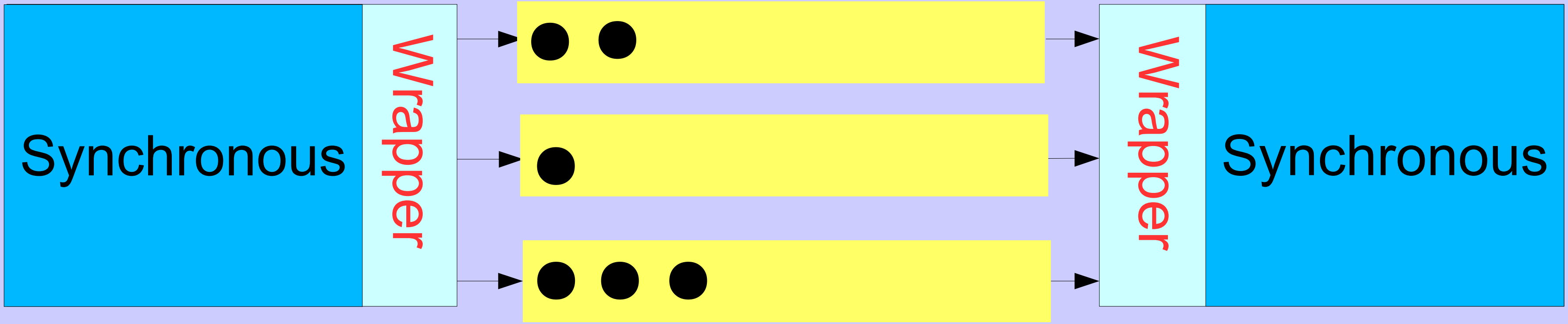
1. Asynchronous determinacy

Is there a unique way to construct reactions ?



2. Synchronous/Asynchronous equivalence

Correctness of the mapping synchronous to GALS



Both problems are undecidable

✔ Histories(Σ) = rational relation.

☹ Emptiness of the intersection of two rational relations is undecidable.

Look for decidable sufficient conditions of properties (1) and (2)

The clock inference game

✍ Benveniste et al. Compositionality in dataflow synchronous languages: specification & distributed code generation. Information and Computation, 163:125-171, 2000.

Two players: environment + system

Given a STS Σ and a state s. Let R_s be the set of non-silent reactions enabled in s.

Position: reaction r, initially r(v) = * for all v

Moves:

System: select a variable v s.t. r(v) = * and a guess $g \in \{\top, \bot\}$

Environment: assign a value $d \in \mathcal{D} \cup \{\bot\}$ to v in r: r := r[d/v], such that $r \leq r'$ for some $r' \in R_s$

Winning conditions:

Environment: The environment contradicts the system:

$g = \top$ and $d = \bot$, or $g = \bot$ and $d \in \mathcal{D}$

System: $r \in R_s$ and the system has not been contradicted

Endochrony

Def 1: STS Σ is endochronous iff for each reachable state s, the system has a winning strategy to the clock inference game.

Lemma 1: Endochrony is decidable in polynomial time on finite STS.

Theorem 1: Endochrony and determinism ⇒ property (1) holds.

Application: SIGNAL compiler

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

Problem: Endochrony is not compositional

Isochrony

Def 2: Σ_1 and Σ_2 are isochronous iff for all (s_1,s_2) reachable state of $\Sigma_1 \times \Sigma_2$, for all r_1 enabled in s_1 and r_2 enabled in s_2,

r_1 and r_2 non-contradictory and non-silent ⇒

r_1 and r_2 synchronizable

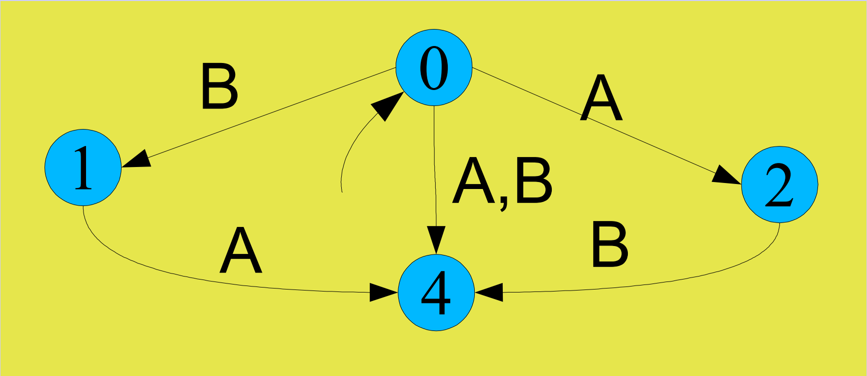
Lemma 2: Isochrony can be checked in polynomial time on pairs of finite STS

Theorem 2: Σ_1 and Σ_2 are isochronous ⇒ property (2) holds

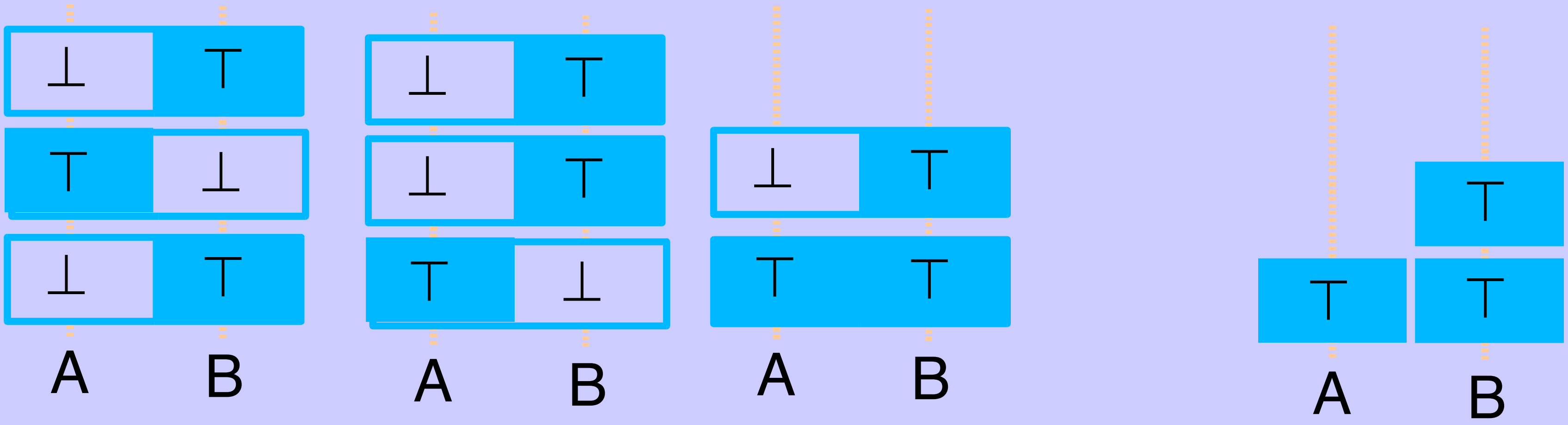
Problem: Isochrony is not compositional

Introducing concurrency…

✍ *Potop et al. Concurrency in Synchronous Systems, ACSD 2004.*



Non-endochronous STS, however...



Three traces with equal histories, reaching the same state in the STS

Weak-endochrony

Def: STS Σ is weakly endochronous iff:

W1. [Deterministic]

$$\begin{tikzcd} & s_1 \\ s \arrow[ur, "r"] \arrow[dr, "r"'] & \\ & s_2 \end{tikzcd} \Rightarrow \; s_1 = s_2$$

W2. [Step]

$$\begin{tikzcd} & s_1 \\ s \arrow[ur, "r_1"] \arrow[dr, "r_2"'] & \\ & s_2 \end{tikzcd} \quad r_1, r_2 \text{ disjoint} \Rightarrow \exists \; s', \quad s \xrightarrow{r_1 \cup r_2} s'$$

W3. [Decomposition]

r_1, r_2 non-contradictory,

$r' = r_1 \cap r_2$

$$\begin{tikzcd} & & s_1 \\ s \arrow[urr, "r_1"] \arrow[r, "r'"] \arrow[drr, "r_2"'] & s' \arrow[ur, "r_1 \setminus r_2"'] \arrow[dr, "r_2 \setminus r_1"] & \\ & & s_2 \end{tikzcd}$$

W4. [Commutation]

r_1, r_2 disjoint

$$s \xrightarrow{r_1} s_1 \xrightarrow{r_2} s_2 \;\Rightarrow\; \exists \; s', \quad s \xrightarrow{r_2} s'$$

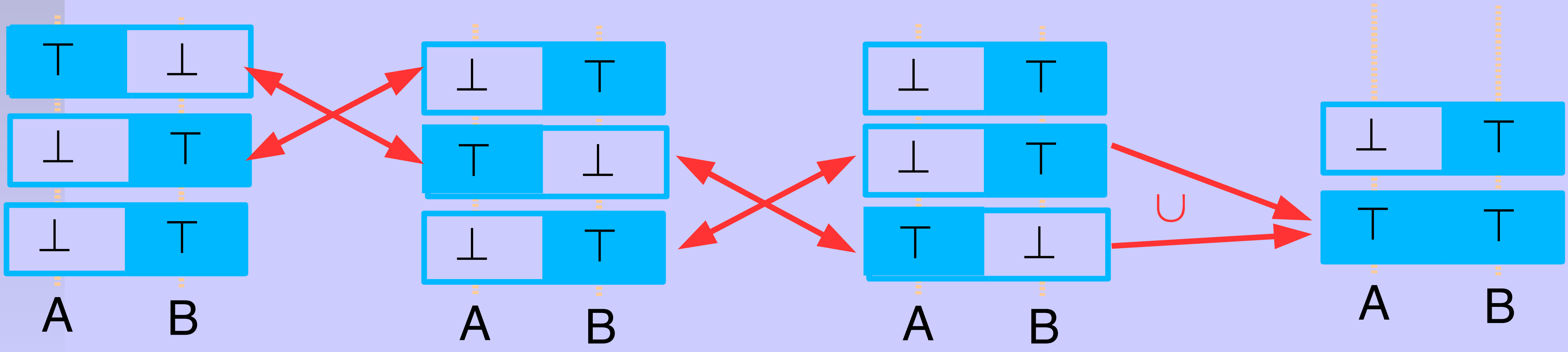
Properties of weak-endochronous STS

Lemma 3: weak-endochrony is compositional and can be checked in polynomial time on finite STS.

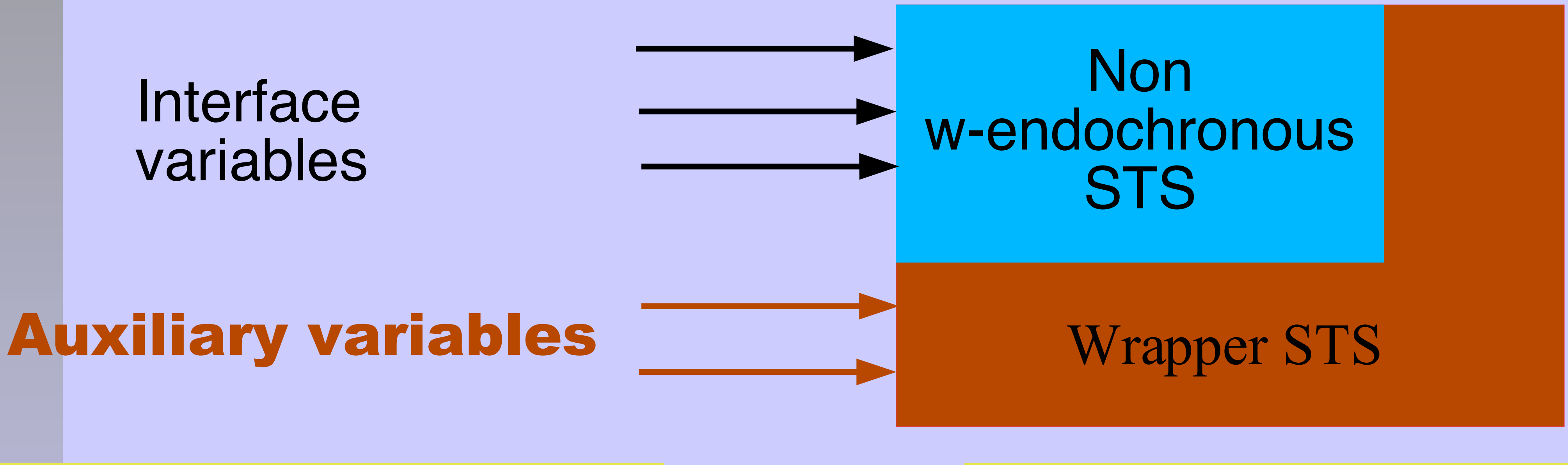
Lemma 4: endochrony ⇒ weak-endochrony, provided states are encoded in interface variables.

Def: Atoms(s) = set of minimal non-silent reactions enabled in s.

Lemma 5: Σ a weak-endochronous STS, Traces(Σ) is closed under commutation and union of disjoint adjacent reactions:



Synthesis of weak-endochrony, while preserving concurrent steps

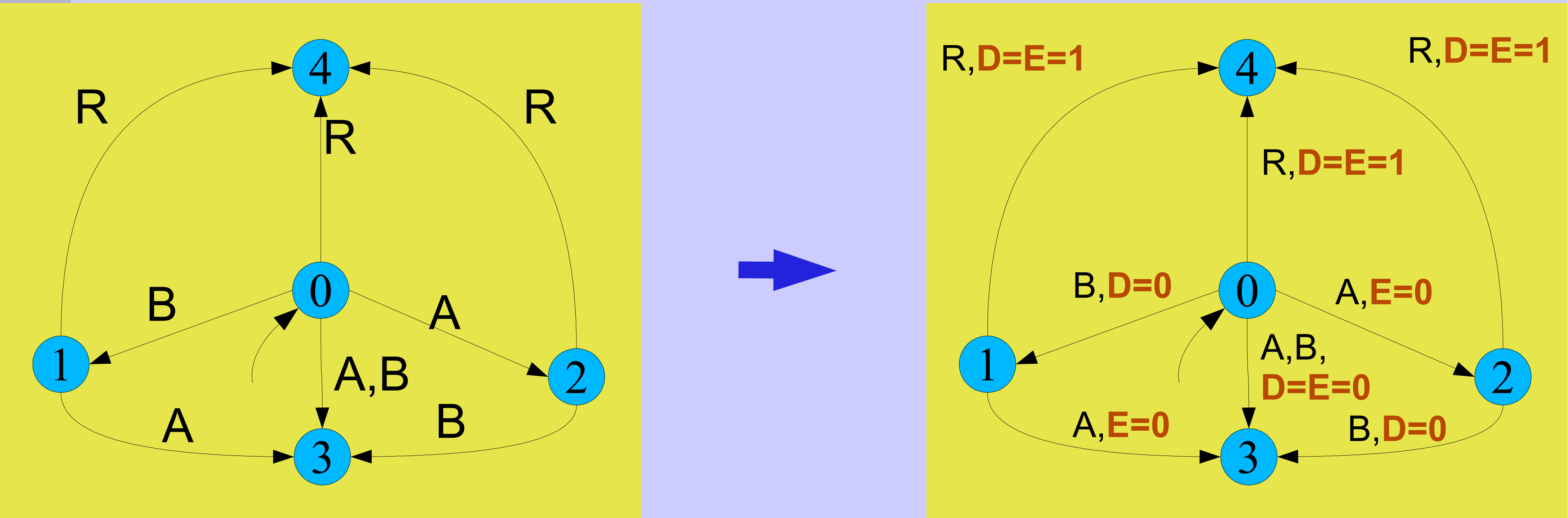


Interface variables

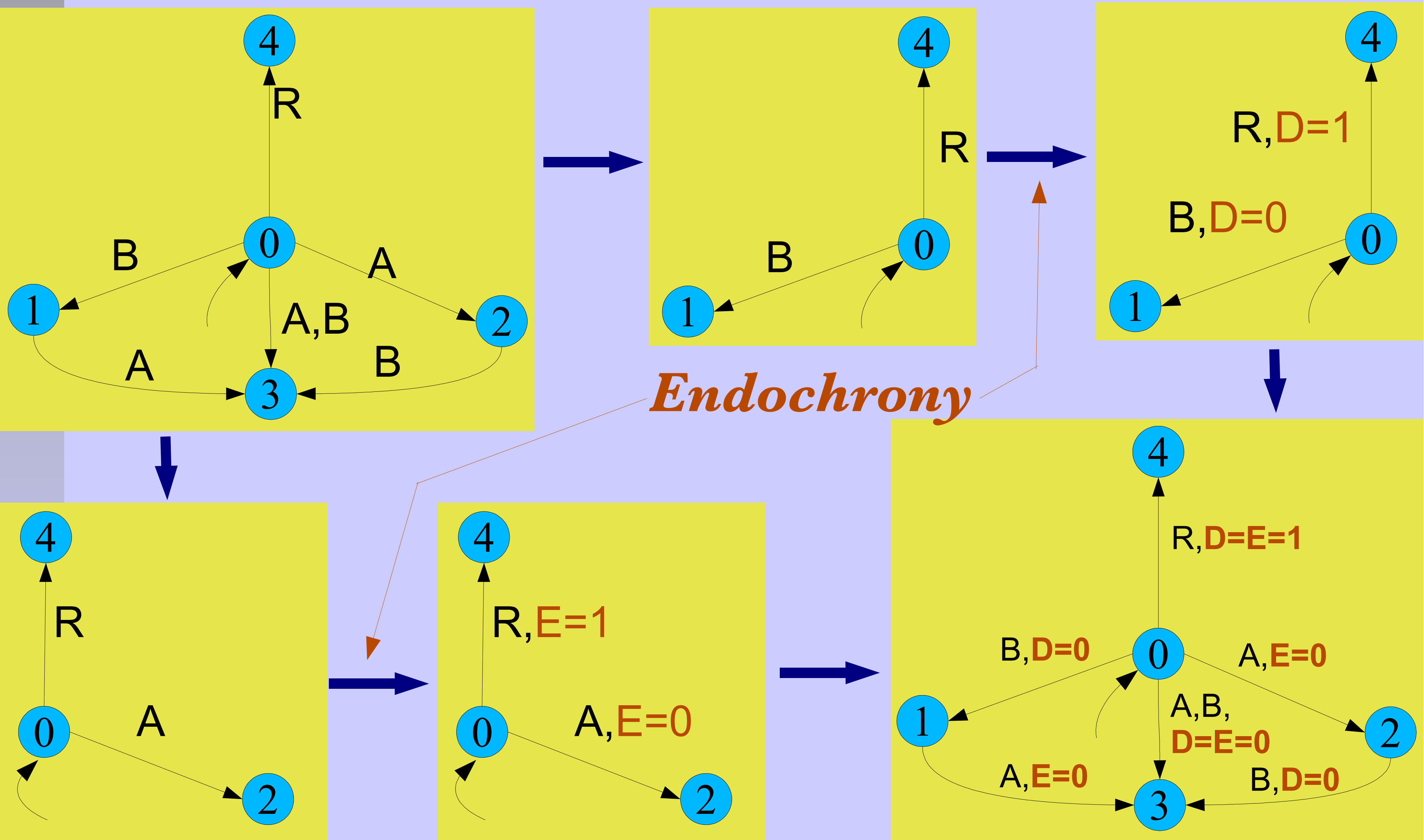
Non w-endochronous STS

Auxiliary variables

Wrapper STS



Synthesis of weak-endochrony through decomposition into sequential components



Endochrony

Weak-Isochrony

Def: $\mathrm{head}_{\Sigma}(s) = \mathrm{head} \circ \delta(\mathrm{Traces}_{\Sigma}(s))$

Def [weak-isochrony]: Σ_1, Σ_2 are weak-isochronous iff for all (s_1,s_2) reachable state of $\Sigma_1 \times \Sigma_2$, for all non-contradictory $r_1 \in \mathrm{head}_{\Sigma 1}(s_1)$, $r_2 \in \mathrm{head}_{\Sigma 2}(s_2)$, there exists r'_1 enabled in s_1 and r'_2 enabled in s_2 such that:

(i) $r'_i \ll r_i$

(ii) r'_1 and r'_2 synchronizable

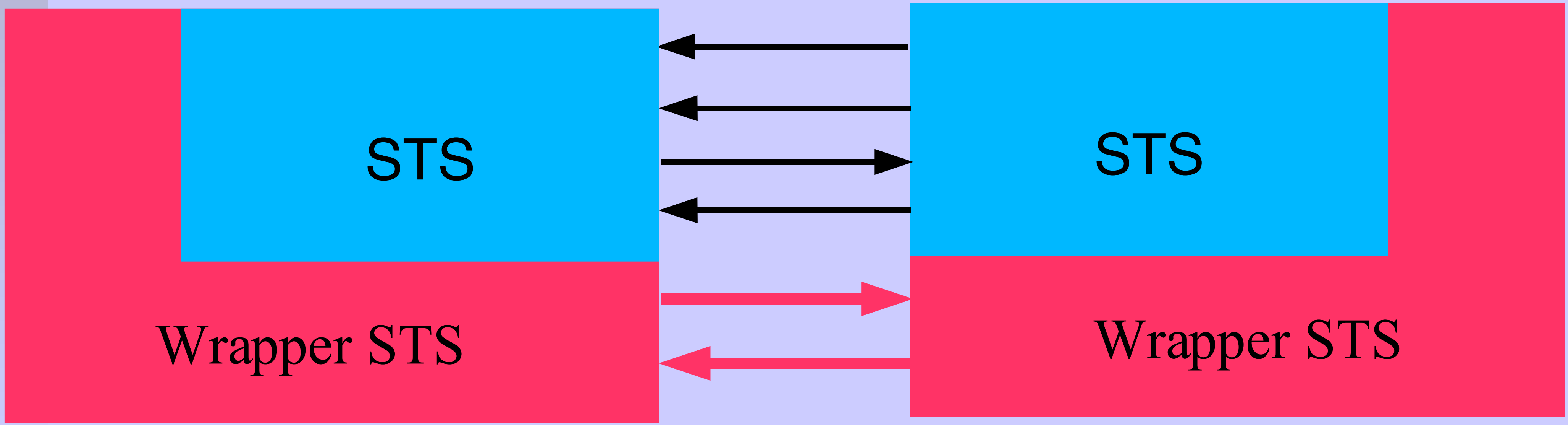
(iii) $r'_1 \vee r'_2$ is non-silent

Lemma 5: weak-isochrony is decidable on fintie STS

Combining weak-endochrony and weak-isochrony

Theorem 1: Σ_1, Σ_2 weak-endochronous and weakisochronous ⇒ Property (2) is satisfied

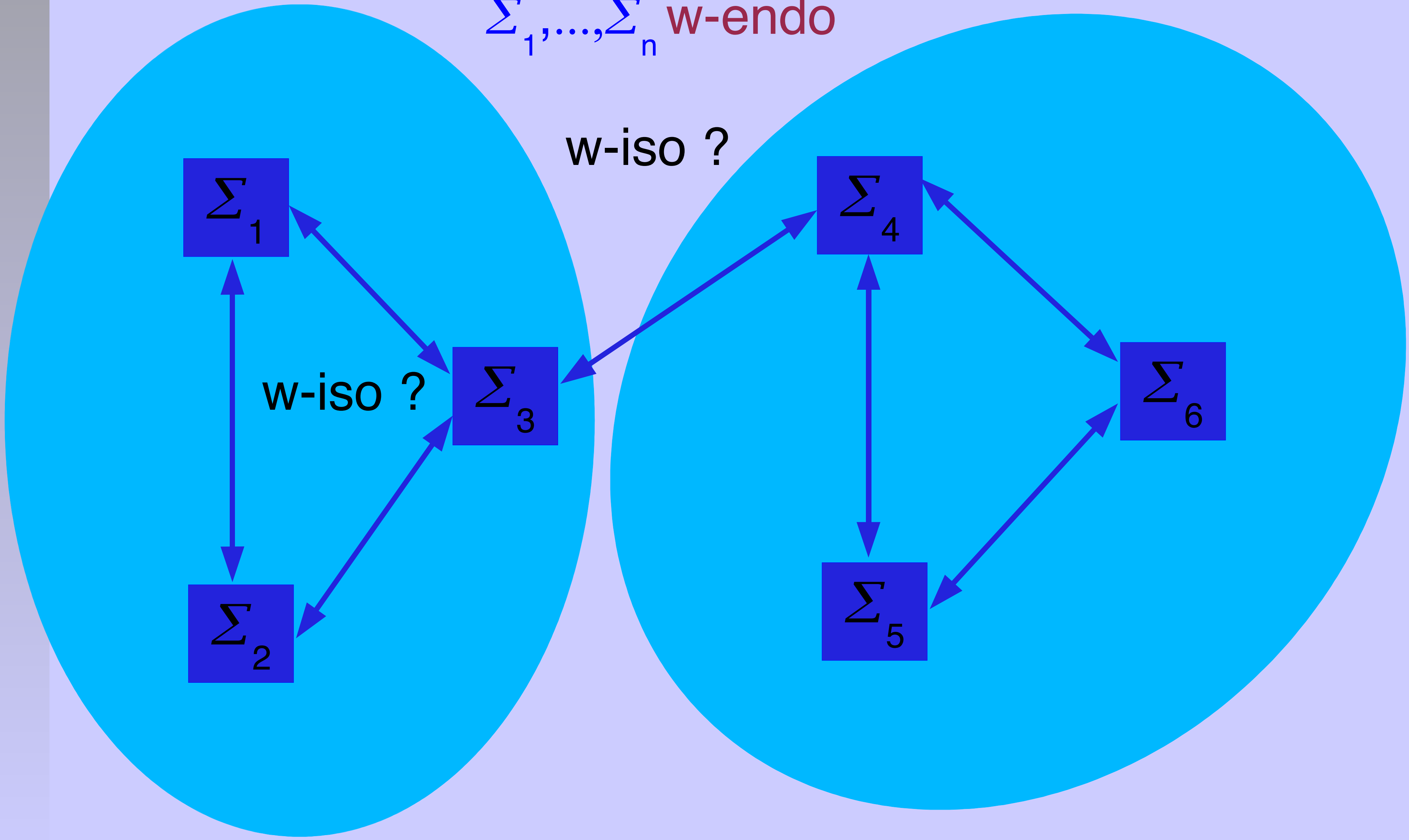
Open problem: synthesis of weak-isochrony



Systems with more than 2 processes

$$\text{Histories}(\Sigma_1 \times \ldots \times \Sigma_n) \overset{?}{=} \text{Histories}(\Sigma_1) \| \ldots \| \text{Histories}(\Sigma_n)$$

$\Sigma_1, \ldots, \Sigma_n$ w-endo



Θ α ν κ

Υ ο υ