A THEORY OF FORMAL CHOREOGRAPHIC LANGUAGES

FRANCO BARBANERA ◉ ^{a}, IVAN LANESE ◉ ^{b}, AND EMILIO TUOSTO ◉ ^{c}

^{a} Dept. of Mathematics and Computer Science, University of Catania (Italy)
  e-mail address: barba@dmi.unict.it

^{b} Focus Team, University of Bologna/INRIA (Italy)
  e-mail address: ivan.lanese@gmail.com

^{c} Gran Sasso Science Institute (Italy)
  e-mail address: emilio.tuosto@gssi.it

ABSTRACT. We introduce a meta-model based on formal languages, dubbed *formal choreographic languages*, to study message-passing systems. Our framework allows us to generalise standard constructions from the literature and to compare them. In particular, we consider notions such as global view, local view, and projections from the former to the latter. The correctness of local views projected from global views is characterised in terms of a closure property. We consider a number of communication properties –such as (dead)lock-freedom– and give conditions on formal choreographic languages to guarantee them. Finally, we show how formal choreographic languages can capture existing formalisms; specifically we consider communicating finite-state machines, choreography automata, and multiparty session types. Notably, formal choreographic languages, differently from most approaches in the literature, can naturally model systems exhibiting non-regular behaviour.

1. Introduction

Choreographic models of message-passing systems are becoming increasingly popular both in academia [BB11, BZ07, CHY12] and industry [KBR$^+$05, OMG11, Bon18]. These models advocate two complementary views of communicating systems. The *global* view can be thought of as an holistic description of the interactions carried out by a number of participants. Dually, the *local* views describe the expected communication behaviour of each participant in isolation.

As discussed in Section 9, the literature offers various choreographic models. Here, we introduce *formal choreographic languages* (FCL) as a meta-model to formalise message-passing systems; existing choreographic models can be conceived as specifications of FCLs. Essentially, this results in the definition of *global* and *local* languages. The words of a global language (g-language for short) consist of (possibly infinite) sequences of *interactions*; an interaction has the form A→B:m and represents the fact that participant A sends message m to participant B, and participant B receives it. The words of a local language (l-language for short) consist of (possibly infinite) sequences of *actions*; actions can take two forms, A B?m and A B!m, respectively representing that participant B receives message m from A and that participant A sends message m to B.

Such languages provide an abstract description of the possible runs of a system in terms of sequences of interactions at the global level, executed through synchronous message-passing at the local level. A word w in a global language represents then a possible run expected of a communicating system. For instance, we can model a continuation of w as another word w' such that $w \cdot w'$ is also in the language. Also, w induces an expected "local" behaviour on each participant A; in fact, the behaviour of A is obtained by *projecting* w, that is by ignoring interactions on the run w not involving A while retaining only the *output* and *input* actions performed by A.

Our language-theoretic treatment is motivated mainly by the need for a general setting immune to syntactic restrictions. This naturally leads us to consider e.g., choreographies represented by context-free languages (cf. Example 3.12). In fact, we strive for generality; basically *prefix-closure* is the only requirement we impose on FCLs. (We discuss some implications of relaxing prefix-closure in Section 10.) The gist is that, if a sequence of interactions or of communications is an observable behaviour of a system, any prefix of the sequence should be observable as well. This allows us to consider partial executions as well as "complete" ones, which can be formalised in terms of maximal words (namely words without continuations). We admit infinite words to account for diverging computations, ubiquitous in communication protocols.

Some g-languages cannot be faithfully executed by distributed components. This can be illustrated with a simple example; consider a g-language containing only the word A→B:m·C→D:n and its prefixes. Such a language – in our interpretation of words concatenation as a "strict sequencing" operator (see e.g. [CM13, SD19] for non-strict interpretations of sequencing) – does specify that the interactions A→B:m and C→D:n must occur only in the given order. Clearly, this is not possible if the participants are distributed or act concurrently because C and D cannot be aware of when the interaction between A and B takes place.

Contributions & structure. We summarise below our main contributions.

Section 2 introduces FCL (g-languages in Definition 2.2, l-languages in Definition 2.4) and adapts standard constructions from the literature. In particular, we render communicating systems and choreographies as, respectively, sets of l-languages (Definition 2.6)

and g-languages, while we take inspiration for projections from choreographies and multi-party session types. We consider synchronous interactions; as discussed in Section 10, the asynchronous case is scope for future work.

Section 3 considers correctness of communicating systems with respect to choreographies (the communicating system "executes" an interaction only if it is specified) and completeness (the communicating system "executes" at least the specified interactions). An immediate consequence of our constructions is the completeness of systems projected from g-languages (Corollary 3.2). Correctness is more tricky and requires to introduce *closure under unknown information* (CUI, cf. Definition 3.3). Intuitively, a g-language is CUI if it contains extensions of words with a single interaction whose participants cannot distinguish the extended word from other words of the language. Theorem 3.7 characterises correctness of projected systems in terms of CUI.

Section 4 shows how FCLs allow us to capture many relevant communication properties in a fairly uniform way.

Section 5 proposes *branch-awareness* (Definition 5.3) to ensure the communication properties defined in Section 4 (Theorem 5.7). Intuitively, branch-awareness requires each participant to "distinguish" words where its behaviour differs. Notably, we separate the condition for correctness (CUI) from the one ensuring the communication properties (branch awareness). Most approaches in the literature instead combine them into a single condition, which takes names such as well-branchedness or projectability [HLV$^+$16]. Thus, these single conditions are stronger than each of CUI and branch-awareness.

We illustrate the generality of FCLs on three case studies (cf. Sections 6 to 8), respectively taken from communicating finite-state machines [BZ83], choreography automata [BLT20] and multiparty session types [SD19]. We remark that FCLs can capture protocols that cannot be represented by regular g-languages such as the "task dispatching" protocol in Example 3.12. To the best of our knowledge this kind of protocols cannot be formalised in other approaches.

Section 9 compares with related work while Section 10 draws some conclusions and discusses future work.

This paper is a revised and extended version of [BLT22]; the main differences are: (i) the results on decidability of CUI and branch-awareness in Section 7.2 are new; (ii) Section 6 yields a new case study; (iii) Section 7 has been significantly extended; (iv) full proofs have been added; (v) the text has been largely revised in order to improve readability.

2. Formal Choreographic Languages

We briefly recall a few basic notions used throughout the paper and fix some notation. Let Σ be an alphabet (i.e., a set of symbols).

We define the sets of finite and infinite words on Σ as

$$\Sigma^\infty = \Sigma^\star \cup \Sigma^\omega$$

where $\Sigma^\star$ is the usual Kleene-star closure on Σ and Σ^ω is the set of infinite words on Σ, that is maps from natural numbers to Σ (aka ω-words [Sta97]).

The concatenation operator is denoted as $\_\cdot\_$ and ε is its neutral element. If w is infinite then $w\cdot w' = w$. We write $a_0\cdot a_1\cdot a_2\cdot\ldots$ for the word mapping i to $a_i \in \Sigma$ for all natural numbers i.

Set	Intuitive definition	ranged over by
$\mathfrak{P}$	Participants	$\mathsf{A}, \mathsf{B}, \mathsf{X} \ldots$
$\mathfrak{M}$	Messages	$\mathsf{m}, \mathsf{x}, \ldots$
finite subsets of $\mathfrak{P}$	Sets of participants	$\mathcal{P}$
Σ_{int}	Interactions	$\alpha, \beta, \ldots$
Σ_{act}	Actions	$\mathsf{a}, \mathsf{b}, \ldots$
$\Sigma_{\mathrm{int}}^{\infty}$	Interaction words	$w, w', \ldots$
$\Sigma_{\mathrm{act}}^{\infty}$	Words of actions	$v, v', \ldots$
$\Sigma_{\mathrm{int}}^{\infty} \cup \Sigma_{\mathrm{act}}^{\infty}$	Words	$z, z', \ldots$
subsets of $\Sigma_{\mathrm{int}}^{\infty}$	Global languages	$\mathcal{L}, \mathcal{L}', \ldots$
subsets of $\Sigma_{\mathrm{act}}^{\infty}$	Local languages	$\mathbb{L}, \mathbb{L}', \ldots$

Table 1: Name conventions for variables

A language L on Σ is a subset of Σ^{∞}, namely $L \subseteq \Sigma^{\infty}$. The prefix-closure of $L \subseteq \Sigma^{\infty}$ is defined as

$$\mathsf{pref}(L) = \{\, z \in \Sigma^{\infty} \mid \text{there is } z' \in L \text{ such that } z \preceq z' \,\}$$

where $\preceq$ is the prefix relation; L is *prefix-closed* if $L = \mathsf{pref}(L)$.

A word z is *maximal* in a language $L \subseteq \Sigma^{\infty}$ if $z \preceq z'$ for $z' \in L$ implies $z' = z$. As usual we shall write $z \prec z'$ whenever $z \preceq z'$ and $z \neq z'$.

We shall deal with languages on particular alphabets[1], namely the alphabets of *interactions* and *actions* whose definition (see below) we borrowed from [BLT20].

Definition 2.1 (Interactions and actions alphabets)**.** Let $\mathfrak{P}$ be a set of *participants* (ranged over by $\mathsf{A}, \mathsf{B}, \mathsf{X}, \ldots$) and $\mathfrak{M}$ (ranged over by $\mathsf{m}, \mathsf{x}, \ldots$) a set of *messages*, such that $\mathfrak{P}$ and $\mathfrak{M}$ are disjoint. We define

$$\Sigma_{\mathrm{int}} = \{\, \mathsf{A}{\to}\mathsf{B{:}m} \mid \mathsf{A} \neq \mathsf{B} \in \mathfrak{P}, \mathsf{m} \in \mathfrak{M} \,\} \qquad \text{ranged over by } \alpha, \beta, \ldots$$
$$\Sigma_{\mathrm{act}} = \{\, \mathsf{A\,B!m}, \mathsf{A\,B?m} \mid \mathsf{A} \neq \mathsf{B} \in \mathfrak{P}, \mathsf{m} \in \mathfrak{M} \,\} \qquad \text{ranged over by } \mathsf{a}, \mathsf{b}, \ldots$$

We call Σ_{int} and Σ_{act}, respectively, *alphabet of interactions* and *alphabet of actions* (over $\mathfrak{P}$ and $\mathfrak{M}$).

Our results are independent of the sets $\mathfrak{P}$ and $\mathfrak{M}$ we consider and hence we do not further specify them. Words on $\Sigma_{\mathrm{int}}^{\infty}$ (ranged over by $w, w', \ldots$) are called *interaction words* while those on $\Sigma_{\mathrm{act}}^{\infty}$ (ranged over by $v, v', \ldots$) are called *words of actions*. Hereafter $z, z', \ldots$ range over $\Sigma_{\mathrm{int}}^{\infty} \cup \Sigma_{\mathrm{act}}^{\infty}$ and we use $\mathcal{L}$ and $\mathbb{L}$ to range over subsets of, respectively, $\Sigma_{\mathrm{int}}^{\infty}$ and $\Sigma_{\mathrm{act}}^{\infty}$.

Function $\mathsf{ptp}(\cdot)$ yields the set of *participants* involved in an interaction or in an action and it is defined on $\Sigma_{\mathrm{int}} \cup \Sigma_{\mathrm{act}}$ as follows:

$$\mathsf{ptp}(\mathsf{A}{\to}\mathsf{B{:}m}) = \mathsf{ptp}(\mathsf{A\,B!m}) = \mathsf{ptp}(\mathsf{A\,B?m}) = \{\, \mathsf{A}, \mathsf{B} \,\}$$

This function extends naturally to (sets of) words. The *subject* of $\mathsf{A\,B!m}$ is the sender A and the subject of $\mathsf{A\,B?m}$ is the receiver B.

We summarise in Table 1 all the name conventions for variables ranging over the sets defined above.

---

[1] These alphabets may be infinite; formal languages over infinite alphabets have been studied, e.g., in [ABB80].

A *global language* specifies the expected interactions of a system while a *local language* specifies the communication behaviour of a participant.

Definition 2.2 (Global language). A *global language* (*g-language* for short) is a prefix-closed language $\mathcal{L}$ on Σ_{int} such that $\mathsf{ptp}(\mathcal{L})$ is finite.

Example 2.3. For any finite subset Σ of Σ_{int}, the set $\Sigma^{\star}$ is a g-language; notice that, if $\mathfrak{P}$ is infinite, then $\Sigma_{\text{int}}^{\star}$ is not a g-language since it encompasses infinitely many participants.

The set $\mathcal{L} = \mathsf{pref}(\{\, \mathsf{C{\to}A{:}w{\cdot}A{\to}B{:}g},\ \mathsf{C{\to}B{:}w{\cdot}A{\to}B{:}g},\ \mathsf{C{\to}A{:}w{\cdot}C{\to}B{:}w} \,\})$ is a g-language (which will be used later on in other examples). It formally describes the following interaction protocol involving Alice, Bob and Carol: Carol can decide to ask either Bob or Alice to work and after that Alice gossips with Bob, unless also Bob is asked to work after Alice is.   ◇

Definition 2.4 (Local language). A *local language* (*l-language* for short) is a prefix-closed language $\mathbb{L}$ on Σ_{act} such that $\mathsf{ptp}(\mathbb{L})$ is finite. An l-language is A-*local* if its words have all actions with subject A.

Example 2.5. The set $\mathbb{L} = \{\, \varepsilon,\ \mathsf{C\,A!w},\ \mathsf{C\,B!w},\ \mathsf{C\,A!w{\cdot}C\,B!w} \,\}$ is a C-local language; in particular, it specifies the local behaviour of Carol with respect to the g-language $\mathcal{L}$ of Example 2.3.   ◇

As discussed in the Introduction, l-languages give rise to *communicating systems*.

Definition 2.6 (Communicating system). Let $\mathcal{P} \subseteq \mathfrak{P}$ be a finite set of participants. A *(communicating) system over* $\mathcal{P}$ is a map $S = (\mathbb{L}_{\mathsf{A}})_{\mathsf{A}\in\mathcal{P}}$ assigning, to each participant $\mathsf{A} \in \mathcal{P}$, an A-local language $\mathbb{L}_{\mathsf{A}} \neq \{\, \varepsilon \,\}$ such that $\mathsf{ptp}(\mathbb{L}_{\mathsf{A}}) \subseteq \mathcal{P}$.

By projecting a g-language $\mathcal{L}$ on a participant A we obtain the A-local language describing the sequence of actions performed by A in the interactions involving A in the words of $\mathcal{L}$. Definition 2.7 below recasts in our setting the notion of projection used in several choreographic formalisms, e.g., in [CHY12, HYC16].

Definition 2.7 (Projection). The *projection on* A of an interaction $\mathsf{B{\to}C{:}m}$ is computed by the function $\_\!\downarrow_{\_} : \Sigma_{\text{int}} \times \mathfrak{P} \to \Sigma_{\text{act}} \cup \{\, \varepsilon \,\}$ defined by:

$$(\mathsf{A{\to}B{:}m}){\downarrow_{\mathsf{A}}} = \mathsf{A\,B!m} \qquad\qquad (\mathsf{A{\to}B{:}m}){\downarrow_{\mathsf{B}}} = \mathsf{A\,B?m} \qquad\qquad (\mathsf{A{\to}B{:}m}){\downarrow_{\mathsf{C}}} = \varepsilon$$

and extended homomorphically to interaction words and g-languages. The *projection of a g-language* $\mathcal{L}$, written $\mathcal{L}{\downarrow}$, is the communicating system $(\mathcal{L}{\downarrow_{\mathsf{A}}})_{\mathsf{A}\in\mathsf{ptp}(\mathcal{L})}$.

Example 2.8. Let $\mathcal{L}$ be the g-language in Example 2.3. By Definition 2.7, we have $\mathcal{L}{\downarrow} = (\mathcal{L}{\downarrow_{\mathsf{X}}})_{\mathsf{X}\in\{\mathsf{A,B,C}\}}$ where $\mathcal{L}{\downarrow_{\mathsf{A}}} = \{\, \varepsilon,\ \mathsf{C\,A?w},\ \mathsf{C\,A?w{\cdot}A\,B!g},\ \mathsf{A\,B!g} \,\}$, $\mathcal{L}{\downarrow_{\mathsf{B}}} = \{\, \varepsilon,\ \mathsf{A\,B?g},\ \mathsf{C\,B?w},$ $\mathsf{C\,B?w{\cdot}A\,B?g} \,\}$, and $\mathcal{L}{\downarrow_{\mathsf{C}}} = \{\, \varepsilon,\ \mathsf{C\,A!w},\ \mathsf{C\,B!w},\ \mathsf{C\,A!w{\cdot}C\,B!w} \,\}$. The latter is the l-language in Example 2.5.   ◇

We consider a *synchronous* semantics of communicating systems, similarly to other choreographic approaches such as [BZ07, CHY12, DGJ$^+$15, SD19]. Intuitively, a choreographic word is in the semantics of a system S iff its projection on each participant A yields a word in the local language assigned by S to A.

Definition 2.9 (Semantics). Given a system S over $\mathcal{P}$, the set

$$[\![S]\!] = \{\, w \in \Sigma_{\text{int}}^{\infty} \mid \mathsf{ptp}(w) \subseteq \mathcal{P} \text{ and for all } \mathsf{A} \in \mathcal{P},\ w{\downarrow_{\mathsf{A}}} \in S(\mathsf{A}) \,\}$$

is the *(synchronous) semantics* of S.

Notice that the above definition coincides with the *join* operation in [FBS04], used in realisability conditions for an asynchronous setting.

By the finiteness condition on the number of participants in a g-language (Definition 2.2) we immediately get the following.

Fact 2.10. Let $\mathcal{L}$ be a g-language. Then $\mathcal{L} \subseteq \Sigma^\star_{\mathrm{int}}$ implies $[\![\mathcal{L}\!\downarrow]\!] \subseteq \Sigma^\star_{\mathrm{int}}$.

Example 2.11. For the system $\mathcal{L}\!\downarrow$ in Example 2.8, we have

$$[\![\mathcal{L}\!\downarrow]\!] = \mathsf{pref}(\{\ \mathsf{C}{\to}\mathsf{A}{:}\mathsf{w}{\cdot}\mathsf{A}{\to}\mathsf{B}{:}\mathsf{g},\ \ \mathsf{C}{\to}\mathsf{B}{:}\mathsf{w}{\cdot}\mathsf{A}{\to}\mathsf{B}{:}\mathsf{g},\ \ \mathsf{A}{\to}\mathsf{B}{:}\mathsf{g},\ \ \mathsf{C}{\to}\mathsf{A}{:}\mathsf{w}{\cdot}\mathsf{C}{\to}\mathsf{B}{:}\mathsf{w}{\cdot}\mathsf{A}{\to}\mathsf{B}{:}\mathsf{g}\ \}) \qquad \diamond$$

Two interactions α and β are *independent* (in symbols $\alpha \parallel \beta$) when $\mathsf{ptp}(\alpha) \cap \mathsf{ptp}(\beta) = \emptyset$. Informally, a language $\mathcal{L}$ describing the behaviour of a system, and containing a word w, does contain also all the words where independent interactions in w are swapped; we say that $\mathcal{L}$ is *concurrency closed*. The notion of concurrency closure in our setting is a delicate one because of the possible presence of infinite words. One in fact has to allow infinitely many swaps of independent interactions while avoiding that the interactions do disappear by pushing them infinitely far away. Technically, we consider Mazurkiewicz's traces [Maz86] on Σ_{int} with independence relation $\alpha \parallel \beta$:

Definition 2.12 (Concurrency closure). Let $\sim$ be the reflexive and transitive closure of the relation $\equiv$ on finite interaction words defined by $w\,\alpha\,\beta\,w' \equiv w\,\beta\,\alpha\,w'$ where $\alpha \parallel \beta$. Following [Gas90, Def. 2.1], $\sim$ extends to $\Sigma^\omega_{\mathrm{int}}$ by defining

$$\text{for all } w, w' \in \Sigma^\omega_{\mathrm{int}}: \qquad w \sim w' \quad \text{iff} \quad w \ll w' \quad \text{and} \quad w' \ll w$$

where $w \ll w'$ iff for each finite prefix w_1 of w there are a finite prefix w'_1 of w' and a g-word $\hat{w} \in \Sigma^\star_{\mathrm{int}}$ such that $w_1{\cdot}\hat{w} \sim w'_1$. A g-language $\mathcal{L}$ is *concurrency closed* if it coincides with its concurrency closure, namely $\mathcal{L} = \{\, w \in \Sigma^\infty_{\mathrm{int}} \mid \text{there is } w' \in \mathcal{L} \text{ such that } w \sim w' \,\}$.

As expected from the discussion above, the semantics of systems is naturally concurrency closed since in a distributed setting independent events can occur in any order. Indeed, relation $\sim$ can be characterised as follows.

Lemma 2.13. *Given a g-language $\mathcal{L}$ and two words $w_1, w_2 \in \mathcal{L}$, $w_1 \sim w_2$ iff $w_1{\downarrow}_{\mathsf{A}} = w_2{\downarrow}_{\mathsf{A}}$ for each $\mathsf{A} \in \mathsf{ptp}(\mathcal{L})$.*

Proof. This follows directly from [Gas90, Proposition 2.2]. $\qquad\square$

Therefore we have

Proposition 2.14. *Let S be a system. Then $[\![S]\!]$ is concurrency closed.*

Proof. Trivial, since closure under swap does not change the projection by Lemma 2.13. $\quad\square$

The intuition that g-languages, equipped with the projection and semantic functions of Definition 2.7 and Definition 2.9, do correspond to a natural syntax and semantics for the abstract notion of choreography, can be strengthened by showing that these functions form a Galois connection.

Let us define $\mathbb{G} = \{\, \mathcal{L} \mid \mathcal{L} \text{ is a g-language} \,\}$ and $\mathbb{S} = \{\, S \mid S \text{ is a system} \,\}$. Moreover, given $S, S' \in \mathbb{S}$, we define $S \subseteq S'$ if $S(\mathsf{A}) \subseteq S'(\mathsf{A})$ for each A.

Proposition 2.15. *The functions $\downarrow$ and $[\![\_]\!]$ form a (monotone) Galois connection between the posets $(\mathbb{G}, \subseteq)$ and $(\mathbb{S}, \subseteq)$, namely, $\downarrow$ and $[\![\_]\!]$ are monotone functions such that, given $\mathcal{L} \in \mathbb{G}$ and $S \in \mathbb{S}$:*

$$\mathcal{L}\!\downarrow\, \subseteq S \iff \mathcal{L} \subseteq [\![S]\!]$$

Proof. Note that $(\_)\!\downarrow$ and $[\![\_]\!]$ are trivially monotone by their definitions.

($\Longrightarrow$) We first observe that

$$
\begin{aligned}
[\![\mathcal{L}\!\downarrow]\!] \;&=\; \{\, w' \in \Sigma_{\mathrm{int}}^{\infty} \;\big|\; \mathsf{ptp}(w') \subseteq \mathsf{ptp}(\mathcal{L}) \text{ and for all } \mathsf{A} \in \mathsf{ptp}(\mathcal{L}),\ w'\!\downarrow_{\mathsf{A}} \in \mathcal{L}\!\downarrow_{\mathsf{A}} \,\} \\
&\subseteq\; \{\, w' \in \Sigma_{\mathrm{int}}^{\infty} \;\big|\; \mathsf{ptp}(w') \subseteq \mathsf{ptp}(\mathcal{L}) \text{ and for all } \mathsf{A} \in \mathsf{ptp}(\mathcal{L}),\ w'\!\downarrow_{\mathsf{A}} \in S(\mathsf{A}) \,\} \\
&=\; [\![S]\!]
\end{aligned}
$$

where the equalities above hold by Definition 2.9 and the inclusion holds by hypothesis $(\mathcal{L}\!\downarrow \subseteq S)$. Hence, given a word $w \in \mathcal{L}$, we get $w \in [\![\mathcal{L}\!\downarrow]\!]$ by construction and $w \in [\![S]\!]$ by the above.

($\Longleftarrow$) We have to show that $\mathcal{L}\!\downarrow_{\mathsf{A}} \subseteq S(\mathsf{A})$ for each A. Let hence $w_{\mathsf{A}} \in \mathcal{L}\!\downarrow_{\mathsf{A}}$. By definition there is $w \in \mathcal{L}$ with $w\!\downarrow_{\mathsf{A}} = w_{\mathsf{A}}$. Then $w \in [\![S]\!]$ by hypothesis and hence, by Definition 2.9, $w\!\downarrow_{\mathsf{A}}(= w_{\mathsf{A}}) \in S(\mathsf{A})$. $\qquad\square$

Notice that, by Proposition 2.15, $\mathcal{L}\!\downarrow \subseteq S$ can be understood as "$\mathcal{L}$ can be realised by S" according to the notion of realisability frequently used in the literature, namely that all behaviours of the choreography are possible for the system.

A *closure operator* is a function cl whose domain and codomain are ordered sets and such that cl is monotone ($x \leq y \implies \mathsf{cl}(x) \leq \mathsf{cl}(y)$), extensive ($x \leq \mathsf{cl}(x)$), and idempotent ($\mathsf{cl}(x) = \mathsf{cl}(\mathsf{cl}(x))$). It is well-known that, given a Galois connection $(f_{\star}, f^{\star})$, $\mathsf{cl} = f^{\star} \circ f_{\star}$ is a closure operator. In our setting $[\![\_\!\downarrow]\!]$ is a closure operator, hence the above boils down to the following corollary:

Corollary 2.16. *For all g-languages* $\mathcal{L}, \mathcal{L}' \in \mathbb{G}$,

monotonicity: $\mathcal{L} \subseteq \mathcal{L}' \implies [\![\mathcal{L}\!\downarrow]\!] \subseteq [\![\mathcal{L}'\!\downarrow]\!]$,

extensiveness: $\mathcal{L} \subseteq [\![\mathcal{L}\!\downarrow]\!]$,

idempotency: $[\![\mathcal{L}\!\downarrow]\!] = [\![[\![\mathcal{L}\!\downarrow]\!]\!\downarrow]\!]$.

As we shall see, extensiveness coincides with completeness (Definition 3.1) and, together with monotonicity, implies *harmonicity* (Definition 4.1).

3. CORRECTNESS AND COMPLETENESS

A g-language specifies the expected communication behaviour of a system made of several components. Definition 2.6 formalises such systems in terms of l-languages. In the present section we deal with properties relating a communicating system with a specification (i.e., a g-language). In particular, we first introduce correctness and completeness of a communicating system with respect a g-language. The latter property follows by the Galois connection discussed in the previous section. Instead in order to prove correctness conditions are introduced in terms of a closure property, CUI; this requires to handle continuity. We highlight the expressiveness of CUI g-languages by showing that there exist non-regular CUI g-languages.

Definition 3.1 (Correctness and completeness)**.** Let $\mathcal{L}$ be a g-language. A system S is *correct with respect to* $\mathcal{L}$ if $[\![S]\!] \subseteq \mathcal{L}$ and it is *complete with respect to* $\mathcal{L}$ if $[\![S]\!] \supseteq \mathcal{L}$.

Correctness and completeness are related to existing notions. For instance, in the literature on multiparty session types (see, e.g., the survey [HLV$^+$16]) correctness is analogous to *subject reduction* and completeness to *session fidelity*.

Notice that, by Proposition 2.15, we can interpret $\mathcal{L}\!\downarrow \subseteq S$ as a characterisation for completeness of S with respect to $\mathcal{L}$. Hence, an immediate result of the Galois connection

defined in Section 2 is that any system projected from a g-language is complete. In fact, completeness coincides with the extensiveness property of the closure operator associated to our Galois connection.

Corollary 3.2. *For any g-language $\mathcal{L}$, $\mathcal{L}\downarrow$ is complete with respect to $\mathcal{L}$.*

It is easy to check that a similar result does not hold for correctness. If we consider the g-language $\mathcal{L}$ of Example 2.3, we have that, as shown in Example 2.11, C→A:w·C→B:w·A→B:g ∈ $[\![\mathcal{L}\downarrow]\!]$ but C→A:w·C→B:w·A→B:g ∉ $\mathcal{L}$. That is $[\![\mathcal{L}\downarrow]\!] \not\subseteq \mathcal{L}$.

Characterising correctness for projected systems. Can we identify conditions on g-languages to ensure correctness of their projections? The answer to this question is positive since correctness can be characterised as a closure property.

Definition 3.3 (CUI)**.** A g-language $\mathcal{L}$ is *closed under unknown information*, in symbols cui($\mathcal{L}$), if for all finite words $w_1 \cdot \alpha, w_2 \cdot \alpha \in \mathcal{L}$ with the same final interaction $\alpha = $ A→B:m $\in \Sigma_{\text{int}}$ we have $w \cdot \alpha \in \mathcal{L}$ for all $w \in \mathcal{L}$ such that $w \downarrow_\mathsf{A} = w_1 \downarrow_\mathsf{A}$ and $w \downarrow_\mathsf{B} = w_2 \downarrow_\mathsf{B}$.

Intuitively, participants cannot distinguish words with the same projection on their role. Hence, if two participants A and B find words w_1 and w_2 compatible with another word w, and interaction A→B:m can occur after both w_1 and w_2, then it should be enabled also after w. Indeed, A cannot know whether the current word is w or w_1 and, likewise, B cannot tell apart w and w_2. Hence, $\mathcal{L}$ should encompass w because, after the execution of w, A and B are willing to take A→B:m, which can thus happen at the system level. Closure under unknown information (CUI for short) lifts this requirement at the level of g-language.

Example 3.4. The language $\mathcal{L}$ in Example 2.3 is not CUI because it contains the words

$$w_1 \cdot \alpha = \mathsf{C{\to}A{:}w{\cdot}A{\to}B{:}g} \qquad w_2 \cdot \alpha = \mathsf{C{\to}B{:}w{\cdot}A{\to}B{:}g} \qquad \text{and} \qquad w = \mathsf{C{\to}A{:}w{\cdot}C{\to}B{:}w}$$

and A cannot distinguish between w_1 and w while B cannot distinguish between w_2 and w; nonetheless $w \cdot$A→B:g $= $ C→A:w·C→B:w·A→B:g $\notin \mathcal{L}$. Notice that, as shown after Corollary 3.2 $\mathcal{L} \not\supseteq [\![\mathcal{L}\downarrow]\!]$.   ◇

The language in Example 3.4 is not the semantics of any system, in fact languages obtained as semantics of a communicating system are always CUI.

Proposition 3.5 (Semantics is CUI)**.** *For all systems S, $[\![S]\!]$ is CUI.*

Proof. Let S be a system over $\mathcal{P}$. In order to show closure of $[\![S]\!]$ under unknown information, let us take words $w_\mathsf{A} \cdot$A→B:m, $w_\mathsf{B} \cdot$A→B:m, $w' \in [\![S]\!]$, such that

$$w_\mathsf{A} \downarrow_\mathsf{A} = w' \downarrow_\mathsf{A} \qquad \text{and} \qquad w_\mathsf{B} \downarrow_\mathsf{B} = w' \downarrow_\mathsf{B}$$

By Definition 3.3, we have to show that $w' \cdot$A→B:m $\in [\![S]\!]$. This in turn, by definition of synchronous semantics, amounts to show that $(w' \cdot$A→B:m$) \downarrow_\mathsf{X} \in S(\mathsf{X})$ for each $\mathsf{X} \in \mathcal{P}$. If $\mathsf{X} \neq \mathsf{A}, \mathsf{B}$ then $(w' \cdot$A→B:m$) \downarrow_\mathsf{X} = w' \downarrow_\mathsf{X} \in S(\mathsf{X})$ since $w' \in [\![S]\!]$. Otherwise $\mathsf{X} \in \{\mathsf{A}, \mathsf{B}\}$; hence, by hypothesis and definition of projection we have

$$(w' \cdot \mathsf{A{\to}B{:}m}) \downarrow_\mathsf{A} = w_\mathsf{A} \downarrow_\mathsf{A} \cdot \mathsf{A\,B!m} \in S(\mathsf{A}) \quad \text{and} \quad (w' \cdot \mathsf{A{\to}B{:}m}) \downarrow_\mathsf{B} = w_\mathsf{B} \downarrow_\mathsf{B} \cdot \mathsf{A\,B?m} \in S(\mathsf{B})$$

where the last two equalities hold by hypothesis.   □

The next property connects finite and infinite words in a language; it corresponds to the closure under the limit operation used in ω-languages [Eil76, Sta97].

Definition 3.6 (Continuity)**.** A language L on an alphabet Σ is *continuous* if $z \in L$ for all $z \in \Sigma^\omega$ such that $\mathsf{pref}(z) \cap L$ is infinite.

This notion of continuity, besides being quite natural, is the most suitable for our purposes among the possible ones [Red86]. Intuitively, a language L is continuous if an ω-word is in L when infinitely many of its approximants (i.e., finite prefixes) are in L. A g-language $\mathcal{L}$ is *standard or continuous* (sc-language, for short) if either $\mathcal{L} \subseteq \Sigma^\star_{\mathrm{int}}$ or $\mathcal{L}$ is continuous. Notice that, for prefix-closed languages, for all $z \in \Sigma^\omega$ we have that $\mathsf{pref}(z) \cap L$ is infinite iff $\mathsf{pref}(z) \subseteq L$.

Closure under unknown information characterises correct projected systems.

Theorem 3.7 (Characterisation of correctness)**.** *If* $\mathcal{L}{\downarrow}$ *is correct with respect to* $\mathcal{L}$ *then* $\mathsf{cui}(\mathcal{L})$ *holds. If* $\mathcal{L}$ *is an sc-language and* $\mathsf{cui}(\mathcal{L})$ *then* $\mathcal{L}{\downarrow}$ *is correct with respect to* $\mathcal{L}$*.*

Proof. We prove the first implication. In order to show closure of $\mathcal{L}$ under unknown information, let us take words $w_{\mathsf{A}} = w'_{\mathsf{A}} \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}$, $w_{\mathsf{B}} = w'_{\mathsf{B}} \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}$, $w' \in \mathcal{L}$, such that

$$w'_{\mathsf{A}}{\downarrow}_{\mathsf{A}} = w'{\downarrow}_{\mathsf{A}} \qquad \text{and} \qquad w'_{\mathsf{B}}{\downarrow}_{\mathsf{B}} = w'{\downarrow}_{\mathsf{B}} \tag{1}$$

By Definition 3.3, we have to show that $w' \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m} \in \mathcal{L}$. Thanks to correctness, it is enough to show that $w' \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m} \in [\![\mathcal{L}{\downarrow}]\!]$. This in turn, by definition of synchronous semantics, amounts to show that $v_{\mathsf{X}} = (w' \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}){\downarrow}_{\mathsf{X}} \in \mathcal{L}{\downarrow}_{\mathsf{X}}$ for each $\mathsf{X} \in \mathcal{P}$. If $\mathsf{X} \neq \mathsf{A}, \mathsf{B}$ then $v_{\mathsf{X}} = (w' \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}){\downarrow}_{\mathsf{X}} = w'{\downarrow}_{\mathsf{X}} \in \mathcal{L}{\downarrow}_{\mathsf{X}}$. Otherwise $\mathsf{X} \in \{\mathsf{A}, \mathsf{B}\}$; we consider only the case $\mathsf{X} = \mathsf{A}$ since the other case is analogous. We have

$$
\begin{aligned}
v_{\mathsf{X}} &= (w' \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}){\downarrow}_{\mathsf{A}} \\
&= w'{\downarrow}_{\mathsf{A}} (\mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}){\downarrow}_{\mathsf{A}} && \text{(by def. of projection)} \\
&= w'_{\mathsf{A}}{\downarrow}_{\mathsf{A}} (\mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}){\downarrow}_{\mathsf{A}} && \text{(by (1))} \\
&= (w'_{\mathsf{A}} \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}){\downarrow}_{\mathsf{A}} \in \mathcal{L}{\downarrow}_{\mathsf{A}} && \text{(by def. of projection and } w'_{\mathsf{A}} \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m} \in \mathcal{L})
\end{aligned}
$$

as required.

We now prove the second implication. By Definition 3.1, we have to show that $[\![\mathcal{L}{\downarrow}]\!] \subseteq \mathcal{L}$; we proceed by contradiction. Fix a word $w \in [\![\mathcal{L}{\downarrow}]\!] \setminus \mathcal{L}$. The only possible cases are:

$\boxed{\mathcal{L} \subseteq \Sigma^\star_{\mathrm{int}}}$ By Fact 2.10, $[\![\mathcal{L}{\downarrow}]\!]$ does not contain infinite g-words. Hence, w is finite and we can take its longest prefix w' that belongs to $\mathcal{L}$. Let $\alpha = \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}$ be the interaction immediately following w' in w. We can choose $w_{\mathsf{A}}, w_{\mathsf{B}} \in \mathcal{L}$ such that $w_{\mathsf{A}}{\downarrow}_{\mathsf{A}} = w{\downarrow}_{\mathsf{A}}$ and $w_{\mathsf{B}}{\downarrow}_{\mathsf{B}} = w{\downarrow}_{\mathsf{B}}$. (Recall that by Definition 2.9 for each $\mathsf{X} \in \mathcal{P}$ there is a word $w_{\mathsf{X}} \in \mathcal{L}$ such that $w{\downarrow}_{\mathsf{X}} = w_{\mathsf{X}}{\downarrow}_{\mathsf{X}}$.) Take the shortest prefixes w'_{A} and w'_{B} of w_{A} and w_{B} respectively such that

$$w'_{\mathsf{A}}{\downarrow}_{\mathsf{A}} = (w' \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}){\downarrow}_{\mathsf{A}} \qquad \text{and} \qquad w'_{\mathsf{B}}{\downarrow}_{\mathsf{B}} = (w' \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}){\downarrow}_{\mathsf{B}}$$

It is then easy to check that w'_{A} and w'_{B} have necessarily the following shapes

$$w'_{\mathsf{A}} = \tilde{w}_{\mathsf{A}} \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m} \qquad\qquad w'_{\mathsf{B}} = \tilde{w}_{\mathsf{B}} \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}$$

where

- $\tilde{w}_{\mathsf{A}}, \tilde{w}_{\mathsf{B}} \in \mathcal{L}$, since $\mathcal{L}$ is prefix-closed;
- $\tilde{w}_{\mathsf{A}}{\downarrow}_{\mathsf{A}} = w'{\downarrow}_{\mathsf{A}}$ and $\tilde{w}_{\mathsf{B}}{\downarrow}_{\mathsf{B}} = w'{\downarrow}_{\mathsf{B}}$.

Now, by $w' \in \mathcal{L}$ and the definition of CUI, we infer that $w' \cdot \mathsf{A}{\to}\mathsf{B}{:}\mathsf{m} \in \mathcal{L}$ against the hypothesis that w' was the longest such prefix of w.

$\boxed{\mathcal{L} \not\subseteq \Sigma^\star_{\mathrm{int}}}$ If the set $\widehat{\mathcal{L}} = \{\, w' \in \Sigma^\star_{\mathrm{int}} \mid w' \preceq w \text{ and } w' \in \mathcal{L} \,\}$ is finite then we take the longest g-word in $\widehat{\mathcal{L}}$ and the proof is as in the previous case. Otherwise we have a contradiction because, by continuity, $w \in \mathcal{L}$. $\qquad\square$

Notice that CUI is defined in terms of g-languages only, hence checking CUI does not require to build the corresponding system. This allows us to study CUI on specific classes of g-languages. For instance, we can show that CUI is decidable on a class of languages accepted by Büchi automata (cf. Section 7). An interesting observation is that strengthening the precondition of Definition 3.3 with the additional requirement $w_1 = w_2$ would invalidate Theorem 3.7. Indeed, $\mathcal{L} \cup \{A{\to}B{:}g\}$ with $\mathcal{L}$ the language in Example 2.3 would become CUI but not correct. The next example shows that the continuity condition in Theorem 3.7 is necessary for languages containing infinite g-words.

Example 3.8 (Continuity matters)**.** The CUI language

$$\mathcal{L} = \mathsf{pref}(\bigcup_{i \geq 0}\{\, A{\to}B{:}l{\cdot}B{\to}C{:}n{\cdot}(C{\to}D{:}n)^i\,\} \cup \{\, A{\to}B{:}r{\cdot}B{\to}C{:}n{\cdot}(C{\to}D{:}n)^\omega\,\})$$

does contain an infinite word but it is not continuous. The projection of $\mathcal{L}$ is not correct because its semantics contains the g-word $A{\to}B{:}l{\cdot}B{\to}C{:}n{\cdot}(C{\to}D{:}n)^\omega$. This word, in fact, does not belong to $\mathcal{L}$ since the projections of C and D can exchange infinitely many messages n due to the infinite g-word of $\mathcal{L}$ regardless whether A and B exchange l or r. $\diamond$

Notice that, since $\mathcal{L} \subseteq [\![\mathcal{L}{\downarrow}]\!]$ always holds, Theorem 3.7 implies that $\mathsf{cui}(\mathcal{L})$ characterises the languages $\mathcal{L}$ such that $\mathcal{L} = [\![\mathcal{L}{\downarrow}]\!]$. Besides, the following corollary descends from Theorem 3.7.

Corollary 3.9. *If $\mathcal{L}$ is an sc-language, $[\![\mathcal{L}{\downarrow}]\!]$ is the smallest CUI sc-language containing $\mathcal{L}$.*

Proof. Let $\mathsf{cl}(\mathcal{L}) = [\![\mathcal{L}{\downarrow}]\!]$. Given an sc-language $\mathcal{L}$, $\mathsf{cui}(\mathsf{cl}(\mathcal{L}))$ holds by Proposition 3.5. Moreover, it is not difficult to check that if $\mathcal{L}$ is an sc-language, so is $\mathsf{cl}(\mathcal{L})$. $\mathcal{L} \subseteq \mathsf{cl}(\mathcal{L})$ holds by extensiveness of cl. Now, in order to show that $\mathsf{cl}(\mathcal{L})$ is smaller or equal than any CUI sc-language containing $\mathcal{L}$, let us consider any sc-language $\tilde{\mathcal{L}}$ such that $\mathsf{cui}(\tilde{\mathcal{L}})$ and $\mathcal{L} \subseteq \tilde{\mathcal{L}}$. By monotonicity of cl, we have that $\mathsf{cl}(\mathcal{L}) \subseteq \mathsf{cl}(\tilde{\mathcal{L}})$ and, by Theorem 3.7, $\mathsf{cl}(\tilde{\mathcal{L}}) = \widehat{\tilde{\mathcal{L}}}$. So $\mathsf{cl}(\mathcal{L}) \subseteq \tilde{\mathcal{L}}$. $\square$

CUI ensures that continuous g-languages are concurrency closed.

Proposition 3.10. *If $\mathcal{L}$ is an sc-language and $\mathsf{cui}(\mathcal{L})$, then $\mathcal{L}$ is concurrency closed.*

Proof. Thanks to Theorem 3.7 $\mathcal{L} = [\![\mathcal{L}{\downarrow}]\!]$, hence it is concurrency closed thanks to Proposition 2.14. $\square$

Hence, an sc-language cannot be CUI unless it is concurrency closed.

As recalled before, in many choreographic formalisms (such as [BDCLT21, HLV+16, CDYP16, BBO12, FBS04]) the correctness and completeness of a projected system, namely $\mathcal{L} = [\![\mathcal{L}{\downarrow}]\!]$ (together with some forms of liveness and deadlock-freedom properties), is guaranteed by *well-branchedness* conditions. Most of such conditions guarantee, informally speaking, that participants reach consensus on which branch to take when choices arise. For instance, a well-branchedness condition could be that, at each choice, there is a unique participant deciding the branch to follow during a computation and that such participant informs each other participant. Such a condition is actually not needed to prove $\mathcal{L} = [\![\mathcal{L}{\downarrow}]\!]$, as shown by the example below.

Example 3.11. The g-language $\mathcal{L}$ of Example 2.11 is CUI, without being well-branched in the above sense. Indeed, after the interaction $C{\to}A{:}w$, there is a branching in the projected

system, since both the interactions $\mathsf{C}{\to}\mathsf{B}{:}\mathsf{w}$ and $\mathsf{A}{\to}\mathsf{B}{:}\mathsf{g}$ can be performed. However, these interactions do not have the same sender.

A key merit of our model is its generality and expressiveness. We show this with an example of a non-regular CUI g-language whose projected system is correct and complete by Theorem 3.7 and Corollary 3.2, respectively.

Example 3.12 (A non-regular CUI g-language)**.** We consider a task dispatching service where, as soon as a Server communicates its $\mathsf{a}$vailability, a Dispatcher sends a $\mathsf{t}$ask to S. The server either processes the task directly and sends back the resulting $\mathsf{d}$ata to D or sends the task to participant H for some pre-processing, aiming at resuming it later on. Indeed, after communicating a result to D, the server can $\mathsf{r}$esume a previous task (if any) from H, process it, and send the result to D. The server eventually stops by sending s to both D and H; this can happen only when all dispatched tasks have been processed. The task dispatching scenario above is faithfully captured by the g-language $\mathcal{L}$ obtained by prefix-closing the (non-regular) language generated by the following context-free grammar.

$$
\begin{array}{lll}
L & ::= & L'{\cdot}\mathsf{S}{\to}\mathsf{D}{:}\mathsf{s}{\cdot}\mathsf{S}{\to}\mathsf{H}{:}\mathsf{s} \\
L' & ::= & \mathsf{S}{\to}\mathsf{D}{:}\mathsf{a}{\cdot}\mathsf{D}{\to}\mathsf{S}{:}\mathsf{t}{\cdot}\mathsf{S}{\to}\mathsf{H}{:}\mathsf{t}{\cdot}L'{\cdot}\mathsf{S}{\to}\mathsf{H}{:}\mathsf{r}{\cdot}\mathsf{H}{\to}\mathsf{S}{:}\mathsf{r}{\cdot}\mathsf{S}{\to}\mathsf{D}{:}\mathsf{d}{\cdot}L' \;\bigm|\; \mathsf{S}{\to}\mathsf{D}{:}\mathsf{a}{\cdot}\mathsf{D}{\to}\mathsf{S}{:}\mathsf{t}{\cdot}\mathsf{S}{\to}\mathsf{D}{:}\mathsf{d}{\cdot}L' \;\bigm|\; \varepsilon
\end{array}
$$

Since S is involved in all the interactions of $\mathcal{L}$, for each pair of words $w, w' \in \mathcal{L}$: $w{\downarrow}_{\mathsf{S}} = w'{\downarrow}_{\mathsf{S}}$ iff $w = w'$. Now, if $w_1\,\alpha, w_2\,\alpha, w \in \mathcal{L}$ satisfy the required conditions for CUI then either $w_1{\downarrow}_{\mathsf{S}} = w{\downarrow}_{\mathsf{S}}$ or $w_2{\downarrow}_{\mathsf{S}} = w{\downarrow}_{\mathsf{S}}$, since $\mathsf{S} \in \mathsf{ptp}(\alpha)$. Hence $\mathsf{cui}(\mathcal{L})$ trivially holds. $\diamond$

The language in Example 3.12 is non-regular since it has the same structure of a language of well-balanced parenthesis. Remarkably, this implies that the g-language cannot be expressed in most of the other choreographic models in the literature. There are models going beyond regular languages for both the binary (e.g., [Dar14, TV16]) and the multi-party case (e.g.,[JY20]). These approaches are based on process algebraic methods. An interesting future research direction would be to study whether the multi-party models give rise to CUI and branch-aware languages. The argument used to show $\mathsf{cui}(\mathcal{L})$ in Example 3.12 proves the following.

Proposition 3.13. *If there exists a participant involved in all the interactions of a g-language* $\mathcal{L}$ *then* $\mathsf{cui}(\mathcal{L})$ *holds.*

4. Communication Properties

Besides correctness and completeness, other properties could be of interest for message-passing systems. For instance, one would like to ensure that participants eventually interact, if they are willing to. More generally, we are interested in some relations between the interactions in a system and the communication actions of its participants. We consider a number of properties, defined as follows.

Harmonicity (HA) requires that each sequence of communications that a participant is able to perform can be executed in some computation of the system.

Definition 4.1 (Harmonicity)**.** A system S on $\mathcal{P}$ is *harmonic* if $S(\mathsf{A}) \subseteq [\![S]\!]{\downarrow}_{\mathsf{A}}$ for each participant $\mathsf{A} \in \mathcal{P}$.

The remaining communication properties rely on the notion of maximal computation (cf. beginning of Section 2 on page 4). *Lock-freedom* (LF) requires that if a participant has pending communications to make on an ongoing computation, then there is a continuation of the computation involving that participant.

Definition 4.2 (Lock-freedom)**.** A system S on $\mathcal{P}$ is *lock free* if, for each finite word $w \in [\![S]\!]$ and participant $\mathsf{A} \in \mathcal{P}$, if $w\!\downarrow_\mathsf{A}$ is not maximal in $S(\mathsf{A})$ then there is a word w' such that $ww' \in [\![S]\!]$ and $w'\!\downarrow_\mathsf{A} \neq \varepsilon$.

Strong lock-freedom (SLF) requires that if a participant has pending communications to make on an ongoing computation, then each maximal continuation of the computation involves that participant.

Definition 4.3 (Strong lock-freedom)**.** A system S on $\mathcal{P}$ is *strongly lock free* if, for each finite word $w \in [\![S]\!]$ and participant $\mathsf{A} \in \mathcal{P}$, if $w\!\downarrow_\mathsf{A}$ is not maximal in $S(\mathsf{A})$ then for each word w' such that ww' is maximal in $[\![S]\!]$ we have $w'\!\downarrow_\mathsf{A} \neq \varepsilon$.

Starvation-freedom (SF) requires that if a participant has pending communications to make on an ongoing computation, then each infinite continuation of the computation involves that participant.

Definition 4.4 (Starvation-freedom)**.** A system S on $\mathcal{P}$ is *starvation free* if, for each finite word $w \in [\![S]\!]$ and participant $\mathsf{A} \in \mathcal{P}$, if $w\!\downarrow_\mathsf{A}$ is not maximal in $S(\mathsf{A})$ then $w'\!\downarrow_\mathsf{A} \neq \varepsilon$ for each infinite word w' such that $ww' \in [\![S]\!]$.

Deadlock-freedom (DF) requires that in all completed computations each participant has no pending actions.

Definition 4.5 (Deadlock-freedom)**.** A system S on $\mathcal{P}$ is *deadlock free* if, for each finite and maximal word $w \in [\![S]\!]$ and participant $\mathsf{A} \in \mathcal{P}$, $w\!\downarrow_\mathsf{A}$ is maximal in $S(\mathsf{A})$.

Barred for harmonicity, these properties appear in the literature under different names in various contexts. For instance, the notion of lock-freedom in [BDCLT21] corresponds to ours, which in turn corresponds to the notion of liveness in [LNTY17, KS10] in a channel-based synchronous communication setting. Likewise, the notion of strong lock-freedom in [SD19] corresponds to ours and, under fair scheduling, to the notion of lock-freedom in [Kob02]. As a final example, the definition of deadlock-freedom in its (equivalent) contrapositive form, coincides with the notion of progress as defined for synchronous processes in [Pad13, GJP+19]. Harmonicity, introduced in the present paper, assures that no behaviour of a participant can be taken out from a system without affecting the overall behaviour of the system itself. Notice that the inverse of harmonicity, $[\![S]\!]\!\downarrow_\mathsf{A} \subseteq S(\mathsf{A})$, holds by construction.

The next proposition highlights the relations among our properties.

Proposition 4.6. *The following relations hold among the properties in Definitions 4.1 to 4.5*
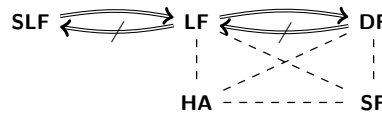
where implication does not hold in any direction between properties connected by dashed lines. Moreover, $\mathsf{DF} \wedge \mathsf{SF} \Leftrightarrow \mathsf{SLF}$.

Proof.

$\boxed{\mathsf{SLF} \implies \mathsf{LF}}$ Let $\mathsf{A} \in \mathcal{P}$, $w \in [\![S]\!]$ be finite, and $w\downarrow_{\mathsf{A}}$ not to be maximal in $S(\mathsf{A})$. By SLF we can infer that also w is not maximal in $[\![S]\!]$. Otherwise, for $w' = \varepsilon$, we would have $w \cdot w'$ to be maximal in $[\![S]\!]$ and, by SLF, we would get $\varepsilon = w'\downarrow_{\mathsf{A}} \neq \varepsilon$. Contradiction. So, if w is not maximal in $[\![S]\!]$, there exists $w' \neq \varepsilon$ such that $w \cdot w'$ is maximal in $[\![S]\!]$. We get hence immediately $w'\downarrow_{\mathsf{A}} \neq \varepsilon$ by SLF.

$\boxed{\mathsf{LF} \implies\!\!\!\!/\ \ \mathsf{SLF}}$ Let us take the communicating system $S = (\mathbb{L}_{\mathsf{X}})_{\mathsf{X} \in \{\mathsf{A},\mathsf{B},\mathsf{C}\}}$, where

$$\mathbb{L}_{\mathsf{A}} = \mathsf{pref}(\{\, \mathsf{A\,B!m \cdot A\,B!m},\ \mathsf{A\,B!m \cdot A\,C!m} \,\}) \quad \mathbb{L}_{\mathsf{B}} = \mathsf{pref}(\{\, \mathsf{A\,B?m \cdot A\,B?m} \,\}) \quad \mathbb{L}_{\mathsf{C}} = \mathsf{pref}(\{\, \mathsf{A\,C?m} \,\})$$

and whose synchronous semantics is

$$[\![S]\!] = \mathsf{pref}(\{\, \mathsf{A{\to}B{:}m \cdot A{\to}B{:}m},\ \mathsf{A{\to}B{:}m \cdot A{\to}C{:}m} \,\})$$

S is lock-free, but not strongly lock-free. In fact, for $\mathsf{A{\to}B{:}m}$ and B, we have that $\mathsf{A{\to}B{:}m \cdot A{\to}C{:}m}$ is maximal, but $(\mathsf{A{\to}C{:}m})\downarrow_{\mathsf{B}} = \varepsilon$.

$\boxed{\mathsf{LF} \implies \mathsf{DF}}$ Let us assume S to be lock-free. By contradiction let us assume S not to be deadlock-free. Then there is a finite and maximal word $w \in [\![S]\!]$ and a participant A such that $w\downarrow_{\mathsf{A}}$ is not maximal in $S(\mathsf{A})$. Since S is lock-free, by definition, there is a word w' such that $ww' \in [\![S]\!]$ and $w'\downarrow_{\mathsf{A}} \neq \varepsilon$. Hence $w' \neq \varepsilon$ and therefore w is not maximal in S, contrary to our assumption.

$\boxed{\mathsf{DF} \implies\!\!\!\!/\ \ \mathsf{LF}}$ Let us take

$$\mathbb{L}_{\mathsf{A}} = \mathsf{pref}(\{\, (\mathsf{A\,B!m})^{\omega} \,\}) \quad \mathbb{L}_{\mathsf{B}} = \mathsf{pref}(\{\, (\mathsf{A\,B?m})^{\omega} \,\}) \quad \mathbb{L}_{\mathsf{C}} = \mathsf{pref}(\{\, \mathsf{A\,C?m} \,\})$$

and consider the communicating system

$$S = (\mathbb{L}_{\mathsf{X}})_{\mathsf{X} \in \{\mathsf{A},\mathsf{B},\mathsf{C}\}} \quad \text{whose synchronous semantics is} \quad [\![S]\!] = \mathsf{pref}(\{\, (\mathsf{A{\to}B{:}m})^{\omega} \,\})$$

It is immediate to check that S is vacuously deadlock free, since there is no finite maximal word in S. However, S is not lock-free. It is enough to take $w = \varepsilon$, which is finite in $[\![S]\!]$ and such that $w\downarrow_{\mathsf{C}} = \varepsilon$ is not maximal in $\mathbb{L}_{\mathsf{C}}$. However there is no w' such that $w \cdot w' \in [\![S]\!]$ and $w'\downarrow_{\mathsf{C}} \neq \varepsilon$.

$\boxed{\mathsf{DF} \implies\!\!\!\!/\ \ \mathsf{SF}}$ Let

$$
\begin{aligned}
\mathbb{L}_{\mathsf{A}} &= \mathsf{pref}(\{\, (\mathsf{A\,B!m})^{\omega},\ \mathsf{A\,B!m \cdot (A\,C!m)}^{\omega} \,\}) \\
\mathbb{L}_{\mathsf{B}} &= \mathsf{pref}(\{\, (\mathsf{A\,B?m})^{\omega} \,\}) \\
\mathbb{L}_{\mathsf{C}} &= \mathsf{pref}(\{\, (\mathsf{A\,C?m})^{\omega} \,\})
\end{aligned}
$$

and consider the communicating system

$$S = (\mathbb{L}_{\mathsf{X}})_{\mathsf{X} \in \{\mathsf{A},\mathsf{B},\mathsf{C}\}}$$

whose synchronous semantics is

$$[\![S]\!] = \mathsf{pref}(\{\, (\mathsf{A{\to}B{:}m})^{\omega},\ \mathsf{A{\to}B{:}m \cdot (A{\to}C{:}m)}^{\omega} \,\})$$

It is immediate to check that S is vacuously deadlock-free, since there is no finite maximal word in S. However, S is not starvation-free. It is enough to take $w = \mathsf{A{\to}B{:}m}$, which is finite in $[\![S]\!]$ and also $w\downarrow_{\mathsf{C}} = \varepsilon$ is not maximal in $\mathbb{L}_{\mathsf{C}}$, but for $w' = (\mathsf{A{\to}B{:}m})^{\omega}$ we have that $w \cdot w' \in [\![S]\!]$ and $w'\downarrow_{\mathsf{C}} = \varepsilon$.

$\boxed{\mathsf{SF} \not\Longrightarrow \mathsf{DF}}$ Let

$$\mathbb{L}_\mathsf{A} = \{\,\varepsilon,\ \mathsf{A\,B!m}\,\} \quad \mathbb{L}_\mathsf{B} = \{\,\varepsilon,\ \mathsf{A\,B?m},\ \mathsf{B\,C!m}\,\} \quad \mathbb{L}_\mathsf{C} = \{\,\varepsilon,\ \mathsf{B\,C?m}\,\}$$

and consider the communicating system

$$S = (\mathbb{L}_\mathsf{X})_{\mathsf{X} \in \{\,\mathsf{A,B,C}\,\}} \quad \text{whose synchronous semantics is} \quad [\![S]\!] = \{\,\varepsilon,\ \mathsf{A{\to}B{:}m},\ \mathsf{B{\to}C{:}m}\,\}$$

It is immediate to check that S is vacuously starvation free, since there is no infinite word in $[\![S]\!]$. However, S is not deadlock free. It is enough to take $w = \mathsf{A{\to}B{:}m}$, which is finite and maximal in $[\![S]\!]$ but $(\mathsf{A{\to}B{:}m}){\downarrow}_\mathsf{C} = \varepsilon$ is not maximal in $\mathbb{L}_\mathsf{C}$.

$\boxed{\mathsf{LF} \not\Longrightarrow \mathsf{HA}}$ Let

$$\mathbb{L}_\mathsf{A} = \mathsf{pref}(\{\,\mathsf{B\,A?m{\cdot}B\,A?m},\ \mathsf{B\,A?m{\cdot}A\,B!y}\,\}) \qquad \text{and} \qquad \mathbb{L}_\mathsf{B} = \mathsf{pref}(\{\,\mathsf{B\,A!m{\cdot}B\,A!m}\,\})$$

and consider the communicating system

$$S = (\mathbb{L}_\mathsf{X})_{\mathsf{X} \in \{\,\mathsf{A,B}\,\}} \quad \text{whose synchronous semantics is} \quad [\![S]\!] = \mathsf{pref}(\{\,\mathsf{B{\to}A{:}m{\cdot}B{\to}A{:}m}\,\})$$

It is easy to check that S is lock-free. However, S is not harmonic, since $\mathsf{B\,A?m \cdot A\,B!y} \in S(\mathsf{A})$ and there is no $w \in [\![S]\!]$ such that $w{\downarrow}_\mathsf{A} = \mathsf{B\,A?m{\cdot}A\,B!y}$.

$\boxed{\mathsf{HA} \not\Longrightarrow \mathsf{LF}}$ Let us consider the system $S = (\mathbb{L}_\mathsf{X})_{\mathsf{X} \in \{\,\mathsf{A,B,C}\,\}}$ where

$$\mathbb{L}_\mathsf{A} = \mathsf{pref}(\{\,\mathsf{C\,A?m{\cdot}A\,C!m}\,\})$$
$$\mathbb{L}_\mathsf{B} = \mathsf{pref}(\{\,\mathsf{B\,C!m}\,\})$$
$$\mathbb{L}_\mathsf{C} = \mathsf{pref}(\{\,\mathsf{C\,A!m},\ \mathsf{B\,C?m{\cdot}C\,A!m{\cdot}A\,C?m}\,\})$$

It is easy to check that

$$[\![S]\!] = \mathsf{pref}(\{\,\mathsf{C{\to}A{:}m},\ \mathsf{B{\to}C{:}m{\cdot}C{\to}A{:}m{\cdot}A{\to}C{:}m}\,\})$$

and that S is harmonic. However, S is not lock-free. In fact, by taking $w = \mathsf{C{\to}A{:}m} \in [\![S]\!]$, we have that $w{\downarrow}_\mathsf{A}$ is not maximal in $\mathcal{L}_\mathsf{A}$, since $\mathsf{C\,A?m{\cdot}A\,C!m} = w{\downarrow}_\mathsf{A}{\cdot}A\,C!m \in \mathcal{L}_\mathsf{A}$. However, there is no word w' such that $w{\cdot}w' \in [\![S]\!]$ and $w'{\downarrow}_\mathsf{A} \neq \varepsilon$.

$\boxed{\mathsf{SF} \not\Longrightarrow \mathsf{LF}}$ Immediate, since otherwise, by $\mathsf{LF} \Longrightarrow \mathsf{DF}$ proved above, we would get SF to imply DF, which we showed not to hold.

$\boxed{\mathsf{LF} \not\Longrightarrow \mathsf{SF}}$ Let us consider $S = (\mathcal{L}_\mathsf{X})_{\mathsf{X} \in \{\,\mathsf{A,B,C}\,\}}$ where

$$\mathbb{L}_\mathsf{A} = \mathsf{pref}(\{\,(\mathsf{A\,B!m})^n{\cdot}\mathsf{A\,C!m}{\cdot}(\mathsf{A\,B!m})^\omega \mid n \in \mathbb{N}\,\} \cup \{\,(\mathsf{A\,B!m})^\omega\,\})$$
$$\mathbb{L}_\mathsf{B} = \mathsf{pref}(\{\,(\mathsf{A\,B?m})^\omega\,\})$$
$$\mathbb{L}_\mathsf{C} = \mathsf{pref}(\{\,\mathsf{A\,C?m}\,\})$$

It is easy to check that

$$[\![S]\!] = \mathsf{pref}(\{\,(\mathsf{A{\to}B{:}m})^n{\cdot}\mathsf{A{\to}C{:}m}{\cdot}(\mathsf{A{\to}B{:}m})^\omega \mid n \in \mathbb{N}\,\} \cup \{\,(\mathsf{A{\to}B{:}m})^\omega\,\})$$

It is not difficult also to check that S is lock-free. However, it is not starvation-free. In fact, by taking the non maximal word $w = \varepsilon$, we have that for the infinite word $w' = (\mathsf{A{\to}B{:}m})^\omega$ and for the participant C we have that $w{\downarrow}_\mathsf{C}$ is non maximal and $ww' \in [\![S]\!]$. However $w'{\downarrow}_\mathsf{C} = \varepsilon$.

$\boxed{\mathsf{SF} \not\Longrightarrow \mathsf{HA}}$ Let

$$\mathbb{L}_{\mathsf{A}} = \mathsf{pref}(\{\, \mathsf{B\,A?m}\cdot\mathsf{B\,A?m},\ \mathsf{B\,A?m}\cdot\mathsf{A\,B!y} \,\}) \qquad \text{and} \qquad \mathbb{L}_{\mathsf{B}} = \mathsf{pref}(\{\, \mathsf{B\,A!m}\cdot\mathsf{B\,A!m} \,\})$$

and consider the communicating system

$$S = (\mathbb{L}_{\mathsf{X}})_{\mathsf{X}\in\{\,\mathsf{A,B}\,\}} \quad \text{whose synchronous semantics is} \quad [\![S]\!] = \mathsf{pref}(\{\, \mathsf{B}{\to}\mathsf{A:m}\cdot\mathsf{B}{\to}\mathsf{A:m} \,\})$$

S is trivially starvation-free, since it contains no infinite word. However, S is not harmonic, since $\mathsf{B\,A?m}\cdot\mathsf{A\,B!y} \in S(\mathsf{A})$ and there is no $w \in [\![S]\!]$ such that $w{\downarrow}_{\mathsf{A}} = \mathsf{B\,A?m}\cdot\mathsf{A\,B!y}$.

$\boxed{\mathsf{HA} \not\Longrightarrow \mathsf{SF}}$ Let us consider $S = (\mathcal{L}_{\mathsf{X}})_{\mathsf{X}\in\{\,\mathsf{A,B,C}\,\}}$ where

$$\mathbb{L}_{\mathsf{A}} = \mathsf{pref}(\{\, (\mathsf{A\,B!m})^n\cdot\mathsf{A\,C!m}\cdot(\mathsf{A\,B!m})^\omega \mid n \in \mathbb{N} \,\} \cup \{\, (\mathsf{A\,B!m})^\omega \,\})$$
$$\mathbb{L}_{\mathsf{B}} = \mathsf{pref}(\{\, (\mathsf{A\,B?m})^\omega \,\})$$
$$\mathbb{L}_{\mathsf{C}} = \mathsf{pref}(\{\, \mathsf{A\,C?m} \,\})$$

It is easy to check that

$$[\![S]\!] = \mathsf{pref}(\{\, (\mathsf{A}{\to}\mathsf{B:m})^n\cdot\mathsf{A}{\to}\mathsf{C:m}\cdot(\mathsf{A}{\to}\mathsf{B:m})^\omega \mid n \in \mathbb{N} \,\} \cup \{\, (\mathsf{A}{\to}\mathsf{B:m})^\omega \,\})$$

It is not difficult also to check that S is harmonic. However, it is not starvation-free. In fact, by taking the non maximal word $w = \varepsilon$, we have that for the infinite word $w' = (\mathsf{A}{\to}\mathsf{B:m})^\omega$ and for the participant C we have that $w{\downarrow}_{\mathsf{C}}$ is non maximal and $ww' \in [\![S]\!]$. However $w'{\downarrow}_{\mathsf{C}} = \varepsilon$.

$\boxed{\mathsf{DF} \not\Longrightarrow \mathsf{HA}}$ Let

$$\mathbb{L}_{\mathsf{A}} = \{\, \varepsilon,\ \mathsf{B\,A?m},\ \mathsf{A\,B!y} \,\} \qquad \text{and} \qquad \mathbb{L}_{\mathsf{B}} = \{\, \varepsilon,\ \mathsf{B\,A!m} \,\}$$

and consider the communicating system

$$S = (\mathbb{L}_{\mathsf{X}})_{\mathsf{X}\in\{\,\mathsf{A,B}\,\}} \quad \text{whose synchronous semantics is} \quad [\![S]\!] = \{\, \varepsilon,\ \mathsf{B}{\to}\mathsf{A:m} \,\}$$

S is deadlock-free, since the only finite maximal word is $\mathsf{B}{\to}\mathsf{A:m}$ whose projection on A and B are both maximal. However, S is not harmonic, since $\mathsf{A\,B!y} \in S(\mathsf{A})$ and there is no $w \in [\![S]\!]$ such that $w{\downarrow}_{\mathsf{A}} = \mathsf{A\,B!y}$.

$\boxed{\mathsf{HA} \not\Longrightarrow \mathsf{DF}}$ Let

$$\mathbb{L}_{\mathsf{A}} = \{\, \varepsilon,\ \mathsf{A\,B!m},\ \mathsf{A\,C!m} \,\} \qquad \text{and} \qquad \mathbb{L}_{\mathsf{B}} = \{\, \varepsilon,\ \mathsf{A\,B?m} \,\} \qquad \text{and} \qquad \mathbb{L}_{\mathsf{C}} = \{\, \varepsilon,\ \mathsf{A\,C?m} \,\}$$

and consider the communicating system

$$S = (\mathbb{L}_{\mathsf{X}})_{\mathsf{X}\in\{\,\mathsf{A,B,C}\,\}} \quad \text{whose synchronous semantics is} \quad [\![S]\!] = \{\, \varepsilon,\ \mathsf{A}{\to}\mathsf{B:m},\ \mathsf{A}{\to}\mathsf{C:m} \,\}$$

It is easy to check that S is harmonic. However, S is not deadlock-free, since for the maximal and finite word $\mathsf{A}{\to}\mathsf{B:m} \in S$ we have that $w{\downarrow}_{\mathsf{C}} (= \varepsilon)$ is not maximal in $\mathbb{L}_{\mathsf{C}}$.

$\boxed{\mathsf{DF} \wedge \mathsf{SF} \Longrightarrow \mathsf{SLF}}$ In order to show SLF, let $w \in [\![S]\!]$ be finite and let $w{\downarrow}_{\mathsf{A}}$ be non maximal in $S(\mathsf{A})$ for $\mathsf{A} \in \mathcal{P}$. Besides, let w' be such that ww' is maximal in $[\![S]\!]$. We have to show that $w'{\downarrow}_{\mathsf{A}} \neq \varepsilon$. We distinguish two cases:

w' **is finite:** Since S is DF we can infer that $w' \neq \varepsilon$ and $w'{\downarrow}_{\mathsf{A}} \neq \varepsilon$.
w' **is infinite:** We get $w'{\downarrow}_{\mathsf{A}} \neq \varepsilon$ immediately by SF.

$\boxed{\mathsf{SLF} \implies \mathsf{DF} \wedge \mathsf{SF}}$ To show DF by contradiction, let us assume to have $w \in \llbracket S \rrbracket$ finite and maximal and such that $w\downarrow_{\mathsf{A}}$ is non maximal in $S(\mathsf{A})$ for $\mathsf{A} \in \mathcal{P}$. By SLF we have that $w'\downarrow_{\mathsf{A}} \neq \varepsilon$ for each w' such that ww' is maximal in $\llbracket S \rrbracket$. Since we assumed w to be maximal, the only possible w' is ε, contradicting $w'\downarrow_{\mathsf{A}} \neq \varepsilon$. To show SF by contradiction, let us assume to have $w \in \llbracket S \rrbracket$ finite and such that $w\downarrow_{\mathsf{A}}$ is non maximal in $S(\mathsf{A})$ for $\mathsf{A} \in \mathcal{P}$. Besides, let us assume that there exists an infinite w' such that $ww' \in \llbracket S \rrbracket$ and $w'\downarrow_{\mathsf{A}} = \varepsilon$. Since w' is infinite, ww' is trivially maximal in $\llbracket S \rrbracket$ and we get a contradiction with SLF.   □

5. Communication Properties by Construction

Harmonicity (cf. Definition 4.1) is the only property guaranteed on any system obtained via projection. This is a simple consequence of Corollary 3.2.

Corollary 5.1. *If $\mathcal{L}$ is a g-language then $\mathcal{L}\downarrow$ is harmonic.*

Proof. By Corollary 3.2, $\mathcal{L} \subseteq \llbracket \mathcal{L}\downarrow \rrbracket$. Now, by monotonicity of projection, we get $\mathcal{L}\downarrow \subseteq \llbracket \mathcal{L}\downarrow \rrbracket\downarrow$, that is harmonicity of $\mathcal{L}\downarrow$.   □

To ensure the other properties on a system $\mathcal{L}\downarrow$ we need to require some conditions on the g-language $\mathcal{L}$. Basically, we will strengthen CUI which is too weak. For instance, $\mathsf{cui}(\mathcal{L})$ does imply neither deadlock-freedom nor lock-freedom for $\mathcal{L}\downarrow$.

Example 5.2 (CUI $\not\Longrightarrow$ DF, LF). It is easy to check that $\mathsf{cui}(\mathcal{L})$ holds for the g-language $\mathcal{L} = \mathsf{pref}(\{ w, w' \})$ where

$$w = \mathsf{A{\rightarrow}C{:}l{\cdot}A{\rightarrow}B{:}m{\cdot}A{\rightarrow}C{:}m} \qquad \text{and} \qquad w' = \mathsf{A{\rightarrow}C{:}r{\cdot}A{\rightarrow}B{:}m{\cdot}B{\rightarrow}C{:}m}$$

Informally, $\mathcal{L}$ is CUI because C can ascertain which of its last actions to execute from the first input. So, Corollary 3.2 and Theorem 3.7 ensure that $\mathcal{L} = \llbracket \mathcal{L}\downarrow \rrbracket$. However, $\mathcal{L}\downarrow$ is not deadlock-free. In particular, $w \in \mathcal{L} = \llbracket \mathcal{L}\downarrow \rrbracket$ is a deadlock since it is a finite maximal word whose projection on B, namely $w\downarrow_{\mathsf{B}} = \mathsf{A\,B?m}$, is not maximal in $\mathcal{L}\downarrow_{\mathsf{B}}$ because $w'\downarrow_{\mathsf{B}} = \mathsf{A\,B?m{\cdot}B\,C!m} \in \mathcal{L}\downarrow_{\mathsf{B}}$. $\mathcal{L}\downarrow$ is non lock-free either by Proposition 4.6.   ◇

In many models (cf. [HLV$^+$16]) in order to ensure, besides other properties, also the correctness of $\mathcal{L}\downarrow$, a condition called *well-branchedness* is required. We identify a notion weaker than well-branchedness, which by analogy we dub *branch-awareness* (BA for short).

Definition 5.3 (Branch-awareness). A participant X *distinguishes* two g-words $w_1, w_2 \in \Sigma_{\mathrm{int}}^{\infty}$ if

$$w_1\downarrow_{\mathsf{X}} \neq w_2\downarrow_{\mathsf{X}} \quad \text{and} \quad w_1\downarrow_{\mathsf{X}} \not\prec w_2\downarrow_{\mathsf{X}} \quad \text{and} \quad w_2\downarrow_{\mathsf{X}} \not\prec w_1\downarrow_{\mathsf{X}}.$$

A g-language $\mathcal{L}$ on $\mathcal{P}$ is *branch-aware* if each $\mathsf{X} \in \mathcal{P}$ distinguishes all maximal words in $\mathcal{L}$ whose projections on X differ.

Example 5.4. The language $\mathcal{L} = \mathsf{pref}(\{ w, w' \})$ with $w = \mathsf{A{\rightarrow}C{:}l{\cdot}A{\rightarrow}B{:}m{\cdot}A{\rightarrow}C{:}m}$ and $w' = \mathsf{A{\rightarrow}C{:}r{\cdot}A{\rightarrow}B{:}m{\cdot}B{\rightarrow}C{:}m}$ from Example 5.2 is not branch-aware, since $w\downarrow_{\mathsf{B}} = \mathsf{A\,B?m}$ and $w'\downarrow_{\mathsf{B}} = \mathsf{A\,B?m{\cdot}B\,C!m}$, hence $w\downarrow_{\mathsf{B}} \neq w'\downarrow_{\mathsf{B}}$ but $w\downarrow_{\mathsf{B}} \prec w'\downarrow_{\mathsf{B}}$.   ◇

Condition $w_1\downarrow_{\mathsf{X}} \neq w_2\downarrow_{\mathsf{X}}$ in Definition 5.3 is not strictly needed to define BA, but it makes the notion of 'distinguishes' more intuitive. Equivalently, as shown in Proposition 5.5 below, a participant X distinguishes two branches if, after a common prefix, X is actively involved in both branches, performing different interactions.

Proposition 5.5. *Participant* X *distinguishes two g-words* $w_1, w_2 \in \Sigma_{int}^{\infty}$ *iff there are* $w_1' \cdot \alpha_1 \preceq w_1$ *and* $w_2' \cdot \alpha_2 \preceq w_2$ *such that* $w_1' \downarrow_{\mathsf{X}} = w_2' \downarrow_{\mathsf{X}}$ *and* $\alpha_1 \downarrow_{\mathsf{X}} \neq \alpha_2 \downarrow_{\mathsf{X}}$.

Proof. Trivial. □

The notions of well-branchedness in the literature [HLV+16] additionally impose that $\alpha_1 \downarrow_{\mathsf{X}}$ and $\alpha_2 \downarrow_{\mathsf{X}}$ in the above proposition are input actions, but for a (unique) participant (a.k.a. the *selector*) which is required to have different outputs. In our case the notion of selector corresponds to the "first" participant that distinguishes two words. Also in our case a selector must be involved in each branch but, due to the perfect symmetry of input and output actions in synchronous communications, its involvement can happen through input or output actions. This is illustrated by the following example.

Example 5.6 (Selector and input actions)**.** Consider the words

$$ w = \mathsf{A} \rightarrow \mathsf{B{:}m} \cdot \mathsf{A} \rightarrow \mathsf{C{:}m} \qquad \text{and} \qquad w' = \mathsf{B} \rightarrow \mathsf{C{:}m} \cdot \mathsf{C} \rightarrow \mathsf{A{:}m} $$

The participant B immediately distinguishes w and w' via its first actions, that is the input from A and the output to C. Hence, B is the selector of the branch made of w and w'. Notice that A and C also distinguish these two words, however this happens "later". Finally, observe that the same would hold if we replace $\mathsf{B} \rightarrow \mathsf{C{:}m}$ with $\mathsf{C} \rightarrow \mathsf{B{:}m}$ in w'. ◇

In our theory, BA is not needed for correctness, but it is nevertheless useful to prove the communication properties presented in Section 4.

Theorem 5.7 (Consequences of BA)**.** *Let* $\mathcal{L}$ *be a branch-aware and CUI sc-language. Then* $\mathcal{L}\downarrow$ *satisfies all the properties in Definitions 4.1 to 4.5.*

Proof. Let $\mathcal{L}$ be a branch-aware sc-language such that $\mathsf{cui}(\mathcal{L})$ holds. We prove the properties separately.

| Harmonicity | Immediate by Corollary 5.1.

| Lock-freedom | By contradiction, let us assume $\mathcal{L}\downarrow$ not to be lock-free. By Definition 4.2 and $\mathsf{cui}(\mathcal{L})$, it follows that there exist a participant $\mathsf{A} \in \mathcal{P}$ and a finite g-word $w \in \mathcal{L}$ such that

- $w \downarrow_{\mathsf{A}}$ is not maximal in $\mathcal{L}\downarrow_{\mathsf{A}}$;
- for all w' if $w \cdot w' \in \mathcal{L}$ then $w' \downarrow_{\mathsf{A}} = \varepsilon$.

By Corollary 5.1, $\mathcal{L}\downarrow$ is harmonic. Hence, by the above and $\mathsf{cui}(\mathcal{L})$, there exists $w'' \in \mathcal{L}$ such that

- $w'' \neq w$;
- $w'' \downarrow_{\mathsf{A}} = w \downarrow_{\mathsf{A}}$;
- there is $\hat{w}$ such that $w'' \cdot \hat{w} \in \mathcal{L}$ and $\hat{w} \downarrow_{\mathsf{A}} \neq \varepsilon$.

This means that, by taking a maximal extension of w and a maximal extension of $w'' \hat{w}$, we would get two non branch-aware words in $\mathcal{L}$, contradicting our hypothesis of $\mathcal{L}$ being branch-aware.

| Deadlock-freedom | Immediate by Proposition 4.6.

| Starvation-freedom | By contradiction, let us assume $\mathcal{L}\downarrow$ not to be starvation-free. By Definition 4.4 and $\mathsf{cui}(\mathcal{L})$, it follows that there exist a participant $\mathsf{A} \in \mathcal{P}$ and a finite g-word $w \in \mathcal{L}$ such that

- $w \downarrow_{\mathsf{A}}$ is not maximal in $\mathcal{L}\downarrow_{\mathsf{A}}$;
- there is an infinite word w' such that $w \cdot w' \in \mathcal{L}$ and $w' \downarrow_{\mathsf{A}} = \varepsilon$.

Now, by harmonicity of $\mathcal{L}{\downarrow}$ (Corollary 5.1), non maximality of $w{\downarrow}_A$ and by $\mathsf{cui}(\mathcal{L})$, it follows that there exist $w'' \in \mathcal{L}$ and a finite word $\hat{w}$ such that

- $w''\hat{w} \in \mathcal{L}$;
- $w''{\downarrow}_A = w{\downarrow}_A$;
- $\hat{w}{\downarrow}_A \neq \varepsilon$.

The above means that by taking ww' and any maximal extension of $w''\hat{w}$ we would get two maximal words in $\mathcal{L}$ which A cannot distinguish, so contradicting our hypothesis of $\mathcal{L}$ being branch-aware.

$\boxed{\text{Strong lock-freedom}}$ Immediate since $\mathsf{SLF} = \mathsf{SF} \wedge \mathsf{DF}$. $\hfill\square$

Example 5.8 (Task dispatching and branch-awareness)**.** In order to show that the g-language $\mathcal{L}$ in Example 3.12 is branch-aware, we first notice that each maximal word in $\mathcal{L}$ ends with the interactions $\mathsf{S{\to}D{:}s{\cdot}S{\to}H{:}s}$. If $\mathcal{L}$ were not branch-aware, there should be two maximal words $w{\cdot}\mathsf{S{\to}D{:}s{\cdot}S{\to}H{:}s}$ and $w'{\cdot}\mathsf{S{\to}D{:}s{\cdot}S{\to}H{:}s}$ and a participant $\mathsf{X} \in \mathsf{ptp}(\mathcal{L})$ such that $(w{\cdot}\mathsf{S{\to}D{:}s{\cdot}S{\to}H{:}s}){\downarrow}_{\mathsf{X}} \prec (w'{\cdot}\mathsf{S{\to}D{:}s{\cdot}S{\to}H{:}s}){\downarrow}_{\mathsf{X}}$. This is impossible, since w and w' are both generated by the non terminal symbol L' and hence cannot contain the message s. $\hfill\diamond$

Proposition 4.6 refines as follows when restricting to projections of g-languages.

Proposition 5.9. *When considering only systems which are projections of g-languages the following relations hold among the properties in Definitions 4.1 to 4.5*
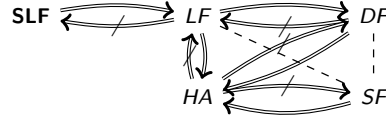
where implication does not hold in any direction between properties connected by dashed lines. Moreover, $\mathsf{DF} \wedge \mathsf{SF} \Leftrightarrow \mathsf{SLF}$*.*

Proof.

$\boxed{\text{SLF} \implies \text{LF}}$ The proof of Proposition 4.6 applies.

$\boxed{\text{LF} \centernot\implies \text{SLF}}$ The proof of Proposition 4.6 applies.

$\boxed{\text{LF} \implies \text{DF}}$ The proof of Proposition 4.6 applies.

$\boxed{\text{DF} \centernot\implies \text{LF}}$ Let us take the language

$$\mathcal{L} = \mathsf{pref}(\{\, \mathsf{A{\to}C{:}n{\cdot}A{\to}B{:}m{\cdot}A{\to}C{:}m}\big(\mathsf{A{\to}C{:}m}\big)^{\omega},\ \mathsf{A{\to}C{:}m{\cdot}A{\to}B{:}m{\cdot}B{\to}C{:}m}\big(\mathsf{A{\to}C{:}m}\big)^{\omega} \,\})$$

The system $\mathcal{L}{\downarrow}$ is trivially DF since its maximal words are all infinite, while it is not LF since the non maximal word $\mathsf{A{\to}C{:}n{\cdot}A{\to}B{:}m} \in [\![\mathcal{L}{\downarrow}]\!]$ is such that its projection on B is not maximal in $\mathcal{L}{\downarrow}_B$ and no continuation in $\mathcal{L}$ of such word contains interactions involving B; in fact for the (only) g-word $\hat{w} = \mathsf{A{\to}C{:}m}\big(\mathsf{A{\to}C{:}m}\big)^{\omega}$ such that $\mathsf{A{\to}C{:}n{\cdot}A{\to}B{:}m}{\cdot}\hat{w} \in [\![\mathcal{L}{\downarrow}]\!]$, we have $\hat{w}{\downarrow}_B = \varepsilon$.

$\boxed{\text{DF} \centernot\implies \text{SF}}$ The counterexample of Proposition 4.6 applies.

$\boxed{\text{SF} \centernot\implies \text{DF}}$ The counterexample of Proposition 4.6 applies.

$\boxed{\text{LF} \implies \text{HA}}$ Trivial since HA always holds thanks to Corollary 5.1.

$\boxed{\text{HA} \centernot\implies \text{LF}}$ Trivial since HA always holds thanks to Corollary 5.1, while this is not the case for LF thanks to Example 5.2.

$\boxed{\text{SF} \centernot\implies \text{LF}}$ The proof of Proposition 4.6 applies.

$\boxed{\text{LF} \centernot\implies \text{SF}}$ The counterexample of Proposition 4.6 applies.

$\boxed{\text{SF} \implies \text{HA}}$ Trivial since HA always holds thanks to Corollary 5.1.

$\boxed{\text{HA} \centernot\implies \text{SF}}$ The counterexample of Proposition 4.6 applies.

$\boxed{\text{DF} \implies \text{HA}}$ Trivial since HA always holds thanks to Corollary 5.1.
$\boxed{\text{HA} \not\Rightarrow \text{DF}}$ Trivial since HA always holds thanks to Corollary 5.1, while this is not the case for DF thanks to Example 5.2.
$\boxed{\text{DF} \wedge \text{SF} \implies \text{SLF}}$ The proof of Proposition 4.6 applies.
$\boxed{\text{SLF} \implies \text{DF} \wedge \text{SF}}$ The proof of Proposition 4.6 applies.   □

It is not difficult to show that branch-awareness actually characterises SLF for systems obtained by projecting CUI languages.

Proposition 5.10 (Branch-awareness characterises SLF). *A CUI g-language $\mathcal{L}$ is branch-aware iff $\mathcal{L}\downarrow$ is strongly lock-free.*

Proof. Necessity follows from Theorem 5.7 while for sufficiency we reason by contraposition. Assume $\mathcal{L}$ not to be branch-aware. Then, by Definition 5.3 there exist a participant X and two maximal words in $\mathcal{L}$ such that their projections on X differ and either $w_1\downarrow_{\mathsf{X}} \prec w_2\downarrow_{\mathsf{X}}$ or $w_2\downarrow_{\mathsf{X}} \prec w_1\downarrow_{\mathsf{X}}$. Let us consider the first case (the second is analogous). Let now w_1' and w_2' be the maximal prefixes (hence finite) of, respectively, w_1 and w_2 such that $w_1'\downarrow_{\mathsf{X}} = w_2'\downarrow_{\mathsf{X}}$. We distinguish the following two cases.

$\boxed{w_1' = w_1}$ In such a case we get that $\mathcal{L}\downarrow$ is not deadlock-free, since, for the finite maximal w we have that $w_1\downarrow_{\mathsf{X}}$ is not maximal. So, by Proposition 5.9, $\mathcal{L}\downarrow$ is not strongly lock-free as well.
$\boxed{w_1' \prec w_1}$ In such a case we get that w_1' is a non-maximal finite word such that $w_1 = w_1' \cdot w_1''$ is maximal, $w_1\downarrow_{\mathsf{X}}$ is non maximal and $w_1''\downarrow_{\mathsf{X}} = \varepsilon$, that is $\mathcal{L}\downarrow$ is not strongly lock-free.   □

6. CFSMs and Choreography Languages

Communicating finite-state machines (CFSMs) have been introduced in [BZ83] as a convenient model to analyse message-passing protocols. Basically, a CFSM is a finite-state automaton (FSA), defined below, whose transitions are actions.

Definition 6.1 (Finite state automaton (FSA)). A *finite state automaton* (FSA) is a tuple $A = \langle \mathcal{S}, q_0, \mathbf{\Lambda}, \rightarrow \rangle$ where
- $\mathcal{S}$ is a finite set of states (ranged over by $q, s, \ldots$) and $q_0 \in \mathcal{S}$ is the *initial state*;
- $\mathbf{\Lambda}$ is a finite set of labels (ranged over by $l, \lambda, \ldots$);
- $\rightarrow \subseteq \mathcal{S} \times \mathbf{\Lambda} \times \mathcal{S}$ is a set of transitions.

We use the usual notation $q_1 \xrightarrow{\lambda} q_2$ for the transition $(q_1, \lambda, q_2) \in \rightarrow$, and $q_1 \rightarrow q_2$ when there exists λ such that $q_1 \xrightarrow{\lambda} q_2$, as well as $\rightarrow^*$ for the reflexive and transitive closure of $\rightarrow$.
The set of *reachable states* of A is $\mathsf{R}(A) = \{\, q \mid q_0 \rightarrow^\star q \,\}$.

Remark 6.2. Our definition of FSA omits the set of *accepting* states since we consider only FSAs where each state is accepting.   ◇

Following the assumption in Remark 6.2, we define the language $\mathbf{L}(A)$ as the union of the finite words accepted by A in the classical sense and the infinite words accepted by A considered as Büchi automaton[2] where all states are accepting.

---

[2]A Büchi automaton accepts an infinite word z when it traverses infinitely often elements of the set of accepting states while consuming the symbols of z.

A CFSM is an FSA labelled in $\Sigma_{\text{act}} \cup \{\varepsilon\}$, where ε is not a symbol in Σ_{act} and it overloads the notation for the empty string to represent internal transitions of CFSMs as usual in automata theory.

Remark 6.3. FSAs, and consequently CFSMs, can be deterministic or not. Deterministic FSAs have no label ε, and transitions from the same state are pairwise different. Given a non-deterministic FSA one can build a deterministic FSA generating the same language. We will assume that each CFSM is deterministic since we are interested in languages and non-deterministic CFSMs can be determinised while preserving their language.

Notice that it is necessary to adopt finer notions of equivalence such as bisimilarity [Par81] in order to tell apart deterministic FSAs from non-deterministic ones. It could be interesting to investigate the possibility of including the notion of nondeterminism in FCLs in the future. ◇

A CFSM is *local* to a participant A (A-local for short) if all its transitions have subject A; we will consider communicating systems where the behaviour of each participant A is specified by an A-local CFSM. Formally, given a finite set $\mathcal{P} \subseteq \mathfrak{P}$ of participants, a *system of CFSMs* is a map $(M_A)_{A \in \mathcal{P}}$ assigning an A-local CFSM M_A to each participant $A \in \mathcal{P}$ such that any participant occurring in a label of a transition of M_A is in $\mathcal{P}$.

The synchronous behaviour of a system of CFSMs $(M_A)_{A \in \mathcal{P}}$ has been defined in [BLT20] as any FSA where states are maps assigning a state in M_A to each $A \in \mathcal{P}$ and transitions are labelled by interactions. Intuitively, given a configuration s, if M_A and M_B have respectively transitions $s(A) \xrightarrow{A\,B!m} q'_A$ and $s(B) \xrightarrow{A\,B?m} q'_B$ then $s \xrightarrow{A \to B:m} s[A \mapsto q'_A, B \mapsto q'_B]$, where $f[x \mapsto y]$ denotes the update of f on x with y. The next definition is a slightly different version of the one in [BLT20], where the semantics of a system of CFSMs **S** is the FSA $(\!|S|\!)$ defined below rather than its language as in our case.

Definition 6.4 (Synchronous semantics of systems of CFSMs). Let $\mathbf{S} = (M_A)_{A \in \mathcal{P}}$ be a system of CFSMs where $M_A = \langle \mathcal{S}_A, q_{0A}, \Sigma_{\text{act}}, \to_A \rangle$ for each participant $A \in \mathcal{P}$. A *synchronous configuration* for **S** is a map $s = (q_A)_{A \in \mathcal{P}}$ assigning a *local state* $q_A \in \mathcal{S}_A$ to each $A \in \mathcal{P}$.

The *synchronous semantics* of **S** is the g-language $\mathbf{L}(\!(\!|S|\!)\!)$ where $(\!|S|\!) = \langle \mathcal{S}, s_0, \Sigma_{\text{int}}, \to \rangle$ is defined as follows:

- $\mathcal{S}$ is the set of configurations of **S**, as defined above, and $s_0 : A \mapsto q_{0A}$ for each $A \in \mathcal{P}$ is the *initial* configuration of $\mathcal{S}$
- $\to$ is the set of transitions $s_1 \xrightarrow{A \to B:m} s_2$ such that
  - $s_1(A) \xrightarrow{A\,B!m} s_2(A)$ in M_A and $s_1(B) \xrightarrow{A\,B?m} s_2(B)$ in M_B, and
  - for all $X \in \mathcal{P} \setminus \{A, B\}$, $s_1(X) = s_2(X)$.

As we will see in Section 7, the synchronous semantics in Definition 6.4 is a *choreography automaton* [BLT20]. Note that this is not the case for the asynchronous semantics of communicating systems which is in general a transition system with infinitely many configurations.

An immediate relation between systems of CFSMs and our communicating systems is that, given a system of CFSMs $\mathbf{S} = (M_X)_{X \in \mathcal{P}}$, we can define the abstract system corresponding to **S** as $\hat{\mathbf{S}} = (\mathbf{L}(M_X))_{X \in \mathcal{P}}$. Unsurprisingly, the semantics of **S** and $\hat{\mathbf{S}}$ do coincide.

Proposition 6.5. *For all systems* $\mathbf{S} = (M_X)_{X \in \mathcal{P}}$ *of CFSMs,* $\mathbf{L}(\!(\!|S|\!)\!) = [\![\hat{\mathbf{S}}]\!]$.

Proof. The proof shows the commutativity of the following diagram

$$
\begin{array}{ccc}
\mathbb{S} & \xrightarrow{\;(\!|\_|\!)\;} & \mathfrak{C} \\
{\scriptstyle\hat{\_}}\big\downarrow & & \big\downarrow{\scriptstyle \mathbf{L}(\_)} \\
\mathfrak{L} & \xrightarrow[\;[\![\_]\!]\;]{} & \mathbb{G}
\end{array}
$$

where $\mathbb{S}$ is the set of systems of CFSMs, $\mathfrak{L}$ is the set of communicating systems (cf. Definition 2.6), $\mathfrak{C}$ is the set of c-automata, and $\mathbb{G}$ is the set of global languages.

Let $M_X = \langle \mathcal{S}_X, q_{0X}, \Sigma_{\mathrm{act}}, \to_X \rangle$ and let $(\!|\mathbf{S}|\!) = \langle \mathcal{S}, s_0, \Sigma_{\mathrm{int}}, \to \rangle$ where $\langle \mathcal{S}, s_0, \Sigma_{\mathrm{int}}, \to \rangle$ is as in Definition 6.4. We prove the equality by proving separately the following two inclusions, for all $\mathbf{S}$.

$\boxed{\mathbf{L}((\!|\mathbf{S}|\!)) \subseteq [\![\hat{\mathbf{S}}]\!]}$ By definition of $\hat{\mathbf{S}}$, it is enough to show that, for any $w \in \Sigma_{\mathrm{int}}^\infty$, if $w \in \mathbf{L}((\!|\mathbf{S}|\!))$, then for any $X \in \mathcal{P}$, we have that $w{\downarrow}_X \in \mathbf{L}(M_X)$. Let $w \in \mathbf{L}((\!|\mathbf{S}|\!))$ and proceed by coinduction on the paths of machine $(\!|\mathbf{S}|\!)$.

If $w = \varepsilon$ the thesis follows immediately. Otherwise, let $w = (\mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}){\cdot}w'$. By definition of recognised word, there is $s_1 \in \mathcal{S}$ such that $s_0 \xrightarrow{\mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}} s_1$ and w' is recognised by the automaton $\langle \mathcal{S}, s_1, \Sigma_{\mathrm{int}}, \to \rangle$.
Necessarily, by Definition 6.4,

(1) $s_0(\mathsf{A}) \xrightarrow{\mathsf{A}\,\mathsf{B}!\mathsf{m}}_\mathsf{A} s_1(\mathsf{A})$ and $s_0(\mathsf{B}) \xrightarrow{\mathsf{A}\,\mathsf{B}?\mathsf{m}}_\mathsf{B} s_1(\mathsf{B})$, and
(2) for all $X \neq \mathsf{A}, \mathsf{B}$, $s_1(X) = s_0(X)$.

It follows that $w' \in \mathbf{L}(\mathbf{S}')$ where $\mathbf{S}' = (\langle \mathcal{S}_X, s_1(X), \Sigma_{\mathrm{act}}, \to_X \rangle)_{X \in \{\mathsf{A},\mathsf{B}\}} \cup (M_X)_{X \in \mathcal{P}\setminus\{\mathsf{A},\mathsf{B}\}}$. The thesis hence follows by coinduction, since $w{\downarrow}_\mathsf{A} = (\mathsf{A}\,\mathsf{B}!\mathsf{m}){\cdot}w'{\downarrow}_\mathsf{A}$, $w{\downarrow}_\mathsf{B} = (\mathsf{A}\,\mathsf{B}?\mathsf{m}){\cdot}w'{\downarrow}_\mathsf{B}$ and, for each $X \in \mathcal{P} \setminus \{\mathsf{A}, \mathsf{B}\}$, $w{\downarrow}_X = w'{\downarrow}_X$.

$\boxed{[\![\hat{\mathbf{S}}]\!] \subseteq \mathbf{L}((\!|\mathbf{S}|\!))}$ For this case we have to prove that for all $w \in \Sigma_{\mathrm{int}}^\infty$, if $w \in [\![\hat{\mathbf{S}}]\!]$ then $w \in \mathbf{L}((\!|\mathbf{S}|\!))$. Let $w \in [\![\hat{\mathbf{S}}]\!]$ and proceed by coinduction. If $w = \varepsilon$ the thesis follows immediately. Otherwise, let $w = (\mathsf{A}{\to}\mathsf{B}{:}\mathsf{m}){\cdot}w'$. Since $w{\downarrow}_\mathsf{A} = (\mathsf{A}\,\mathsf{B}!\mathsf{m}){\cdot}w'{\downarrow}_\mathsf{A}$, $w{\downarrow}_\mathsf{B} = (\mathsf{A}\,\mathsf{B}?\mathsf{m}){\cdot}w'{\downarrow}_\mathsf{B}$ and, for each $X \in \mathcal{P} \setminus \{\mathsf{A}, \mathsf{B}\}$, $w{\downarrow}_X = w'{\downarrow}_X$, and since, for each $X \in \mathcal{P}$, we have $w{\downarrow}_X \in \mathbf{L}(M_X)$, it follows, by definition of recognised word and by Definition 2.9, that $w' \in [\![\hat{\mathbf{S}}']\!]$, with $\hat{\mathbf{S}}' = (\mathbf{L}(\langle \mathcal{S}_X, s_1(X), \Sigma_{\mathrm{act}}, \to_X \rangle))_{X \in \{\mathsf{A},\mathsf{B}\}} \cup (\mathbf{L}(M_X))_{X \in \mathcal{P}\setminus\{\mathsf{A},\mathsf{B}\}}$, where

(1) $s_0(\mathsf{A}) \xrightarrow{\mathsf{A}\,\mathsf{B}!\mathsf{m}}_\mathsf{A} s_1(\mathsf{A})$ and $s_0(\mathsf{B}) \xrightarrow{\mathsf{A}\,\mathsf{B}?\mathsf{m}}_\mathsf{B} s_1(\mathsf{B})$, and
(2) for all $X \neq \mathsf{A}, \mathsf{B}$, $s_1(X) = s_0(X)$.

The thesis hence follows by coinduction. $\qquad\square$

Notice that the proof of Proposition 6.5 does not require CFSMs to be deterministic; indeed, the result above holds for non-deterministic CFSMs too.

The communication properties of a system of CFSMs $\mathbf{S}$ on $\mathcal{P}$ considered in [BLT20] are liveness, lock-freedom, and deadlock-freedom. Intuitively

- $\mathbf{S}$ is *live* when each reachable configuration where a participant $\mathsf{A} \in \mathcal{P}$ can execute a communication has a continuation where A is involved;
- $\mathbf{S}$ is *lock-free* when in all computations starting from a reachable configuration where a participant $\mathsf{A} \in \mathcal{P}$ can execute, A is involved;

- **S** is *deadlock-free* if in none of its reachable configurations without outgoing transitions there exists $A \in \mathcal{P}$ willing to communicate.

The next definition formalises properties of systems of CFSMs.

Definition 6.6 (Communication properties of systems of CFSMs [BLT20])**.** Let $\mathbf{S} = (M_X)_{X \in \mathcal{P}}$ be a system of CFSMs.

- **Liveness**: **S** is *live* if for each configuration $s \in \mathsf{R}(\lVert\mathbf{S}\rVert)$ and each $A \in \mathcal{P}$ such that $s(A)$ has some outgoing transition in M_A, there exists a run of $\lVert\mathbf{S}\rVert$ from s including a transition involving A.
- **Lock freedom**: a configuration $s \in \mathsf{R}(\lVert\mathbf{S}\rVert)$ is a *lock* if there is $A \in \mathcal{P}$ with an outgoing transition from $s(A)$ in M_A and there exists a run of $\lVert\mathbf{S}\rVert$ starting from s, maximal with respect to prefix order and containing no transition involving A. System **S** is *lock-free* if for each $s \in \mathsf{R}(\lVert\mathbf{S}\rVert)$, s is not a lock.
- **Deadlock freedom**: a configuration $s \in \mathsf{R}(\lVert\mathbf{S}\rVert)$ is a *deadlock* if s has no outgoing transitions in $\lVert\mathbf{S}\rVert$, yet there exists $A \in \mathcal{P}$ such that $s(A)$ has an outgoing transition in M_A. System **S** is *deadlock-free* if for each $s \in \mathsf{R}(\lVert\mathbf{S}\rVert)$, s is not a deadlock.

It is the case that lock-freedom, strong lock-freedom, and deadlock-freedom of $\hat{\mathbf{S}}$ (in the sense of Definitions 4.3 to 4.5) respectively imply liveness, lock-freedom, and deadlock-freedom of **S** as stated in Proposition 6.8 below, which relies on the following lemma.

Lemma 6.7. *Let* $\mathbf{S} = (M_X)_{X \in \mathcal{P}}$ *be a system of CFSMs. If* $w \in \mathbf{L}(\lVert\mathbf{S}\rVert)$ *is recognised by a run of* $\lVert\mathbf{S}\rVert$ *from* s_0 *to* s *then, for each* $A \in \mathcal{P}$, $w{\downarrow}_A \in \mathbf{L}(M_A)$ *and* $w{\downarrow}_A$ *is recognised by a run of* M_A *from* $s_0(A)$ *to* $s(A)$.

Proof. By induction on the length of w using Definition 6.4. $\square$

Proposition 6.8. *For all systems of CFSMs* $\mathbf{S} = (M_X)_{X \in \mathcal{P}}$

  (i) $\hat{\mathbf{S}}$ *lock-free iff* **S** *live;*
 (ii) $\hat{\mathbf{S}}$ *strong lock-free iff* **S** *lock-free;*
(iii) $\hat{\mathbf{S}}$ *deadlock-free iff* **S** *deadlock-free.*

Proof. (i)
$\boxed{\Rightarrow}$ Let $\hat{\mathbf{S}}$ be lock-free. Following Definition 6.6, in order to show the liveness of **S**, let us consider $s \in \mathsf{R}(\lVert\mathbf{S}\rVert)$, and let $s(A) \xrightarrow{\mathsf{a}}$ be a transition in M_A (for a certain $A \in \mathcal{P}$). Let now w be the trace of a run of $\lVert\mathbf{S}\rVert$ from s_0 to s. By Proposition 6.5 $w \in \llbracket\hat{\mathbf{S}}\rrbracket$ and, by Lemma 6.7, $w{\downarrow}_A \in \mathbf{L}(M_A)$ and $w{\downarrow}_A$ is the trace of a run from $s_0(A)$ to $s(A)$ in M_A. Now, to prove that **S** is live, we have to show that there exists a run of $\lVert\mathbf{S}\rVert$ from s such that one of its transitions has a component transition from $s(A)$. We have that $w{\downarrow}_A{\cdot}\mathsf{a} \in \mathbf{L}(M_A)$. Hence, by lock-freedom of $\hat{\mathbf{S}}$, there is w' such that $w{\cdot}w' \in \llbracket\hat{\mathbf{S}}\rrbracket$ and $w'{\downarrow}_A \neq \varepsilon$. By Proposition 6.5 w' is also a run of $\lVert\mathbf{S}\rVert$ from s and, by Lemma 6.7 (considering the CFSMs $\mathbf{S}' = (M_X')_{X \in \mathcal{P}}$ where each M_X' is like M_X but with $s(X)$ as initial state), such a run has a transition from $s(A)$ as component transition.
$\boxed{\Leftarrow}$ By contraposition, let us assume $\hat{\mathbf{S}}$ not to be lock-free. Then there exists a participant A and a word $w \in \llbracket\hat{\mathbf{S}}\rrbracket$ such that

- $w{\downarrow}_A$ is not maximal in $\hat{\mathbf{S}}(A)$; and
- for all w', $w{\cdot}w' \in \llbracket\hat{\mathbf{S}}\rrbracket$ implies $w'{\downarrow}_A = \varepsilon$

Let s be the configuration in $(\!|\mathbf{S}|\!)$ reached by recognising w. Since $w{\downarrow_{\mathsf{A}}}$ is not maximal, we get, by Lemma 6.7 and determinism of the automata in $\mathbf{S}$, that $s(\mathsf{A})$ has at least an outgoing transition. If $\mathbf{S}$ were live, then there would be a run of $(\!|\mathbf{S}|\!)$ from s – and hence a word of the form $w{\cdot}w' \in \mathbf{L}((\!|\mathbf{S}|\!)) = [\![\hat{\mathbf{S}}]\!]$ – including a transition involving A, so contradicting the liveness of $\mathbf{S}$. So $\mathbf{S}$ is not live and we are done.

The proof of (ii) and (iii) are similar to (i). $\qquad\qquad\square$

Remark 6.9. Proposition 6.8 does not hold in case we consider non-determistic CFSMs. For instance, let us consider the following system of CFSMs. $\mathbf{S} = (M_{\mathsf{X}})_{\mathsf{X}\in\{\mathsf{A},\mathsf{B}\}}$ where
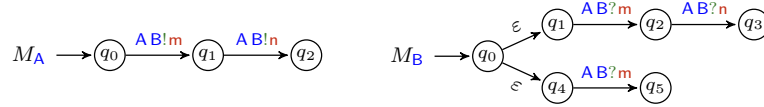
The corresponding communicating systems is $\hat{\mathbf{S}} = (\mathbb{L}_{\mathsf{X}})_{\mathsf{X}\in\{\mathsf{A},\mathsf{B}\}}$ where

$$\mathbb{L}_{\mathsf{A}} = \{\,\varepsilon,\ \mathsf{A\,B!m},\ \mathsf{A\,B!m{\cdot}A\,B!n}\,\} \qquad \mathbb{L}_{\mathsf{B}} = \{\,\varepsilon,\ \mathsf{A\,B?m},\ \mathsf{A\,B?m{\cdot}A\,B?n}\,\}$$

It is easy to check that $\hat{\mathbf{S}}$ is deadlock-free. However $\mathbf{S}$ is not so, since the system can reach the stuck configuration (q_1, q_5) and there is an outgoing transition from q_1 in M_{A}. Likewise, the communicating system made of M_{A} and either of the following machines
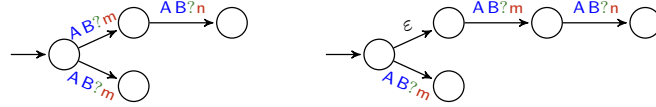
also reaches a deadlock configuration.

The FCL formalism, by abstracting from the notion of state, is in fact intrinsically deterministic.

7. Choreography Automata and Choreography Languages

We advocated *choreography automata* (c-automata) [BLT20] as an expressive and flexible model of global specifications. Essentially c-automata are finite-state automata (FSAs) whose transitions are labelled with interactions. This yields an immediate connection between g-languages and c-automata in terms of the languages the latter accept. This section explores such connection, as well as the connection between the projection operation on c-automata in [BLT20] and the projection on g-languages.

7.1. The g-languages of c-automata.

Choreography automata (c-automata for short, ranged over by $\mathbb{C}A$, $\mathbb{C}B$, etc.) are defined in [BLT20] as deterministic FSAs with labels in the set Σ_{int} of interactions. (Observe that the set of participants occurring in a c-automaton is necessarily finite.) We can see c-automata as a tool for specifying g-languages as shown by the next proposition.

Proposition 7.1. *Given a c-automaton* $\mathbb{C}A$, $\mathbf{L}(\mathbb{C}A)$ *is a continuous g-language.*

Proof. To show that $\mathbf{L}(\mathbb{C}A)$ is a g-language we need to show prefix closure. It follows from the fact that all the states are accepting: if a word w is in $\mathbf{L}(\mathbb{C}A)$ then any prefix of w can be generated by taking the corresponding prefix of the computation generating w.

We now need to show that the language is continuous. We need to show that $\mathbf{L}(\mathbb{C}A)$ contains an infinite word if it contains all its finite prefixes. Note that, thanks to determinism, words which are prefixes one of the other are generated by computations which are prefixes one of the other as well. Let w be an infinite word whose prefixes are in $\mathbf{L}(\mathbb{C}A)$. The infinite run obtained as the limit of the runs generating the prefixes of w generates w. This concludes the proof. $\qquad\square$

Interestingly, c-automata have an immediate relation with CFSMs [BZ83], introduced in the previous section, which can be adopted to model the local behaviour of distributed components. Indeed, the local behaviour of a participant of a c-automaton $\mathbb{C}A$ can be algorithmically obtained directly from $\mathbb{C}A$. Formally[3]

Definition 7.2 (Projection of c-automata [BLT20])**.** Let $\mathbb{C}A = \langle \mathcal{S}, q_0, \Sigma_{\text{int}}, \rightarrow \rangle$ be a c-automaton and A be a participant. The *projection of* $\mathbb{C}A$ *on* A is the CFSM $\mathbb{C}A{\downarrow}_A$ obtained by determinising up-to-language equivalence the *intermediate* automaton

$$\mathbf{A}_A = \langle \mathcal{S}, q_0, \Sigma_{\text{act}} \cup \{\, \varepsilon \,\}, \{\, q \xrightarrow{\lambda{\downarrow}_A} q' \mid q \xrightarrow{\lambda} q' \,\} \rangle$$

The *projection of* $\mathbb{C}A$, written $\mathbb{C}A{\downarrow}$, is the system of CFSMs $(\mathbb{C}A{\downarrow}_A)_{A \in \mathcal{P}}$.

The l-language of a projection of a c-automaton $\mathbb{C}A$ coincides with the projection of the language of $\mathbb{C}A$:

Proposition 7.3. *If* $\mathbb{C}A$ *is a c-automaton on* $\mathcal{P}$ *then for all* $X \in \mathcal{P}$, $\mathbf{L}(\mathbb{C}A){\downarrow}_X = \mathbf{L}(\mathbb{C}A{\downarrow}_X)$.

Proof. By definition of projection and since determinisation preserves the language, we have $\mathbf{L}(\mathbb{C}A{\downarrow}_X) = \mathbf{L}(\mathbf{A}_X)$, where $\mathbf{A}_X$ is the intermediate automaton (cf. Definition 7.2). Since $\mathbf{A}_X$ is obtained by taking the projection of every transition in $\mathbb{C}A$, $\mathbf{L}(\mathbf{A}_X) = \mathbf{L}(\mathbb{C}A){\downarrow}_X$. The thesis follows by transitivity. $\qquad\square$

Observing that $\mathbb{C}A{\downarrow}$ is ε-free, the LTS of $\mathbb{C}A{\downarrow}$ is a c-automaton and its language coincides with the g-language of the system $(\mathbf{L}(\mathbb{C}A{\downarrow}_X))_{X \in \mathcal{P}}$ thanks to Proposition 6.5.

The notion of well-formedness we provided in [BLT20] was aimed to guarantee correctness and completeness as well as the communication properties of projected systems (cf. Definition 6.6). We discovered later on that our notion of well-formedness in [BLT20] was flawed. In fact it does not correctly handle the interplay between concurrent transitions and choices. This is shown by the following example.

Example 7.4. The c-automaton $\mathbb{C}A$ below
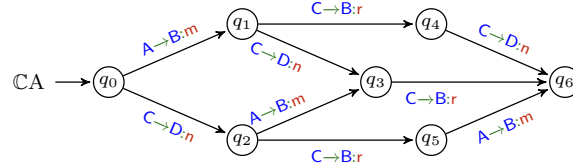
is well-formed according to [BLT20, Def. 4.12]. However, the system $\mathbb{C}A{\downarrow}$ admits the run C→B:r C→D:n which is not a word accepted by $\mathbb{C}A$. $\qquad\diamond$

___
[3]Overloading the projection operator of Definition 2.7 does not introduce confusion and avoids the introduction of further notation.

In our setting, the problem of the above $\mathbb{C}A$ is that $\mathbf{L}(\mathbb{C}A)$ is not CUI. In fact, by setting $w_1 = \mathsf{A{\to}B{:}m}$ and $w_2 = \mathsf{C{\to}D{:}n}$, we have that $w_1{\cdot}\mathsf{C{\to}B{:}r}, w_2{\cdot}\mathsf{C{\to}B{:}r} \in \mathbf{L}(\mathbb{C}A)$. Now, by taking $w = \varepsilon$, we have that $w_1{\downarrow}_\mathsf{C} = w{\downarrow}_\mathsf{C}$ and $w_2{\downarrow}_\mathsf{B} = w{\downarrow}_\mathsf{B}$, but $w{\cdot}\mathsf{C{\to}B{:}r} = \mathsf{C{\to}B{:}r} \notin \mathbf{L}(\mathbb{C}A)$.

7.2. **Deciding CUI and Branch-Awareness.** In this section we show that CUI and BA are decidable when restricting to c-languages associated to c-automata. The approach of the present paper can hence be used to prove the communication properties considered in [BLT20] (namely those in Definition 6.6) by showing the corresponding ones in the FCL setting.

We begin by unveiling an interesting interplay between c-automata and FCLs. Languages accepted by c-automata are continuous closures of regular languages, so making conditions like CUI and branch-awareness decidable.

Theorem 7.5. *CUI is decidable on languages accepted by c-automata.*

Proof. Let $\mathcal{L}$ be a language accepted by a c-automaton $\mathbb{C}A$. We show that $\neg\mathsf{cui}(\mathcal{L})$ is decidable by reducing the problem to a search in the FSAs $\mathbb{C}A$, $\mathbb{C}A{\downarrow}_\mathsf{A}$, and $\mathbb{C}A{\downarrow}_\mathsf{B}$.

By definition of CUI we have $\neg\mathsf{cui}(\mathcal{L})$ iff there are an interaction $\alpha = \mathsf{A{\to}B{:}m}$ and finite words $w_1, w_2, w \in \mathcal{L}$ such that

$$w{\cdot}\alpha \notin \mathcal{L} \qquad w{\downarrow}_\mathsf{A} = w_1{\downarrow}_\mathsf{A} \qquad w{\downarrow}_\mathsf{B} = w_1{\downarrow}_\mathsf{B} \qquad \text{while} \qquad w_1{\cdot}\alpha, w_2{\cdot}\alpha \in \mathcal{L}$$

This amounts to find a state q of $\mathbb{C}A$ and two states[4] Q_A and Q_B respectively in $\mathbb{C}A{\downarrow}_\mathsf{A}$ and $\mathbb{C}A{\downarrow}_\mathsf{B}$ such that

$$q \overset{\alpha}{\not\to} \quad q \in Q_\mathsf{A} \cap Q_\mathsf{B} \quad \text{while there are} \quad q_\mathsf{A} \in Q_\mathsf{A}, q_\mathsf{B} \in Q_\mathsf{B} \text{ s.t. } q_\mathsf{A} \overset{\alpha}{\to} \text{ and } q_\mathsf{B} \overset{\alpha}{\to}$$

Since all the involved FSAs are finite an exhaustive search can determine if such states exist for each interaction. $\qquad\square$

Theorem 7.6. *BA is decidable on languages accepted by c-automata.*

Proof. Let $\mathcal{L}$ be a language accepted by a c-automaton $\mathbb{C}A$ and let $\mathbf{A}_\mathsf{X}$ be the intermediate automaton obtained as in Definition 7.2 on a participant X of $\mathbb{C}A$. Given a state p of an FSA A, we let $[A]_p$ denote the FSA obtained by replacing the initial state of A with p and let $[A]^p$ denote the FSA obtained by making p the only final state in A (i.e., setting the final states of A to the singleton $\{ p \}$).

In order to get a decision procedure for $\mathsf{ba}(\mathcal{L})$, we observe that the following equivalences do hold.

$\neg\mathsf{ba}(\mathcal{L})$

iff there are $\mathsf{X} \in \mathcal{P}$ and two maximal words $w_1 \neq w_2 \in \mathcal{L}$ such that $w_1{\downarrow}_\mathsf{X} \prec w_2{\downarrow}_\mathsf{X}$

iff there are $\mathsf{X} \in \mathcal{P}$, two states $p \neq q$ of $\mathbb{C}A$ such that

a) $\mathbf{L}([\mathbf{A}_\mathsf{X}]^p) \cap \mathbf{L}([\mathbf{A}_\mathsf{X}]^q) \neq \emptyset$;

b) there are two maximal words $w_1' \in \mathbf{L}([\mathbb{C}A]_p)$ and $w_2' \in \mathbf{L}([\mathbb{C}A]_q)$ such that $w_1'{\downarrow}_\mathsf{X} = \varepsilon$ and $w_2'{\downarrow}_\mathsf{X} \neq \varepsilon$.

Intuitively, the first equivalence above states that for a participant X it is not possible to distinguish when a computation halts with word w_1 or continues as w_2. Transferring this condition on automata yields the last equivalence, which basically requires that $\mathbf{L}([\mathbf{A}_\mathsf{X}]^p) \cap \mathbf{L}([\mathbf{A}_\mathsf{X}]^q) \neq \emptyset$. Then from its initial state $\mathbb{C}A$ has two paths from the initial state respectively

---

[4]Recall that by definition of projection the states of $\mathbb{C}A{\downarrow}_\mathsf{A}$ and $\mathbb{C}A{\downarrow}_\mathsf{B}$ are subsets of the states of $\mathbb{C}A$ due to the determinisation of the intermediate automata.

reaching p and q with two words whose projection on X coincide. Therefore there are two runs from p and q respectively such that X does not occur in the run from p while it occurs in the run from q, as required by condition b).

We now notice that condition a) is decidable because the intersection of ω-regular languages is computable. In order to decide condition b), instead, we can proceed as follows. We perform a breath-first search on $[\mathbf{A}_{\mathsf{X}}]^p$ stopping either when a state does not have outgoing transitions or when it has been already visited. Condition b) holds iff the resulting tree has a path from p to a leaf made of ε-transitions only. Finiteness of automata guarantees that this procedure terminates. Likewise we can check the existence of a word $w'_2 \in \mathbf{L}([\mathbb{C}\mathrm{A}]_q)$ such that $w'_2\!\downarrow_{\mathsf{X}} \neq \varepsilon$ (as required by condition b)). $\qquad\qquad\square$

8. Global Types as Choreographic Languages

The global types of [SD19] are our last case study. Informally, a global type $\mathsf{A} \to \mathsf{B} : \{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n}$ specifies a protocol where participant A must send to B a message m_i for some $1 \leq i \leq n$ and then, depending on which m_i was chosen by A, the protocol continues as G_i. Global types and multiparty sessions are defined in [SD19] in terms of the following grammars for, respectively, *pre-global types*, *pre-processes*, and *pre-multiparty sessions* (we adapted some of the notation to our setting):

$$\mathsf{G} \ ::=^{co} \ \begin{array}{l} \mathtt{end} \\ | \quad \mathsf{A} \to \mathsf{B} : \{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n} \end{array} \qquad P \ ::=^{co} \ \begin{array}{l} \mathbf{0} \\ | \quad \mathsf{A}?\{\mathsf{m}_i.P_i\}_{1 \leq i \leq n} \\ | \quad \mathsf{A}!\{\mathsf{m}_i.P_i\}_{1 \leq i \leq n} \end{array} \qquad \mathcal{M} \ ::= \ \begin{array}{l} \mathsf{A} \triangleright P \\ | \quad \mathcal{M} \mid \mathcal{M} \end{array}$$

where we assume messages m_i to be pairwise distinct in sets $\{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n}$ and $\{\mathsf{m}_i.P_i\}_{1 \leq i \leq n}$, to whose elements we call *branches*. The first two grammars are interpreted coinductively, that is their solutions are both minimal and maximal fixpoints (the latter corresponding to infinite trees). A pre-global type G (resp. pre-process P) is a *global type* (resp. *process*) if its tree representation is *regular*, namely it has finitely many distinct sub-trees. A *multiparty session* (MPS for short) is a pre-multiparty session such that (a) in $\mathsf{A} \triangleright P$, participant A does not occur in process P and (b) in $\mathsf{A}_1 \triangleright P_1 \mid \ldots \mid \mathsf{A}_n \triangleright P_n$, participants A_i are pairwise different.

The semantics of global types is the LTS induced by the following two rules:

$$\frac{}{\mathsf{A} \to \mathsf{B} : \{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n} \xrightarrow{\mathsf{A}\to\mathsf{B}:\mathsf{m}_i} \mathsf{G}_i} \ \text{(GT1)}$$

$$\frac{\text{for each } 1 \leq i \leq n \quad \mathsf{G}_i \xrightarrow{\mathsf{A}\to\mathsf{B}:\mathsf{m}} \mathsf{G}'_i \quad \{\mathsf{A}, \mathsf{B}\} \cap \{\mathsf{R}, \mathsf{S}\} = \emptyset}{\mathsf{R} \to \mathsf{S} : \{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n} \xrightarrow{\mathsf{A}\to\mathsf{B}:\mathsf{m}} \mathsf{R} \to \mathsf{S} : \{\mathsf{m}_i.\mathsf{G}'_i\}_{1 \leq i \leq n}} \ \text{(GT2)}$$

Rule (GT2) allows out-of-order execution, namely interaction $\mathsf{A}\to\mathsf{B}:\mathsf{m}$ between participants A and B can happen even if an interaction between other participants is syntactically occurring before, provided that $\mathsf{A}\to\mathsf{B}:\mathsf{m}$ occurs in all branches. As usual, we let $\mathsf{G} \xrightarrow{\alpha_0 \cdots \alpha_n} \mathsf{G}'$ to denote $\mathsf{G} \xrightarrow{\alpha_0} \mathsf{G}_1 \cdots \mathsf{G}_n \xrightarrow{\alpha_n} \mathsf{G}'$ and always assume that $\mathsf{G} \xrightarrow{\varepsilon} \mathsf{G}$ holds.

The semantics for MPSs is the LTS defined by the following rule

$$\mathsf{A} \triangleright \mathsf{B}!(\{\mathsf{m}.P\} \uplus \Lambda) \ \big| \ \mathsf{B} \triangleright \mathsf{A}?(\{\mathsf{m}.P'\} \uplus \Lambda') \ \big| \ \mathcal{M} \ \xrightarrow{\mathsf{A} \to \mathsf{B}:\mathsf{m}} \ \mathsf{A} \triangleright P \ \big| \ \mathsf{B} \triangleright P' \ \big| \ \mathcal{M} \quad (1)$$

where $\{\mathsf{m}_i : P_i\}_{i \in I} \uplus \{\mathsf{m}'_j : P'_j\}_{j \in J} = \{\mathsf{m}_i : P_i\}_{i \in I} \cup \{\mathsf{m}'_j : P'_j\}_{j \in J}$ if $\{\mathsf{m}_i\}_{i \in I} \cap \{\mathsf{m}'_j\}_{j \in J} = \emptyset$, otherwise $\{\mathsf{m}_i : P_i\}_{i \in I} \uplus \{\mathsf{m}'_j : P'_j\}_{j \in J}$ is undefined.

Rule (1) applies only if the messages in Λ' include those in Λ, which is the case for MPSs obtained by projection, defined below.

Definition 8.1 (Projection [SD19, Definition 3.4]). The *projection* of G on a participant X such that the depths of its occurrences in G are bounded is the partial function $\mathsf{G} {\restriction}_\mathsf{X}$ coinductively defined by $\mathbf{end} {\restriction}_\mathsf{X} = \mathbf{0}$ and, for a global type $\mathsf{G} = \mathsf{A} \to \mathsf{B} : \{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n}$, by:

$$\mathsf{G} {\restriction}_\mathsf{X} = \begin{cases} \mathbf{0} & \text{if } \mathsf{X} \text{ is not a participant of } \mathsf{G} \\ \mathsf{B}!\{\mathsf{m}_i.\mathsf{G}_i {\restriction}_\mathsf{X}\}_{1 \leq i \leq n} & \text{if } \mathsf{X} = \mathsf{A} \\ \mathsf{A}?\{\mathsf{m}_i.\mathsf{G}_i {\restriction}_\mathsf{X}\}_{1 \leq i \leq n} & \text{if } \mathsf{X} = \mathsf{B} \\ \mathsf{G}_1 {\restriction}_\mathsf{X} & \text{if } \mathsf{X} \notin \{\mathsf{A}, \mathsf{B}\} \text{ and } n = 1 \\ \mathsf{S}?(\Lambda_1 \uplus \ldots \uplus \Lambda_n) & \text{if } \mathsf{X} \notin \{\mathsf{A}, \mathsf{B}\},\ n > 1,\ \text{and for all } 1 \leq i \leq n, \mathsf{G}_i {\restriction}_\mathsf{X} = \mathsf{S}?\Lambda_i \end{cases}$$

The global type G is *projectable*[5] if $\mathsf{G} {\restriction}_\mathsf{X}$ is defined for all participants X of G, in which case $\mathsf{G} {\restriction}$ denotes the corresponding MPS.

The projection on X is partial because if we have a choice where X is not involved in the first interaction then only the last clause in Definition 8.1 could apply, but this clause requires X to start each branch with an input action from the same sender S.

Let $\widehat{\mathcal{L}}(\mathsf{G})$ be the language coinductively defined as follows:

$$\widehat{\mathcal{L}}(\mathbf{end}) = \{\varepsilon\} \qquad \text{and} \qquad \widehat{\mathcal{L}}(\mathsf{A} \to \mathsf{B} : \{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n}) = \bigcup_{1 \leq i \leq n} \{\mathsf{A} \to \mathsf{B}:\mathsf{m}_i \cdot w \mid w \in \widehat{\mathcal{L}}(\mathsf{G}_i)\}$$

The g-language $\mathcal{L}(\mathsf{G})$ associated to a global type G is the concurrency and prefix closure of $\widehat{\mathcal{L}}(\mathsf{G})$, that is

$$\mathcal{L}(\mathsf{G}) = \mathsf{pref}(\{w \in \Sigma_{\text{int}}^\infty \mid \text{there is } w' \in \widehat{\mathcal{L}}(\mathsf{G}) \text{ such that } w \sim w'\})$$

We define the l-language $\mathbb{L}(\mathsf{B} \triangleright P)$ associated to a named process $\mathsf{B} \triangleright P$ as the prefix closure of $\widehat{\mathbb{L}}(\mathsf{B} \triangleright P)$ which, letting $\star \in \{?, !\}$, is defined by

$$\widehat{\mathbb{L}}(\mathsf{B} \triangleright \mathbf{0}) = \{\varepsilon\} \quad \text{and} \quad \widehat{\mathbb{L}}(\mathsf{B} \triangleright \mathsf{A} \star \{\mathsf{m}_i.P_i\}_{1 \leq i \leq n}) = \bigcup_{1 \leq i \leq n} \{\mathsf{AB} \star \mathsf{m}_i \cdot w \mid w \in \widehat{\mathbb{L}}(P_i)\}$$

The system associated to an MPS is defined as the following map:

$$S(\mathsf{A}_1 \triangleright P_1 \mid \ldots \mid \mathsf{A}_n \triangleright P_n) = \{\mathsf{A}_i \mapsto \mathbb{L}(\mathsf{A}_i \triangleright P_i) \mid 1 \leq i \leq n\}$$

Our constructions capture relevant properties of the global types in [SD19].

We establish correspondences between the two frameworks in Propositions 8.2 and 8.5.

Proposition 8.2. *Given a projectable global type* G, $\mathcal{L}(\mathsf{G}) = \{w \mid \mathsf{G} \xrightarrow{w}\}$.

Proof. We show the two inclusions.

$\boxed{\subseteq}$ We begin by showing $\widehat{\mathcal{L}}(\mathsf{G}) \subseteq \{w \mid \mathsf{G} \xrightarrow{w}\}$. If $\mathsf{G} = \mathbf{end}$ then the thesis trivially follows since $\mathsf{G} \xrightarrow{\varepsilon}$. If $\mathsf{G} = \mathsf{A} \to \mathsf{B} : \{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n}$ then each word in $\widehat{\mathcal{L}}(\mathsf{G})$ begins with an interaction

---

[5]In [SD19], projectability embeds well-branchedness.

$\mathsf{A} \to \mathsf{B}{:}\mathsf{m}_i$ with $1 \leq i \leq n$. By the reduction semantics of global types, $\mathsf{G} \xrightarrow{\mathsf{A} \to \mathsf{B}{:}\mathsf{m}_i} \mathsf{G}_i$. By coinduction [KS17] we obtain the thesis. Each word in $\mathcal{L}(\mathsf{G})$ is a prefix of a concurrency equivalent word $w \in \widehat{\mathcal{L}}(\mathsf{G})$. Hence we obtain the thesis by observing that $\mathsf{G} \xrightarrow{w}$ from the previous argument and by (possibly repeated) application of rule (GT2) of the semantics of global types.

$\boxed{\supseteq}$ If $\mathsf{G} \xrightarrow{w}$ then, for some G' we have $\mathsf{G} \xrightarrow{\alpha} \mathsf{G}' \xrightarrow{w'}$ where $w = \alpha \cdot w'$. If $\mathsf{G} \xrightarrow{\alpha} \mathsf{G}'$ is an instance of the first rule of the operational semantics of global types then the thesis follows by coinduction. Otherwise, assuming $\mathsf{G} = \mathsf{R} \to \mathsf{S} : \{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n}$, for each $1 \leq i \leq n$ we have (i) $\mathsf{G}_i \xrightarrow{\alpha} \mathsf{G}'_i$, (ii) $\mathsf{ptp}(\alpha) \cap \{\mathsf{R}, \mathsf{S}\} = \emptyset$, and (iii) $\mathsf{G}' = \mathsf{R} \to \mathsf{S} : \{\mathsf{m}_i.\mathsf{G}'_i\}_{1 \leq i \leq n} \xrightarrow{w'}$. By coinduction, $w' \in \mathcal{L}(\mathsf{G}')$ hence $w = \alpha \cdot w' \in \mathcal{L}(\mathsf{G})$. $\qquad\square$

We now relate projectability (cf. Definition 8.1) and our properties.

Proposition 8.3. *If G is a projectable global type then $\mathcal{L}(\mathsf{G})$ is a CUI and branch-aware sc-language.*

Proof. Let $\mathsf{G} = \mathsf{A} \to \mathsf{B} : \{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n}$ be a global type projectable on X. By construction, $\mathcal{L}(\mathsf{G}) \subseteq \Sigma^\star_{\mathrm{int}}$ or $\mathcal{L}(\mathsf{G})$ is continuous.

To show that $\mathcal{L}(\mathsf{G})$ is branch-aware we proceed by contradiction: assume that $\mathcal{L}(\mathsf{G})$ is not branch-aware. Then there exist two distinct maximal words in $\mathcal{L}(\mathsf{G})$, say $w = \alpha_0 \cdot \alpha_1 \cdots$ and $w' = \alpha'_0 \cdot \alpha'_1 \cdots$, such that $w \not\sim w'$ and, for a participant X, we have $w{\downarrow}_\mathsf{X} \prec w'{\downarrow}_\mathsf{X}$, that is $w{\downarrow}_\mathsf{X}$ is a proper prefix of $w'{\downarrow}_\mathsf{X}$. Note that X must occur in G otherwise, $\mathsf{G}{\downarrow}_\mathsf{X} = \mathbf{0}$. By definition of $\mathcal{L}(\mathsf{G})$, there are $\widehat{w} = \alpha_0 \cdot \alpha_1 \cdots$ and $\widehat{w'} = \alpha'_0 \cdot \alpha'_1 \cdots$ in $\widehat{\mathcal{L}}(\mathsf{G})$, such that $w \sim \widehat{w}$ and $w' \sim \widehat{w'}$; we have $\widehat{w}{\downarrow}_\mathsf{X} \prec \widehat{w'}{\downarrow}_\mathsf{X}$.

By Proposition 8.2, there are two runs

$$\mathsf{G} \xrightarrow{\alpha_0} \mathsf{G}_1 \xrightarrow{\alpha_2} \cdots \qquad \text{and} \qquad \mathsf{G} \xrightarrow{\alpha'_0} \mathsf{G}'_1 \xrightarrow{\alpha'_2} \cdots$$

Let h be the index such that $\alpha_h \neq \alpha'_h$ and $\alpha_i = \alpha'_i$ for all $0 \leq i < h$; note that h must exist otherwise $\widehat{w} = \widehat{w'}$, contradicting $w \not\sim w'$. Necessarily, X does not occur in α_h otherwise $(\alpha_0 \cdots \cdot \alpha_h){\downarrow}_\mathsf{X} \not\prec (\alpha'_0 \cdots \cdot \alpha'_h){\downarrow}_\mathsf{X}$ contradicting $\widehat{w}{\downarrow}_\mathsf{X} \prec \widehat{w'}{\downarrow}_\mathsf{X}$.

Take the last index, say n, of the interaction in $\widehat{w}$ involving X, which necessarily exists otherwise $\widehat{w}{\downarrow}_\mathsf{X}$ cannot be a proper prefix of $\widehat{w'}{\downarrow}_\mathsf{X}$. This contradicts the projectability of G since its subtree G_h has two branches initiating with α_h and α'_h on which the projections for X differ. Hence, none of the cases of Definition 8.1 can be applied.

To show CUI we have to prove that if there are words $w_1 \cdot \mathsf{A} \to \mathsf{B}{:}\mathsf{m}, w_2 \cdot \mathsf{A} \to \mathsf{B}{:}\mathsf{m}, w \in \mathcal{L}(\mathsf{G})$ such that $w_1 {\downarrow}_\mathsf{A} = w {\downarrow}_\mathsf{A}$ and $w_2 {\downarrow}_\mathsf{B} = w {\downarrow}_\mathsf{B}$ then $w \cdot \mathsf{A} \to \mathsf{B}{:}\mathsf{m} \in \mathcal{L}(\mathsf{G})$.

By definition of $\mathcal{L}(\mathsf{G})$, from $w_1 \cdot \mathsf{A} \to \mathsf{B}{:}\mathsf{m}, w_2 \cdot \mathsf{A} \to \mathsf{B}{:}\mathsf{m}, w \in \mathcal{L}(\mathsf{G})$ we can infer that there exist $\hat{w}_1, \hat{w}_2, \hat{w} \in \widehat{\mathcal{L}}(\mathsf{G})$ and $w'_1, w'_2, w' \in \mathcal{L}(\mathsf{G})$ such that

- $\hat{w}_1 \sim w'_1 \qquad \hat{w}_2 \sim w'_2 \qquad \hat{w} \sim w'$;
- $w'_i = w_i \cdot \mathsf{A} \to \mathsf{B}{:}\mathsf{m} \cdot w''_i \quad$ with $\mathsf{ptp}(w''_i) \cap \{\mathsf{A}, \mathsf{B}\} = \emptyset \qquad (i \in \{1, 2\})$;
- $w' = w \cdot w'' \quad$ with $\mathsf{ptp}(w'') \cap \{\mathsf{A}, \mathsf{B}\} = \emptyset$.

Let us now consider the following cases.

$\boxed{\mathsf{ptp}(w') \cap \{\mathsf{A}, \mathsf{B}\} = \emptyset}$ From the hypothesis w'_1 contains exactly one interaction involving A and such an interaction is $\mathsf{A} \to \mathsf{B}{:}\mathsf{m}$. We proceed now by coinduction on the projection of $\mathsf{G} = \mathsf{R} \to \mathsf{S} : \{\mathsf{m}_i.\mathsf{G}_i\}_{1 \leq i \leq n}$ on A (cf. Definition 8.1). Since the first action of A is an output we necessarily have a sequence of co-inductive cases where we apply the 4-th case of Definition 8.1 followed by a case where the 2-nd case of Definition 8.1 is applied. Hence, G

has no branch before the interaction $A{\to}B{:}m$ which must occur in all the maximal words, as desired.

$\boxed{\mathsf{ptp}(w') \cap \{\,A, B\,\} \neq \emptyset}$ Let $C{\to}D{:}x$ be the first interaction in w' (hence also in $\hat{w}$) such that $\{\,C, D\,\} \cap \{\,A, B\,\} \neq \emptyset$. With no loss of generality, assume $A = C$; the other cases are analogous and therefore omitted.

By $w_1 \downarrow_A = w \downarrow_A$, the interaction $A{\to}D{:}x$ is present also in w'_1 and it is the first one having A as participant. Moreover, the same holds for $\hat{w}_1$. Let us proceed by coinduction on the projection of $G = S \to R : \{\,m_i.G_i\,\}_{1 \leq i \leq n}$ on A. The case $R = A$ follows by coinduction since both $\hat{w}$ and $\hat{w}_1$ are generated from the same branch since $w\downarrow_A = w_1\downarrow_A$. The case $A \notin \{R, S\}$ and $n = 1$ follows by coinduction.

Let us consider the case $\{\,S, R\,\} \cap \{\,A, B\,\} = \emptyset$, and assume that $\hat{w}_1$ and $\hat{w}$ begin, respectively, with $S \to R : m_u$ and $S \to R : m_v$ with $u \neq v$ (the case $u = v$ is simpler). Let now $G_u \upharpoonright_A = D?\Lambda_u$, and $G_v \upharpoonright_A = D?\Lambda_v$ (which are defined by projectability of G). By what said previously, we have that $\Lambda_u = \{\,x.P' \uplus \dots\,\}$ and $\Lambda_u = \{\,x.P'' \uplus \dots\,\}$. Now, since $G \upharpoonright_A = D?\{\,\Lambda_u \uplus \Lambda_v \uplus \dots\,\}$, we can infer that $P' = P''$. This implies that $w \cdot A{\to}B{:}m \in \mathcal{L}(G)$. This concludes the proof. $\qquad\square$

The proof of Proposition 8.5 below uses the following auxiliary lemma.

Lemma 8.4. *If $G \upharpoonright_A$ is defined then $w \in \widehat{\mathcal{L}}(G)$ implies $w\downarrow_A \in \widehat{\mathbb{L}}(G \upharpoonright_A)$.*

Proof. By coinduction on G. If $G = \texttt{end}$ the thesis trivially hold. If $G = A \to B : \{m_i.G_i\}_{1 \leq i \leq n}$ or $G = B \to A : \{m_i.G_i\}_{1 \leq i \leq n}$ then the thesis immediately follows by coinduction observing that $w = A{\to}B{:}m_i \cdot w'$ or $w = B{\to}A{:}m_i \cdot w'$ for some $1 \leq i \leq n$ and $w' \in \widehat{\mathcal{L}}(G_i)$. If $G = R \to S : \{m_i.G_i\}_{1 \leq i \leq n}$ with $A \neq R$ and $A \neq S$ then $w = R{\to}S{:}m_i \cdot w'$ for some $1 \leq i \leq n$ and $w' \in \widehat{\mathcal{L}}(G_i)$; hence $w \upharpoonright_A = w' \upharpoonright_A$. By Definition 8.1, either (a) $n = 1$ and $G \upharpoonright_A = G_1 \upharpoonright_A$ or (b) $n > 1$ and $G \upharpoonright_A = S?(\Lambda_1 \uplus \dots \uplus \Lambda_n)$ where, for all $1 \leq i \leq n$, $G_i \upharpoonright_A = S?\Lambda_i$ and Λ_i of the form $\{m_1^i : G_1^i, \dots, m_{n_i}^i : G_{n_i}^i\}$. In case (a) the thesis immediately follows by coinduction observing that $i = n = 1$. In case (b), $w' \upharpoonright_A = S?m_{ij} \cdot v$ with $1 \leq j \leq n_i$ and, coinductively, $v \in \widehat{\mathbb{L}}(A \triangleright G_i \upharpoonright_A)$ the projection on A of some word in $\widehat{\mathcal{L}}(G_i)$ otherwise we would violate the projectability of G on A. $\qquad\square$

Proposition 8.5. *Given a projectable global type G, $[\![S(G \upharpoonright)]\!] = \{\, w \mid G \upharpoonright \xrightarrow{w} \,\}$.*

Proof. We get $[\![S(G \upharpoonright)]\!] = \{\, w \mid G \upharpoonright \xrightarrow{w} \,\}$ by showing that

$$S(G \upharpoonright) = \mathcal{L}(G)\downarrow \tag{2}$$

Indeed, $[\![S(G \upharpoonright)]\!] = [\![\mathcal{L}(G)\downarrow]\!]$ by Proposition 8.2. Proposition 8.3 ensures $S(G \upharpoonright)$ is correct and complete, hence $[\![S(G \upharpoonright)]\!] = \mathcal{L}(G)$ by Theorem 3.7 and Corollary 3.2. So, assume $G \upharpoonright = A_1 \triangleright P_1 \mid \dots \mid A_n \triangleright P_n$, then

$$
\begin{aligned}
\mathcal{L}(G)\downarrow &= \{\, w \mid G \xrightarrow{w} \,\}\downarrow && \text{by Proposition 8.2} \\
&= \{\, A_i \mapsto \{\, w\downarrow_{A_i} \mid G \xrightarrow{w} \,\} \mid 1 \leq i \leq n \,\} && \text{by Definition 2.7} \\
&= \{\, A_i \mapsto \{\, w\downarrow_{A_i} \mid w \in \mathsf{pref}(\{\, w' \in \widehat{\mathcal{L}}(G) \mid w \sim w' \,\}) \,\} \mid 1 \leq i \leq n \,\} && \text{by def. of } \mathcal{L}(G) \\
&= \{\, A_i \mapsto \mathbb{L}(A_i \triangleright P_i) \mid 1 \leq i \leq n \,\} && \text{by Lemma 8.4}
\end{aligned}
$$

Finally $\{\, A_i \mapsto \mathbb{L}(A_i \triangleright P_i) \mid 1 \leq i \leq n \,\} = S(G \upharpoonright)$ by definition of $S(\_)$. $\qquad\square$

Projectable global types are proved strongly lock-free in [SD19]. The following result corresponds to [SD19, Theorem 4.7].

Corollary 8.6. $S(\mathsf{G}{\upharpoonright})$ *is strongly lock-free for any projectable* G.

The symmetry between senders and receivers in CUI and branch-awareness allows for an immediate generalisation of the projection in Definition 8.1 by extending the last case with the clause:

$$\mathsf{S}!(\Lambda_1 \uplus \ldots \uplus \Lambda_n) \qquad \text{if } \mathsf{X} \notin \{\mathsf{A}, \mathsf{B}\}, \ n > 1, \text{ and for all } 1 \leq i \leq n, \mathsf{G}_i{\upharpoonright}_{\mathsf{X}} = \mathsf{S}!\Lambda_i$$

Corollary 8.6 still holds for this generalised definition of projection.

9. Related Work

The use of formal language theories for the modelling of concurrent systems dates back to the theory of traces [Maz86]. A trace is an equivalence class of words that differ only for swaps of independent symbols. Closure under concurrency corresponds on finite words to form traces, as we noted after Definition 2.12. An extensive literature has explored a notion of realisability whereby a language of traces is realisable if it is accepted by some class of finite-state automata. Relevant results in this respect are the characterisations in [Zie87, Dub86] (and the optimisation in [GM06]) for finite words and the ones in [Gas90, Gas91, GPZ91] for infinite ones. A key difference of our framework with respect to this line of work is that we aim to stricter notions of realisability: in our context it is not enough that the runs of the language may be faithfully executed by a certain class of finite-state automata. Rather we are interested in identifying conditions on the g-languages that guarantee well-behaved executions in "natural" realisations.

Other abstract models of choreographies, such as Conversation protocols (CP) [FBS04] and c-automata [BLT20], have some relation with our FCL. We discussed in depth the relation with c-automata in Section 7.

CP, probably the first automata-based model of choreographies, are non-deterministic Büchi automata whose alphabet resembles a constrained variant of our Σ_{int}. A comparison with the g-languages accepted by CPs is not immediate as CPs are based on asynchronous communications (although some connections are evident as noted below Definition 2.9).

Other proposals ascribable to choreographic settings (cf. [HLV$^+$16]) define global views that can be seen as g-languages. We focus on synchronous approaches because our current theory needs to be extended to cope with asynchrony.

In [BZ07, LGMZ08] the correctness of implementations of choreographies (called *chore-ography conformance*) is studied in a process algebraic setting. The other communication properties we consider here are not discussed there.

The notion of choreography implementation in [BZ07] corresponds to our correctness plus a form of existential termination. It is shown that one can decide whether a system is an implementation of a given choreography, since both languages are generated by finite-state automata, hence language inclusion and existential termination are decidable.

In [LGMZ08] three syntactic conditions (connectedness, unique points of choice and causality safety) ensure bisimilarity (hence trace equivalence) between a choreography and its projection. Connectedness rules out systems which are not concurrency closed, while we conjecture that unique points of choice and connectedness together imply our CUI and BA.
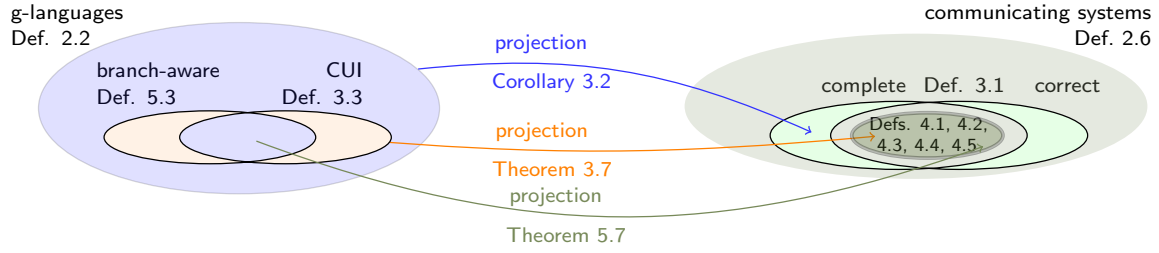
Figure 10.1: Main contributions

Causality safety, immaterial in our case, is needed in [LGMZ08] due to the existence of an explicit parallel composition operator in their process algebra.

Many multiparty session type systems [HLV+16] have two levels of types (global and local) and one implementation level (local processes). This is the case also for synchronous session type systems such as [KY14, DGJ+15]. The approach in [SMD22] instead merges the two levels of types into a single one to allow for composition, while having a level of local processes for implementation. Our approach, like the session type systems in [SD19, BDCLT21], considers only (two) abstract descriptions, g-languages and l-languages. The literature offers several behavioural types featuring correctness-by-construction principles through conditions (known as projectability or well-branchedness) more demanding than ours. For instance, relations similar to those in Section 8 can be devised for close formalisms, such as [BDCLT21] whose notion of projection is more general than the one in [SD19], yet its notion of projectability still implies CUI and BA.

There is a connection between CUI and the closure property CC2 [AEY03] on message-sequence charts (MSCs) [ITU96]. On finite words CC2 and CUI coincide. Actually, CUI can be regarded as a step-by-step way to ensure CC2 on finite words. The relations between our properties and CC3, also used in MSCs, are still under scrutiny.

10. CONCLUDING REMARKS

We introduced formal choreography languages as a general and abstract theory of choreographies based on formal languages. In this theory we recasted known properties and constructions such as projections from global to local specifications.

One of our contributions is the characterisation of systems' correctness in terms of closure under unknown information. Other communication properties can be ensured by additionally requiring branch awareness. A synopsis of our main contributions is in Figure 10.1.

We demonstrated the versatility of FCL by considering three existing models. We showed some relations between FCL and two automata models, communicating finite-state machines [BZ83] and c-automata [BLT20]. These models are close to FCL, given that the relations retrace the well-known connection between automata and formal language theories. The last model captured with FCL is the variant of MPSTs presented in [SD19]. Being based on behavioural types, this model radically differs from FCL.

Future work. Our investigation proposes a new point of view for choreography formalisms and the related constructions. As such, a number of extensions and improvements need to be analysed, to check how they may fit in our setting. We list below the most relevant.

Our framework considers only point-to-point synchronous communications. Possible generalisations could contemplate other interaction models such as those e.g., in [CMSY17, Wad14] and those based on *asynchronous* communications. Another possible generalisation is to consider *nominal* FCLs in the line of nominal languages [KST12, BBKL12, KMPS15]. While the general approach should apply, it is not immediate how to extend CUI in order to characterise correctness for an asynchronous semantics. This is somehow confirmed by the results in [Alu05, AEY01] on the realisability of MSCs showing that in the asynchronous setting this is a challenging problem.

A second direction is analysing how to drop prefix-closure, so allowing for specifications where the system (and single participants) may stop their execution at some points but not at others; a word would hence represent a complete computation, not only a partial one.

A further direction would unveil the correspondence between closure properties and subtyping relations used in many multiparty session types approaches.

Additionally, it would be good to understand whether the composition and decomposition operators or the partial multiparty sessions defined in, respectively, [BDCLT21] and [SMD22], for multiparty session types can be rephrased in the more general setting of FCLs.

Another intriguing research direction is the application of FCLs to capture properties that are not usually considered in behavioural type frameworks. Specifically, the notions of *receptiveness* and *responsiveness* have been defined in [tBCHK17, tBCHP21, tBHK20] to formalise the properties of communicating systems where some components can always succeed to send or receive messages. An interesting initial question to address is whether receptiveness and responsiveness can be characterised in terms of FCLs. A starting point to address it could be to give FCL models for the dynamic logic used in [tBCHP23] to characterise receptiveness.

Finally, our approach strives for generality neglecting efficiency and practical applicability. An interesting research direction is to identify classes of languages ensuring that our properties, such as CUI, could be checked efficiently.

References

[ABB80]   Jean-Michel Autebert, Joffroy Beauquier, and Luc Boasson. Langages sur des alphabets infinis. *Discrete Applied Mathematics*, 2(1):1 − 20, 1980. `doi:10.1016/0166-218X(80)90050-5`.

[AEY01]   Rajeev Alur, Kousha Etessami, and Mihalis Yannakakis. Realizability and verification of MSC graphs. In Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen, editors, *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, volume 2076 of *Lecture Notes in Computer Science*, pages 797–808. Springer, 2001. `doi:10.1007/3-540-48224-5_65`.

[AEY03]   Rajeev Alur, Kousha Etessami, and Mihalis Yannakakis. Inference of Message Sequence Charts. *IEEE Trans. Software Eng.*, 29(7):623–633, 2003.

[Alu05]   Rajeev Alur. The benefits of exposing calls and returns. In Martín Abadi and Luca de Alfaro, editors, *CONCUR 2005 - Concurrency Theory, 16th International Conference, CONCUR 2005, San Francisco, CA, USA, August 23-26, 2005, Proceedings*, volume 3653 of *Lecture Notes in Computer Science*, pages 2–3. Springer, 2005. `doi:10.1007/11539452_2`.

[BB11]   Samik Basu and Tevfik Bultan. Choreography conformance via synchronizability. In Sadagopan Srinivasan, Krithi Ramamritham, Arun Kumar, M. P. Ravindra, Elisa Bertino, and Ravi Kumar, editors, *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*, pages 795–804. ACM, 2011. `doi:10.1145/1963405.1963516`.

[BBKL12]   Mikolaj Bojanczyk, Laurent Braud, Bartek Klin, and Slawomir Lasota. Towards nominal computation. In John Field and Michael Hicks, editors, *Proceedings of the 39th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2012, Philadelphia, Pennsylvania, USA, January 22-28, 2012*, pages 401–412. ACM, 2012. `doi:10.1145/2103656.2103704`.

[BBO12]   Samik Basu, Tevfik Bultan, and Meriem Ouederni. Deciding choreography realizability. In John Field and Michael Hicks, editors, *Proceedings of the 39th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2012, Philadelphia, Pennsylvania, USA, January 22-28, 2012*, pages 191–202, 2012. `doi:10.1145/2103656.2103680`.

[BDCLT21] Franco Barbanera, Mariangiola Dezani-Ciancaglini, Ivan Lanese, and Emilio Tuosto. Composition and decomposition of multiparty sessions. *Journal of Logical and Algebraic Methods in Programming*, 119:100620, 2021. `doi:10.1016/j.jlamp.2020.100620`.

[BLT20]   Franco Barbanera, Ivan Lanese, and Emilio Tuosto. Choreography automata. In Simon Bliudze and Laura Bocchi, editors, *Coordination Models and Languages - 22nd IFIP WG 6.1 International Conference, COORDINATION 2020, Held as Part of the 15th International Federated Conference on Distributed Computing Techniques, DisCoTec 2020, Valletta, Malta, June 15-19, 2020, Proceedings*, volume 12134 of *Lecture Notes in Computer Science*, pages 86–106. Springer, 2020. `doi:10.1007/978-3-030-50029-0\_6`.

[BLT22]   Franco Barbanera, Ivan Lanese, and Emilio Tuosto. Formal choreographic languages. In Maurice H. ter Beek and Marjan Sirjani, editors, *Coordination Models and Languages - 24th IFIP WG 6.1 International Conference, COORDINATION 2022, Held as Part of the 17th International Federated Conference on Distributed Computing Techniques, DisCoTec 2022, Lucca, Italy, June 13-17, 2022, Proceedings*, volume 13271 of *LNCS*, pages 121–139. Springer, 2022. `doi:10.1007/978-3-031-08143-9\_8`.

[Bon18]   Jonas Bonér. *Reactive Microsystems - The Evolution Of Microservices At Scale*. O'Reilly, 2018.

[BZ83]   Daniel Brand and Pitro Zafiropulo. On communicating finite-state machines. *J. ACM*, 30(2):323–342, 1983. `doi:10.1145/322374.322380`.

[BZ07]   Mario Bravetti and Gianluigi Zavattaro. Towards a unifying theory for choreography conformance and contract compliance. In Markus Lumpe and Wim Vanderperren, editors, *Software Composition, 6th International Symposium, SC 2007, Braga, Portugal, March 24-25, 2007, Revised Selected Papers*, volume 4829 of *Lecture Notes in Computer Science*, pages 34–50. Springer, 2007. `doi:10.1007/978-3-540-77351-1_4`.

[CDYP16]   Mario Coppo, Mariangiola Dezani-Ciancaglini, Nobuko Yoshida, and Luca Padovani. Global progress for dynamically interleaved multiparty sessions. *Mathematical Structures in Computer Science*, 26(2):238–302, 2016. `doi:10.1017/S0960129514000188`.

[CHY12]   Marco Carbone, Kohei Honda, and Nobuko Yoshida. Structured communication-centered programming for web services. *ACM Trans. Program. Lang. Syst.*, 34(2):8:1–8:78, 2012. `doi:10.1145/2220365.2220367`.

[CM13]   Marco Carbone and Fabrizio Montesi. Deadlock-freedom-by-design: Multiparty asynchronous global programming. *SIGPLAN Not.*, 48(1):263–274, January 2013. `doi:10.1145/2480359.2429101`.

[CMSY17]   Marco Carbone, Fabrizio Montesi, Carsten Schürmann, and Nobuko Yoshida. Multiparty session types as coherence proofs. *Acta Informatica*, 54(3):243–269, 2017. `doi:10.1007/s00236-016-0285-y`.

[Dar14]   Ornela Dardha. Recursive session types revisited. *Electronic Proceedings in Theoretical Computer Science*, 162:27–34, August 2014. `doi:10.4204/eptcs.162.4`.

[DGJ$^+$15]   Mariangiola Dezani-Ciancaglini, Silvia Ghilezan, Svetlana Jaksic, Jovanka Pantovic, and Nobuko Yoshida. Precise subtyping for synchronous multiparty sessions. In Simon Gay and Jade Alglave, editors, *Proceedings Eighth International Workshop on Programming Language Approaches to Concurrency- and Communication-cEntric Software, PLACES 2015, London, UK, 18th April 2015*, volume 203 of *EPTCS*, pages 29–43, 2015. `doi:10.4204/EPTCS.203.3`.

[Dub86]   Christine Duboc. Mixed product and asynchronous automata. *TCS*, 48(3):183–199, 1986. `doi:10.1016/0304-3975(86)90094-0`.

[Eil76]   Samuel Eilenberg. *Automata, languages, and machines, B*. Pure and applied mathematics. Academic Press, 1976. URL: `https://www.worldcat.org/oclc/310535259`.

[FBS04]   Xiang Fu, Tevfik Bultan, and Jianwen Su. Conversation protocols: a formalism for specification and verification of reactive electronic services. *TCS*, 328(1-2):19–37, 2004. `doi:10.1016/j.tcs.2004.07.004`.

[Gas90]   Paul Gastin. Infinite traces. In Irène Guessarian, editor, *Semantics of Systems of Concurrent Processes, LITP Spring School on Theoretical Computer Science, La Roche Posay, France, April 23-27, 1990, Proceedings*, volume 469 of *Lecture Notes in Computer Science*, pages 277–308. Springer, 1990. `doi:10.1007/3-540-53479-2_12`.

[Gas91]   Paul Gastin. Recognizable and rational languages of finite and infinite traces. In Christian Choffrut and Matthias Jantzen, editors, *STACS 91, 8th Annual Symposium on Theoretical Aspects of Computer Science, Hamburg, Germany, February 14-16, 1991, Proceedings*, volume 480 of *Lecture Notes in Computer Science*, pages 89–104. Springer, 1991. `doi:10.1007/BFb0020790`.

[GJP+19]   Silvia Ghilezan, Svetlana Jaksic, Jovanka Pantovic, Alceste Scalas, and Nobuko Yoshida. Precise subtyping for synchronous multiparty sessions. *J. Log. Algebraic Methods Program.*, 104:127–173, 2019. `doi:10.1016/j.jlamp.2018.12.002`.

[GM06]   Blaise Genest and Anca Muscholl. Constructing exponential-size deterministic zielonka automata. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 565–576. Springer, 2006. `doi:10.1007/11787006\_48`.

[GPZ91]   Paul Gastin, Antoine Petit, and Wieslaw Zielonka. A kleene theorem for infinite trace languages. In Javier Leach Albert, Burkhard Monien, and Mario Rodríguez-Artalejo, editors, *Automata, Languages and Programming, 18th International Colloquium, ICALP91, Madrid, Spain, July 8-12, 1991, Proceedings*, volume 510 of *Lecture Notes in Computer Science*, pages 254–266. Springer, 1991. `doi:10.1007/3-540-54233-7_139`.

[HLV+16]   Hans Hüttel, Ivan Lanese, Vasco T. Vasconcelos, Luís Caires, Marco Carbone, Pierre-Malo Deniélou, Dimitris Mostrous, Luca Padovani, António Ravara, Emilio Tuosto, Hugo Torres Vieira, and Gianluigi Zavattaro. Foundations of session types and behavioural contracts. *ACM Comput. Surv.*, 49(1):3:1–3:36, 2016. `doi:10.1145/2873052`.

[HYC16]   Kohei Honda, Nobuko Yoshida, and Marco Carbone. Multiparty asynchronous session types. *Journal of the ACM*, 63(1):9:1–9:67, 2016. Extended version of a paper presented at POPL'08. `doi:10.1145/2827695`.

[ITU96]   ITU Telecommunication Standardization Sector. *ITU-T recommendation Z.120. Message Sequence Charts (MSC'96)*, 1996.

[JY20]   Sung-Shik Jongmans and Nobuko Yoshida. Exploring type-level bisimilarity towards more expressive multiparty session types. In Peter Müller, editor, *Programming Languages and Systems*, pages 251–279, Cham, 2020. Springer. `doi:10.1007/978-3-030-44914-8_10`.

[KBR+05]   Nickolas Kavantzas, David Burdett, Gregory Ritzinger, Tony Fletcher, Yves Lafon, and Charlton Barreto. Web services choreography description language version 1.0. Technical report, W3C, 2005. `http://www.w3.org/TR/ws-cdl-10/`.

[KMPS15]   Dexter Kozen, Konstantinos Mamouras, Daniela Petrişan, and Alexandra Silva. Nominal kleene coalgebra. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *International Colloquium on Automata, Languages, and Programming*, pages 286–298, Kyoto, Japan, 2015. `doi:10.1007/978-3-662-47666-6_23`.

[Kob02]   Naoki Kobayashi. A Type System for Lock-Free Processes. *Information and Computation*, 177:122–159, 2002. `doi:10.1006/inco.2002.3171`.

[KS10]   Naoki Kobayashi and Davide Sangiorgi. A hybrid type system for lock-freedom of mobile processes. *ACM Trans. Program. Lang. Syst.*, 32(5):16:1–16:49, 2010. `doi:10.1145/1745312.1745313`.

[KS17]   Dexter Kozen and Alexandra Silva. Practical Coinduction. *Mathematical Structures in Computer Science*, 27(7):1132–1152, 2017.

[KST12]   Alexander Kurz, Tomoyuki Suzuki, and Emilio Tuosto. On nominal regular languages with binders. In Lars Birkedal, editor, *Foundations of Software Science and Computational Structures - 15th International Conference, FOSSACS 2012, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2012, Tallinn, Estonia, March 24 - April 1, 2012. Proceedings*, volume 7213 of *Lecture Notes in Computer Science*, pages 255–269. Springer, 2012. `doi:10.1007/978-3-642-28729-9\_17`.

[KY14]   Dimitrios Kouzapas and Nobuko Yoshida. Globally governed session semantics. *Log. Methods Comput. Sci.*, 10(4), 2014. `doi:10.2168/LMCS-10(4:20)2014`.

[LGMZ08]   Ivan Lanese, Claudio Guidi, Fabrizio Montesi, and Gianluigi Zavattaro. Bridging the gap between interaction- and process-oriented choreographies. In Antonio Cerone and Stefan Gruner, editors, *Software Engineering and Formal Methods, SEFM 2008*, pages 323–332, Cape Town, South Africa, 2008. `doi:10.1109/SEFM.2008.11`.

[LNTY17]   Julien Lange, Nicholas Ng, Bernardo Toninho, and Nobuko Yoshida. Fencing off go: liveness and safety for channel-based programming. In Giuseppe Castagna and Andrew D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, pages 748–761. ACM, 2017. `doi:10.1145/3009837.3009847`.

[Maz86]   Antoni W. Mazurkiewicz. Trace theory. In Wilfried Brauer, Wolfgang Reisig, and Grzegorz Rozenberg, editors, *Petri Nets: Central Models and Their Properties, Advances in Petri Nets 1986, Part II, Proceedings of an Advanced Course, Bad Honnef, Germany, 8-19 September 1986*, volume 255 of *Lecture Notes in Computer Science*, pages 279–324. Springer, 1986. `doi:10.1007/3-540-17906-2_30`.

[OMG11]   OMG. *Business Process Model and Notation (BPMN), Version 2.0*, January 2011. `https://www.omg.org/spec/BPMN`.

[Pad13]   Luca Padovani. From lock freedom to progress using session types. In Nobuko Yoshida and Wim Vanderbauwhede, editors, *Proceedings 6th Workshop on Programming Language Approaches to Concurrency and Communication-cEntric Software, PLACES 2013, Rome, Italy, 23rd March 2013*, volume 137 of *EPTCS*, pages 3–19, 2013. `doi:10.4204/EPTCS.137.2`.

[Par81]   David Park. Concurrency and Automata on Infinite Sequences. In *Theoretical Computer Science, 5th GI-Conf.*, volume 104 of *LNCS*, pages 167–183. Springer, Karlsruhe, March 1981.

[Red86]   Roman R. Redziejowski. Infinite-word languages and continuous mappings. *TCS*, 43:59–79, 1986. `doi:10.1016/0304-3975(86)90166-0`.

[SD19]   Paula Severi and Mariangiola Dezani-Ciancaglini. Observational equivalence for multiparty sessions. *Fundam. Informaticae*, 170(1-3):267–305, 2019. `doi:10.3233/FI-2019-1863`.

[SMD22]   Claude Stolze, Marino Miculan, and Pietro Di Gianantonio. Composable partial multiparty session types for open systems. *Software and Systems Modeling*, 2022. `doi:10.1007/s10270-022-01040-x`.

[Sta97]   Ludwig Staiger. ω-languages. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages, Volume 3: Beyond Words*, pages 339–387. Springer, 1997. `doi:10.1007/978-3-642-59126-6\_6`.

[tBCHK17]   Maurice H. ter Beek, Josep Carmona, Rolf Hennicker, and Jetty Kleijn. Communication requirements for team automata. In Jean-Marie Jacquet and Mieke Massink, editors, *Coordination Models and Languages*, pages 256–277, Cham, 2017. Springer International Publishing. `doi:10.1007/978-3-319-59746-1_14`.

[tBCHP21]   Maurice H. ter Beek, Guillermina Cledou, Rolf Hennicker, and José Proença. Featured team automata. In Marieke Huisman, Corina Păsăreanu, and Naijun Zhan, editors, *Formal Methods*, pages 483–502, Cham, 2021. Springer International Publishing. `doi:10.1007/978-3-030-90870-6_26`.

[tBCHP23]   Maurice H. ter Beek, Guillermina Cledou, Rolf Hennicker, and José Proença. Can we communicate? using dynamic logic to verify team automata. In Marsha Chechik, Joost-Pieter Katoen, and Martin Leucker, editors, *Formal Methods - 25th International Symposium, FM 2023, Lübeck, Germany, March 6-10, 2023, Proceedings*, volume 14000 of *Lecture Notes in Computer Science*, pages 122–141. Springer, 2023. `doi:10.1007/978-3-031-27481-7\_9`.

[tBHK20]   Maurice H. ter Beek, Rolf Hennicker, and Jetty Kleijn. Compositionality of safe communication in systems of team automata. In Violet Ka I. Pun, Volker Stolz, and Adenilso Simao, editors, *Theoretical Aspects of Computing – ICTAC 2020*, pages 200–220, Cham, 2020. Springer International Publishing. `doi:10.1007/978-3-030-64276-1_11`.

[TV16]   Peter Thiemann and Vasco T. Vasconcelos. Context-free session types. In Jacques Garrigue, Gabriele Keller, and Eijiro Sumii, editors, *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming, ICFP 2016, Nara, Japan, September 18-22, 2016*, pages 462–475. ACM, 2016. `doi:10.1145/2951913.2951926`.

[Wad14]   Philip Wadler. Propositions as sessions. *J. Funct. Program.*, 24(2-3):384–418, 2014. `doi:10.1017/S095679681400001X`.

[Zie87]   Wieslaw Zielonka. Notes on finite asynchronous automata. *RAIRO Theor. Informatics Appl.*, 21(2):99–135, 1987. `doi:10.1051/ita/1987210200991`.