Concurrency in Synchronous Systems

Dumitru Potop-Butucaru, Benoit Caillaud, Albert Benveniste

Concurrency in synchronous systems *

Dumitru Potop-Butucaru Benoît Caillaud Albert Benveniste
IRISA, Campus de Beaulieu, 35042 Rennes, France
{Dumitru.Potop,Benoit.Caillaud,Albert.Benveniste}@irisa.fr

9th February 2004

Abstract

In this paper we introduce the notion of weak endochrony, which extends to a synchronous setting the classical theory of Mazurkiewicz traces. The notion is useful in the synthesis of correct-by-construction communication protocols for globally asynchronous, locally synchronous (GALS) systems. The independence between various computations can be exploited here to provide communication schemes that do not restrict the concurrency while still guaranteeing correctness. Such communication schemes are then lighter and more flexible than their latency-insensitive or endo/isochronous counterparts.
Keywords: Synchrony, distribution, desynchronization, globally asynchronous locally synchronous (GALS), concurrency, trace theory

1 Introduction

The *notion of time* has been a crucial aspect of electronic system design for years. Dealing with concurrency, time and causality has become increasingly difficult as the complexity of the design grew. The *synchronous programming model* [3] has had major successes at the specification level because it provides a simpler way to access the power of concurrency in functional specification. Synchronous languages like ESTEREL [4], LUSTRE [10], and SIGNAL [14], the quasi-synchronous STATECHARTS modeling methodology [12], and design environments like SIMULINK/STATEFLOW all benefit from the simplicity of the *synchronous assumption*, *i.e.*: (1) the system evolves through an infinite sequence of successive atomic reactions indexed by a *global logical clock*, (2) during a reaction each component of the system computes new events for all its output signals based on its internal state and on the values of its input signals, and (3) the communication of all events between components occur *synchronously* during each reaction.

However, if the synchronous assumption simplifies system specification and verification, the problem of deriving a correct physical implementation from it does remain [3]. In particular, difficulties arise when the target implementation architecture has a distributed nature that does not match the synchronous assumption because of large variance in computation and communication speeds and because of the difficulty of

*Work supported by the ARTIST and COLUMBUS IST European projects. Extended version of a paper submitted to ACSD2004

maintaining a global notion of time. This is increasingly the case in complex microprocessors and Systems-on-a-Chip (SoC), and for many important classes of embedded applications in avionics, industrial plants, and the automotive industry.

For instance, many industrial embedded applications consist of multiple processing elements, operating at different rates, distributed over an extended area, and connected via communication buses (*e.g.* CAN for automotive applications, ARINC for avionics, and Ethernet for industrial automation). To use a synchronous approach in the development of such applications, one solution is to replace the asynchronous buses with communication infrastructures[1] that comply with a notion of global synchronization. However, such a fully synchronous implementation must be conservative, forcing the global clock to run as slow as the slowest computation/communication process. In consequence, the overhead implied by time-triggered architectures and synchronous implementations is often large enough to convince designers to use asynchronous communication architectures such as the ones implemented by the buses mentioned above.

Gathering advantages of both the synchronous and asynchronous approaches, the Globally Asynchronous Locally Synchronous (GALS) architectures are emerging as the architecture of choice for implementing complex specifications in both hardware and software. In a GALS system, locally-clocked synchronous components are connected through asynchronous communication lines. Thus, unlike for a purely asynchronous design, the existing synchronous tools can be used for most of the development process, while the implementation can exploit the more efficient/unconstrained/required asynchronous communication schemes. We further pursue, in this paper, our quest for correct-by-construction deployment of synchronous designs over GALS architectures.

1.1 Informal discussion of the issues

In the *synchronous paradigm* [11, 3], programs progress along a sequence of *reactions*. Thus, a synchronous run, also called *trace*, is a sequence $r_1, r_2, \ldots$, where each reaction r_i assigns values to the set of variables U of the considered program. Not all variables need to be involved in each reaction. However, this is taken into account by extending the domain of values of all variables with an extra symbol $\perp$, which denotes absence. Thus, absence can be tested and used to exercise control.

In contrast, no global clock exists in the *asynchronous* paradigm, and therefore no notion of reaction. Asynchronous runs of a program, also called *histories*, are tuples of signals, where a signal assigns a sequence of values to each variable. Absence has no meaning and cannot be sensed. Thus the extra value $\perp$ is not allowed.

As illustrated in fig. 1, relaxing the synchronous communication to obtain an asynchronous or GALS implementation consists of removing signal absence events ($\perp$) and synchronization boundaries of the reactions (the vertical gray boxes). Since time is only logic (and not metric), there remains no way of ordering events belonging to different variables. The result of the *desynchronization* operation is therefore a purely asynchronous run, represented by the second diagram. Clearly, relaxing synchrony is not invertible: the three signals associated with X, Y, and Z can be reorganized in successive reactions in many ways.

[1] like the family of Timed-Triggered Architectures introduced and promoted by H. Kopetz [13]

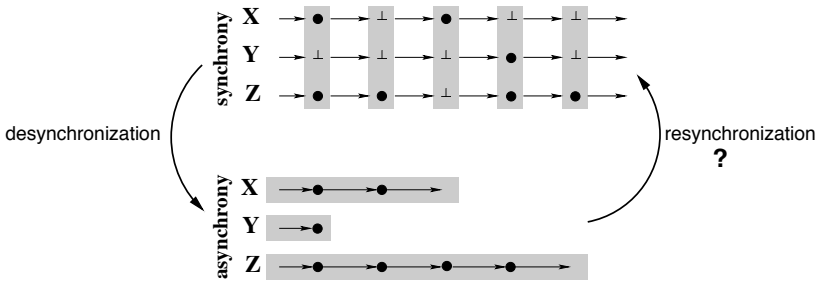


Figure 1: Relaxing synchrony. Symbols X, Y, Z denote variables, vertical gray boxes represent reactions, and horizontal gray boxes represent signals

The desynchronization problem addressed in this paper is informally illustrated in fig. 2, which shows how a channel of a small synchronous model is substituted in the implementation process by asynchronous communication lines. Note that immerging the synchronous modules in an asynchronous environment requires the development of schedulers that (1) decide when synchronous reactions are computed by each component (*e.g.* when all locally-needed input is available) and (2) reconstruct synchronous input events from their desynchronized counterparts, and feed them as input to the associated synchronous components.

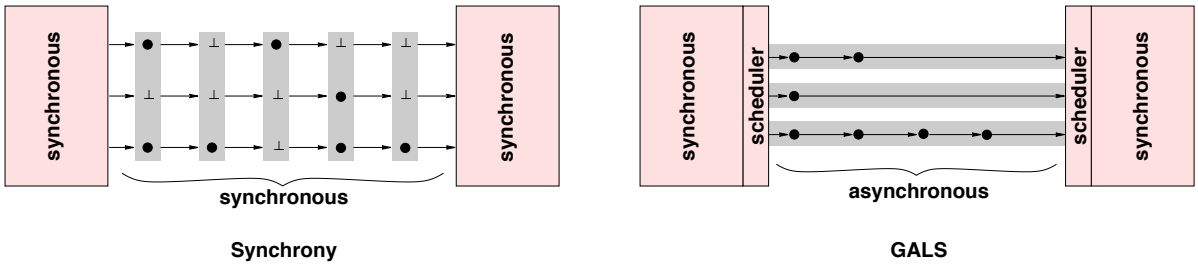


Figure 2: From synchrony to GALS.

Metric time is not so much an issue, as revealed by fig. 3, which depicts a run of a synchronous program in which all variables are present in all reactions. The effect of an asynchronous medium typically results in offsets between the dates of arrival of the variables that are part of the same reaction. However, this can be easily corrected at the receiver end thanks to a proper buffering mechanism. Such a technique has been

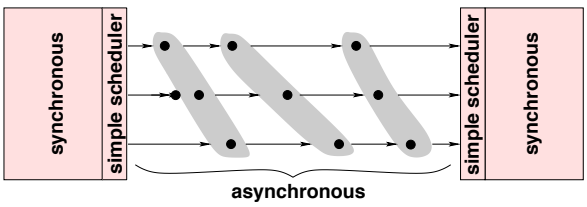


Figure 3: From synchrony to GALS: latency-insensitive programs.

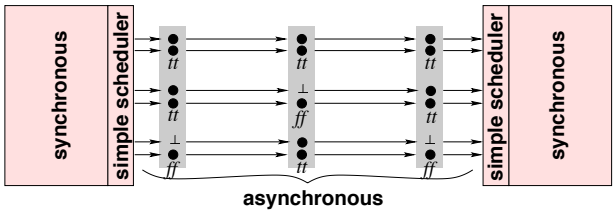


Figure 4: From synchrony to GALS: additional signaling for programs involving absence.

systematically used in [7] for hardware circuits, where it is referred to as *latency insensitive design*.

Unfortunately, this simple method does not extend to the general case of fig. 2: Since some variables can be absent in a nondeterministic way in each reaction, the buffering mechanism used in [7] does not ensure the correct reconstruction of reactions. A first solution to the general problem, illustrated in fig. 4, would consist in attaching to each variable v an additional Boolean variable $[v]$ which indicates at each reaction the presence/absence status of v. However, this simple solution has two drawbacks: It results in twice as many communications and, more importantly, the components of the resulting distributed system are running at the same pace. This is inappropriate whenever components have dissimilar activation rates.

Correctness and *efficiency* are the two main issues in the distributed implementation of synchronous specifications. Correctness is important because the advantages of synchrony lie with specification and verification. We would therefore like that each asynchronous history of the GALS implementation is covered by the verification of the initial synchronous model. In consequence, we will require that each history is the desynchronization of a trace of the initial synchronous specification.

At the same time, we also expect a GALS implementation *not* to restrict the functionality of the system, for instance through tough scheduling policies aimed at ensuring correctness in a simple way. The correct desynchronization criterion is therefore:

Criterion 1 (semantics preservation, informal) *The asynchronous observation of the initial synchronous system and of its GALS implementation produces to the same set of asynchronous histories.*

Efficiency, in terms of communication and execution speed, or number of exchanged messages, is also important, because GALS embedded systems often run on architectures with few resources. The issue is obviously subject to many trade-offs between different criteria, but exploiting properties like independence between computations or identifying execution modes to minimize communication, power consumption, or to allow multi-rate computation proves useful in most cases (thus offering criteria for comparing different approaches to solving the distribution problem).

1.2 Previous work

The most general **analysis** of the distributed implementation problem is due to Benveniste *et al.* [2]. There, heterogeneous models such as GALS architectures are formalized along with heterogeneous parallel composition operators. Then, by forgeting about causality aspects, a comprehensive notion of correct deployment

of a heterogenous system over a GALS architecture (a form of semantics preservation) can be defined, which covers the distributed implementation of synchronous specifications.

Previous approaches to **implementing** synchronous specifications over GALS architectures are respectively based on *latency-insensitive systems*, on *endo/isochronous systems*, and on *Kahn process networks* (KPN). All these approaches (described later in this section) follow a general pattern by trying to transform the components of the initial synchronous specification into "equivalent" synchronous components that have been "normalized" (by modifying their interface) in such a way as to make trivial schedulers (like those needed in fig. 3 and 4) correct and efficient. Every such approach is basically defined by two properties, which in turn determine the scope and complexity of the "normalizing" synthesis methods and the exact type of schedulers to consider:

scheduling-independence The first property characterizes the "normalized", easily-schedulable synchronous components. The behavior of such components is scheduling-independent, meaning that they know how to read their inputs, so that we can assume them self-clocked. The scheduling-independence properties of the previously-mentioned approaches are respectively latency-insensitivity, endochrony, and I/O determinism.

semantics preservation The second property is the actual semantics-preservation criterion. It ensures that enough signaling has been added between the self-clocked components as to prohibit asynchronous executions not corresponding to synchronous ones. Such a property is isochrony; the KPN-based approach does not specify one, not covering correctness aspects; finally, latency-insensitivity itself is highly constrained, ensuring by itself the preservation of semantics.

In the *latency-insensitive systems* of Carloni *et al.* [7], each synchronous component reads each input and writes every output at each execution instant (see fig. 3). Thus, the communication protocols effectively simulate a single-clocked system. This is very inefficient, but simplifies the implementation.

An essential improvement is brought by previous work by Benveniste *et al.* on *endo/isochronous systems* [1]. Informally speaking, a synchronous component is endochronous when the presence/absence of each variable can be inferred incrementally during each reaction from the values of state variables and present input variables. A pair of synchronous components is isochronous if there exists no pair of reactions that are not synchronizable, yet not contradictory (*i.e.* both present with different value on some common variable). Both endochrony and isochrony can be model-checked and even synthesized [1]. Unlike latency-insensitivity, the endo/isochronous approach is able to take into account execution modes and independence between components in order to minimize communication and allow multi-rate computation. The problem of the approach is that endochrony is not compositional [6], mainly due to poor handling of concurrency within components. This leads to inefficient synthesis for systems formed of more than 2 components.

While incomplete from the point of view of the semantics preserving criterion, we cite here the approach based on *Kahn process networks* [15, 16] because it is the only one formulated in a causal framework. Here, by requiring that each component has a deterministic input/output function, the determinism of the global system (and thus the independence from the scheduling scheme) is guaranteed. Giving the approach its

strength, the determinism is also its main drawback, as non-determinism is often useful in the specification and analysis of concurrent systems.

1.3 Contribution

Our work follows the same pattern, but brings an essential improvement over latency-insensitive design and endo/isochrony by allowing operations within a component to run independently when synchronization is not necessary. Being formulated in a non-causal framework, our approach is also less constrained than the KPN-based one, allowing nondeterminism in the less abstract causal model. The scheduling-independence criterion is in our case *weak endochrony*, while *weak isochrony* ensures the preservation of semantics. The two properties form together a correct desynchronization criterion that is decidable on finite synchronous systems. Moreover, transforming a general synchronous system to satisfy them is easy (although making it in an efficient way is a difficult, yet unsolved problem).

Our main contribution is the definition of weak endochrony, which represents a non-trivial extension to a synchronous setting of the classical trace theory [8].

The approach potentially supports signalization schemes that are simpler and more efficient than their latency-insensitive and endo/isochronous counterparts. This is due (1) to the fact that weak isochrony is able to exploit concurrency to provide lighter communication schemes and (2) to the fact that weak endochrony is compositional, leading to simpler synthesis schemes.

1.4 Outline

The remainder of the paper is organized as follows: Section 2 defines the formal framework used throughout the paper. Section 3 is on *weak endochrony*. After formally and intuitively introducing the notion, we state and prove here the decomposition and normal form results, thus showing the strong relation between weak endochrony and the classical trace theory [8]. Section 4 formally defines our desynchronization problem, introduces the notion of weak isochrony, and shows that weak endochrony and weak isochrony form a decidable criterion that guarantees the correct desynchronization. A technical comparison between our approach and existing ones is given in section 5, and we conclude in section 6.

2 Definitions

We formally define in this section the notions used throughout the paper: reactions, traces, and histories, a non-causal model of synchronous systems (the labelled synchronous transition systems), and the parallel composition operators.

2.1 Variables, values, and reactions

A *finite set* $\mathcal{V}$ of *variables* represents the communication channels connecting the synchronous components. Without losing generality, we shall assume that all the values exchanged between components belong to a

single domain $\mathcal{D}$. Each variable $v \in \mathcal{V}$ has at every moment one value $x \in \overline{\mathcal{D}} = \mathcal{D} \cup \{\perp, *\}$. A value $x \in \mathcal{D}$ represents a communication on the channel associated with v, $x = \perp$ represents the absence of communication, and $x = *$ tells us that the status of the channel is not known.

The interaction between a synchronous component and its environment consists in a sequence of *reactions*, which are mappings $r \in Reactions = \mathcal{V} \to \overline{\mathcal{D}}$. The *signature* of a reaction r is $sig(r) = \{v \mid r(v) \neq *\}$. The reactions of a given synchronous component will all have the same signature, given by the set of communication variables of the component. We denote with $Reactions(U)$ the set of reactions of signature $U \subseteq \mathcal{V}$. The *image* of a reaction r through the signature U is the reaction $r_{|U} \in Reactions(U)$ which equals r over variables common to U and $sig(r)$ and equals $\perp$ over $U \setminus sig(r)$. The *support* of a reaction r is $supp(r) = \{v \mid r(v) \in \mathcal{D}\}$. Note that $supp(r) \subseteq sig(r)$ for all $r \in Reactions$. We denote with $\perp_U$ the *stuttering reaction* of empty support and signature U.

On $\overline{\mathcal{D}}$ we define two partial orders. The first, denoted with $\leq$, is the least partial order such that $\forall x \in \mathcal{D} : * \leq \perp \leq x$. The second, denoted with $\sqsubseteq$, is the least partial order such that $* \sqsubseteq \perp$ and $\forall x \in \mathcal{D} : * \sqsubseteq x$. Each of the two relations induces *least upper bound* and *greatest lower bound* operators. We denote them, respectively, with $\vee$ and $\wedge$ (for $\leq$), and $\sqcup$ and $\sqcap$ (for $\sqsubseteq$). The operators $\wedge$ and $\sqcap$ are totally defined, while $\vee$ and $\sqcup$ are only partial. Note that $r_1 \vee r_2$ is defined whenever $r_1 \sqcup r_2$ is defined, and in this case the two values are equal. The order relations $\leq$ and $\sqsubseteq$ are extended variable-wise to reactions, and we denote with $\vee$, $\wedge$, $\sqcup$, and $\sqcap$ the associated least upper bound and greatest lower bound operators on reactions. Note that for any $r_1, r_2 \in Reactions$ and $\theta \in \{\vee, \wedge, \sqcup, \sqcap\}$, $r_1 \theta r_2$ exists whenever $r_1(v) \theta r_2(v)$ exists for all $v \in \mathcal{V}$, and in this case $\forall v \in \mathcal{V} : (r_1 \theta r_2)(v) = r_1(v) \theta r_2(v)$.

We will say that the reactions r_1 and r_2 are *non-contradictory* whenever $r_1 \vee r_2$ exists. For any two non-contradictory reactions r_1 and r_2 having the same signature U we define $r_1 \setminus r_2 = \bigwedge \{r \in Reactions(U) \mid r \vee (r_1 \wedge r_2) = r_1\}$. Othewise said, $\forall v \in \mathcal{V}$, $(r_1 \setminus r_2)(v) = *$ if $v \notin U$, $(r_1 \setminus r_2)(v) = r_1(v)$ if $v \in U$ and $r_2(v) = \perp$, and $(r_1 \setminus r_2)(v) = \perp$ otherwise. We also define in this case $r_1 \Delta r_2 = (r_1 \setminus r_2) \vee (r_2 \setminus r_1)$.

When $r_1 \sqcup r_2$ exists, we say that r_1 and r_2 are *synchronizable*.

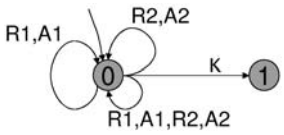
2.2 Traces and histories

A *synchronous trace*, simply called *trace* in the sequel, is a sequence of reactions having the same signature. For $U \subseteq \mathcal{V}$, we denote with $Traces(U)$ the set of traces of signature U. More exactly, $Traces(U) = Reactions(U)^\omega = \{(r_i)_{1 \leq i < n} \mid n \leq \infty \wedge \forall i : sig(r_i) = U\}$. We also denote with $Traces$ the set of all traces. Given a trace $\tau = (r_i)_{1 \leq i < n} (n \leq \infty)$ we denote with $length(\tau) = n - 1$ its length; we also denote with $\tau[i]$ its i^{th} element r_i. We denote $\tau[i, j]$ the sub-trace of τ, such that $\tau[i, j] =_{def} r_i r_{i+1} \ldots r_j$ if $i \leq j$, and $\tau[i, j] =_{def} \epsilon$, otherwise. The suffix of τ is defined has follows: $\tau[i \ldots] =_{def} (r_j)_{i \leq j < n}$. Note that any reaction is a trace of length 1.

The order relation $\sqsubseteq$ and the operators $\sqcup$, and $\sqcap$ are component-wise extended to pairs of traces of the same length. Given two traces τ_1 and τ_2 we shall say that they are *synchronizable* if $\tau_1 \sqcup \tau_2$ exists. We also extend component-wise to traces the image operator $\tau_{|U}$.

Two traces τ_1 and τ_2 can be concatenated if they have the same signature and if $length(\tau_1) < \infty$. The

```
module M1:input R1,R2,K;output A1,A2;
relation R1#K R2#K;
abort
  loop await R1 ; emit A1 end
||
  loop await R2 ; emit A2 end
when K
end module
```



(a) (b)

Figure 5: A small synchronous program (a) and its LSTS (b). We omitted the stuttering transitions and the $\perp$ values in the other transitions.

trace τ_1 is a prefix of τ_2 (written $\tau_1 \preceq \tau_2$) if, by definition, $\tau_2 = \tau_1\tau_3$ for some τ_3. The prefix relation is a partial order over traces.

A *history* is an asynchronous observation of one or more synchronous components. In a history, the synchronization constraints are forgotten, so that only the communications/values can be observed. Formally, a history is any mapping $\chi \in Histories = \mathcal{V} \to \mathcal{D}^\omega$. The concatenation operator over $Histories$ is the variable-wise extension of the concatenation operator over $\mathcal{D}^\omega$. The associated prefix order, denoted with $\preceq$, induces a greatest lower bound $\sqcap$ and a least upper bound $\sqcup$ operator. The *support* of a history χ is $supp(\chi) = \{v \in \mathcal{V} \mid \chi(v) \neq \epsilon\}$, where ϵ denotes the empty word over $\mathcal{D}$. In the sequel, we shall also denote with ϵ the trace of length 0 and the empty history. The length of a history χ is $length(\chi) = \max_{v\in\mathcal{V}} length(\chi(v))$.

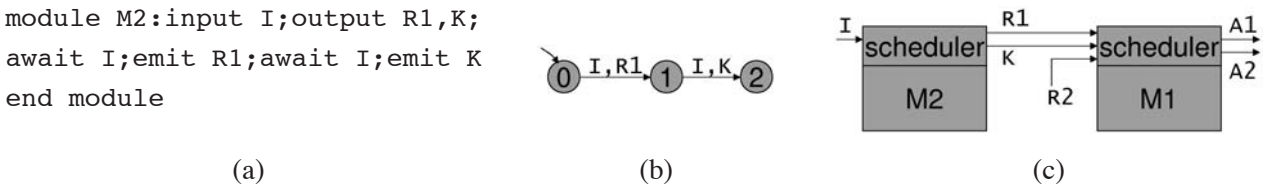
The *desynchronization morphism* $\delta : Traces \to Histories$ associates an asynchronous observation to every synchronous trace by forgetting the synchronization values $*$ and $\perp$: (i) $\delta(\epsilon) = \epsilon$; (ii) $\forall r \in Reactions$, $v \in \mathcal{V}$, $\delta(r)(v) = r(v)$, if $r(v) \in \mathcal{D}$, and $\delta(r)(v) = \epsilon$, otherwise; (iii) $\delta(\tau_1\tau_2) = \delta(\tau_1)\delta(\tau_2)$. The morphism preserves the concatenation and least upper bound operators.

For $\chi \in Histories$ and $U \subseteq \mathcal{V}$, we denote with $head_U(\chi)$ the maximal reaction of signature U such that its desynchronization is a prefix of χ.

2.3 Synchronous transition systems

To represent our synchronous systems we shall use throughout this paper the LSTS formalism. A *labelled synchronous transition system* (LSTS) is a synchronous automaton $\Sigma = (U, S, \to, \hat{s})$, where $U \subseteq \mathcal{V}$ is the set of variables of Σ (its signature), S is the set of *states*, $\to \subseteq S \times Reactions(U) \times S$ is the transition relation, and $\hat{s}$ is the initial state of the system. The notation $s \xrightarrow[\Sigma]{r} s'$ shall be used in the sequel to represent the transition $(s, r, s') \in \to$. For $s \in S$ and $\rho \in Reactions(U)$ we denote $Reactions_\Sigma(s) = \{r \mid \exists s' : s \xrightarrow[\Sigma]{r} s'\}$ and $Reactions_\Sigma(s, \rho) = \{r \mid \exists s' : s \xrightarrow[\Sigma]{r} s' \wedge r \leq \rho\}$. When confusion is not possible, the name of the LSTS can be omitted from these notations.

The LSTS $\Sigma = (U, S, \to, \hat{s})$ *accepts the synchronous trace* $\tau \in Traces(U)$ *in the state* s if, by definition,



(a) (b) (c)

Figure 6: The M2 synchronous module (a), its LSTS (b), and a small GALS system (c) formed by composing M1 and M2

there exist the states $s_1 = s, s_2, \ldots$ such that for all i $s_i \xrightarrow{\tau[i]} s_{i+1}$. When τ is of finite length n, we shall also write in this case $s \stackrel{\tau}{\Longrightarrow} s_n$. We denote with $Traces_\Sigma(s)$ the language formed by the traces accepted by Σ in the state s. A state $s' \in S$ is reachable from $s \in S$ if there exists τ such that $s \stackrel{\tau}{\Longrightarrow} s'$. We denote with $RSS_s(\Sigma)$ the set of states that are reachable from s. The *reachable state space* of Σ is $RSS(\Sigma) = RSS_{\hat{s}}(\Sigma)$.

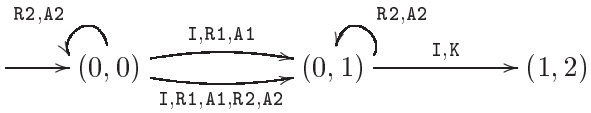
We shall only consider in the sequel *stuttering-invariant* systems that can take time while doing nothing. Formally, the LSTS $\Sigma = (U, S, \rightarrow, \hat{s})$ is stuttering-invariant if $s \xrightarrow{\perp_U} s$ is a transition of Σ for all $s \in RSS(\hat{s})$. Later in the paper we shall also require that systems do not change their state without interacting with their environment, so that $\perp_U$ only labels stuttering transitions.

Figure 5 gives a small synchronous program, written in Esterel[4], and its LSTS representation. When started, the program answers to various R1 and R2 request events with corresponding A1 and A2 answers. The process is aborted and the program terminates when the kill event K occurs. The program is stuttering-invariant, but the stuttering transitions have been omitted, for the sake of clarity.

2.4 Parallel composition

The composition of LSTSs is defined by means of a *synchronized product*. Given the LSTSs $\Sigma_i = (U_i, S_i, \rightarrow_i, \hat{s}_i), i = \overline{1,n}$, their product is the LSTS: $\Sigma_1 \times \ldots \times \Sigma_n = (\bigcup_{i=1}^n U_i, \prod_{i=1}^n S_i, \rightarrow, (\hat{s}_i)_{i=\overline{1,n}})$, where $(s_i)_{i=\overline{1,n}} \xrightarrow[\Sigma_1 \times \ldots \times \Sigma_n]{r} (s'_i)_{i=\overline{1,n}}$ if $\forall i : s_i \xrightarrow[\Sigma_i]{r_{|U_i}} s'_i$. The product operator is associative and commutative, and: $Traces_{\Sigma_1 \times \ldots \times \Sigma_n}(s_1, \ldots, s_n) = \{\bigsqcup_{i=1}^n \tau_i \mid \tau_i \in Traces_{\Sigma_i}(s_i)\}$, for all $s_i \in S_i, i = \overline{1,n}$.

For instance, the product of the LSTSs M1 (fig. 5(b)) and M2 (fig. 6(b)) is the LSTS:



The function of the M2 module is here to produce one request R1 and then emit K, thus requiring that M1 terminates. One trace of the product is, for instance, $(\mathtt{I}, \mathtt{R1}, \mathtt{A1})(\mathtt{I}, \mathtt{K}) \in Traces_{\mathtt{M1} \times \mathtt{M2}}((0,0))$.

3 Weakly endochronous systems

To extend the theory of Mazurkiewicz traces to our synchronous setting, we introduce the notion of *weak endochrony*. Weakly endochronous systems have trace languages that are closed under commutation of inde-

pendent reactions, where independence is given by the non-overlapping of supports. Moreover, to fully take into account the synchronous setting the languages are also closed under unification of independent reactions and overlapping non-contradictory reactions can be decomposed into atomic reactions that commute. As we shall see, such systems satisfy the classical commutation and normal form properties of the Mazurkiewicz traces. Unfortunately, our experience showed that one cannot use the existing theory to prove these results, as the dificculties mainly arise from the interpretation of the independence alphabet over synchronous reactions. Hence, the complexity of the proofs.

Definition 1 (weak endochrony) *We say that LSTS* $\Sigma = (U, S, \rightarrow, \hat{s})$ *is weakly endochronous if the following properties are satisfied for all* $s, s_1, s_2 \in RSS(\Sigma)$*, and for all* $r, r_1, r_2 \in Reactions(U)$*:*

W1. *Determinism:* $s \xrightarrow{r} s_1 \ \wedge \ s \xrightarrow{r} s_2 \ \Rightarrow s_1 = s_2$

W2. *Transitions with disjoint signal support commute and can be joined to form larger transitions. Formally, if* $supp(r_1) \cap supp(r_2) = \emptyset$ *then:*

a. $\left.\begin{array}{l} s \xrightarrow{r_1} s_1 \\ s \xrightarrow{r_2} s_2 \end{array}\right\} \Rightarrow \exists s' : \quad s \xrightarrow{r_1 \vee r_2} s'$

b. $s \xrightarrow{r_1} s_1 \xrightarrow{r_2} s_2 \ \Rightarrow \exists s' : \ s \xrightarrow{r_2} s'$

W3. *Non-contradictory overlapping reactions can be fragmented into reactions of smaller supports:*

$$\left.\begin{array}{l} s \xrightarrow{r_1} s_1 \\ s \xrightarrow{r_2} s_2 \\ r_1 \vee r_2 \ \textit{exists} \end{array}\right\} \Rightarrow \exists s' : \quad s \xrightarrow{r_1 \wedge r_2} s' \ \begin{array}{ll} \nearrow^{r_1 \setminus r_2} & s_1 \\ \searrow_{r_2 \setminus r_1} & s_2 \end{array}$$

The **intuition** behind our definition is that we are looking for systems where:

- all causality relations are implied by sequencing of messages on some communication channel (the only way to observe it in an asynchronous environment)
- a choice (priority, exclusiveness) between behaviors is always visible from the exterior as a choice over the value (and not presence/absence status) of some communication channel.

All internal decisions of such a system can therefore be observed or determined from the exterior even after desynchronization, meaning that there is no ambiguity concerning the construction of the schedulers (the system is self-clocked). Moreover, unlike latency-insensitive and endochronous systems, which ensure scheduling-independence by prohibiting all internal concurrency, weakly endochronous components only expose meaningful causality and choices.

Latency-insensitive systems [7] and endochronous systems [1] satisfy the axioms of weak endochrony. Moreover, unlike classical endochrony, weak endochrony is preserved by parallel composition:

Theorem 1 (Composition) *If Σ_1 and Σ_2 are weakly endochronous LSTSs, then $\Sigma_1 \times \Sigma_2$ is weakly endochronous.*

Proof: Let $\Sigma_i = (U_i, S_i, \rightarrow_i, \hat{s}_i), i = 1, 2$ be weakly endochronous LSTSs. To prove our theorem, we have to prove that the weak endochrony axioms **W1–W3** are satisfied by $\Sigma_1 \times \Sigma_2 = (U_1 \cup U_2, S_1 \times S_2, \rightarrow, (\hat{s}_1, \hat{s}_2))$, where $(s_1, s_2) \xrightarrow{r} (s_1', s_2')$ *iff* $s_i \xrightarrow[\Sigma_i]{r_{|U_i}} s_i'$, $i = 1, 2$.

W1: Let $s_i, s_i', s_i'' \in RSS(\Sigma_i), i = 1, 2$, and $r \in Reactions(U_1 \cup U_2)$ such that $(s_1, s_2) \xrightarrow{r} (s_1', s_2')$ and $(s_1, s_2) \xrightarrow{r} (s_1'', s_2'')$. Then, from the definition of $\Sigma_1 \times \Sigma_2$, we have $s_i \xrightarrow[\Sigma_i]{r_{|U_1}} s_i'$ and $s_i \xrightarrow[\Sigma_i]{r_{|U_i}} s_i''$ for all $i \in \{1, 2\}$. Then, by using **W1** on the endochronous systems Σ_1 and Σ_2 we obtain respectively $s_1' = s_1''$ and $s_2' = s_2''$. This implies $(s_1', s_2') = (s_1'', s_2'')$, so that **W1** is satisfied by $\Sigma_1 \times \Sigma_2$.

W3: Let $s_i, s_i', s_i'' \in RSS(\Sigma_i), i = 1, 2$, and $r', r'' \in Reactions(U_1 \cup U_2)$ such that $r' \vee r''$ exists and such that $(s_1, s_2) \xrightarrow{r'} (s_1', s_2')$ and $(s_1, s_2) \xrightarrow{r''} (s_1'', s_2'')$. Then, from the definition of $\Sigma_1 \times \Sigma_2$, we have $s_1 \xrightarrow[\Sigma_1]{r'_{|U_1}} s_1'$ and $s_1 \xrightarrow[\Sigma_1]{r''_{|U_1}} s_1''$, and since $(r'_{|U_1}) \vee (r''_{|U_1})$ exists (being equal to $(r' \vee r'')_{|U_1}$), we can apply **W3** and obtain:

$$\exists t_1 : \quad s_1 \xrightarrow[\Sigma_1]{(r'_{|U_1}) \wedge (r''_{|U_1})} t_1 \quad \begin{array}{ll} \xrightarrow[\Sigma_1]{(r'_{|U_1}) \setminus (r''_{|U_1})} & s_1' \\ \xrightarrow[\Sigma_1]{(r''_{|U_1}) \setminus (r'_{|U_1})} & s_1'' \end{array}$$

Similarly,

$$\exists t_2 : \quad s_2 \xrightarrow[\Sigma_2]{(r'_{|U_2}) \wedge (r''_{|U_2})} t_2 \quad \begin{array}{ll} \xrightarrow[\Sigma_2]{(r'_{|U_2}) \setminus (r''_{|U_2})} & s_2' \\ \xrightarrow[\Sigma_2]{(r''_{|U_2}) \setminus (r'_{|U_2})} & s_2'' \end{array}$$

Finally, by taking into account that for all i $(r'_{|U_i}) \wedge (r''_{|U_i}) = (r' \wedge r'')_{|U_i}$, $(r'_{|U_i}) \setminus (r''_{|U_i}) = (r' \setminus r'')_{|U_i}$, and $(r''_{|U_i}) \setminus (r'_{|U_i}) = (r'' \setminus r')_{|U_i}$, and by composing corresponding transitions we obtain:

$$\exists (t_1, t_2) : \quad (s_1, s_2) \xrightarrow{r' \wedge r''} (t_1, t_2) \quad \begin{array}{ll} \xrightarrow{r' \setminus r''} & (s_1', s_2') \\ \xrightarrow{r'' \setminus r'} & (s_1'', s_2'') \end{array}$$

Therefore, **W3** is satisfied.

The same proof scheme, consisting in mapping the hypothesis on the components Σ_1 and Σ_2 and then recomposing transitions of $\Sigma_1 \times \Sigma_2$, holds in the demonstration of **W2**. □

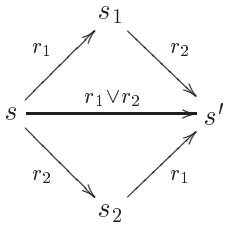
3.1 Atoms

The first step in establishing the relation with Mazurkiewicz trace theory is to define the alphabet of letters and the independence relation. In our case, the letters are the *atomic* reactions, of which all the other reactions are composed. Indeed, *atomic* reactions are those least, but not silent, reactions enabled in a given state of the system. Axiom W3 is the means by which atoms are constructed, by fragmenting larger reactions. The independence relation is the non-overlapping of supports of atoms. We define $Atoms_\Sigma(s)$ to be the set of atomic reactions enabled in state s of LSTS Σ: $Atoms_\Sigma(s) = \{r \in Reactions_\Sigma(s) \mid \forall r' \in Reactions_\Sigma(s),\ r \not> r'\}$. We also define the set of atoms smaller than a given reaction (as usual, we can omit Σ from the notations when no confusion is possible): $Atoms_\Sigma(s, \rho) = \{r \in Atoms_\Sigma(s) \mid r \leq \rho\}$.

The remainder of this section presents some basic properties of the weakly endochronous systems and proves that the atoms of $Atoms_\Sigma(s)$ indeed generate all the reactions of $Reactions_\Sigma(s)$.

Lemma 1 (Full diamond) *Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS, $s \in RSS(\Sigma)$, and $s \xrightarrow{r_i} s_i$, $i = 1, 2$. Then:*

$$supp(r_1) \cap supp(r_2) = \emptyset \Rightarrow \exists s' :$$



Proof: Consider s, s_1, s_2, r_1, r_2 satisfying the hypothesis of the lemma. Then, thanks to axiom **W2.1**, we can build s' and the transition $s \xrightarrow{r_1 \vee r_2} s'$. The transitions that close the diamond ($s_1 \xrightarrow{r_2} s'$ and $s_2 \xrightarrow{r_1} s'$) are simply built using axiom **W3** respectively on the pairs of transitions ($s \xrightarrow{r_1} s_1$, $s \xrightarrow{r_1 \vee r_2} s'$) and ($s \xrightarrow{r_2} s_2$, $s \xrightarrow{r_1 \vee r_2} s'$). □

More generally, the property holds for any number of transitions with mutually independent labels, which form a full diamond (with diagonals). In particular, the property holds for any set of independent atoms in a given state.

Corollary 1 (Commutation) *Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS, and $s \in RSS(\Sigma)$.*

a. If $a, b, r_i \in Reactions(U), i \geq 1$ such that $supp(a) \cap supp(b) = \emptyset$, then:

(1) $r_1 \dots r_k a b r_{k+1} \dots \in Traces_\Sigma(s) \Leftrightarrow r_1 \dots r_k b a r_{k+1} \dots \in Traces_\Sigma(s)$

(2) $r_1 \dots r_k a b r_{k+1} \dots \in Traces_\Sigma(s) \Rightarrow r_1 \dots r_k (a \vee b) r_{k+1} \dots \in Traces_\Sigma(s)$

b. Let, in addition, $\tau = r_1 r_2 \dots \in Traces_\Sigma(s)$ and $a \in Reactions_\Sigma(s)$ such that $\forall i : supp(a) \cap supp(r_i) = \emptyset$. Then, $a\tau \in Traces_\Sigma(s)$.

Proof: The first point is a direct application of the diamond lemma 1. The proof of the second point is performed in two simple steps. The first consists of proving the lemma for traces $r_1 r_2 \dots r_n$ of finite length

(induction over $n \geq 0$ using lemma 1). The second step considers infinite traces $r_1 r_2 \ldots$, builds the partial finite traces $a r_1 \ldots r_n, n \geq 0$, and then uses the determinism of Σ (axiom **W1**) to deduce that they are increasing prefixes of the infinite trace $a r_1 r_2 \ldots$ □

Lemma 2 (Decomposition into atoms) *Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS, $s \in RSS(\Sigma)$, and $r \in Reactions_\Sigma(s)$. Then:*

$$r = \bigvee_{a \in Atoms_\Sigma(s,r)} a$$

Proof: Let $r \in Reactions_\Sigma(s)$ and let $r' = r \setminus \bigvee_{a \in Atoms_\Sigma(s,r)} a$. Then, from axiom **W3**, $r' \in Reactions_\Sigma(s)$ such that $\forall a \in Atoms_\Sigma(s) : a \not\leq r'$. But this can only be when $r' = \bot_U$, in which case $r = \bigvee_{a \in Atoms_\Sigma(s,r)} a$, *q.e.d.*. □

Corollary 2 (Generation) *Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS, and $s \in RSS(\Sigma)$. Then:*

$$Reactions_\Sigma(s) = \{ \bigvee_{i=1}^{n} a_i \mid n \geq 0 \wedge \forall i : a_i \in Atoms_\Sigma(s) \wedge \forall i \neq j : supp(a_i) \cap supp(a_j) = \emptyset \}$$

Moreover, the decomposition of a reaction into atoms is unique up to permutation.

Proof: The inclusion $\subseteq$ is given by lemma 2. As $Atoms_\Sigma(s) \subseteq Reactions_\Sigma(s)$ and since $Reactions_\Sigma(s)$ is closed under union of reactions whose support is disjoint (axiom **W2.1**), we also have the reverse inclusion. The uniqueness is a direct consequence of axiom **W3**, which tells us that any two atoms that are non-contradictory and have non-overlapping supports are identical. □

Lemma 3 (Atom preservation) *Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS, let $s \in RSS(\Sigma)$, $\rho \in Reactions(U)$, and $r \leq \rho$ such that $s \xrightarrow{r} s'$. Then, $Atoms_\Sigma(s', \rho \setminus r) = Atoms_\Sigma(s, \rho \setminus r)$.*

Proof: According to lemma 1, any atom $a \in Atoms_\Sigma(s)$ such that $supp(a) \cap supp(r) = \emptyset$ is a transition of $Reactions_\Sigma(s')$. Therefore: $Reactions_\Sigma(s', \rho \setminus r) \supseteq \{r' \in Atoms(s, \rho) \mid r' \leq \rho \setminus r\} = Atoms_\Sigma(s, \rho \setminus r)$ Then, using corollary 2 we obtain:

$$Reactions_\Sigma(s', \rho \setminus r) \supseteq Reactions_\Sigma(s, \rho \setminus r) \tag{1}$$

Conversely, any atom of $Atoms_\Sigma(s, \rho \setminus r)$ commutes with r, and thus is a reaction of $Reactions_\Sigma(s)$, so that $Atoms_\Sigma(s', \rho \setminus r) \subseteq \{r' \in Reactions_\Sigma(s, \rho) \mid r' \leq \rho \setminus r\} = Reactions_\Sigma(s, \rho \setminus r)$. By using corollary 2 we obtain $Reactions_\Sigma(s', \rho \setminus r) \subseteq Reactions_\Sigma(s, \rho \setminus r)$, and from equation 1 we deduce $Reactions_\Sigma(s', \rho \setminus r) = Reactions_\Sigma(s, \rho \setminus r)$. *Q.e.d.* □

Lemma 4 (technical lemma) *Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS, $s \in RSS(\Sigma)$, and $V \subseteq U$. Suppose that $\tau, \theta \in Traces_\Sigma(s)$ such that:*

$$\delta(\theta)(v) = \begin{cases} \delta(\tau)(v), \text{ if } v \in V \\ \epsilon, \text{ otherwise} \end{cases}$$

Then, $(\tau[1]_{|V})_{|U}, (\tau[1]_{|(U \setminus V)})_{|U} \in Reactions_\Sigma(s)$.

Proof: Consider the unique decomposition of $\tau[1]$ into atoms of $Atoms_\Sigma(s)$: $\tau[1] = \bigvee_{a \in Atoms_\Sigma(s,\delta(\tau[1]))} a$. If all the atoms $a \in Atoms_\Sigma(s, \delta(\tau[1]))$ have the support fully included in either V or $U \setminus V$, then:

$$(\tau[1]_{|V})_{|U} = \bigvee \{a \in Atoms_\Sigma(s, \delta(\tau[1])) \mid supp(a) \in V\} \in Reactions_\Sigma(s)$$

$$(\tau[1]_{|(U \setminus V)})_{|U} = \bigvee \{a \in Atoms_\Sigma(s, \delta(\tau[1])) \mid supp(a) \in U \setminus V\} \in Reactions_\Sigma(s)$$

and the lemma is proved. Assume to the contrary that there exists $a \in Atoms_\Sigma(s, \delta(\tau[1]))$ such that $supp(a) \cap V \neq \emptyset$ and $supp(a) \cap (U \setminus V) \neq \emptyset$. Consider, in this case, the derivation of $\theta \in Traces_\Sigma(s)$: $s = s_0 \xrightarrow{\theta[1]} s_1 \xrightarrow{\theta[1]} \cdots$. We shall prove next, by induction over $i \geq 0$ that $a \in Atoms_\Sigma(s_i)$ and $supp(a) \cap supp(\theta_{i+1}) = \emptyset$. Given the choice of a, this implies in turn that there exists $v \in V$ such that: $\delta(\tau)(v) \succeq \delta(a)(v) \neq \epsilon = \delta(\theta)(v)$ which contradicts the hypothesis on τ and θ. Therefore, there exists no a such that $supp(a) \cap V \neq \emptyset$ and $supp(a) \cap (U \setminus V) \neq \emptyset$.

The induction: For $i = 0$, we obviously have $a \in Atoms_\Sigma(s) = Atoms_\Sigma(s_0)$. Also, a and $\theta[1]$ are non-contradictory, and from axiom **W3** we have $a \wedge \theta[1] \in Reactions_\Sigma(s)$. Since a is an atom not included in $\theta[1]$, we conclude $supp(a) \cap supp(\theta[1]) = \emptyset$.

Assume now the property established for $i \leq k$. We prove it for $i = k + 1$. Since $supp(a) \cap supp(\theta[k+1]) = \emptyset$ we have, by applying the atom preservation lemma 3, $a \in Atoms_\Sigma(s_{k+1})$. But since the induction hypothesis implies $supp(\delta(\theta[1, k+1])) \cap supp(a) = \emptyset$, we will have $\theta[k+2]_{|supp(a)} \leq head_{supp(a)}(\delta(\theta))_{|supp(a)}$, so that a and $\theta[k+2]$ are non-contradictory. Then, $a \wedge \theta[k+2] \in Reactions_\Sigma(s)$, and since a is an atom not included in $\theta[k+2]$, we have $supp(a) \cap supp(\theta[k+2]) = \emptyset$. □

Lemma 5 (technical lemma) *Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS, $s \in RSS(\Sigma)$, $\tau \in Traces_\Sigma(s)$, and $r \in Reactions_\Sigma(s)$ such that $\delta(r) \preceq \delta(\tau)$. Then, there exists $\tau' \in Traces_\Sigma(s)$ such that $\delta(\tau') = \delta(\tau)$ and $\tau'[1] = r$.*

Proof: Let $r = \bigvee_{i=1}^{l} a_i$ be the unique decomposition of r into atoms of $Atoms_\Sigma(s)$. We prove our property by induction over $l \geq 0$. When $l = 0$ we have $r = \bot_U$, and we can always insert a stuttering transition in the beginning of a trace, so that $\tau' =_{def} \bot_U \tau \in Traces_\Sigma(s)$ satisfies our requirements.

Assume the property proved for $l = k$. We prove it for $l = k + 1$. From the induction hypothesis, there exists $\tau'' \in Traces_\Sigma(s)$ such that $\delta(\tau'') = \delta(\tau)$ and $\tau''[1] = \bigvee_{i=1}^{k} a_i$. Let $s = s_0, s_1, \ldots$ be the sequence of states in the derivation of τ''. Since $\delta(a_{k+1}) \preceq \delta(r) \preceq \delta(\tau) = \delta(\tau'')$ and $supp(\delta(a_{k+1})) \cap supp(\tau''[1]) = \emptyset$, there exists $m \geq 2$ such that $supp(a_{k+1}) \cap supp(\tau''[m]) \neq \emptyset$. We can assume that m is minimal with this property. Then, by applying the atom preservation lemma 3 we have $a_{k+1} \in Atoms_\Sigma(s_{m-1})$. The minimality of m also implies $supp(\delta(\tau''[1, m-1])) \cap supp(a) = \emptyset$, which in turn implies that for all $v \in supp(a)$ we have $\delta(\tau''[m \ldots])(v) = \delta(\tau'')(v)$. Then, since $a_{k+1} \leq \tau''[1]$, we deduce that a_{k+1} and $\tau''[m]$ are non-contradictory. By taking into account that $supp(a_{k+1}) \cap supp(\tau''[m]) \neq \emptyset$ we obtain then $a_{k+1} \leq \tau''[m]$. Then, since $supp(\tau''[1, m-1]) \cap supp(a_{k+1}) = \emptyset$ we can use the commutation corollary 1 to bring a_{k+1} in the first transition, which means that: $\tau' = (\bigvee_{i=1}^{k+1} a_i)\, \tau''[2, m-1]\, (\tau''_m \setminus a_{k+1})\, \tau''[m+1 \ldots] \in Traces_\Sigma(s)$. The induction is then proved, and so is the lemma. □

Theorem 2 (Disjoint support) *Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS, $s \in RSS(\Sigma)$, and $V \subseteq U$. Suppose that $\tau, \theta \in Traces_\Sigma(s)$ such that:*

$$\delta(\theta)(v) = \begin{cases} \delta(\tau)(v), \text{ if } v \in V \\ \epsilon, \text{ otherwise} \end{cases}$$

Then, the traces $\tau', \tau'' \in Traces(U)$ defined for all $i \geq 1$ by:

$$\tau'[i] = (\tau[i]_{|V})_{|U} \qquad \tau''[i] = (\tau[i]_{|U \setminus V})_{|U}$$

are traces of $Traces_\Sigma(s)$.

Proof: We shall prove here that for all $i \geq 0$, $\tau'[1,i], \tau''[1,i] \in Traces_\Sigma(s)$. Then, since Σ is deterministic, we have $\tau', \tau'' \in Traces_\Sigma(s)$, and the theorem is proved. The demonstration is based on the inductive construction of $\tau^i, \theta^i \in Traces_\Sigma(s)$ such that for all i, the following equations are satisfied: (a) $\delta(\tau^i) = \delta(\tau)$ and $\delta(\theta^i) = \delta(\theta)$, (b) $\tau^i[1,i] = \theta^i[1,i] = \tau'[1,i]$, (c) $\tau^i[i+1, 2i] = \tau''[1,i]$, and (d) $\tau^i[2i+1\ldots] = \tau[i+1\ldots]$. From this inductive definition, it is obvious that $\tau'[1,i] = \tau^i[1,i] \in Traces_\Sigma(s)$. Moreover, from $\forall i, j : supp(\tau'[i]) \cap supp(\tau''[j]) = \emptyset$ and by using the commutation corollary 1 on $\tau^i[1,2i]$,we can also deduce that $\tau''[1,i] \in Traces_\Sigma(s)$.

The construction is performed inductively. Obviously, $\tau^0 = \tau$ and $\theta^0 = \theta$ satisfy the conditions (a)-(d) for $i = 0$. Assume now τ^k and θ^k constructed. We build τ^{k+1} and θ^{k+1}. Let s_k, t_k be defined by $s \xRightarrow{\tau[1,k]} s_k$ and $s \xRightarrow{\theta[1,k]=\tau'[1,k]} t_k$. Note that, by construction, we also have: $t_k \xRightarrow{\tau''[1,k]} s_k$. From the commutation corollary 1, and since $supp(\delta(\tau'[1,i])) \cap supp(\delta(\tau''[1,i])) = \emptyset$, we have $s \xRightarrow{\tau''[1,k]\theta^k[1,k]} s_k$. The same lemma applied to the traces of disjoint support $\tau''[1,k]$ and θ^k gives $\tau''[1,k]\theta^k \in Traces_\Sigma(s)$. This means that $\theta^k[k+1\ldots] \in Traces_\Sigma(s_k)$, so that we can consider the sequence of states $s_k, s_{k+1}, s_{k+2}, \ldots$ in the derivation of $\theta^k[k+1\ldots]$.

Then, the traces $\tau^k[k+1\ldots], \theta^k[k+1\ldots] \in Traces_\Sigma(s_k)$ satisfy the hypothesis of lemma 4, so that $\tau[k+1]_{|V|U}, \tau[k+1]_{|U \setminus V|U} \in Traces_\Sigma(s_k)$, which, by definition, means that $\tau'[k+1], \tau''[k+1] \in Traces_\Sigma(s_k)$. Since $\tau[k+1] = \tau'[k+1] \vee \tau''[k+1]$, and since $supp(\tau''[k+1]) \cap supp(\tau'[k+1]) = \emptyset$, by applying the commutation lemma 1, we obtain $\tau''[1,k]\tau'[1,k]\tau'[k+1]\tau''[k+1]\tau[k+2\ldots] \in Traces_\Sigma(s)$ Since $supp(\delta(\tau''[1,k])) \cap supp(\delta(\tau'[1,k+1])) = \emptyset$, and by applying again the commutation corollary 1 we have $\tau^{k+1} =_{def} \tau''[1,k+1]\tau'[1,k+1]\tau[k+2\ldots] \in Traces_\Sigma(s)$. Obviously, τ^{k+1} satisfies the properties (a)-(d). To build θ^{k+1}, note that $\tau'[1,k+1] = \tau^{k+1}[1,k+1] \in Traces_\Sigma(s)$, so that $\tau'[k+1] \in Reactions_\Sigma(t_k)$. But we also have $\theta^k[k+1\ldots] \in Traces_\Sigma(t_k)$ and $\delta(\tau'[k+1]) \preceq \delta(\theta^k[k+1\ldots])$, so that (from lemma 5), there exists $\theta' \in Traces_\Sigma(t_k)$ such that $\theta'[1] = \tau'[k+1]$ and $\delta(\theta') = \delta(\theta^k[k+1\ldots])$. Then, $\theta^{k+1} = \theta^k[1,k]\theta'$ satisfies our requirements, and the proof is completed. □

Corollary 3 (Disjoint support) *Under the hypothesis of theorem 2, there exists $\phi \in Traces_\Sigma(s)$ such that:*

$$\delta(\phi)(v) = \begin{cases} \delta(\tau)(v), \text{ if } v \in U \setminus V \\ \epsilon, \text{ otherwise} \end{cases}$$

3.2 Normal form

Given an execution of a weakly endochronous system, we can put it in *normal form*, where the atomic operations are re-combined to form largest transitions, so that each atom is executed as early as possible. Like for the Cartier-Foata normal form, putting a trace in normal form keeps unchanged the causal relations between atomic operations (determined by support overlapping). Unlike in the classical trace theory, however, our synchronous setting facilitates the understanding and the manipulations, as the normal form is a synchronous trace, and not a sequence of cliques of atoms/letters.

The normal form can be computed from the desynchronized version of an execution. Even more, constructing a normal form execution from a general history χ (one that does not necessarily correspond to a trace) results in an execution τ which is maximal such that $\delta(\tau) \preceq \chi$. This maximal execution is unique upto commutation of independent atoms and any smaller execution can be "completed" to one that is maximal. The remainder of the section is dedicated to the formalization and proof of our normal form results.

This section is structured as follows: We first explain how the normal form associated with a history is constructed. Then, we introduce the notion of downward commutation, which will play an important role in subsequent developments. The last and longest part of the section is dedicated to the definition and proof of our normal form results, which form the theoretical core of our paper.

3.2.1 Definition

Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS. Then, for all $s \in RSS(\Sigma)$ and $\rho \in Reactions$ we define:

$$\lfloor \rho \rfloor_s = \bigvee_{a \in Atoms(s,\rho)} a$$

the *unique* maximal reaction smaller than ρ.

Let $\chi \in Histories$ such that $supp(\chi) \subseteq U$. We shall denote with $NF_\Sigma(s, \chi)$ the trace of Σ obtained by starting in state s and performing at each step the maximal reaction enabled by the remainder of the history χ. When no confusion is possible, we will also write $NF(s, \chi)$. Formally, $NF_\Sigma(s, \chi) = r_1 r_2 \ldots$, where $s_1 = s$, $\chi_1 = \chi$, and for all $i \geq 1$: $r_i = \lfloor head_U(\chi_i) \rfloor_{s_i}$, $\chi_{i+1} = \chi_i \setminus \delta(r_i)$, and $s_i \xrightarrow{r_i} s_{i+1}$.

Note that our normal form construction may involve histories whose support is not restricted to the signature of the considered LSTSs. This induces no problems, as values of variables not belonging to the signature are simply ignored by the considered system. In the sequel, we shall exploit this property by only considering throughout the proofs histories whose support is included in the signature of the LSTS.

3.2.2 Downward commutation

We call *downward commutation* the transformation of traces, by means of commutation, that changes the position of atoms from their initial reactions to reactions of lesser index. Consider the trace $\tau \in Traces_\Sigma(s)$, its derivation $s = s_1 \xrightarrow{\tau[1]} s_2 \tau[2] \longrightarrow \cdots$, and $a \in Atoms_\Sigma(s_i)$ such that $a \leq \tau[i]$. A downward commutation of a to the reaction of index $j < i$ is possible if $supp(a) \cap \tau[k] = \emptyset$ for $j \leq k < i$. In this case,

lemma 3 implies that for all $j \leq k < i$ we have $a \in Atoms_\Sigma(s_k)$, and by applying property **W3** and the commutation corollary 1 we obtain

$$\tau[1] \dots \tau[k-1](\tau[k] \vee a)\tau[k+1] \dots \tau[i-1](\tau[i] \setminus a)\tau[i+1] \dots \in Traces_\Sigma(s)$$

We call such a transformation a *downward commutation step*.

3.2.3 Normal form results

Theorem 3 (Normal form and confluence) *Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS. For all $\chi \in Histories$, $s \in RSS(\Sigma)$, $\tau \in Traces_\Sigma(s)$ such that $\delta(\tau) \preceq \chi$, we have $\delta(\tau) \preceq \delta(NF_\Sigma(s, \chi))$. Suppose, in addition, that χ is of finite length and $s \stackrel{NF_\Sigma(s,\chi)}{\Longrightarrow} s'$. Then, if τ is chosen such that $\delta(\tau)$ is maximal, we have $\delta(\tau) = \delta(NF_\Sigma(s, \chi))$ and $s \stackrel{\tau}{\Longrightarrow} s'$.*

Proof: Case 1: finite histories. In this case, we prove the stronger equality property by induction over $\mid \chi \mid =_{def} \sum_{x \in \mathcal{V}} length(\chi(x))$. The inequality property is a direct consequence.

When $\mid \chi \mid = 0$, we have $\chi = \epsilon$, and the property is clearly satisfied. Let now $\chi \in Histories$, finite, $\chi \neq \epsilon$. We assume the normal form properties satisfied for all $\chi' \in Histories$ such that $\mid \chi' \mid < \mid \chi \mid$, and we prove it for χ. Let $s \in RSS(\Sigma)$ and $\tau \in Traces_\Sigma(s)$ such that $\delta(\tau) \preceq \chi$. We assume that τ is chosen such that $\delta(\tau)$ is maximal. Let s', s'_1, and $\overline{s}$ be respectively defined by $s \stackrel{NF_\Sigma(s,\chi)}{\Longrightarrow} s'$, $s \xrightarrow{NF_\Sigma(s,\chi)[1]} s'_1$, $s \xrightarrow{\tau} \overline{s}$.

Case 1.a: $\tau[1] = NF_\Sigma(s, \chi)[1]$. Let in this case $\tau_1 = \tau[2, length(\tau)]$ and $\chi_1 = \chi \setminus \delta(\tau[1])$. By construction, we have $NF_\Sigma(s'_1, \chi_1) = NF_\Sigma(s, \chi)[2, length(NF_\Sigma(s, \chi))]$. But the maximality of τ and the construction also imply that $\delta(\tau_1)$ is maximal such that $\delta(\tau_1) \preceq \chi_1$. Then, $\tau_1 \in Traces_\Sigma(s'_1)$ has the property that $\delta(\tau_1)$ is maximal such that $\delta(\tau_1) \preceq NF_\Sigma(s'_1, \chi_1)$. By applying the induction hypothesis, we obtain $\delta(\tau_1) = \delta(NF_\Sigma(s'_1, \chi_1))$ and $s'_1 \stackrel{\tau_1}{\Longrightarrow} s'$. By inserting the initial transition $s \xrightarrow{\tau[1]} s'_1$, we obtain $\delta(\tau) = \delta(NF_\Sigma(s, \chi))$ and $s \stackrel{\tau}{\Longrightarrow} s'$. *Q.e.d.*

Case 1.b: $\tau[1] \neq NF_\Sigma(s, \chi)[1]$. In this case, $\tau[1] \leq NF_\Sigma(s, \chi)[1]$. However, some atoms $a_1, \dots, a_k \in Atoms_\Sigma(s, head_U(\chi))$ are not part of $\tau[1]$. Consider the derivation of τ: $s \xrightarrow{\tau[1]} s_1 \xrightarrow{\tau[2]} s_2 \xrightarrow{\tau[3]} \cdots$. Then, according to lemma 3, each of the missing atoms a_i will be included in the first reaction $\tau[k_i]$ such that $a_i \wedge \tau[k_i] \neq \bot_U$ (a_i remains an atom and blocks the channels belonging to its support until executed). As the trace τ is maximal, each a_i is eventually executed, so that we can use downward commutation to transform τ into τ', where:

$$\tau'[i] = \begin{cases} \tau[1] \vee \bigvee_{j=1}^{k} a_j = NF_\Sigma(s, \chi)[1], & \text{if} \quad i = 1 \\ \tau[i] \setminus \bigvee_{k_j = i} a_j, & \text{otherwise} \end{cases}$$

Note that the transformation is done by pushing each a_i in the first reaction through a downward commutation step.

Then, by applying case 1.a, we obtain $\delta(\tau') = \delta(NF_\Sigma(s, \chi))$ and $s \stackrel{\tau'}{\Longrightarrow} s'$, which implies $\delta(\tau) = \delta(NF_\Sigma(s, \chi))$ and $s \stackrel{\tau}{\Longrightarrow} s'$. This completes the induction step, meaning that the theorem holds in the finite case.

Case 2: infinite histories. Let $\chi \in Histories$ s.t. $length(\chi) = \infty$, $s \in RSS(\Sigma)$, and $\tau \in Traces_\Sigma(s)$ s.t. $\delta(\tau) \preceq \chi$.

Case 2.a: $length(NF_\Sigma(s,\chi))$ **finite.** Let then τ' be a finite prefix of τ and let χ' be the *finite* history defined by $\chi' = \delta(NF_\Sigma(s,\chi)) \sqcup \delta(\tau')$. Note that $NF_\Sigma(s,\chi') = NF_\Sigma(s,\chi)$ by construction.

By applying the first case of the theorem, we obtain $\delta(\tau') \preceq \delta(NF_\Sigma(s,\chi'))$. But $NF_\Sigma(s,\chi') = NF_\Sigma(s,\chi)$ (by construction), so that τ' is a trace with $\delta(\tau') \preceq \delta(NF_\Sigma(s,\chi))$. This inequality holds for all prefix τ' of τ, and by taking the union of the prefixes we obtain $\delta(\tau) \preceq \delta(NF_\Sigma(s,\chi))$, *q.e.d.*.

Case 2.b: $length(NF_\Sigma(s,\chi))$ **infinite.** As $\delta(NF_\Sigma(s,\chi)) \preceq \chi$, we can define $\overline{\chi} = \chi \setminus \delta(NF_\Sigma(s,\chi))$. Note that for all $x \in \mathcal{V}$, we have $\overline{\chi}(x) \neq \epsilon \Rightarrow length(\delta(NF_\Sigma(s,\chi))(x)) < \infty$. Then, we can define $\psi_i \in Reactions(U)$ and $\Psi_i \in Traces(U), i \geq 1$ with $\Psi_i = \psi_1 \dots \psi_i$ and such that for all $x \in U$:

$$\psi_i(x) = \begin{cases} NF_\Sigma(s,\chi)[i](x) & \text{if} NF_\Sigma(s,\chi)[i](x) \neq \bot \\ (\chi \setminus \delta(\Psi_{i-1}))(x)[1] & \text{if} NF_\Sigma(s,\chi)[i](x) = \bot \\ & \text{and} (\chi \setminus \delta(\Psi_{i-1}))(x) \neq \epsilon \\ \bot & \text{otherwise} \end{cases}$$

Also let $\Psi = \psi_1\psi_2\dots$. In other words, after finishing all the values of $NF_\Sigma(s,\chi)$ for a given variable x, Ψ considers one value of χ per reaction until no more values are available. This ensures good fairness properties, so that $\delta(\Psi) = \chi$. Moreover, the construction ensures that for all i we have:

$$NF_\Sigma(s,\delta(\Psi_i)) = NF_\Sigma(s,\chi)[1,i] \tag{2}$$

Let now $\tau \in Traces_\Sigma(s)$ such that $\delta(\tau) \preceq \chi$. By taking into account relation 2, we deduce that for all i there exists k_i such that $\delta(\tau[1,i]) \preceq \delta(\Psi_{k_i})$. By applying the finite case of our theorem to these, we obtain for any i: $\delta(\tau[1,i]) \preceq \delta(NF_\Sigma(s,\delta(\Psi_{k_i})))$. Then, by taking into account relation 2 we deduce that $\forall i, \delta(\tau[1,i]) \preceq \delta(NF_\Sigma(s,\chi)[1,k_i]) \preceq \delta(NF_\Sigma(s,\chi))$, so that $\delta(\tau) \preceq \delta(NF_\Sigma(s,\chi))$. *Q.e.d.* □

Lemma 6 (Monotonicity and limit on histories) *Let $\Sigma = (U,S,\rightarrow,\hat{s})$ be a weakly endochronous LSTS, $s \in RSS(\Sigma)$, and $\chi,\chi',\chi_i \in Histories, i \geq 1$. Then:*

a. If $\chi' \preceq \chi$ then $\delta(NF_\Sigma(s,\chi')) \preceq \delta(NF_\Sigma(s,\chi))$

b. If $(\chi_k)_{k\geq 1}$ is monotonous such that $\bigsqcup_{k\geq 1}\chi_k = \chi$ then $\bigsqcup_{k\geq 1}\delta(NF_\Sigma(s,\chi_k)) = \delta(NF_\Sigma(s,\chi))$

Proof: Property 1: From $\chi' \preceq \chi$ we deduce that $\delta(NF_\Sigma(s,\chi')) \preceq \chi$, which implies (using theorem 3) that $\delta(NF_\Sigma(s,\chi')) \preceq \delta(NF_\Sigma(s,\chi))$.

Property 2: From property 1 we have:

$$\delta(NF_\Sigma(s,\chi_1)) \preceq \delta(NF_\Sigma(s,\chi_2)) \preceq \dots \preceq \delta(NF_\Sigma(s,\chi)) \tag{3}$$

Consider now $i \geq 1$. As $\bigsqcup_{i\geq 1}\chi_i = \chi$ and $\delta(NF_\Sigma(s,\chi)[1,i]) \preceq \chi$ is finite, there exists k_i such that $\delta(NF_\Sigma(s,\chi)[1,i]) \preceq \chi_{k_i}$. Then, $\delta(NF_\Sigma(s,\chi)[1,i]) \preceq \delta(NF_\Sigma(s,\chi_{k_i}))$ and thus, for all i: $\delta(NF_\Sigma(s,\chi)[1,i]) \preceq \bigsqcup_{k\geq 1}\delta(NF_\Sigma(s,\chi_k))$, which implies by taking the union over i: $\delta(NF_\Sigma(s,\chi)) \preceq \bigsqcup_{k\geq 1}\delta(NF_\Sigma(s,\chi_k))$. Along with relation 3 this implies the second statement of our lemma. □

Lemma 7 (Completion) *Let $\Sigma = (U, S, \rightarrow, \hat{s})$ be a weakly endochronous LSTS, $s \in RSS(\Sigma)$, $\tau \in Traces_\Sigma(s)$, finite, and $\chi \in Histories$ such that $\delta(\tau) \preceq \chi$. Then, if $\delta(\tau) \neq \delta(NF_\Sigma(s, \chi))$, there exists a reaction $r \in Reactions(U)$, $r \neq \bot_U$ such that $\tau r \in Traces_\Sigma(s)$ and $\delta(\tau r) \preceq \chi$.*

Proof: Case 1: τ is a normal form: In this case, $\tau[i] = NF_\Sigma(s, \delta(\tau))$ for all i. Since $\delta(\tau) \neq \delta(NF_\Sigma(s, \chi))$, we have $\tau \neq NF_\Sigma(s, \chi)$ and there exists an index i such that $\tau[i] \neq NF_\Sigma(s, \chi)[i]$. Let i_0 be the smallest such index. Since $\delta(\tau) \setminus \delta(\tau[1, i_0 - 1]) \preceq \delta(\chi) \setminus \delta(\tau[1, i_0 - 1])$, we will have $\tau[i_0] < NF_\Sigma(s, \chi)[i_0]$.

Let $s = s_0, s_1, \ldots$ be the sequence of states in the derivation of τ and let s_{i_0}' be the first different state in the derivation of $NF_\Sigma(s, \chi)$. Let $a \in Atoms_\Sigma(s_{i_0 - 1})$ be one of the atoms that form the reaction $NF_\Sigma(s, \chi)[i_0]$ and is not part of $\tau[i_0]$. Then, $supp(a) \cap supp(\tau[i_0]) = \emptyset$ and, according to lemma 3, it remains an atom in each of $s_{i_0}, s_{i_0+1}, \ldots$, but cannot be taken (doing so would mean that τ is not in normal form). Moreover, $supp(a) \cap supp(\tau[j]) = \emptyset$ for all $j \geq i_0$. Then, τa is a trace of Σ with the desired properties.

Case 2: τ is not a normal form: By applying the first case to $NF_\Sigma(s, \delta(\tau))$ and χ, we find r with the desired properties. Then, given that $\delta(\tau) = \delta(NF_\Sigma(s, \delta(\tau)))$ we can use the commutation corollary 1(1) to transform the trace $NF_\Sigma(s, \delta(\tau))r$ into τr. □

4 Application to GALS systems

Traditionally, the synchronous paradigm has been used in hardware, where the clock-driven execution model and the instantaneous communication abstraction are natural. Later, the synchronous approach has been introduced in the development of safety-critical embedded software, where the deterministic concurrency of the model results in better verification capabilities. Synchronous languages like Esterel [4], Lustre [10], or Signal [14] have been developed, along with compilation methods able to generate efficient *monolithic* implementations in both sequential software and digital circuits.

The implementation of a synchronous specification, however, may have to be *distributed* to some extent, due to architectural constraints. This is obviously true when the target is a distributed software system. Less obvious, this is increasingly the case in complex microprocessors and Systems-on-a-Chip (SoCs). As complexity and speed grow, controlling such a chip using a single global clock becomes increasingly difficult, so that future large-scale circuits will likely be composed of several timing domains (each domain being either asynchronous, or localy clocked).

Globally Asynchronous Locally Synchronous (GALS) architectures are emerging as the architecture of choice for implementing complex specifications in both hardware and software. In a GALS system, locally-clocked (synchronous) components are connected through asynchronous communication lines. Thus, unlike for a purely asynchronous design, the existing synchronous tools can be used for most of the development process.

Starting from synchronous specifications, our objective is to derive GALS implementations where the synchronous components are connected through bounded lossless FIFOs – communication lines that can be easily implemented in both hardware and software. We exemplify on the small system of fig. 6c. The

previously-defined modules `M1` and `M2` are connected here through FIFOs that transmit the signals `R1` and `K` as they are emitted by `M2`[2]. As the synchronous modules cannot be directly run in an asynchronous environment, small executives are used to read the inputs and schedule them into sequences of synchronous reactions (thus defining the local clock of each module).

The work presented in this paper aims at automatically synthesizing such executives that are both efficient (in terms of execution time and number of exchanged messages) and correct, where correctness is defined by the following two properties:

a. for any asynchronous execution trace (history) χ of the GALS system, there exists a synchronous trace τ of the global synchronous model such that $\delta(\tau) = \chi$ (recall that the desynchronization morphism δ retains on each channel the sequence of messages exchanged between the emitter and the receiver).

b. for any synchronous trace τ of the synchronous system, its desynchronization $\delta(\tau)$ is a history of the GALS system.

This semantics-preservation criterion ensures that (1) the verification of the synchronous model covers all the executions of the GALS system and that (2) the executives (the communication protocols) do not restrict the functionality with respect to the synchronous model.

Note that not any schedule is a good one. In our example (fig. 6c), the module M2 produces the outputs `R1` and `K` that are transmitted on separate channels. If `M1` is scheduled so that `K` is read before `R1` (*e.g.* (`K`)(`R1`)), then `M1` terminates execution before reading `R1`. Moreover, `R1` will never be read, the system deadlocks, and the execution history does not correspond to a synchronous trace.

4.1 Towards a synthesis strategy.

We propose in this paper a decomposition of the synthesis problem in 3 well-defined steps that result into systems satisfying our requirements. The steps correspond to 3 essential issues: (1) reconstruction of synchronous instants, (2) communication and, (3) causality.

In the **first step**, we consider the problem of defining partial executives that are able to *reconstruct, upto commutation of independent operations, a synchronous trace τ from its desynchronized version $\delta(\tau)$*. Moreover, we will require that *any* trace reconstructed from $\delta(\tau)$ is equivalent upto commutation of independent operations with τ.

Our solution to this problem is based on weak endochrony. In a weakly endochronous system, non-independent atomic operations (*i.e.* operations that do not fully commute because they are conflictual or causally ordered) share common channels whose values make the reconstruction decisions possible. Independent atoms share no common channel and need no synchronization, allowing the use of lighter executives.

Making a system weakly endochronous may involve the introduction of new interface signals that allow the reconstruction to be done. In `M1`, for instance, a supplementary signal carrying a Boolean value is needed in order to decide which of the non-independent actions "`await R1`" and "`await K`" must be executed.

[2]For brevity, we considered here data-less programs, so that the FIFOs carry only *present* values. In real examples, however, the signals may also carry more complex data like integers. The signal presence acts in these cases as a *data ready* indicator.

Note that the setting is non-causal, meaning that the signal can either be produced by the environment (telling the process what signal to expect) or by `M1` (telling the environment what signal is expected). Making a module weakly endochronous results in another synchronous module with richer interface and able to run in an asynchronous environment.

The **second step** synchronizes the transition systems of the different components to make sure that the weakly endochronous reconstruction of one component cannot break the synchronization barriers defined by another (again, upto commutation of independent operations). Connecting the resulting systems through FIFOs results into a GALS system that can deadlock (due to causality issues) but has no deadlock-free histories not corresponding to a synchronous trace. Pairs of synchronous components satisfying these constraints are called *weakly isochronous* (formal definition in section 4.4). The following two LSTSs are *not* weakly isochronous:

By connecting through FIFOs the corresponding programs we obtain a system that has no synchronous trace, but has one asynchronous history with no deadlock.

The **third step** takes into account the causality to synthesize the actual communication protocols and make sure that deadlocks cannot occur due, for instance, to emitted signals that cannot be interpreted by the receiver.

4.2 Semantics preservation criterion

Our contribution concerns the first two steps, which can be performed without taking causality into account (*e.g.* on LSTSs). In this framework, the operation representing the asynchronous composition through FIFOs of arbitrary length is the upper bound operator on histories $\sqcup$.

Furthermore, we do not use $\sqcup$ to compose arbitrary executions. More exactly, we overlook deadlocking pairs of traces where emitted signals may remain unread, as such executions are prohibited by the actual communication protocols in the 3^{rd} step. We only consider here deadlock-free executions that complete with empty FIFOs, and to identify such executions we introduce the notion of asynchronous composability.

Definition 2 (asynchronous composability) *The synchronous traces τ_i of $\Sigma_i = (U_i, S_i, \rightarrow_i, \hat{s}_i), i = 1, 2$ are asynchronously composable, denoted $\tau_1 \bowtie \tau_2$, if, by definition, $\delta(\tau_1)(v) = \delta(\tau_2)(v)$ for all $v \in U_1 \cap U_2$.*

With this notation, the formal semantics preservation criterion two systems (Σ_1, Σ_2) must satisfy after the first two synthesis steps (a variant of that defined in Benveniste *et al.*[1]) is:

Criterion 2 (Benveniste *et al.*, 2000) *For all $(s_1, s_2) \in RSS(\Sigma_1 \times \Sigma_2)$:*

$$\delta(Traces_{\Sigma_1 \times \Sigma_2}(s_1, s_2)) = \{\delta(\tau_1) \sqcup \delta(\tau_2) \mid \tau_i \in Traces_{\Sigma_i}(s_i) \wedge \tau_1 \bowtie \tau_2\}$$

Unfortunately, criterion 2 cannot serve as an effective semantics preservation criterion, as it is undecidable for LSTSs that are weakly endochronous and finite.

Theorem 4 (Undecidability) *Criterion 2 is undecidable, even on finite weakly endochronous LSTS with variables taking their values in finite domains.*

To prove this theorem, we first recall a classical theorem in language theory, which claims the undecidability of several problems on rational relations. The reader is refered to [9] and [5] (page 90) for the proof of this theorem.

Theorem 5 (Fischer-Rosenberg) *Let X, Y be alphabets with at least two letters. Given two rational subsets $L_1, L_2 \subseteq X^\star \times Y^\star$, it is undecidable to determine the validity of any of the following statements (i) $L_1 \cap L_2 = \emptyset$, (ii) $L_1 \subseteq L_2$, (iii) $L_1 = L_2$, (iv) $L_1 = X^\star \times Y^\star$, (v) $X^\star \times Y^\star \setminus L_1$ is finite, and (vi) L_1 is recognizable.*

Proof:(of theorem 4) The proof relies on the undecidability of equality (i) in Theorem 5: Let X, Y be alphabets with at least two letters, and two rational subsets $L_1, L_2 \subseteq X^\star \times Y^\star$.

For $i = 1, 2$, language L_i is generated by some finite deterministic automaton $A_i = (S_i, \hat{s}_i \in S_i, F_i \subseteq S_i, T_i \subseteq S_i \times (X \uplus \{\varepsilon\}) \times (Y \uplus \{\varepsilon\}) \times S_i)$, where S_i is a finite set of states, $\hat{s}_i$ is an initial state, F_i is a set of final states, and T_i is a transition relation, labelled by pairs of words of length atmost 1.

Two LSTS $B_i = (C_i = S_i \uplus \{\$\}, \hat{s}_i, V = \{w, x, y, z_1, z_2\}, \Delta_i)$, $i = 1, 2$ are derived from A_1 and A_2. LSTS B_1 has states in $C_1 = S_1 \uplus \{\$\}$. Initial state is state $\hat{s}_1$. The transition relation Δ_1 is the least synchronous transition relation such that:

$$
\begin{array}{l}
\forall s, s' \in S_1, u \in X, v \in Y, \\
\quad \text{If} \quad (s, u, v, s') \in T_1 \text{ then} \\
\qquad \forall r, \text{ if } r(w) = \top, r(x) = u, r(y) = v, r(z_1) = 0, r(z_2) \neq \bot, \text{ then } (s, r, s') \in \Delta_1 \\
\quad \text{If} \quad (s, \varepsilon, v, s') \in T_1 \text{ then} \\
\qquad \forall r, \text{ if } r(w) = \top, r(x) = \bot, r(y) = v, r(z_1) = 1, r(z_2) \neq \bot, \text{ then } (s, r, s') \in \Delta_1 \\
\quad \text{If} \quad (s, u, \varepsilon, s') \in T_1 \text{ then} \\
\qquad \forall r, \text{ if } r(w) = \top, r(x) = u, r(y) = \bot, r(z_1) = 2, r(z_2) \neq \bot, \text{ then } (s, r, s') \in \Delta_1 \\
\quad \text{If} \quad (s, \varepsilon, \varepsilon, s') \in T_1 \text{ then} \\
\qquad \forall r, \text{ if } r(w) = \top, r(x) = \bot, r(y) = \bot, r(z_1) = 3, r(z_2) \neq \bot, \text{ then } (s, r, s') \in \Delta_1 \\
\quad \text{If} \quad s \in F_1 \text{ then} \\
\qquad \forall r, \text{ if } r(w) = \top, r(x) = r(y) = \$, r(z_1) = 0, r(z_2) \neq \bot, \text{ then } (s, r, \$) \in \Delta_1 \\
\quad \forall r, \text{ if } r(w) = r(x) = r(y) = \bot, r(z_1) = 3, r(z_2) \neq \bot, \text{ then, } (\$, r, \$) \in \Delta_1
\end{array}
$$

The transition relation of LSTS B_2 is the least transition relation such that:

$\forall s, s' \in S_2, u \in X, v \in Y,$

If $(s, u, v, s') \in T_2$ then

$\forall r$, if $r(w) = \bot, r(x) = u, r(y) = v, r(z_2) = 0, r(z_1) \neq \bot$, then $(s, r, s') \in \Delta_2$

If $(s, \varepsilon, v, s') \in T_2$ then

$\forall r$, if $r(w) = \bot, r(x) = \bot, r(y) = v, r(z_2) = 1, r(z_1) \neq \bot$, then $(s, r, s') \in \Delta_2$

If $(s, u, \varepsilon, s') \in T_2$ then

$\forall r$, if $r(w) = \bot, r(x) = u, r(y) = \bot, r(z_2) = 2, r(z_1) \neq \bot$, then $(s, r, s') \in \Delta_2$

If $(s, \varepsilon, \varepsilon, s') \in T_2$ then

$\forall r$, if $r(w) = \bot, r(x) = \bot, r(y) = \bot, r(z_2) = 3, r(z_1) \neq \bot$, then $(s, r, s') \in \Delta_2$

If $s \in F_2$ then

$\forall r$, if $r(w) = \bot, r(x) = r(y) = \$, r(z_2) = 0, r(z_1) \neq \bot$, then $(s, r, \$) \in \Delta_2$

$\forall r$, if $r(w) = \top, r(x) = r(y) = \bot, r(z_2) = 3, r(z_1) \neq \bot$, then, $(\$, r, \$) \in \Delta_2$

These two LSTS have been defined so that they both satisfy the axioms of weak-endochrony: Axiom **W1** holds since automata A_1 and A_2 are deterministic, and the tranformation defined above preserves determinism. Axiom **W2** holds because all transitions of $B_i, i = 1, 2$ are labelled by a reaction in which variable z_i is present. Axiom **W3** holds because, in LSTS B_1 and B_2, any two distinct reactions enabled in a given state are contradictory.

The behaviour of the asynchronous composition of LSTS B_1 and B_2 encodes the intersection of languages L_1 and L_2: $\{\delta(\tau_1) \sqcup \delta(\tau_2) \,|\, \forall i = 1, 2,\ \tau_i \in Traces_{B_i}(\hat{s}_i) \wedge \tau_1 \bowtie \tau_2\}_{|x,y} = \{(\varepsilon, \varepsilon)\} \cup (L_1 \cap L_2) \cdot \{(\$, \$)\}$. On the contrary, the synchronous composition of the two LSTS leads to a deadlock, right from the initial state: $\delta(Traces_{B_1 \times B_2}(\hat{s}_1, \hat{s}_2))_{|x,y} = \{(\varepsilon, \varepsilon)\}$. Therefore, Criterion 2 is satisfied if and only if L_1 and L_2 have empty intersection, which is undecidable, from equality (i) of Theorem 5. □

Taking into account this undecidability result, our goal is here to find sufficient conditions of Criterion 2 able to handle meaningful classes of LSTSs, while being decidable.

4.3 Composition of weakly endochronous systems

The first step in this direction is the remark that criterion 2 can be largely simplified when we apply it to systems whose components are weakly endochronous.

Criterion 3 *For all* $(s_1, s_2) \in RSS(\Sigma_1 \times \Sigma_2)$ *and for all* $\tau_i \in Traces_{\Sigma_i}(s_i), i = 1, 2$ *such that* $\tau_1 \bowtie \tau_2$ *and* $\delta(\tau_1) \sqcup \delta(\tau_2) \neq \epsilon$*, the following holds:*

$$\exists r_i \in Reactions_{\Sigma}(s_i), i = 1, 2 : \begin{cases} r_1 \vee r_2 \neq \bot_{U_1 \cup U_2} \\ \delta(r_i) \preceq \delta(\tau_i), i = 1, 2 \\ r_1 \sqcup r_2 \text{exists} \end{cases}$$

The equivalence between the two criteria is stated by the following result:

Theorem 6 (equivalence) *Criteria 2 and 3 are equivalent for weakly endochronous LSTSs.*

The result is important, as it replaces the synchronization of full traces (impossible to compute for infinite traces) with the existence of a pair of synchronizable initial transitions.

Proof: Criterion 1 $\Rightarrow$ Criterion 2.

Consider s_1, s_2, τ_1, τ_2 satisfying the hypothesis of criterion 3. Then $\delta(\tau_1) \sqcup \delta(\tau_2) \in \{\delta(\tau_1) \sqcup \delta(\tau_2) \mid \tau_i \in Traces_{\Sigma_i}(s_i) \wedge \tau_1 \bowtie \tau_2\}$, and by applying criterion 2 we obtain $\delta(\tau_1) \sqcup \delta(\tau_2) \in \delta(Traces_{\Sigma_1 \times \Sigma_2}(s_1, s_2))$. This means that there exists $\tau \in Traces_{\Sigma_1 \times \Sigma_2}(s_1, s_2)$ such that $\delta(\tau) = \delta(\tau_1) \sqcup \delta(\tau_2)$. From the definition of the synchronous composition we obtain $\tau_i' = \tau_{|U_i} \in Traces_{\Sigma_i}(s_i), i = 1, 2$. Moreover, $\tau_1 \bowtie \tau_2$ implies that $\delta(\tau_i') = \delta(\tau_{|U_i}) = \delta(\tau_i), i = 1, 2$. As $\delta(\tau_1') \sqcup \delta(\tau_2') = \delta(\tau_1) \sqcup \delta(\tau_2) \neq \epsilon$, there exists $j \geq 1$ such that $\tau_1'[j] \sqcup \tau_2'[j] \neq \perp_{U_1 \cup U_2}$. Let j_0 be the smallest such j. Then, since the stuttering transitions do not change the state, we have $\tau_i'[j_0 \ldots] \in Traces_{\Sigma_i}(s_i), i = 1, 2$, and $r_i =_{def} \tau_i'[j_0], i = 1, 2$ satisfy the conclusion of criterion 3.

Criterion 2 $\Rightarrow$ Criterion 1.

Consider s_1, s_2 satisfying the hypothesis of criterion 2 (*i.e.* $(s_1, s_2) \in RSS(\Sigma_1 \times \Sigma_2)$). As any trace of $Traces_{\Sigma_1 \times \Sigma_2}(s_1, s_2)$ determines synchronously composable traces in Σ_1 and Σ_2, the inclusion $\delta(Traces_{\Sigma_1 \times \Sigma_2}(s_1, s_2)) \subseteq \{\delta(\tau_1) \sqcup \delta(\tau_2) \mid \tau_i \in Traces_{\Sigma_i}(s_i) \wedge \tau_1 \bowtie \tau_2\}$ is obvious. We still have to prove the reverse inclusion.

Let then $\tau_i \in Traces_{\Sigma_i}(s_i), i = 1, 2$ such that $\tau_1 \bowtie \tau_2$. To prove the reverse inclusion we have to prove that there exist $\tau_i' \in Traces_{\Sigma_i}(s_i), i = 1, 2$, synchronously composable and such that $\delta(\tau_i') = \delta(\tau_i), i = 1, 2$. Furthermore, if we denote $\chi = \delta(\tau_1) \sqcup \delta(\tau_2) = \delta(\tau_1' \sqcup \tau_2')$, then proving that $\tau_i', i = 1, 2$ exist is equivalent to proving that:

$$\delta(NF_{\Sigma_1 \times \Sigma_2}((s_1, s_2), \chi)) = \chi \tag{4}$$

This is a direct consequence of theorem 3, which based on $\tau_1' \sqcup \tau_2' \in Traces_{\Sigma_1 \times \Sigma_2}(s_1, s_2)$ and $\delta(\tau_1' \sqcup \tau_2') \preceq \chi$, tells us that $\delta(\tau_1' \sqcup \tau_1') \preceq \delta(NF_{\Sigma_1 \times \Sigma_2}((s_1, s_2), \chi)) \preceq \chi$. If $\tau_i', i = 1, 2$ exist such that $\delta(\tau_1' \sqcup \tau_1') = \chi$, then equation 4 is satisfied. Conversely, if equation 4 is satisfied, then the projections $NF_{\Sigma_1 \times \Sigma_2}((s_1, s_2), \chi)_{|U_i}$, with $i = 1, 2$, respectively satisfy the conditions required on $\tau_i', i = 1, 2$.

Before proving equation 4, we introduce some more notations: Let $(s_1, s_2) = (s_1^0, s_2^0), (s_1^1, s_2^1), (s_1^2, s_2^2), \ldots$ be the sequence of states in the derivation of $NF_{\Sigma_1 \times \Sigma_2}((s_1, s_2), \chi)$. Also let $\theta_i, i = 1, 2$ be the projections of $NF_{\Sigma_1 \times \Sigma_2}((s_1, s_2), \chi)$ respectively onto $\Sigma_i, i = 1, 2$: $\theta_i = NF_{\Sigma_1 \times \Sigma_2}((s_1, s_2), \chi)_{|U_i}$. From the definition of the synchronized product, we have $\theta_i \in Traces_{\Sigma_i}(s_i), i = 1, 2$. Also, let χ' be the remainder of χ after consuming the normal form: $\chi' = \chi \setminus \delta(NF_{\Sigma_1 \times \Sigma_2}((s_1, s_2), \chi))$. Note that for all $v \in \mathcal{V}$, $\chi'(v) \neq \epsilon$ implies that $\delta(NF_{\Sigma_1 \times \Sigma_2}((s_1, s_2), \chi))(v)$ is finite [3], so that there exists $i_v \geq 1$ such that $\forall i \geq i_v : NF_{\Sigma_1 \times \Sigma_2}((s_1, s_2), \chi)[i](v) = \perp$. Since $\mathcal{V}$ is finite, we can define $i_{s_1,s_2}(\tau_1, \tau_2) = \max_{v \in \mathcal{V}} i_v$, which has the property:

$$\chi'(v) \neq \epsilon \Rightarrow \forall i \geq i_{s_1,s_2}(\tau_1, \tau_2) : NF_{\Sigma_1 \times \Sigma_2}((s_1, s_2), \chi)[i](v) = \perp \tag{5}$$

[3] due to a basic property on words: $w_1 w_2 \in A^\omega \wedge w_2 \neq \epsilon \Rightarrow w_1 \in A^*$

Note that χ, χ', and the θ_i also depend on $\tau_i, i = 1, 2$ and on $s_i, i = 1, 2$. Unlike for $i_{s_1,s_2}(\tau_1, \tau_2)$, however, we did not make this clear at the notation level. As we shall see, this simplification of the notation allows us to shorten the proof while not affecting its clarity.

With these notations, to prove equation 4 (and our theorem) we still have to prove that $\chi' = \epsilon$. We are doing this by using induction over $k \geq 1$ to prove the proposition:

P[k] The previous construction leads to a void χ' for all $(s_1, s_2) \in RSS(\Sigma_1 \times \Sigma_2)$ and all $\tau_i \in Traces_{\Sigma_i}(s_i)$ with $\tau_1 \bowtie \tau_2$ and $i_{s_1,s_2}(\tau_1, \tau_2) = k$

Initial case: P[1] ($i_{s_1,s_2}(\tau_1, \tau_2) = 1$).

In this case, equation 5 implies that $\chi'(v) \neq \epsilon \Rightarrow \delta(NF_{\Sigma_1\times\Sigma_2}((s_1, s_2), \chi))(v) = \epsilon$, so that $supp(\chi') \cap supp(\delta(NF_{\Sigma_1\times\Sigma_2}((s_1, s_2), \chi))) = \emptyset$, which, in turn, implies $supp(\chi') \cap supp(\delta(\theta_i)) = \emptyset, i = 1, 2$. Then, since for all $v \in U_i$ we have $\chi'(v) = \chi(v) \setminus \delta(NF_{\Sigma_1\times\Sigma_2}((s_1, s_2), \chi))(v) = \chi(v) \setminus \delta(\theta_i)(v)$, we obtain $\delta(\theta_i)(v) = \delta(\tau_i)(v)$, if $v \in supp(\theta_i)$, and $\delta(\theta_i)(v) = \epsilon$, otherwise. We can then apply the *disjoint support* corollary 3 to the traces θ_i and τ_i to obtain $\tau_i' \in Traces_{\Sigma_i}(s_i)$ with $\delta(\tau_i') = \delta(\tau_i) \setminus \delta(\theta_i), i = 1, 2$.

Recall that $\tau_1 \bowtie \tau_2$, so that $\forall v \in U_1 \cap U_2 : \delta(\tau_1)(v) = \delta(\tau_2)(v)$, which in turn implies that $\forall v \in U_i, \chi(v) = (\delta(\tau_1) \sqcup \delta(\tau_2))(v) = \delta(\tau_i)(v)$. Then $\forall v \in U_i, \chi'(v) = \chi(v) \setminus \delta(NF_{\Sigma_1\times\Sigma_2}((s_1, s_2), \chi))(v) = \delta(\tau_i)(v) \setminus \delta(\theta_i)(v) = \delta(\tau_i')(v)$, and by taking the union: $\chi' = \delta(\tau_1') \sqcup \delta(\tau_2')$, so that proving the initial case is equivalent to proving that $\delta(\tau_1') \sqcup \delta(\tau_2') = \epsilon$.

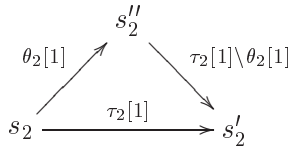
Assume to the contrary that $\delta(\tau_1') \sqcup \delta(\tau_2') \neq \epsilon$. Then, s_1, s_2, τ_1' and τ_2' satisfy the hypothesis of the semantics preservation criterion 3, so that there exist $r_i \in Reactions_{\Sigma_i}(s_i), i = 1, 2$ such that (i) $r_1 \sqcup r_2$ exists, (ii) $r_1 \sqcup r_2 \neq \perp_{U_1\cup U_2}$, and (iii) $\delta(r_i) \preceq \delta(\tau_i'), i = 1, 2$. Since $\forall i : supp(r_i) \subseteq supp(\chi')$, we have $\forall i, j : supp(r_i) \cap supp(\theta_j) = \epsilon$, and we can apply the commutation corollary 1 to deduce that $r_i\theta_i \in Traces_{\Sigma_i}(s_i), i = 1, 2$. The traces $r_1\theta_1$ and $r_2\theta_2$ being synchronously composable, we can define $\tau = (r_1\theta_1) \sqcup (r_2\theta_2) = (r_1 \sqcup r_2)(\theta_1 \sqcup \theta_2) \in Traces_{\Sigma_1\times\Sigma_2}(s_1, s_2)$. Then, by remembering once more that $\forall i, j : supp(r_i) \cap supp(\theta_j) = \epsilon$, we have: $\delta(\tau) = \delta(r_1 \sqcup r_2) \sqcup \delta(\theta_1 \sqcup \theta_2) \succ \delta(\theta_1 \sqcup \theta_2) = \delta(NF_{\Sigma_1\times\Sigma_2}((s_1, s_2), \chi))$, which is impossible (according to the normal form theorem 3, since by construction $\delta(\tau) \preceq \chi$). We have a contradition with the assumption that $\delta(\tau_1') \sqcup \delta(\tau_2') \neq \epsilon$. P[1] is proved.

Induction step: P[k]⇒P[k+1].

First note that a direct consequence of the normal form theorem is $\delta(\tau_i) = \delta(NF_{\Sigma_i}(s_i, \delta(\tau_i))), i = 1, 2$. We can therefore assume, without losing generality, that $\tau_i, i = 1, 2$ are in normal form. If they are not, we simply replace them, respectively, with $NF_{\Sigma_i}(s_i, \delta(\tau_i)), i = 1, 2$, as part of the construction process. The transformation does not affect the construction of χ, χ', and $\theta_i, i = 1, 2$, so that it does not change the value of $i_{s_1,s_2}(\tau_1, \tau_2)$.

By construction, we have $\theta_1[1] \in Reactions_{\Sigma_1}(s_1)$ and $\delta(\theta_1[1]) \preceq \delta(\tau_1)$. The trace τ_1 being in normal form, we have $\theta_1[1] \leq \tau_1[1]$, and we can apply axiom **W3** to divide the reaction of τ_1 in two reactions with the same global effect. More exactly, if s_1' is defined by $s_1 \xrightarrow{\tau_1[1]} s_1'$, then there exists s_1'' such that $s_1 \xrightarrow{\theta_1[1]} s_1'' \xrightarrow{\tau_1[1]\setminus\theta_1[1]} s_1'$, which in turn implies that $\tau_1' = \theta_1[1](\tau_1[1] \setminus \theta_1[1])\tau_1[2\ldots] \in Traces_{\Sigma_1}(s_1)$.

Similarly, we define s_2'' and s_2' through:



and we have $\tau_2' = \theta_2[1](\tau_2[1] \setminus \theta_2[1])\tau_2[2\ldots] \in Traces_{\Sigma_2}(s_2)$. As $\theta_1[1]$ and $\theta_2[1]$ are synchronously composable, we have $(s_1'', s_2'') \in RSS(\Sigma_1 \times \Sigma_2)$ and $\tau_i'[2\ldots] \in Traces_{\Sigma_i}(s_i''), i = 1, 2$, such that $\tau_1'[2\ldots] \bowtie \tau_2'[2\ldots]$.

Let $\chi_k = \delta(\tau_1'[2\ldots]) \sqcup \delta(\tau_2'[2\ldots])$. To prove our induction step, we shall prove next that:

$$i_{s_1'', s_2''}(\tau_1'[2\ldots], \tau_2'[2\ldots]) = k \tag{6}$$

This relation tells us that we can use P[k] to deduce $\delta(NF_{\Sigma_1\times\Sigma_2}((s_1'', s_2''), \chi_k)) = \chi_k$, which implies $\delta((\theta_1 \sqcup \theta_2)NF_{\Sigma_1\times\Sigma_2}((s_1'', s_2''), \chi_k)) = \chi$, and since, by construction, $\delta((\theta_1 \sqcup \theta_2)NF_{\Sigma_1\times\Sigma_2}((s_1'', s_2''), \chi_k)) \preceq \delta(NF_{\Sigma_1\times\Sigma_2}((s_1, s_2), \chi))$, we finally obtain $\delta(NF_{\Sigma_1\times\Sigma_2}((s_1, s_2), \chi)) = \chi$. P[k+1] is proved.

To prove relation 6, note that $\chi_k = \chi \setminus \delta(\theta_1 \sqcup \theta_2) = \chi \setminus \delta(NF_{\Sigma_1\times\Sigma_2}((s_1, s_2), \chi)[1])$, which implies, given the construction of the normal form, $NF_{\Sigma_1\times\Sigma_2}((s_1'', s_2''), \chi_k) = NF_{\Sigma_1\times\Sigma_2}((s_1, s_2), \chi)[2\ldots]$. From the definition of $i_{s_1'', s_2''}(\tau_1'[2\ldots], \tau_2'[2\ldots])$, we have: $i_{s_1'', s_2''}(\tau_1'[2\ldots], \tau_2'[2\ldots]) = i_{s_1, s_2}(\tau_1, \tau_2) - 1 = k$. □

4.4 Weak isochrony

The undecidability of criterion 2 on weakly endochronus systems and the equivalence between criteria 2 and 3 implies the undecidability of criterion 3. Intuitively, this is due to the quantification over asynchronously composable traces. To derive a decidable criterion, we over-approximate by quantifying over traces whose *asynchronous prefixes* of length 1 are synchronously composable. Formally, we start by denoting for all $\Sigma = (U, S, \rightarrow, \hat{s})$, and $s \in RSS(s)$ the set of asynchronous prefixes of depth 1: $head_\Sigma(s) = \{head_U(\delta(\tau)) \mid \tau \in Traces_\Sigma(s)\}$. Note that the elements of $head_\Sigma(s)$ are not necessarily reactions of Σ, but that $head_\Sigma(s)$ can be computed for any finite LSTS Σ, for instance through a depth-first search. With this definition, the semantics preservation property that will imply correctness criterion 3 shall be:

Criterion 4 (weak isochrony) *Two LSTSs Σ_1 and Σ_2 are weakly isochronous if, by definition, for all $(s_1, s_2) \in RSS(\Sigma_1 \times \Sigma_2)$ and for all $r_i \in head_{\Sigma_i}(s_i), i = 1, 2$ such that $r_1 \sqcup r_2$ exists we have:*

$$\exists \overline{r}_i \in Reactions(s_i), i = 1, 2 : \begin{cases} \overline{r}_i \leq r_i, i = 1, 2 \\ \overline{r}_1 \sqcup \overline{r}_2 \text{ exists} \\ \overline{r}_1 \vee \overline{r}_2 \neq \bot_{U_1 \cup U_2} \end{cases}$$

It is obvious that weak isochrony implies criterion 3, and therefore the main result of our paper:

Theorem 7 (correct desynchronization) *Let Σ_1 and Σ_2 be weakly endochronous LSTSs such that (Σ_1, Σ_2) is weakly isochronous. Then, Σ_1 and Σ_2 satisfy Benveniste's correct desynchronization criterion 2.*

Note that weak isochrony can only be used in conjunction with weak endochrony to form a correctness criterion. In the following example the LSTSs are weakly isochronous, but not weakly endochronous, and the semantics preservation criterion is not satisfied:

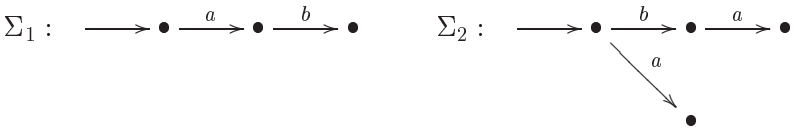
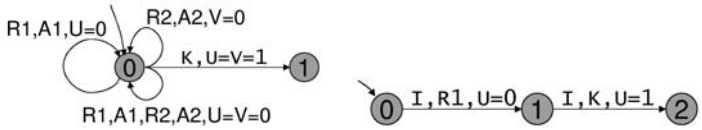


Fig. 7 shows a possible solution to the simple distribution problem defined in fig. 6. As earlier explained, making the system `M1` weakly endochronous requires the use of supplementary variables needed in the scheduling decisions (in our case, the variables are `U` and `V`). System `M2` is by definition weakly endochronous, but we need to enrich its interface to make the pair $(\texttt{M1}, \texttt{M2})$ weakly isochronous.



(a) (b)

Figure 7: Versions of the LSTSs M1 (a) and M2 (b) that are weakly endochronous and weakly isochronous

5 Comparison with existing work

5.1 Weak endochrony and trace theory

As the results of section 3 show, weak endochrony is indeed an extension to a synchronous setting of the classical trace theory. More exactly, if we consider the formalization of [8], chapter 11, the dependence alphabet is interpreted in our case over synchronous reactions, while the dependence relation is given by a syntactic relation over such reactions (the non-overlapping of supports). The traces of a weakly endochronous system correspond to real dependency graphs having atoms as vertices, and the trace language of a system is closed under commutation of independent atoms.

Easily defined, the extension is nevertheless non-trivial. The new *synchrony* relation leads, when joined to the classical *independence* and *causality*, to stronger results, but also complexifies proofs. This is particularly obvious in the case of the normal form result: The formulation is in our case stronger, in the sense where the normal form of a trace is indeed another trace of the weakly endochronous system, and not a decomposition of the initial trace into a sequence of cliques of independent letters[4]. Moreover, weak endochrony also covers state-related determinism aspects through its strong confluence properties (theorem 3). On the other side, the proof of the normal form theorem has been challenging, as the classical results of [8], which do not cover the synchrony relation, cannot be used.

[4]the relation between the two normal forms is straightforward: the decomposition as a sequence of cliques (the trace theory normal form) is given by the sequence of reactions of the weakly endochronous normal form, each reaction being transformed into the clique of atoms that generate it.

Going beyond classical trace properties, our normal form operator is also continuous with respect to the c.p.o. (complete partial order) structures on the sets of traces and histories (cf. lemma 6). We can therefore consider that networks of weakly endochronous components satisfy a relational form of the Kahn principle [15, 16] where the determinism and continuity of the normal form operator on components implies the determinism (upto commutation of independent operations) of the system evolution for a given history.

5.2 Desynchronization results

As mentioned in section 1.2, the **latency-insensitive paradigm** [7] features no concurrency, nor execution modes. The communication protocols of a latency-insensitive system simulate a single-clocked system where all signals are transmitted during each reaction. This is very inefficient, but has the advantage of easily supporting causality and simple synthesis algorithms. By comparison, our approach takes into account execution modes and independence between computations, and allows multi-rate computation. This means that it potentially supports lighter communication protocols that minimize communication and power consumption.

The **endo/isochronous systems** of Benveniste *et al.* [1, 2] take into account execution modes and independence between system components in order to minimize communication and allow multi-rate computation. Meanwhile, there is no independence between computations at component level. Our work improves over the endo/isochronous approach by allowing operations within a component to run independently when no synchronization is necessary. For instance, the two systems in fig. 8 are both weakly endochronous and weakly isochronous, meaning that no further synchronization is necessary. However, making Σ_2 endochronous, and then isochronous with Σ_1 would require the removal of the non-determinism either by removing transitions of Σ_2 or by introducing supplementary signalization channels.

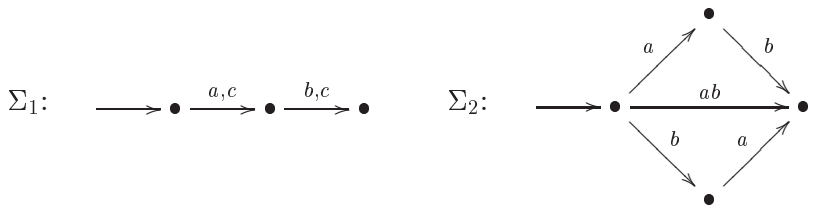


Figure 8: The synchronous systems Σ_1 and Σ_2 are both weakly endochronous and weakly isochronous, but Σ_2 is not endochronous because the independent operations a and b can be executed in any order

Weak endochrony generalizes endochrony, which in turn generalizes the notion of latency insensitivity. On the other hand, neither endochrony, nor weak endochrony take into account causality in the computation of reactions, and efficient synthesis algorithms have yet to be defined for both of them. Last, but not least, weak endochrony is compositional, while endochrony is not [6][5].

In a **Kahn process network**, the input/output determinism of each component implies the determinism of the global system, and thus the independence from the scheduling scheme is guaranteed. Giving the

[5] In consequence, synthesizing endo-isochronous communication protocols for systems composed of more than 2 synchronous components involves "endochronization" steps resulting in heavy synchronization

approach its strength, the determinism is also its main drawback, as non-determinism is often useful in the specification and analysis of concurrent systems. By comparison, weakly endochronous systems also guarantee the deterministic re-synchronization upto commutation, but in a non-causal setting. Thus, oracles can be used (in the more concrete causal model) as soon as the environment is informed about the non-deterministic internal decisions. As mentioned in the previous section, weak endochrony can be seen as a generalization of the Kahn processes to a relational (non-causal) setting, giving the actual implementations a supplementary degree of freedom that can be exploited by more flexible, lighter protocols.

6 Conclusion

In this paper we introduced the notion of weak endochrony and we explained how it can be used to advance the state of the art in correct-by-construction implementation of synchronous specifications over GALS architectures. Weak endochrony generalizes over previous work on endochronous and latency-insensitive systems by introducing a notion of independence (derived from the classical trace theory) between operations within a synchronous component. This potentially allows the use of lighter, more efficient synchronization schemes in the synthesis of GALS implementations. The properties of weak endochrony and weak isochrony form a criterion guaranteeing the correct distribution of synchronous specitications in the sense of Benveniste *et al.* [2]. Moreover, the criterion can be decided on general finite LSTSs.

The current paper only represents a first step in our quest for *effective* methods for the implementation of synchronous specifications over GALS architectures. Our future research directions will include (1) the derivation of efficient synthesis algorithms ensuring the properties of weak endochrony and weak isochrony and (2) the extension of the model to take into account causality (and thus support the synthesis of actual GALS implementations).

References

[1] A. Benveniste, B. Caillaud, and P. Le Guernic. Compositionality in dataflow synchronous languages: Specification and distributed code generation. *Information and Computation*, (163):125–171, 2000.

[2] A. Benveniste, L. Carloni, P. Caspi, and A. Sangiovanni-Vincentelli. Heterogenous reactive systems modeling and correct-by-construction deployment. In *Proceedings of EMSOFT'03*, Philadelphia, Pennsylvania, USA, 2003.

[3] A. Benveniste, P. Caspi, S. Edwards, N. Halbwachs, P. Le Guernic, and R. de Simone. The Synchronous Languages Twelve Years Later. *Proceedings of the IEEE*, 2003.

[4] G. Berry and G. Gonthier. The Esterel synchronous programming language: Design, semantics, implementation. *Science of Computer Programming*, 19(2):87–152, 1992.

[5] J. Berstel. *Transductions and Context-Free Languages*. Teubner Studienbücher, Stuttgart, Germany, 1979.

[6] B. Caillaud, D. Potop-Butucaru, and A. Benveniste. Erratum to: A. Benveniste, B. Caillaud, P. Le Guernic. Compositionality in Dataflow Synchronous Languages, Specification and Distributed Code Generation. Information and Computation 163, 125-171 (2000), September 2003. http://www.irisa.fr/prive/Benoit.Caillaud/erratum-ic-2003.pdf.

[7] C. Carloni, K. McMillan, and A. Sangiovanni-Vincentelli. The theory of latency-insensitive design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 20(9):1059–1076, 9 2001.

[8] V. Diekert and G. Rozenberg, editors. *The Book of Traces*. World Scientific, 1995.

[9] P. C. Fischer and A. L. Rosenberg. Multitape one-way nonwriting automata. *Journal of Computer and System Sciences (JCSS)*, 2:88–101, 1968.

[10] N. Halbwachs, P. Caspi, P. Raymond, and D. Pilaud. The synchronous dataflow programming language Lustre. In *Proceedings of the IEEE*, volume 79, pages 1305–1320, 1991.

[11] Nicolas Halbwachs. *Synchronous programming of reactive systems*. Kluwer Academic Publishers, 1993.

[12] D. Harel. Statecharts: A visual formulation for complex systems. *Science of Computer Programming*, 8(3):231–274, 1987.

[13] H. Kopetz. *Real-Time Systems, Design Principles for Distributed Embedded Applications*. Kluwer Academic Publishers, 1997.

[14] P. Le Guernic, T. Gauthier, M. Le Borgne, and C. Le Maire. Programming real-time applications with Signal. In *Proceedings of the IEEE*, volume 79, pages 1321–1336, 1991.

[15] N. Lynch and E. Stark. A proof of the Kahn principle for input/output automata. *Information and Computation*, 82(1):81–92, 1989.

[16] D. Potop-Butucaru. The Kahn principle for networks of endochronous programs. In *Proceedings FMGALS2003*, Pisa, Italy, 2003.