Project 22703: SYRF

Deliverable 4/5.1

Integrating synchrony & asynchrony: fundamentals

Abstract

Here we provide the fundamentals to understand what synchronous program desynchronisation means, and how it can be performed.

Keywords : synchronous languages, desynchronisation.

1 The essentials of the synchronous paradigm

There has been several attempts to characterize the essentials of the synchronous paradigm. With some experience and after attempts to address the issue of moving from synchrony to asynchrony (and back), we feel the following features are indeed essential for characterizing this paradigm :

1. Programs progress via an infinite sequence of *reactions* :

   $$P = R^{\omega}$$

   where R denotes the family of possible reactions[1].

2. Within a reaction, decisions can be taken on the basis of the *absence* of some events, as exemplified by the following typical statements, taken from Esterel, Lustre, and Signal respectively :

   ```
   present S else ‘stat’
   y = current x
   y := u default v
   ```

   The first statement is selfexplanatory. The "`current`" operator delivers the most recent value of `x` at the clock of the considered node, it thus has to test for absence of `x` before producing `y`. The "`default`" operator delivers its first argument when it is present, and otherwise its second argument.

3. When it is defined, parallel composition is always given by taking the conjunction of associated reactions :

   $$P_1 \| P_2 = (R_1 \wedge R_2)^{\omega}$$

[1] In fact, "reaction" is a slightly restrictive term, as we shall see in the sequel that "reacting to the environment" is not the only possible kind of interaction a synchronous system may have with its environment.

> Typically, if specifying is the intention, then the above formula is a perfect definition of parallel composition. In contrast, if programming was the intention, then the need for this definition to be compatible with an operational semantics very much complicates the "when it is defined" prerequisite[2].

Of course, such a characterization of what the synchronous paradigm is makes the class of "synchrony–compliant" formalisms much larger than usually considered. But it has been our experience that these were the key features for the techniques we have developed so far to work.

In the next section we introduce the model of Synchronous Transition Systems as a simplest possible model to capture the essence of the synchrony paradigm.

2 Synchronous Transition Systems (STS)

Synchronous Transition Systems (STS). We assume a vocabulary $\mathcal{V}$ which is a set of typed variables. All types are implicitly extended with a special element $\perp$ to be interpreted as "absent". Some of the types we consider are the type of *pure signals* with domain $\{\mathrm{T}\}$, and *booleans* with domain $\{\mathrm{T}, \mathrm{F}\}$ (recall both types are extended with the distinguished element $\perp$).

We define a *state* s to be a type-consistent interpretation of $\mathcal{V}$, assigning to the family of all variables, a value for it over its domain. We denote by S the set of all states. For a subset of variables $V \subseteq \mathcal{V}$, we define a V-state to be a type-consistent interpretation of V. Thus a V-state assigns to the family V a value $s[V]$ for it over its domain; also, for $v \in V$ a variable, we denote by $s[v]$ its interpretation by state s.

Define a *Synchronous Transition System* (STS) to be a system

$$\Phi = \langle V, \Theta, \rho \rangle$$

consisting of the following components:

- V is a finite set of typed *variables,*
- $\Theta(V)$ is an assertion characterizing *initial states.*
- $\rho = \rho(V^-, V)$ is the *transition relation* relating past and current states s^- and s, by referring to both past[3] and current versions of variables V^- and V. For example the assertion $x = x^- + 1$ states that the value of x in s is greater by 1 than its value in s^-. If $\rho(s^-[V], s[V]) = \mathrm{T}$, we say that state s^- is a *ρ-predecessor* of state s.

A *run* $\sigma : s_0, s_1, s_2, \ldots$ is a sequence of states such that

$$s_0 \models \Theta(s_0) \ \bigwedge \ \forall i > 0 \ , \ (s_{i-1}, s_i) \models \rho(s_{i-1}, s_i) \tag{1}$$

[2]For instance, most of the effort related to the semantics of ESTEREL has been directed toward solving this issue satisfactorily.

[3]Usually, variables and *primed* variables are used to refer to current and *next* states. This is equivalent to our present notation. We have preferred to consider s^- and s, just because the formulæ we shall write mostly involve current variables, rather than past ones. Using the standard notation would have resulted in a burden of primed variables in the formulæ.

The *composition* of two STS $\Phi = \Phi_1 \bigwedge \Phi_2$ is defined as follows:

$$\begin{array}{rcl} V & = & V_1 \cup V_2 \\ \Theta & = & \Theta_1 \wedge \Theta_2 \\ \rho & = & \rho_1 \wedge \rho_2 \ , \end{array}$$

the composition is thus the pairwise conjunction of initial and transition relations.

Notations for STS: we shall use the following generic notations in the sequel:

- $c, v, w, \ldots$ denote STS *variables.*
- for v a variable, $h_v \in \{\mathrm{T}, \perp\}$ denotes its *clock*:

  $$[h_v \neq \perp] \quad \Leftrightarrow \quad [v \neq \perp]$$

- for v a variable, ξ_v denotes its associated *state variable,* defined by:

  $$\begin{array}{lll} \textbf{if} \;\; h_v & \textbf{then} & \xi_v = v \\ & \textbf{else} & \xi_v = \xi_v^- \end{array} \tag{2}$$

  with the convention that $s_0[\xi_v^-] = \perp$, i.e., ξ_v is absent before the 1st occurrence of v, and is always present after the 1st occurrence of v. State variable ξ_v is assumed to be *local,* i.e., is never shared among different STS, and thus state variables play no role for STS composition.

Examples of Transition Relations :

- a selector:

  $$\textbf{if} \;\; b \;\; \textbf{then} \;\; z = u \;\; \textbf{else} \;\; z = v \ .$$

  Note that the "**else**" part corresponds to the property "$[b = \mathrm{F}] \ \vee \ [b = \perp]$".

- decrementing a register:

  $$\textbf{if} \;\; h_z \;\; \textbf{then} \;\; v = \xi_z^- - 1 \;\; \textbf{else} \;\; v = \perp \quad ,$$

  where ξ_z is the state variable associated with z as in (2), and ξ_z^- denotes its past value. The more intuitive interpretation of this statement is: $(v_n = z_{n-1} - 1)$, where index "n" denotes the instants at which both v and z are present (their clocks are specified to be equal). The specification of a register would simply be:

  $$\textbf{if} \;\; h_z \;\; \textbf{then} \;\; v = \xi_z^- \;\; \textbf{else} \;\; v = \perp \quad .$$

  Note that both statements imply the equality of clocks:

  $$h_z \quad \equiv \quad h_v \ .$$

- testing for a property :

$$\textbf{if} \quad h_v \quad \textbf{then} \quad b = (v \leq 0) \quad \textbf{else} \quad b = \perp \quad .$$

  Note that a consequence of this definition is, again,

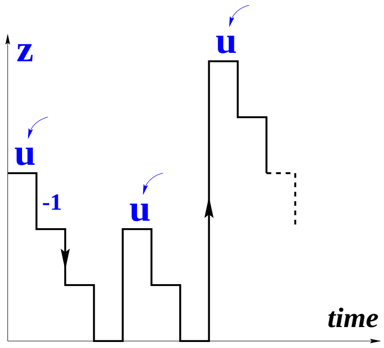
$$h_v \quad \equiv \quad h_b \ .$$

- a synchronization constraint :

$$(b = \text{T}) \quad \equiv \quad (h_u = \text{T}) \ ,$$

  meaning that the clock of u is the set of instants at which boolean variable b is true.

Putting things together yields the STS :



$$\begin{array}{ll} & \textbf{if } b \textbf{ then } z = u \textbf{ else } z = v \\ \bigwedge & \textbf{if } h_z \textbf{ then } v = \xi_z^- - 1 \textbf{ else } v = \perp \\ \bigwedge & \textbf{if } h_v \textbf{ then } b = (v \leq 0) \textbf{ else } b = \perp \\ \bigwedge & h_v \equiv h_z \equiv h_b \\ \bigwedge & (b = \text{T}) \equiv (h_u = \text{T}) \end{array}$$

A run of this STS for the z variable is depicted on the figure above. Each time u is received, z is reset and gets the value of u. Then z is decremented by one at each activation cycle of the STS, until it reaches the value 0. Immediately after this latter instant, a fresh u can be read, and so on. Note the schyzophrenic nature of the "inputs" of this STS. While the value carried by u is an input, the instant at which u is read is not : reading of the input is on demand-driven mode. This is reflected by the fact that inputs of this STS are the pair {activation clock h, value of u when it is present}.

Using these primitives, dataflow synchronous languages such as LUSTRE and SIGNAL are easily encoded.

Open environments and modularity. As modularity is wanted, it is desirable that the pace of an STS is local to it rather than global. Since any STS is subject to further composition in some yet unknown environment, this makes the requirement of having a global pace quite inconvenient. This is why *we prohibit the use of clocks that are always present.* This has several consequences. First, it is not possible to consider the "complement of a clock" or the "negation of a clock" : this would require referring to the always present clock. Thus, as will be revealed by our examples, clocks will always be variables, and we shall be able to relate clocks only using $\wedge$ (intersection of instants of presence), $\vee$ (union of instants of presence), and $\setminus$ (set difference of instants of presence).

Stuttering. In the same vein, it should be permitted, for an STS, to do nothing while the environment is possibly working. This feature has been yet identified in the litterature and is known as *stuttering* invariance or robustness. It is central to TLA, where it is understood that a transition with no event at all and no change of states is always legal.

For an STS Φ, stuttering invariance is defined as follows: if

$$\sigma \ : s_0, s_1, s_2, \ldots$$

is a run of Φ, so is

$$\sigma' \ : s_0, \underbrace{\perp_0, \ldots, \perp_0}_{0 \leq \ \#\{\perp_0\} < \infty}, s_1, \perp_1, \ldots, \perp_1, s_2, \perp_2, \ldots, \perp_2, \ldots \ , \tag{3}$$

where symbol $\perp_i$ denotes a *silent* state in which all state variables keep the value they had at state s_i, while other variables take the value $\perp$. The number of inserted silent states is ≥ 0 but finite. This models that the considered STS can do nothing for any arbitrary but finite "duration".

Stuttering invariance is not hardwired into our STS formalism, but a quick inspection of all the statements we have introduced in our example reveals that any composition of them is stuttering invariant. More generally, any STS not involving the always present clock (be it directly or indirectly) is stuttering invariant.

3 Formal study of desynchronization

3.1 Desynchronizing STS

From the definition (1) of a run for an STS, we can say that a run is a sequence of t-uples of values in domains extended with the extra symbol $\perp$. Desynchronising a run amounts to loosing the synchronisation barriers defining the successive reactions. Hence, for each variable $v \in V$, we only know the ordered sequence of *present* values. Thus

> desynchronising a run amounts to mapping a sequence of t-uples of values in domains extended with the extra symbol $\perp$, into a t-uple of sequences of present values, one sequence per each variable.

This is formalized next. For $\sigma \ : s_0, s_1, s_2, \ldots$ a run for Φ, we decompose state s_k as

$$s_k \quad = \quad (s_k[v])_{v \in V}$$

Thus we can rewrite run σ as follows:

$$\begin{array}{rcl} \sigma & = & (\sigma[v])_{v \in V} \ , \text{ where} \\ \sigma[v] & = & s_0[v] \, , \, s_1[v] \, , \, \ldots \, , \, s_k[v] \, , \, \ldots \ . \end{array}$$

Now, compress each $\sigma[v]$ by deleting those $s_k[v]$ that are equal to $\perp$. Formally, denote by $k_0, k_1, k_2, \ldots$ the subsequence of $j = 0, 1, 2, \ldots$ such that $s_j[v] \neq \perp$. Then we set

$$\begin{array}{rcl} \sigma^a & = & (\sigma^a[v])_{v \in V} \ , \text{ where} \\ \sigma^a[v] & = & s_{k_0}[v] \, , \, s_{k_1}[v] \, , \, s_{k_2}[v] \, , \, \ldots \ , \end{array}$$

this define our *desynchronisation mapping*

$$\sigma \quad \mapsto \quad \sigma^a \ . \tag{4}$$

For $\Phi = \langle V, \Theta, \rho \rangle$ an STS, we define

$$\Phi^a \quad =_{\text{def}} \quad \langle V, \Sigma^a \rangle \ , \tag{5}$$

where Σ^a is the family of all σ^a, for σ ranging over the set of runs of Φ. For $\Phi_i = \langle V_i, \Theta_i, \rho_i \rangle \ , i = 1, 2$, we define

$$\Phi_1^a \overset{a}{\bigwedge} \Phi_2^a \quad =_{\text{def}} \quad \langle V, \Sigma^a \rangle \ , \text{where} \left\{ \begin{array}{rcl} V & = & V_1 \cup V_2 \\ \Sigma^a & = & \Sigma_1^a \ \wedge^a \ \Sigma_2^a \end{array} \right. \tag{6}$$

and $\wedge^a$ denotes conjunction of sets of asynchronous runs, which we define now. For $\sigma_i^a \in \Sigma_i^a, i = 1, 2$, we say that σ_1^a and σ_2^a are *compatible,* written

$$\sigma_1^a \quad \bowtie^a \quad \sigma_2^a \ , \tag{7}$$

if the following condition holds:

$$\forall v \in V_1 \cap V_2 \quad : \quad \sigma_1^a[v] = \sigma_2^a[v] \text{ holds.}$$

If condition (7) holds, then we define $\sigma^a =_{\text{def}} \sigma_1^a \ \wedge^a \ \sigma_2^a$ as

$$\begin{array}{rcl} \forall v \in V_1 \cap V_2 & : & \sigma^a[v] = \sigma_1^a[v] = \sigma_2^a[v] \\ \forall v \in V_1 \setminus V_2 & : & \sigma^a[v] = \sigma_1^a[v] \\ \forall v \in V_2 \setminus V_1 & : & \sigma^a[v] = \sigma_2^a[v] \end{array}$$

Finally, Σ^a is the set of the so defined σ^a.

Synchrony vs. Asynchrony ? At this point two natural questions arise, namely:

1. *Is resynchronisation feasible and uniquely defined ?* More precisely, is it possible to uniquely reconstruct the original run σ for our STS from its desychronised version σ^a as defined in (4).

2. *Does communication behave equivalently for both the synchronous and asynchronous compositions ?* More precisely, does the following property hold:

$$\Phi_1^a \overset{a}{\bigwedge} \Phi_2^a \quad = \quad \left(\Phi_1 \bigwedge \Phi_2 \right)^a \ , \tag{8}$$

If point 1 was true, then we could desynchronise a run of the considered STS, and then still recover the original synchronous run. Thus point 1 would guarantee the preserving of the synchronous semantics when performing desynchronisation, for a single STS.

On the other hand, if (8) was true, then we could interpret our STS composition equivalently as synchronous or asynchronous.

Unfortunately, neither 1 nor 2 are true in general, due to the possibility to exercise control by the way of absence in synchronous composition $\bigwedge$. In the following section, we show that these properties indeed hold under certain sufficient conditions, in which the two notions of *endochrony* (for point 1) and *isochrony* (for point 2) play a central role[4].

[4] Endochronous, from ancient greek $\varepsilon\nu\delta o$–inside and $\chi\rho o\nu o\varsigma$–time; Isochronous, from ancient greek $\iota\sigma o$–identical and $\chi\rho o\nu o\varsigma$–time. It's sometimes nice to remember that ancient greeks used to be great scientists, and thus honor them by reusing their words in our context.

3.2 Endochrony and re-synchronisation

In this section, we use notations from section 2. For $\Phi = \langle V, \Theta, \rho \rangle$ an STS, and s an accessible state of Φ, we denote by s^h the clock-abstraction of s, defined by

$$\forall v \in V \ : \ s^h[v] \in \{\bot, \top\}, \text{ and } s^h[v] = \bot \Leftrightarrow s[v] = \bot \tag{9}$$

For $\Phi = \langle V, \Theta, \rho \rangle$ an STS, and $W' \subseteq W \subseteq V$, we say that W' is a *clock inference* of W, written

$$W' \hookrightarrow W \ , \tag{10}$$

if, for each accessible state s of Φ, knowing the presence/absence and actual value carried by each variable belonging to W' allows us to determine exactly the presence/absence for each variable belonging to W. In other words,

$$s[W'] \text{ determines } s^h[W] \, . \tag{11}$$

If $W' \hookrightarrow W_1$ and $W' \hookrightarrow W_2$ hold, then $W' \hookrightarrow (W_1 \cup W_2)$ follows, thus there exists a greatest W such that $W' \hookrightarrow W$ holds. Hence we can consider the unique increasing chain

$$\emptyset = V(0) \hookrightarrow V(1) \hookrightarrow V(2) \hookrightarrow \dots \tag{12}$$

of subsets of V such that, for each k, $V(k)$ is the greatest set of variables such that $V(k-1) \hookrightarrow V(k)$ holds. As $\emptyset = V(0)$, $V(1)$ consists of the subset of variables that are present as soon as the considered STS gets activated [5]. Of course chain (12) must become stationary at some finite $k_{\max}$: $V(k_{\max}+1) = V(k_{\max})$. In general, we only know that $V(k_{\max}) \subseteq V$. Chain (12) is called the *synchronization chain* of Φ.

Definition 1 (endochrony) STS Φ *is said to be* endochronous *if* $V(k_{\max}) = V$*, i.e., if the following condition is satisfied : the synchronization chain*

$$\textbf{(E)} \qquad \emptyset = V(0) \hookrightarrow V(1) \hookrightarrow V(2) \hookrightarrow \dots \textit{ converges to } V. \tag{13}$$

Condition (13) expresses that presence/absence of all variables can be inferred *incrementally* from already known values carried by present variables and state variables of the STS in consideration. Hence no test for presence/absence on the environment is needed. The following theorem justifies our approach :

Theorem 1 *Consider an* STS $\Phi = \langle V, \Theta, \rho \rangle$*. Then, condition (a) implies condition (b) :*

(a) Φ *is endochronous.*

(b) For each $\delta \in \Sigma^a$*, we can reconstruct the corresponding synchronous run* σ *such that* $\sigma^a = \delta$*, in a unique way up to silent reactions.*

[5] Of course we assume here that no variable is absent in every accessible state.

Proof: We prove the result by induction. Pick a $\delta \in \Sigma^a$, and assume for the moment that we were able to decompose it as:

$$\underbrace{s_1\ ,\ s_2\ ,\ \dots\ ,\ s_n}_{n-\text{initial segment of } \sigma}\quad ,\ \delta_n \tag{14}$$

i.e., into a finite n-lentgh sequence of non-silent states s_i (the head of the synchronous run σ we wish to reconstruct), followed by the tail of the asynchronous run δ, which we denote by δ_n, and we assume that such a decomposition is unique. Then we claim that

$$(14) \text{ is also valid with } n \text{ substituted by } n+1. \tag{15}$$

To prove (15), we note that, when STS Φ gets activated, then we know that variables belonging to $V(1)$ will be present in the considered state. By assumption, the clock-abstracted state $s^h_{n+1}[V(1)]$, having $V(1)$ as variables, is uniquely determined. In the sequel we write $s^h_{n+1}(1)$ for short instead of $s^h_{n+1}[V(1)]$. Thus, presence/absence of variables for state $s_{n+1}(1)$ is known, it remains to determine the values carried by present variables.

For $v \in V_1$, we simply pick the value carried by the minimal element of the sequence associated with variable v in δ_n. Values carried by corresponding state variables are updated accordingly. Thus we know all of $s_{n+1}(1)$.

Next we move on constructing $s_{n+1}(2)$. From $s_{n+1}(1)$ we know $s^h_{n+1}(2)$. Thus we know how to split V_2 into present and absent variables for the considered state. Pick the present ones, and repeat the same argument as before to get $s_{n+1}(2)$.

Repeat this argument until $V(k) = V$ for some finite k (by endochrony assumption). This proves claim (15).

Given the initial condition for δ, we get from (15), by induction, the desired proof that *(a)* $\Rightarrow$ *(b)*. $\diamond$

COMMENTS.

1. The proof of this theorem in fact provides an effective algorithm for the on-the-fly reconstruction of the successive reactions, for a desynchronized run of an endochronous program.

2. In fact, it is "nearly true" that *(b)* implies *(a)*. We conjecture that it would be feasible to slightly tighten our notion of a clock inference and synchronisation chain in order to take this problem into account, but at the price of an increasing complexity. We felt that this burden is not really justified, as the property which is checked in practice in the SIGNAL compiler is even slightly stronger than our present notion.

3.3 Isochrony, and synchronous and *a*synchronous compositions

The next result addresses the question of when property (8) holds true. We are given two STS $\Phi_i = \langle V_i, \Theta_i, \rho_i \rangle, i = 1,2$. Denote by $W = V_1 \cap V_2$ the set of their common variables, and by $\Phi = \Phi_1 \bigwedge \Phi_2$ their synchronous composition. For s an accessible state in Φ, we denote by $s_1 =_{\text{def}} s[V_1]$ and $s_2 =_{\text{def}} s[V_2]$ the restrictions of state s to Φ_1 and Φ_2, respectively. Note that, for $i = 1,2$, s_i is an accessible state for Φ_i. Corresponding notations s^-, s^-_1, s^-_2 for past states will be used accordingly.

Definition 2 (isochrony) *The pair* (Φ_1, Φ_2) *is called* isochronous *if the following condition* **(I)** *is satisfied:*

(I) *For each pair* (s_i^-, s_i) *of successive states for* $\Phi_i, i = 1, 2$ *satisfying the following three conditions:*

1. $s_1^- = s^-[V_1]$ *and* $s_2^- = s^-[V_2]$ *holds for some accessible state* s^- *for* Φ, *in particular* s_1^- *and* s_2^- *are unifiable;*
2. *none of the states* $s_i, i = 1, 2$ *are silent on the common variables, i.e., it is not the case that, for some* $i = 1, 2$: $s_i[v] = \perp$ *holds* $\forall v \in W$;
3. s_1 *and* s_2 *coincide over the set of* present *common variables*[6], *i.e.:*

$$\forall v \in W \ : \ (\, s_1[v] \neq \perp \text{ and } s_2[v] \neq \perp\,) \Rightarrow s_1[v] = s_2[v] \ ;$$

Then it follows that states s_1 *and* s_2 *coincide over the* whole set *of common variables, i.e., states* s_1 *and* s_2 *are unifiable, i.e.,*

$$s_1 = s[V_1] \text{ and } s_2 = s[V_2] \text{ holds for some accessible state } s \text{ for } \Phi \ .$$

Roughly speaking, condition **(I)** expresses that unifying over *present* common variables is enough to guarantee the unification of the two considered states s_1 and s_2. The following theorem justifies introducing this notion of isochrony.

Theorem 2

1. *If the pair* (Φ_1, Φ_2) *is isochronous, then it satisfies property (8).*
2. *Conversely, assume in addition that* Φ_1 *and* Φ_2 *are both endochronous. If the pair* (Φ_1, Φ_2) *satisfies property (8), then it is isochronous.*

Thus, isochrony is sufficient for (8) to hold, and it is also in fact necessary when the components are endochronous.

COMMENTS: We already discussed the importance of guaranteing property (8). Now, why is this theorem interesting? Mainly because it replaces condition (8), which involves infinite runs, by condition **(I)** of isochrony, which only involves a single reaction for the considered pair of STS.

Proof: We successively prove points 1 and 2.

Isochrony implies property (8). We proceed into two steps.

1. Denote by Φ^a the desynchronisation of Φ, defined by (5), and denote by δ a run of Φ^a. For each $\delta \in \Sigma^a$, there is at least one corresponding synchronous run σ for Φ such that $\delta = \sigma^a$. Any such σ is clearly the synchronous composition of two compatible runs σ_1 and σ_2 for Φ_1 and Φ_2, respectively. Hence associated asynchronous runs σ_1^a

[6] By convention this is satisfied if the set of present common variables is empty.

and σ_2^a are also compatible, and their asynchronous composition $\sigma_1^a \wedge^a \sigma_2^a$ belongs to $\Sigma_1^a \wedge^a \Sigma_2^a$. Thus we always have the inclusion

$$\Phi_1^a \overset{a}{\bigwedge} \Phi_2^a \supseteq \left(\Phi_1 \bigwedge \Phi_2\right)^a , \tag{16}$$

which proves the first part of (8). So far we have only used the definition of desynchronisation and asynchronous composition, isochrony has not yet been used.

2. To prove the opposite inclusion, we need to prove that, when moving from asynchronous composition to synchronous one, the additional need for a reaction-per-reaction matching of compatible runs will not result in rejecting pairs of runs that otherwise would be compatible in the asynchronous sense. This is where condition **(I)** of isochrony enters the game.

   Pick a pair (δ_1, δ_2) such that $\delta_1 \bowtie^a \delta_2$ (cf. (7)): they can be combined while performing the asynchronous composition $\Phi_1^a \wedge^a \Phi_2^a$ to form some δ (cf. (6)), this is denoted by $\delta_1 \wedge^a \delta_2 = \delta$. By definition of desynchronisation (cf. appendix 3.1), there exist a (synchronous) run σ_1 for Φ_1, and a (synchronous) run σ_2 for Φ_2, such that δ_i is obtained by desynchronising σ_i, $i = 1, 2$ (as we do not assume endochrony at this point, run σ_i is not uniquely determined). Thus each run σ_i is a succession of states. Clearly, inserting finitely many silent states between successive states of σ_i would also provide valid candidates for recovering δ_i after desynchronisation. We shall show, by induction over successive states, that:

   properly inserting such a silent state in the appropriate
   component will provide two runs which are (17)
   unifiable in the synchronous sense.

   This will show that, from a pair (δ_1, δ_2) such that $\delta_1 \bowtie^a \delta_2$, we can reconstruct (at least) one pair (σ_1, σ_2) of runs for Φ_1 and Φ_2 that are unifiable in the synchronous sense, and thus will prove the alternative inclusion

$$\Phi_1^a \overset{a}{\bigwedge} \Phi_2^a \subseteq \left(\Phi_1 \bigwedge \Phi_2\right)^a . \tag{18}$$

   From (16) and (18) we then deduce property (8). Thus we move on proving (17) by induction over successive states.

   Assume that pair (σ_1, σ_2) of runs for Φ_1 and Φ_2 are decomposed via sequence as:

$$\underbrace{s_{i,1}\,,\ s_{i,2}\,,\ \ldots\,,\ s_{i,n}}_{n-\text{initial segment of } \sigma_i}\,,\ \sigma_{i,n} \tag{19}$$

   into a finite n-lentgh sequence of states $s_{i,k}$, $k = 1, \ldots, n$, followed by the tail $\sigma_{i,n}$ of the considered run, in such a way that

$$\forall k = 1, \ldots, n \ : \quad \text{states } s_{1,k} \text{ and } s_{2,k} \text{ are unifiable.} \tag{20}$$

Next, decompose the tail $\sigma_{i,n}$ into

$$\sigma_{i,n} \quad = \quad s_{i,n+1}\,,\,\sigma_{i,n+1}\;.$$

It is generally not true that states $s_{1,n+1}$ and $s_{2,n+1}$ are unifiable. The following cases can occur:

CASE 1 : none of the two states $s_{1,n+1}$ and $s_{2,n+1}$ is silent over the common W variables. Concentrate on those $v \in W$ variables that are present in both states $s_{1,n+1}$ and $s_{2,n+1}$. As $\delta_1 \bowtie^a \delta_2$ holds, then we must have $s_{1,n+1}[v] = s_{2,n+1}[v]$ for any such v. Thus points 1,2,3 of condition **(I)** of isochrony are satisfied. Hence $s_{1,n+1}$ and $s_{2,n+1}$ are indeed unifiable in this case, by isochrony.

CASE 2 : both states $s_{1,n+1}$ and $s_{2,n+1}$ are silent over the common W variables. They are unifiable.

CASE 3 : one and only one of the two states $s_{1,n+1}$ and $s_{2,n+1}$ is silent over the common W variables, say $\forall v \in W \;:\; s_{1,n+1}[v] = \perp$. In this case we unify state $s_{1,n+1}$ with the silent state for Φ_2, i.e., we redefine and stretch $\sigma_{2,n}$ as follows:

$$\sigma_{2,n} \quad := \quad \perp\,,\; \underbrace{s_{2,n+1}\,,\,\sigma_{2,n+1}}_{\text{redefined tail } \sigma_{2,n+1}}\;,$$

where $\perp$ denotes the silent state.

This proves that hypothesis (19) can be extended to $n+1$.

Under endochrony of the components, property (8) implies isochrony. This is easy. From Theorem 1 we know that, in our argument for proving point 1 of theorem 2, the synchronous runs σ_i are uniquely defined, up to silent states, from their desynchronised respective versions σ_i^a. Now, focus on CASE 1 of this argument. If isochrony is not satisfied, then, for some pair $\sigma_1^a \bowtie^a \sigma_2^a$ of compatible asynchronous runs, and some decomposition (19) of them, it follows that points 1,2,3 of condition **(I)** of isochrony are satisfied, but states $s_{1,n+1}$ and $s_{2,n+1}$ are *not* unifiable. As our only possibility is to try to insert silent states for one of the two components – not feasible in CASE 1 – our process of incremental unification on a per reaction basis fails. Thus (18) is violated, and so is property (8). This finishes the proof of the theorem. $\diamond$