# Semantics of global view of choreographies ${ }^{\star \pi}$ 

Emilio Tuosto ${ }^{\mathrm{a}, *}$, Roberto Guanciale ${ }^{\mathrm{b}, * *}$<br>${ }^{a}$ University of Leicester, UK<br>${ }^{b}$ KTH, Sweden


#### Abstract

We propose two abstract semantics of the global view of choreographies given in terms of partial orders. The first semantics is formalised as pomsets of communication events while the second one is based on hypergraphs of events. These semantics can accommodate different levels of abstractions. We discuss the adequacy of our models by considering their relation with communicating machines, that we use to formalise the local view. Our approach increases expressiveness and allows us to overcome some limitations that affect alternative semantics of global views. This will be illustrated by discussing some interesting examples. Finally, we show that the two semantics are equivalent and have different merits. More precisely, the semantics based on pomsets yields a more elegant presentation, but it is less suitable for implementation. The semantics based on hypergraphs instead is amenable to a straightforward implementation.


Keywords: Choreography, communicating finite-state machines, global graphs, hypergraphs, pomsets, semantics.

## 1. Introduction

Distributed applications are nowadays widespread. Rarely applications are stand-alone anymore: software is today conceived to dynamically interact with other applications. The combination of ubiquitous connectivity and the evolution of portable or wearable devices (such as smart phones or watches) practically changed the nature of software and it is also determining new approaches to software development [28]. Big vendors as well as small software companies, have to satisfy the appetite of users who want data and applications 'always handy'. Also, software is becoming more and more important as it increasingly deals with delicate societal and economical aspects. Distributed applications are used in many aspects of our lives, from handling commercial transactions

[^0]to social interaction, to providing e-health and e-government services. For such (and other) reasons, developers are required to carefully design their applications so that unintended behaviours do not happen at runtime.

The problem. It is widely accepted that distributed systems and applications are not easy to design, implement, verify, deploy, and maintain. A key issue to tackle is the coordination of distributed components. For this, two primary (and in our view, complementary) approaches have been identified: orchestration and choreography. We focus here on the latter. More precisely, we propose a new semantics of a model of choreographies for message-passing software. We argue that our semantics offer a framework that ( $i$ ) generalises existing linguistic constructs of choreographies by removing some unnecessary constraints adopted elsewhere, (ii) allows architects to design the coordination of distributed applications without forcing them to consider low level details, and (iii) enables the possibility of tool support at earlier stages of the development.

Choreographies have been advocated as a suitable methodology for the design and analysis of distributed applications since, unlike models based on orchestration, they do not require an explicit coordinator. Roughly, a choreography describes how two or more distributed components coordinate with each other by exchanging messages. Among the possible interpretations of what choreographies are (see [2] for a discussion and references), we embrace the one suggested by W3C's [23]:

Using the Web Services Choreography specification, a contract containing a global definition of the common ordering conditions and constraints under which messages are exchanged, is produced that describes, from a global viewpoint [...] observable behaviour [...]. Each party can then use the global definition to build and test solutions that conform to it. The global specification is in turn realised by combination of the resulting local systems [...]

This description conceptualises two views, a global and a local one, which enable the relations represented by the following diagram:

where the operation of 'projection' produces the local view from the global one and the operation 'comply' verifies that the behaviour of each component adheres to the one of the corresponding local view.

For diagram (1) to make sense, precise semantics should be fixed for the global and the local views. The semantics of the latter is well understood: it directly emanates from the adopted communication model. In fact, the local view details how communications take place. For instance, in a channel-based communication model, the local view may specify what is the behaviour of each component in terms of its send/receive actions.

What is instead "the semantics of the global view"? We investigate such question here. And, after making it more precise, we propose a new semantic framework for global views and discuss its advantages over existing frameworks.

Before continuing, we remark that the relations among views and systems of choreographies are richer than those depicted in diagram (1). For instance, local views can also be 'compiled' into template code for the local components and the projection operation may have an "inverse" (cf. [26]). Those aspects are not in the scope of this paper though.

A view of global views. The W3C description above is intriguing, however it is not very enlightening to understand what a global view is; basically it says that a global view has to describe the observable behaviour from a global viewpoint...a bit too much circularity for a definition!

We will consider global views as high-level descriptions of systems abstracting away some aspects in order to offer a holistic understanding of the communication behaviour of distributed systems. (This is still vague, but will become precise in the forthcoming sections.) In a global view, components are not taken anymore in isolation. Rather they are specified together, while forgetting some details. For us, this will mean to describe the protocol of interaction of a system in a way that makes it explicit how messages are actually exchanged among components. For instance, in our example based on channels, the global view may abstract away from send/receive actions and use interactions as the unit of coordination 7].

The idea depicted in diagram (1) is beautiful. To our best knowledge, it has been firstly formally pursued in 19] (later refined in [21]) and then followed by others. The main reason that makes diagram (1) attractive is the interplay between global and local artefacts ${ }^{1}$ as it fosters some of the best principles of computer science and of software engineering:

Separation of concerns The intrinsic logic of the distributed coordination is expressed in and analysed on global artefacts, while the local artefacts refine such logic at lower levels of abstraction.

Modular software development life-cycle The W3C description above yields a distinctive element of choreographies which makes them appealing (also to practitioners). Choreographies allow independent development: components can harmoniously interact when proven compliant to their corresponding local view. Global and local views yield the "blueprints" of systems as a whole and of each component, respectively.

Principled design A choreographic framework orbits around the following implication:
if cond(global artefact) then behave(projection(global artefact))

[^1]that is, proving that a correctness condition cond holds on an abstraction (the global artefacts) guarantees that the system is well behaved, provided that the local artefacts are "compiled" from the global ones via a projection operation that preserves behaviour.

Therefore, providing suitable semantics for global artefacts is worthwhile: it gives precise algorithms and establishes precise relations between specifications of distributed systems (the global artefacts) and their refinements (the local artefacts). It is worth remarking that the languages we adopt here for global and local views are specification languages. In particular, we abstract away from concrete programming mechanisms (e.g., adopting non-deterministic constructs), avoiding to specify local computations (that is computations that do not require the interactions of components), and abstract away from data, in fact the term "message" here has to be understood as "data type" rather than actual valu¢ ${ }^{2}$.

We have previously proposed in [18] a hypergraph semantics of choreographies, which is apt to be implemented (we are currently embedding the hypergraph semantics in ChorGram [25], a toolchain for choreographic modelling). The new pomset semantics proposed here is more abstract (and, we believe, more elegant) than the hypergraph semantics. In particular, the use of pomsets simplifies the notion of projection and generalises well-sequencedness and well-branchedness. Besides, defining the pomsets semantics and showing its equivalence with the hypergraph-based semantics, in this paper we have revised and simplified the presentation in 18 .
Contributions Our main technical contributions are two semantics of global views of choreographies, modelled as global graphs [15]. Our first semantic framework relies on pomsets [29, [16, which provide a simple and elegant theoretical presentation of concurrency and distribution. We note that pomsets have been adopted also in [22] as a semantic framework of message-sequence charts. Our second semantic framework relies on a model of hypergraphs. This framework is more suitable for developing analysis tools. As we show in the paper, the two semantic models are equivalent, in the sense that they yield the same causality relations of choreographies. To the best of our knowledge, our semantics generalise existing approaches; we demonstrate this by giving examples of global views usually discarded in other approaches.

We note that in [14 (grammars of) hypergraphs have been used as a formal operational semantics of distributed computations. Here we use hypergraphs as the semantic codomain in a denotational style.
Outline Section 2 highlights the advantages of abstract semantics of global views. The syntax of our language of global artefacts is in Section 3 with some examples used in later sections to explain some of our constructions. Section 4 presents the abstract semantics of global artefacts by capturing their causal dependencies using pomsets. A first technical advantage of our semantics is provided by the definition of well-branched choices, explained through some illustrative

[^2]examples. Our semantics is used to identify all licit traces of a choreography, so to precisely characterise the behaviour expected by the specification. Section 5 first recalls the communicating finite state machines (that are used to formalise the local behaviours) and then defines the projection of global artefacts on communicating machines. The main technical results establish that well-branched choreographies yield deadlock-free systems (Theorem 1) whose behaviour is included in the behaviours specified by the global view (Theorem 2). Section 6 presents an alternative semantics based on hypergraphs and introduces the notion of reflection, crucial for our model. Theorem 3 establishes equivalence between the two semantics (detailed proofs are reported in 31). Section 7 draws some conclusions.

## 2. Why going abstract?

We propose a denotational semantics of global views of choreographies based on partial orders. Our new semantics is more abstract than existing ones as it is not based on traces and it makes minimal assumptions on how messages are exchanged at lower levels. Conceptually this is easy to achieve. We fix a specification language of global artefacts and we interpret a specification as a set of "minimal and natural" causal dependencies among the messages. We then define when a global artefact is sound, namely when its causal dependencies are consistent so that they are amenable to be executed distributively by some local artefacts, regardless of the underlying message passing semantics. We illustrate the advantages of our approach by adopting a rather liberal language of global artefacts inspired by global graphs [15]. We then show the relation of such language on a local view featuring local artefacts as communicating machines [5].

As discussed in Section 1, many authors have adopted the idea in diagram (1) and several semantics of (models of) the global view have been introduced. We distinguish two broad classes ${ }^{3}$

The first class consists of the approaches where the semantics of the global view is obtained by composing the semantics of the local view. For instance, this class includes the seminal work on global types [20]. The idea is to give an explicit semantics of local artefacts and define the semantics of global artefacts in terms of the semantics of their projections. In the case of global types, the projection yields local types, that are process algebras equipped with an operational semantics. This approach is ubiquitous in the literature based on behavioural types and it has also been adopted in [26] where global artefacts are global graphs [15] and local artefacts are communicating finite-state machines [5].

In the other class, the semantics of global view is defined explicitly. A first such semantics appeared in [6] where a language of global views syntactically equivalent to ours has been considered (the semantics is however different from ours since in [6] synchrony is assumed both at the global and local level). Also, the global types of [20] are refined and equipped with an operational semantics in 21].

[^3]In [9] an operational interleaving semantics is defined for global specifications while in [3] a trace-based semantics is given. In both cases, the idea is to "split" the interactions in the global view into its constituent send/receive actions. A slightly more abstract interleaving operational semantics of global types is given in [8] where interleaving is attained by swapping independent interactions, rather than manipulating traces of send/receive actions. A denotational tracebased model of global views is given in [30] (unlike ours, this semantics is based on synchronous communication and respects the sequential compositions of projections on well-formed global views (Theorem 2 in [30]). In this category we also put approaches like [12, 13] where global artefacts become global programs with an operational semantics. In this area authors have proposed to mitigate the limitations usually present in languages for global views using operational approaches [6, 24, 13, 8].

The classes above contain perfectly reasonable approaches. After all, from a theoretical perspective, we just need a semantics for the global view; whatever "fits" with the semantics of the local view would do. We believe that an abstract semantics of global view like the one we propose here makes it easier reason about global artefacts and to develop the algorithmic and tool support for such reasoning $4_{4}^{4}$ Trace-based (denotational or operational) semantics of global views are less suitable in this respect because they require the designer to consider the dynamics of the interactions in terms of (an approximation of) their execution. In other words, existing semantics of global views are "too concrete" and require to manipulate traces in order to capture relevant aspects of the design. For instance, many approaches take traces up-to permutations of non causally-related events to "mimick" asynchrony at the global level. Another drawback present in some of the approaches based on traces is that they make the semantics of the global view a dependent variable of the semantics of the local one brings in the following issues. For instance, to guarantee some properties it is sometimes necessary to adopt (syntactic) restrictions imposed by the underlying communication infrastructure. In several cases these restrictions limit the expressiveness of the language at hand (for instance, languages featuring the parallel composition of global artefacts do not allow components involved in more than one parallel thread). Another example of such dependency is the introduction in the semantics of the global views of low-level elements (e.g., to guarantee order-preserving asynchronous outputs).

To sum up, we suggest that an abstract semantics of global views yields a more suitable framework to foster the interplay between global and local views described in Section 1

[^4]
## 3. Global views as graphs

Let $\mathcal{P}$ be a set of participants (ranged over by $\mathrm{A}, \mathrm{B}$, etc.), $\mathcal{M}$ a set of messages (ranged over by $\mathrm{m}, \mathrm{x}$, etc.), and $\mathcal{K}$ a set of control points (ranged over by $\mathrm{i}, \mathrm{j}$, etc.). We take $\mathcal{P}, \mathcal{M}$, and $\mathcal{K}$ pairwise disjoint. The participants of a choreography exchange messages to coordinate with each other. In the global view, this is modelled with interactions $\rrbracket^{5} \mathrm{~A} \xrightarrow{m} \mathrm{~B}$, representing the fact that participant A sends message $m$ to participant $B$, which is expected to receive $m$. A global choreography (g-choreography for short) is a term G derived by the following grammar (recursion/iteration is omitted for simplicity as discussed in Section 7 )

$$
\begin{equation*}
\mathrm{G}::=\mathbf{0}|\mathrm{i}: \mathrm{A} \xrightarrow{\mathrm{~m}} \mathrm{~B}| \mathrm{G} ; \mathrm{G}^{\prime}\left|\mathrm{i}:\left(\mathrm{G} \mid \mathrm{G}^{\prime}\right)\right| \mathrm{i}:\left(\mathrm{G}+\mathrm{G}^{\prime}\right) \tag{2}
\end{equation*}
$$

A g-choreography can be empty, a simple interaction, the sequential or parallel composition of g-choreographies, or the choice between two g-choreographies. We implicitly assume $A \neq B$ in interactions $\mathrm{i}: A \xrightarrow{m} B$. In (2), control points, denoted by $i$, tag nodes of $g$-choreographies that correspond to events of interest. For instance, the control point of an interaction identifies its output and input events, the one of a fork (resp. choice) identifies when a g-choreography splits in more threads (resp. branches). Note that the empty g-choreography is not tagged with a control point because it does not have any event; likewise for sequential composition G ; $\mathrm{G}^{\prime}$ whose events are already completely determined by the control points of $G$ and $\mathrm{G}^{\prime}$. We assume that in a g-choreography $G$ any two control points occurring in different positions are different, e.g., we cannot write $\mathrm{i}:(\mathrm{j}: \mathrm{A} \xrightarrow{\mathrm{m}} \mathrm{B} \mid \mathrm{i}: \mathrm{C} \xrightarrow{\mathrm{y}} \mathrm{D})$. Let $\mathcal{G}$ be the set of g-choreographies and, for $\mathrm{G} \in \mathcal{G}$, let $\mathrm{cp}(\mathrm{G})$ denote the set of control points in $G$. Control points are a technical device and could be avoided $]^{6}$ The concrete representation of control point is immaterial; throughout the paper, for all $G \in \mathcal{G}$, we will assume that $\mathrm{cp}(\mathrm{G})$ are strictly positive integers. Also, we may omit control points when immaterial, e.g., writing $G+G^{\prime}$ instead of $i:\left(G+G^{\prime}\right)$.

The syntax in (22) captures the structure of a visual language of acyclic ${ }^{7}$ directed graphs. In fact, each g-choreography G can be represented as a rooted graph with a single "enter" (resp. "exit") node; namely, G has a distinguished source (resp. sink) node that can reach (resp. be reached by) any other node in G. Before commenting on our visual notation in Fig. 1, we remark that each fork or branch gate with control point i in our pictures will have a corresponding join and merge gate with control point - i ; note that negative control points will appear only in the visual notation of a global graph and not in its textual representation. This will also be used in Section 6 .

In Fig. 1, dotted edges connect nodes • to boxed G to identify source and sink nodes of G. More precisely, a dotted edge from • to a boxed G means

[^5]

Figure 1: Our graphs: $\circ$ is the source node, © the sink one; other nodes are drawn as $\bullet$
that • is the source of G; similarly, a dotted edge from a boxed G to • means that $\bullet$ is the sink of $G$. For instance, the dotted edges entering and leaving the g-choreography G in the sequential composition in Fig. 1 respectively identify the source and sink nodes of $G$ while the other dotted edges identifies the source and sink nodes of $\mathrm{G}^{\prime}$; in other words, the sequential composition of G and $\mathrm{G}^{\prime}$ is obtained by coalescing the sink of $G$ with the source of $\mathrm{G}^{\prime}$. Figs. 2a and 2b give an example of this construction for sequential and parallel composition respectively (in the latter figure we omitted the control points of interactions for readability). Fig. 2a shows why there is no need to assign a control point to sequential composition: there is no interesting event at the coalescing • node. Indeed, the pattern $\rightarrow \bullet \rightarrow$ in a graph does not yield any important event to trace and in the following we will often replace it with a simple edge $\rightarrow$ ). Note that the graph $]^{8} G_{\sqrt{2 b}}$ in Fig. 2 b represents a choreography where $A$ sends $B$ messages $m$ and $n$ in any order. Akin to BPMN [17] diagrams, our graphs yield a visual description of the distributed coordination of communicating components. In this respect, control points mark the nodes of the graph where communication and distributed work flow activities may happen. This is similar to BPMN where tasks (corresponding to our communication activities) and control gates (corresponding to our fork/join and branch/merge gates) have a special standing in the graphical notation.

Our graphs resemble the ones in [15, 26] the only differences being that

- by construction, forking and branching control points i have a corresponding join and merge control point -i;
- there is a unique sink control point with a unique incoming edge (as in [15, 26, there is also a unique source control point with a unique outgoing edge).

[^6]

Figure 2: Examples of choreographies

We now consider a few examples to give an intuition of g-choreographies and of the problems arising when defining their semantics. Indeed, not all gchoreographies of our language have a defined semantics. In particular, sequential and non-deterministic composition of choreographies can lead to choreographies that are impossible to realise. We will also use these examples through the paper to highlights some aspects of our constructions.

The problem of giving semantics to the sequential composition can be illustrated with the graph $G_{[2 a)}$. The graph specifies that the interaction between A and B must be followed (in some sense) by an interaction between C and D. Intuitively, this is not directly realisable in a distributed setting. In fact, neither the participant $C$ nor $D$ are informed about the termination of the interaction $A \xrightarrow{m} B$, making it impossible to respect the order of the interactions without further communication between A or B and C or D. In Section 4.2 we introduce the notion of well-sequencedness (cf. Definition 6) to identify sequential composition of choreographies that are meaningful. Typically, this problem is addressed by imposing syntactic constraints on sequential composition so to obtain the intended causal order on output and input events. Our notion of well-sequencedness is given instead at the semantic level so to accommodate more cases. Intuitively, we will require that $\mathrm{G} ; \mathrm{G}^{\prime}$ is defined when the causal order it generates allows each participant to ascertain when its execution in G is completed and the one in $\mathrm{G}^{\prime}$ can start.

We now turn our attention to non-deterministic g-choreographies. The graph $\mathrm{G}_{\sqrt[3]{3}}$ contains a sound choice: participant A decides which message (between $x$ and $y$ ) is delivered to $B$. A further example of correct choreography is $G^{3 b}$. Here, participant A decides which branch is taken and informs either B or C, which in turn notifies the third participant. Intuitively, participant B can wait for a message coming from A or C and identify the selected branch accordingly. Let us first consider two correct non-deterministic g-choreographies that are only partially accommodated in the literature. The first interesting case is $G_{\text {3c }}$. Intuitively, no difference can be observed between $G_{\boxed{3 c}}$ and $A \xrightarrow{m} B$, thus every participant complying with $A \xrightarrow{m} B$ is also complying with $G_{3 \mathrm{Bc}}$. Another

(a) $G_{\sqrt[3 a]{ }}=i:(\underbrace{A \xrightarrow{x} B}_{G_{1}}+\underbrace{A \xrightarrow{y} B}_{G_{2}})$
(b) $\mathrm{G}_{\sqrt{3 \mathrm{~b}}}=\underbrace{\mathrm{A} \xrightarrow{\mathrm{x}} \mathrm{B} ; \mathrm{B} \xrightarrow{\mathrm{z}} \mathrm{C}}_{\mathrm{G}_{1}}+\underbrace{\mathrm{A} \xrightarrow{\mathrm{y}} \mathrm{C} ; \mathrm{C} \xrightarrow{\mathrm{w}} \mathrm{B}}_{\mathrm{G}_{2}}$

(c) $G_{(3 \mathrm{C})}=\mathrm{i}:(\underbrace{\mathrm{A} \xrightarrow[m]{m} B}_{\mathrm{G}_{1}}+\underbrace{\mathrm{A} \rightarrow}_{\mathrm{G}_{1}} \mathrm{~B})$
(d) $\mathrm{G}_{\underline{3 \mathrm{~d}}}=\underbrace{\mathrm{A} \xrightarrow{m} \mathrm{~B} ; \mathrm{B} \xrightarrow{x} \mathrm{C}}_{\mathrm{G}_{1}}+\underbrace{\mathrm{A} \xrightarrow{\mathrm{m}} \mathrm{B} ; \mathrm{B} \xrightarrow{\mathrm{y}} \mathrm{C}}_{\mathrm{G}_{2}}$

Figure 3: Correct non-deterministic g-choreographies
example of correct choreography is the graph $\mathrm{G}_{\sqrt[3 \mathrm{~d}]{ }}$, where the two branches have the same first interaction. Here $A$ has the same behaviour in both $G_{1}$ and $G_{2}$, $B$ decides which message is delivered to $C$, and $C$ is informed about the choice taken by B via the reception of different messages. These simple cases are sometimes tackled in the literature by merging the branches of a choice. However, these type of (partial) operations have been defined in literature by constraining the parallelism in the branches of choices [7]. Note also that in our framework in G ; $\mathrm{G}^{\prime}$ and in $\mathrm{G}+\mathrm{G}^{\prime}$ both G and $\mathrm{G}^{\prime}$ can be parallel choreographies, matching the level of generality of the choreographic languages in [6, 7] and in [24, however, our notion of well-branchedness is more general than the corresponding notions there.

The following examples discuss and anticipate some of the problems related to non-deterministic composition of $g$-choreographies. Choreography $G_{[4 a]}$ provides an example of unsound choices. Participant A decides to which participant the message $x$ is delivered. The participant that is not selected by A has no way to identify if/when the choice has been made. This will force that participant to wait indefinitely for the message $x$. Another example of incorrect choreography

(a) $G \xrightarrow[4 a]{ }=i:(\underbrace{A \xrightarrow{x} B}_{G_{1}}+\underbrace{A \stackrel{\times}{\rightarrow} C}_{G_{2}})$

(b) $G_{4 \mathrm{~b}}=\mathrm{i}:(\underbrace{\mathrm{A} \xrightarrow{\mathrm{x}} \mathrm{B}}_{\mathrm{G}_{1}}+\underbrace{\mathrm{C}}_{\mathrm{G}_{2}} \stackrel{\times}{\rightarrow} \mathrm{D})$

Figure 4: Incorrect non-deterministic choices
is $G_{(4 b}$, since it requires both $A$ and $C$ to commit to a distributed choice without any communication among them.

The choreography $\mathbb{G}_{5}$ illustrates a complex choreography with nested choices. Here, the participant making all the choices is $A$. However, $B$ and $C$ receive different information about these choices. The behaviour of $B$ is uniform in all branches of $G_{1}$ (where it always receives I from $A$ ) as well as in all those of $G_{2}$ (where it always receives $r$ from $A$ ). The behaviour of $C$ is more complex. The first message sent by $A$ is either $h$ or $y_{1}$. If $h$ is received, $C$ discovers that the left branch of either $G_{1}$ or $G_{2}$ has been selected: in both cases all the tasks of $C$ in the choreography have been completed. If instead $y_{1}$ is received, $C$ discovers that the right branch of either $G_{1}$ or $G_{2}$ has been selected: in all cases $C$ waits for the reception of the message $y_{2}$ from $A$ and uses the subsequent message $\left(z_{1}\right.$, $z_{2}$ or $z_{3}$ ) to identify the choice made by $A$.

In Section 4.2 we introduce the notion of well-branchedness to identify nondeterministic composition of choreographies that are meaningful.

## 4. Pomset-based semantics of choreographies

The basic idea to provide a semantics to g-choreographies is to use collections of pomsets in order to account for the casual dependencies among communications that are introduced by different alternatives of choices.

### 4.1. Preliminaries: pomsets for global graphs

In our framework, interactions are based on actions that happen on channels, which we identify by the names of the participants involved in the communication. Formally, a channel is an element of the set $\mathcal{C}=\mathcal{P}^{2} \backslash\{(A, A) \mid A \in \mathcal{P}\}$ and we abbreviate $(A, B) \in \mathcal{C}$ as $A B$. The set of labels $\mathcal{L}$ is defined by

$$
\mathcal{L}=\mathcal{L}^{!} \cup \mathcal{L}^{?} \quad \text { where } \quad \mathcal{L}^{!}=\mathcal{C} \times\{!\} \times \mathcal{M} \quad \text { and } \quad \mathcal{L}^{?}=\mathcal{C} \times\{?\} \times \mathcal{M}
$$



$$
\begin{aligned}
& \mathrm{G}^{5}=\mathrm{i}:\left(\mathrm{G}_{1}+\mathrm{G}_{2}\right)
\end{aligned}
$$

$$
\begin{aligned}
& \mathrm{G}_{2}=\quad \mathrm{i}^{\prime \prime}:\left(\begin{array}{c}
\mathrm{A} \xrightarrow{\mathrm{r}} \mathrm{~B} ; \mathrm{A} \xrightarrow{\mathrm{~h}} \mathrm{C} \\
+ \\
\mathrm{i}^{\prime \prime \prime}:\left(\mathrm{G}_{2 a}+\mathrm{G}_{2 b}\right)
\end{array}\right) \\
& \mathrm{G}_{2 a}=\quad \begin{array}{l}
\mathrm{A} \xrightarrow{\mathrm{r}} \mathrm{~B} ; \mathrm{A} \xrightarrow{\mathrm{y}_{1}} \mathrm{C} ; \\
\mathrm{A} \xrightarrow{y_{2}} \mathrm{C} ; \mathrm{A} \xrightarrow{\mathrm{z}_{2}} \mathrm{C}
\end{array} \\
& \mathrm{G}_{2 b}=\begin{array}{l}
\mathrm{A} \xrightarrow{\mathrm{r}} \mathrm{~B} ; \mathrm{A} \xrightarrow{\mathrm{y}_{1}} \mathrm{C} ; \\
\mathrm{A} \xrightarrow{\mathrm{y}_{2}} \mathrm{C} ; \mathrm{A} \xrightarrow{\mathrm{z}_{2}} \mathrm{C}
\end{array}
\end{aligned}
$$

Figure 5: A complex choreography

The elements of $\mathcal{L}^{!}$and $\mathcal{L}^{\text {? }}$, outputs and inputs, respectively represent sending and receiving actions; we shorten $(A B,!, m)$ as $A B!m$ and $(A B, ?, m)$ as $A B ? m$ and let $l, l^{\prime}, \ldots$ range over $\mathcal{L}$. The subject of an action is defined by

$$
\operatorname{sbj}(A B!m)=A \quad(\text { the sender }) \quad \text { and } \quad \operatorname{sbj}(A B ? m)=B \quad(\text { the receiver })
$$

In the following, we write $l \in G$ when there is an interaction $A \xrightarrow{m} B$ in the g-choreography $G$ such that $l \in\{\mathrm{AB}!\mathrm{m}, \mathrm{AB}$ ? m$\}$, and accordingly $\tilde{l} \subseteq G$ means that $l \in \mathrm{G}$ for all $l \in \tilde{l}$.

We reuse the formalisation of partially-ordered multi-set of 16 later used by [22] to give semantics to message-sequence charts.

Definition 1. A labelled partially-ordered set (lposet) is a triple $(\mathcal{E}, \leq, \lambda)$, with $\mathcal{E}$ a set of events, $\leq \subseteq \mathcal{E} \times \mathcal{E}$ a reflexive, anti-symmetric, and transitive relation on $\mathcal{E}$, and $\lambda: \mathcal{E} \rightarrow \mathcal{L}$ a labelling function.

For $e \neq e^{\prime}, \lambda(e)=\lambda\left(e^{\prime}\right)$ means that $e$ and $e^{\prime}$ model different occurrences of the same action. Intuitively, $\leq$ represents causality; for $e \neq e^{\prime}$, if $e \leq e^{\prime}$ and both events occur then $e^{\prime}$ is caused by $e$. In the following, we use $\varepsilon$ to denote the empty lposet and $e \rightarrow e^{\prime}$ to denote that $e \leq e^{\prime}$.
Definition 2. Two lposets $(\mathcal{E}, \leq, \lambda)$ and $\left(\mathcal{E}^{\prime}, \leq^{\prime}, \lambda^{\prime}\right)$ are isomorphic iff there exists a bijection $\phi: \mathcal{E} \rightarrow \mathcal{E}^{\prime}$ such that $e \leq e^{\prime}$ iff $\phi(e) \leq^{\prime} \phi\left(e^{\prime}\right)$ and $\lambda=\lambda^{\prime} \circ \phi$.
Definition 3. A partially-ordered multi-set (of actions), pomset for short, is an isomorphism class of lposets.


Figure 6: Examples of pomsets

The main benefit of using pomset in place of lposets is that we can abstract from the mathematical structure used to define the events $\mathcal{E}$, freeing us from introducing canonical representations of lposets. In the following, $[\mathcal{E}, \leq, \lambda]$ denotes the isomorphism class of $(\mathcal{E}, \leq, \lambda)$, we let $r, r^{\prime}, \ldots$ (resp. $\mathcal{R}, \mathcal{R}^{\prime}, \ldots$ ) range over pomsets (resp. sets of pomsets), and we assume that any $r$ contains at least one lposet which will possibly be referred to as $\left(\mathcal{E}_{r}, \leq_{r}, \lambda_{r}\right)$. Examples of pomsets are depicted in Fig. 6, where events are not explicitly shown as their labels univocally determine them. Intuitively, $r_{[6 a]}$ establishes a total causal order from the left-most to the bottom right-most event; $r_{\sqrt{6 b}}$ represents the pomset of the parallel execution of two threads. In Fig. 6. $\mathcal{\mathcal { R }}_{\sqrt{6 c}}$ represents a set of two pomsets, where each pomset defines the causal order of events that corresponds to a choice made by the participant $A$; in one case $A$ sends message $x$, in the other case A sends message $y$.

In the following, given a natural number $n, \mathbf{n}$ represents the singleton $\{n\}$. Also, we use $X \uplus Y$ to represent the disjoint union of two sets $X$ and $Y$ : $X \uplus Y=(X \times \mathbf{1}) \cup(Y \times \mathbf{2})$. Finally, given a function $f$ on $X$, we define $f \otimes \mathbf{n}=\{(x, n) \mapsto f(x) \mid x \in X\}$ as the function extending $f$ to $X \times \mathbf{n}$; analogously, for a relation $R \subseteq X \times Y$, we let $R \otimes \mathbf{n}=\{((x, n),(y, n)) \mid(x, y) \in R\}$ be the relation extending $R$ to $(X \times \mathbf{n}) \times(Y \times \mathbf{n})$.

We now define two important constructions to compose pomsets in parallel and sequentially.

Definition 4. Let $r=[\mathcal{E}, \leq, \lambda]$ and $r^{\prime}=\left[\mathcal{E}^{\prime}, \leq^{\prime}, \lambda^{\prime}\right]$ be two pomsets. The parallel composition of $r$ and $r^{\prime}$ is:

$$
\operatorname{par}\left(r, r^{\prime}\right)=\left[\mathcal{E} \uplus \mathcal{E}^{\prime},(\leq \otimes \mathbf{1}) \cup\left(\leq^{\prime} \otimes \mathbf{2}\right),(\lambda \otimes \mathbf{1}) \cup\left(\lambda^{\prime} \otimes \mathbf{2}\right)\right]
$$

For a pomset $r$ and a participant $\mathrm{A} \in \mathcal{P}$, let $\mathcal{E}_{r, \mathrm{~A}}=\left\{e \in \mathcal{E}_{r} \mid \operatorname{sbj}\left(\lambda_{r}(e)\right)=\mathrm{A}\right\}$ be the set of events of A in $\mathcal{E}_{r}$. The sequential composition of $r$ and $r^{\prime}$ is:

$$
\operatorname{seq}\left(r, r^{\prime}\right)=\left[\mathcal{E} \uplus \mathcal{E}^{\prime}, \leq_{\text {seq }},(\lambda \otimes \mathbf{1}) \cup\left(\lambda^{\prime} \otimes \mathbf{2}\right)\right]
$$

where

$$
\leq_{s e q}=\left((\leq \otimes \mathbf{1}) \cup\left(\leq^{\prime} \otimes \mathbf{2}\right) \cup \bigcup_{\mathrm{A} \in \mathcal{P}}\left(\left(\mathcal{E}_{r, \mathrm{~A}} \times \mathbf{1}\right) \times\left(\mathcal{E}_{r^{\prime}, \mathrm{A}} \times \mathbf{2}\right)\right)\right)^{\star}
$$

and $\star$ is the reflexive-transitive closure.


Figure 7: Examples of sequential composition

Both parallel and sequential composition preserve the causal dependencies of its constituents $\leq$ and $\leq^{\prime}$. However, while there is no new dependency introduced by the parallel composition, the sequential composition of two pomsets adds to those in $\leq$ and $\leq^{\prime}$ the dependencies among events in $r$ and $r^{\prime}$ with the same subject. Basically, a causal relation is induced whenever a participant performing a communication in $r$ also performs a communication in $r^{\prime}$. Fig. 7 depicts the sequential compositions of two pomsets, say $r$ and $r^{\prime}$. The former pomset corresponds to the interaction $A \xrightarrow{\times} B$, while the second ranges over the interactions

$$
\mathrm{A} \xrightarrow{\mathrm{y}} \mathrm{C}, \quad \mathrm{~B} \xrightarrow{\mathrm{y}} \mathrm{C}, \quad \mathrm{C} \xrightarrow{\mathrm{y}} \mathrm{~B}, \quad \mathrm{~A} \xrightarrow{\mathrm{y}} \mathrm{~B}, \quad \text { and } \quad \mathrm{C} \xrightarrow{\mathrm{y}} \mathrm{D}
$$

Simple arrows represent the dependencies induced by the subjects and dotted arrows represent dependencies induced by the sequential composition (the meaning of stroken arrows will be explained in Section 4.2).

### 4.2. Semantics of choreographies

The semantics of a choice-free g-choreography $G \in \mathcal{G}$ (i.e. a choreography that does not contain $+_{+}$terms) can be expressed using a partial order, which represents the causal dependencies of the communication actions specified by G. Choices are a bit more tricky. Intuitively, the semantics of $G+G^{\prime}$ consists of two (sets of) partial orders, one representing the causal dependencies of the communication actions of G and the other of those of $\mathrm{G}^{\prime}$. Therefore, the semantics
of a g-choreography is a family of pomsets defined as

$$
\begin{aligned}
\llbracket \mathbf{0} \rrbracket & =\{\varepsilon\} \\
\llbracket \mathrm{A} \stackrel{\mathrm{~m}}{\rightarrow} \mathrm{~B} \rrbracket & =\left\{\left[\left(\left\{e_{1}, e_{2}\right\},\left\{\left(e_{1}, e_{1}\right),\left(e_{2}, e_{2}\right),\left(e_{1}, e_{2}\right)\right\}, \lambda\right)\right]\right\} \text { where } \lambda: \begin{cases}e_{1} \mapsto \mathrm{AB}!\mathrm{m} \\
e_{2} \mapsto \mathrm{AB} ? \mathrm{~m}\end{cases} \\
\llbracket \mathrm{G} \mid \mathrm{G}^{\prime} \rrbracket & = \begin{cases}\left\{\operatorname{par}\left(r, r^{\prime}\right) \mid\left(r, r^{\prime}\right) \in \llbracket \mathrm{G} \rrbracket \times \llbracket \mathrm{G}^{\prime} \rrbracket\right\} & \text { if } w f\left(\mathrm{G}, \mathrm{G}^{\prime}\right) \\
\perp & \text { otherwise }\end{cases} \\
\llbracket \mathrm{G} ; \mathrm{G}^{\prime} \rrbracket & = \begin{cases}\left\{\operatorname{seq}\left(r, r^{\prime}\right) \mid\left(r, r^{\prime}\right) \in \llbracket \mathrm{G} \rrbracket \times \llbracket \mathrm{G}^{\prime} \rrbracket\right\} & \text { if } w s\left(\mathrm{G}, \mathrm{G}^{\prime}\right) \\
\perp & \text { otherwise }\end{cases} \\
\llbracket \mathrm{G}+\mathrm{G}^{\prime} \rrbracket & = \begin{cases}\llbracket \mathrm{G} \rrbracket \cup \llbracket \mathrm{G}^{\prime} \rrbracket & \text { if } w b\left(\mathrm{G}, \mathrm{G}^{\prime}\right) \\
\perp & \text { otherwise }\end{cases}
\end{aligned}
$$

where $w f\left(\mathrm{G}, \mathrm{G}^{\prime}\right), w s\left(\mathrm{G}, \mathrm{G}^{\prime}\right)$, and $w b\left(\mathrm{G}, \mathrm{G}^{\prime}\right)$ in the last three clauses check that parallel, sequential, and non-deterministic composition of choreographies are meaningful. Before defining those predicates we discuss the other cases. The semantics of the empty g-choreography $\mathbf{0}$ and of interaction $A \xrightarrow{m} B$ are straightforward; for the latter, the send part $A B!m$ of the interaction must precede its receive part $A B ? m$.

For the parallel composition $G \mid G^{\prime}$ we take the union of the dependencies of every possible execution, thus allowing the concurrent occurrence of the events of each thread. We also require that input events of $G$ and $\mathrm{G}^{\prime}$ are disjoint. Formally,

Definition 5. Pomsets $r=[\mathcal{E}, \leq, \lambda]$ and $r^{\prime}=\left[\mathcal{E}^{\prime}, \leq^{\prime}, \lambda^{\prime}\right]$ are well-forked if

$$
\lambda(\mathcal{E}) \cap \lambda^{\prime}\left(\mathcal{E}^{\prime}\right) \cap \mathcal{L}^{?}=\emptyset
$$

We write $w f\left(r, r^{\prime}\right)$ when $r$ and $r^{\prime}$ are well-forked and, for $\mathrm{G}, \mathrm{G}^{\prime} \in \mathcal{G}, w f\left(\mathrm{G}, \mathrm{G}^{\prime}\right)$ when $\llbracket \mathrm{G} \rrbracket \neq \perp \wedge \llbracket \mathrm{G}^{\prime} \rrbracket \neq \perp \wedge \forall r \in \llbracket \mathrm{G} \rrbracket, r^{\prime} \in \llbracket \mathrm{G}^{\prime} \rrbracket: w f\left(r, r^{\prime}\right)$

Parallel composition of the pomsets of $G_{1}$ and $G_{2}$ preserves the order relations of parallel threads. Additionally, well-forkedness ensures that the actions corresponding to the events in one thread cannot be confused with those in other threads. When well-forkedness does not hold, the pomsets induced by $G_{1}$ and $G_{2}$ would yield an order on shared inputs that would be too strict. This is illustrated by considering the the choreography $G_{[8]}=G_{\sqrt[8 a]{ }} \mid G_{\sqrt[8 b]{ }}$, where $G_{\sqrt[8 a]{ }}$ and $G_{\sqrt[8 b]{ }}$ are given in Fig. 8 together with their semantics. In the parallel composition, the interaction $A \xrightarrow{x} B$ is shared between the two threads. The semantics of $G_{8}$ should be the parallel composition of $\llbracket G_{\boxed{8 a}} \rrbracket$ and $\llbracket G_{[8 b]} \rrbracket$, without any further dependencies among the events of the two pomsets. Notice that, independently of the interleaving of the constituent threads, the event AC! $r_{1}$ must always precede the event $B C!r_{2}$. However, this property cannot be enforced because an implementation of $\mathrm{G}_{8}$ can proceed as follows

1. the left thread of $A$ executes $A C!l_{1}$ and $A B!x$


Figure 8: A non well-forked choreography
2. the right thread of $B$ executes $A B$ ? $x$, "stealing" the message $\times$ generated by the left thread of $A$ and meant for the left thread of $B$
3. the right thread of $B$ executes $B C!r_{2}$.

Informally, the participant $B$ cannot ascertain if the reception of $A B$ ?m belongs to the first or second thread, therefore he cannot decide which message between $I_{2}$ and $r_{2}$ should be delivered.

The semantics of sequential composition G; $\mathrm{G}^{\prime}$ establishes happens-before relations as computed by seq $\left(r, r^{\prime}\right)$ provided that they cover the dependencies between the output events of G and the input events of $\mathrm{G}^{\prime}$. Formally,

Definition 6. Pomsets $r=[\mathcal{E}, \leq, \lambda]$ and $r^{\prime}=\left[\mathcal{E}^{\prime}, \leq^{\prime}, \lambda^{\prime}\right]$ are well-sequenced if

$$
\leq_{\mathrm{seq}\left(r, r^{\prime}\right)} \supseteq\left(\left\{e \in \mathcal{E} \mid \lambda(e) \in \mathcal{L}^{!}\right\} \times \mathbf{1}\right) \times\left(\left\{e \in \mathcal{E}^{\prime} \mid \lambda^{\prime}(e) \in \mathcal{L}^{?}\right\} \times \mathbf{2}\right)
$$

We write ws $\left(r, r^{\prime}\right)$ when $r$ and $r^{\prime}$ are well-sequenced and, for $\mathrm{G}, \mathrm{G}^{\prime} \in \mathcal{G}$, $w s\left(\mathrm{G}, \mathrm{G}^{\prime}\right)$ when $\llbracket \mathrm{G} \rrbracket \neq \perp \wedge \llbracket \mathrm{G}^{\prime} \rrbracket \neq \perp \wedge \forall r \in \llbracket \mathrm{G} \rrbracket, r^{\prime} \in \llbracket \mathrm{G}^{\prime} \rrbracket:$ ws $\left(r, r^{\prime}\right)$.

Well-sequencedness ensures the soundness of sequential composition; when it does not hold, there is a participant $A$ in $\mathrm{G}^{\prime}$ that cannot ascertain when to start its execution in $\mathrm{G}^{\prime}$. All examples in Fig. 7 are sound, bar the one in Fig. 7 7 , where the stroken edge depicts the missing causal dependency.

The semantics of a choice $G+\mathrm{G}^{\prime}$ is defined provided that the well-branched condition holds. The definition of well-branchedness relies on the notions of active and passive participants (defined below) that single out those participants that (internally) select which branch to execute and those participants that have to "understand" what choice was made according to the messages they receive, instead of internally choosing a branch.

Definition 7. Two g-choreographies G and $\mathrm{G}^{\prime}$ are well-branched when
(i) there is at most one active participant in $G+\mathrm{G}^{\prime}$ and
(ii) all the other participants of $\mathrm{G}+\mathrm{G}^{\prime}$ are passive.

We write $w b\left(\mathrm{G}, \mathrm{G}^{\prime}\right)$ to denote that G and $\mathrm{G}^{\prime}$ are well-branched.

As said earlier, in the semantics of $G+\mathrm{G}^{\prime}$ no additional dependencies are introduced other than the ones induced by either $G$ or $G^{\prime}$. In fact, either the actions of the first branch or the actions of the second one will be performed. The behaviour of two branches can be the same up to a point of divergence where some participants start to behave differently on the two branches. It is at points of divergence where we discriminate if a participant is active or passive.

As usual, $\left.\right|_{X}$ denotes the restriction of a function to a subset $X$ of its domain. In order to identify a point of divergence we need to consider the part of a pomset involving a participant. In the following, we fix a participant $A \in \mathcal{P}$.

Definition 8. Let $r=[\mathcal{E}, \leq, \lambda]$ be a pomset. The pomset

$$
\left.r\right|_{\mathrm{A}}=\left[\mathcal{E}_{r, \mathrm{~A}}, \leq \cap\left(\mathcal{E}_{r, \mathrm{~A}} \times \mathcal{E}_{r, \mathrm{~A}}\right),\left.\lambda\right|_{\mathcal{E}_{r, \mathrm{~A}}}\right]
$$

is the pomset of A in $r$; - $\mathrm{l}_{\mathrm{A}}$ extends to sets of pomsets element-wise.
A way of defining points of divergence of two branches is in terms of the events that differ in the two branches after "a common prefix", where a prefix of a pomset $r$ is a pomset $r^{\prime}$ on a subset of the events of $r$ that preserves the order and labelling of $r$; formally (following [22])

Definition 9. A pomset $r^{\prime}=\left[\mathcal{E}^{\prime}, \leq^{\prime}, \lambda^{\prime}\right]$ is a sub-pomset of pomset $r=[\mathcal{E}, \leq, \lambda]$ if

$$
\mathcal{E}^{\prime} \subseteq \mathcal{E} \quad \text { and } \quad \leq^{\prime}=\leq \cap\left(\mathcal{E}^{\prime} \times \mathcal{E}^{\prime}\right) \quad \text { and } \quad \lambda^{\prime}=\left.\lambda\right|_{\mathcal{E}^{\prime}}
$$

A sub-pomset $r^{\prime}$ of $r$ is a prefix of $r$ if $\leq \cap\left(\left(\mathcal{E} \backslash \mathcal{E}^{\prime}\right) \times \mathcal{E}^{\prime}\right)=\emptyset$.
Definition 10. The suffix of a pomset $r$ with respect to one of its prefixes $r^{\prime}$, denoted as $r-r^{\prime}$, is the pomset $\left[\mathcal{E}^{\prime}, \leq_{r} \cap\left(\mathcal{E}^{\prime} \times \mathcal{E}^{\prime}\right)\right.$, $\left.\lambda_{r} \mid \mathcal{E}^{\prime}\right]$, where $\mathcal{E}^{\prime}=\mathcal{E}_{r} \backslash \mathcal{E}_{r^{\prime}}$.

The semantics of each branch in a choice $G_{1}+G_{2}$ is, in general, a set of pomsets. Our definition relies on finding a "common part" of the branches for which participants behave uniformly in both branches $G_{1}$ and $G_{2}$ of the choice. To identify such common part we consider the behaviour of each participant A on the branches of the choice "in isolation"; namely, we analyse $\llbracket G_{1} \rrbracket l_{A}$ and $\llbracket G_{2} \rrbracket l_{A}$ that is the events on the branches that involve A. A complication in finding common prefixes is due to the fact that a pomset in $\llbracket G_{1} \rrbracket l_{A}$ could be "matched" by several pomsets in $\llbracket \mathrm{G}_{2} \rrbracket \mathrm{l}_{\mathrm{A}}$. To address this problem, we need to identify how the pomsets in $\llbracket G_{1} \rrbracket L_{A}$ correspond to those in $\llbracket G_{2} \rrbracket L_{A}$. For instance, for the participant $C$ in the example $G_{5 \text { ( }}$ (page 12 ), we need to relate the right branch of $i^{\prime}$ to both branches of $i^{\prime \prime \prime}$ because of the common prefix consisting of the reception of $y_{1}$ followed by the reception of $y_{2}$.

Definition 11. The pair of functions $(\phi, \psi)$ is an $A$-prefix map of $\mathrm{G}_{1}$ and $\mathrm{G}_{2}$ if

- $\operatorname{dom} \phi$ and $\operatorname{cod} \phi$ are partition $\Phi^{9}$ of $\llbracket \mathrm{G}_{1} \rrbracket l_{\mathrm{A}}$ and $\llbracket \mathrm{G}_{2} \rrbracket l_{\mathrm{A}}$ respectively, $\phi$ is bijective, and

[^7]- $\operatorname{dom} \psi=\operatorname{dom} \phi$ and $\operatorname{cod} \psi$ is a set of pomsets
such that, for all $\mathcal{R} \in \operatorname{dom} \phi$ and $\left(r, r^{\prime}\right) \in \mathcal{R} \times \phi(\mathcal{R})$, the pomset $\psi(\mathcal{R})$ is a prefix pomset of both $r$ and $r^{\prime}$.

Definition 12. Let $\tilde{l}_{1}$ and $\tilde{l}_{2}$ be two subsets of $\mathcal{L}$, we say that $\left(\tilde{l}_{1}, \tilde{l}_{2}\right)$ is the divergence point of $\mathrm{G}_{1}$ and $\mathrm{G}_{2}$ (or that $\mathrm{G}_{1}$ and $\mathrm{G}_{2}$ diverge at $\left(\tilde{l}_{1}, \tilde{l}_{2}\right)$ ) with respect to the A-prefix map $(\phi, \psi)$, denoted $\operatorname{div}_{\mathrm{A}}^{\phi, \psi}\left(\mathrm{G}_{1}, \mathrm{G}_{2}\right)=\left(\tilde{l}_{1}, \tilde{l}_{2}\right)$ if

$$
\tilde{l}_{1}=\bigcup_{\substack{\mathcal{R} \in \operatorname{dom} \phi, r \in \mathcal{R}}} \lambda_{r}(\min (r-\psi(\mathcal{R}))) \quad \text { and } \quad \tilde{l}_{2}=\bigcup_{\substack{\mathcal{R} \in \operatorname{cod} \phi, r \in \mathcal{R}}} \lambda_{r}\left(\min \left(r-\psi\left(\phi^{-1}(\mathcal{R})\right)\right)\right)
$$

where, for a pomset $r, \min r=\left\{e \in \mathcal{E}_{r} \mid \nexists e^{\prime} \in \mathcal{E}_{r}: e^{\prime} \neq e \wedge e^{\prime} \leq_{r} e\right\}$.
The bijection $\phi$ "matches" sets of executions (pomsets) in $\llbracket G_{1} \rrbracket l_{A}$ with those in $\llbracket G_{2} \rrbracket L_{A}$ that have an initial common behaviour determined by the common prefix identified by the mapping $\psi$. Thus, $\tilde{l}_{1}$ (resp. $\tilde{l}_{2}$ ) is the union of the labels at the point of divergence for $\llbracket G_{1} \rrbracket l_{A}$ (resp. $\left.\llbracket G_{2} \rrbracket l_{A}\right)$, i.e. the union of the labels of the minimal events of the pomsets in $\llbracket G_{1} \rrbracket L_{A}$ (and $\llbracket G_{2} \rrbracket L_{A}$ ) after removing the common prefixes. We use the examples of Sections 3 and 4 to demonstrate the notion of divergence points.

Example $\mathrm{G}_{\text {Ba }}$. For participant A we have $\mathcal{R}_{1}=\llbracket \mathrm{G}_{1} \rrbracket \mathrm{~L}_{\mathrm{A}}=\{[\mathrm{AB}!\mathrm{x}]\}$ and $\mathcal{R}_{2}=$ $\left.\llbracket G_{2} \rrbracket\right\rfloor_{A}=\{[A \bar{B}!y]\}$. Since $\mathcal{R}_{1}$ and $\mathcal{R}_{2}$ are singletons, each of them has exactly one partition. Let $\phi=\mathcal{R}_{1} \mapsto \mathcal{R}_{2}$ and $\psi=\mathcal{R}_{1} \mapsto \varepsilon$ (i.e. the empty pomset, which is the only common prefix) then $\operatorname{div}_{A}^{\phi, \psi}\left(G_{1}, G_{2}\right)=(\{A B!x\},\{A B!y\})$. For participant $\mathrm{B}, \mathcal{R}_{1}=\llbracket \mathrm{G}_{1} \rrbracket \mathrm{~L}_{\mathrm{B}}=\{[\mathrm{AB} ? \mathrm{x}]\}$ and $\mathcal{R}_{2}=\llbracket \mathrm{G}_{2} \rrbracket \mathrm{~L}_{\mathrm{B}}=\{[\mathrm{AB}$ ? y$]\}$. Let $\phi=\mathcal{R}_{1} \mapsto \mathcal{R}_{2}$ and $\psi=\mathcal{R}_{1} \mapsto \varepsilon$ then $\operatorname{div}_{B}^{\phi, \psi}\left(\mathrm{G}_{1}, \mathrm{G}_{2}\right)=(\{\mathrm{AB} ? \mathrm{x}\},\{\mathrm{AB} ? \mathrm{y}\})$.

Example $\mathrm{G}_{\text {(3b }}$. Similarly to the previous example, participant A diverges at $(\{A B!x\},\{A C!y\})$. For participant $B, \mathcal{R}_{1}=\llbracket G_{1} \rrbracket L_{B}=\{[A B ? x \rightarrow B C!z]\}$ and $\mathcal{R}_{2}=\llbracket \mathrm{G}_{2} \rrbracket \mathrm{~L}_{\mathrm{B}}=\{[\mathrm{CB} ? \mathrm{w}]\}$, with $\phi=\mathcal{R}_{1} \mapsto \mathcal{R}_{2}$ and $\psi=\mathcal{R}_{1} \mapsto \varepsilon$ we obtain $\operatorname{div}_{\mathrm{B}}^{\phi, \psi}\left(\mathrm{G}_{1}, \mathrm{G}_{2}\right)=(\{\mathrm{AB}$ ? $\times\},\{\mathrm{CB}$ ? w $\})$. For participant $\mathrm{C}, \mathcal{R}_{1}=\llbracket \mathrm{G}_{1} \rrbracket \mathrm{l}_{\mathrm{c}}=\{[\mathrm{BC} ? \mathrm{z}]\}$ and $\mathcal{R}_{2}=\llbracket \mathrm{G}_{2} \rrbracket \mathrm{Lc}=\{[\mathrm{AC}$ ? $\mathrm{y} \rightarrow \mathrm{CB}!\mathrm{w}]\}$. Let $\phi=\mathcal{R}_{1} \mapsto \mathcal{R}_{2}$ and $\psi=\mathcal{R}_{1} \mapsto \varepsilon$ then $\operatorname{div}_{\mathrm{B}}^{\phi, \psi}\left(G_{1}, G_{2}\right)=(\{B C ? z\},\{A C ? y\})$.

Example $\mathrm{G}_{\text {(3c) }}$. For participant A, we have two cases depending on which prefix map we consider. Note that $\mathcal{R}_{1}=\mathcal{R}_{2}=\llbracket \mathrm{G}_{1} \rrbracket \mathrm{~L}_{\mathrm{A}}=\llbracket \mathrm{G}_{2} \rrbracket \mathrm{~L}_{\mathrm{A}}=\{r\}$, where $r=[\mathrm{AB}!\mathrm{m}]$, and let $\phi=\mathcal{R}_{1} \mapsto \mathcal{R}_{2}$. If we choose the empty prefix (that is $\psi=\mathcal{R}_{1} \mapsto \boldsymbol{\varepsilon}$ ) the point of divergence can be computed as $\operatorname{div}_{A}^{\phi, \psi}\left(G_{1}, G_{2}\right)=(\{A B!m\},\{A B!m\})$. $A$ different point of divergence can be computed using the prefix $r$ : let $\psi=\mathcal{R}_{1} \mapsto r$ then $\operatorname{div}_{A}^{\phi, \psi}\left(G_{1}, G_{2}\right)=(\emptyset, \emptyset)$. Similarly, for the participant $B, \mathcal{R}_{1}=\mathcal{R}_{2}=$ $\llbracket G_{1} \rrbracket \rrbracket_{B}=\llbracket G_{2} \rrbracket \rrbracket_{B}=\{r\}$, where $r=[\mathrm{AB}$ ? m$]$. According with the prefix chosen, two different points of divergence can be computed. For the non-empty prefix: let $\phi=\mathcal{R}_{1} \mapsto \mathcal{R}_{2}$ and $\psi=\mathcal{R}_{1} \mapsto r$ then $\operatorname{div}_{B}^{\phi, \psi}\left(\mathrm{G}_{1}, \mathrm{G}_{2}\right)=(\emptyset, \emptyset)$.

Example $\mathrm{G}_{\sqrt[3 d]]{ }}$. Here the two branches have the same first interaction. For participant A , we have $\mathcal{R}_{1}=\mathcal{R}_{2}=\llbracket \mathrm{G}_{1} \rrbracket \mathrm{l}_{\mathrm{A}}=\llbracket \mathrm{G}_{2} \rrbracket \mathrm{~L}_{\mathrm{A}}=\{[\mathrm{AB}!\mathrm{m}]\}$; hence $\phi=$ $\mathcal{R}_{1} \mapsto \mathcal{R}_{2}$ is the unique candidate function for $\phi$ while either $\psi=\mathcal{R}_{1} \mapsto \varepsilon$ or $\psi=\mathcal{R}_{1} \mapsto[\mathrm{AB}!\mathrm{m}]$; choosing the latter, we obtain that $\operatorname{div}_{\mathrm{A}}^{\phi, \psi}\left(\mathrm{G}_{1}, \mathrm{G}_{2}\right)=(\emptyset, \emptyset)$. For participant $C, \mathcal{R}_{1}=\llbracket G_{1} \rrbracket L_{c}=\{[B C ? x]\}$ and $\mathcal{R}_{2}=\llbracket G_{2} \rrbracket L_{c}=\{[B C ? y]\}$. Let $\phi=\mathcal{R}_{1} \mapsto \mathcal{R}_{2}$ and $\psi=\mathcal{R}_{1} \mapsto \varepsilon$ (i.e. the only possible choice, since there is no common prefix) then $\operatorname{div}_{C}^{\phi, \psi}\left(G_{1}, G_{2}\right)=(\{B C ? x\},\{B C ? y\})$. Finally, for participant $\mathrm{B}, \mathcal{R}_{1}=\left.\llbracket \mathrm{G}_{1} \rrbracket\right|_{\mathrm{B}}=\{[\mathrm{AB} ? \mathrm{~m} \mapsto \mathrm{BC}!\mathrm{x}]\}$ and $\mathcal{R}_{2}=\llbracket \mathrm{G}_{1} \rrbracket \mathrm{~L}_{\mathrm{B}}=\{[\mathrm{AB} ? \mathrm{~m} \mapsto \mathrm{BC}!\mathrm{y}]\}$. Let $\phi=\mathcal{R}_{1} \mapsto \mathcal{R}_{2}$ and $\psi=\mathcal{R}_{1} \mapsto[A B ? \mathrm{~m}]$ then $\operatorname{div}_{\mathrm{B}}^{\phi, \psi}\left(\mathrm{G}_{1}, \mathrm{G}_{2}\right)=(\{\mathrm{BC}!\mathrm{x}\},\{\mathrm{BC}!\mathrm{y}\})$.

Example $G_{[4 a)}$. Participant $A$ sends either $A B!\times$ or $A C!y$ and the point of divergence can be computed similarly to the case $G_{\sqrt{3 b}}$. The divergence points for the other participants are more interesting. We consider participant $B$ only as the case for $C$ is similar. We have $\mathcal{R}_{1}=\left.\llbracket G_{1} \rrbracket\right|_{\mathrm{B}}=\{[A B ? \times]\}$ and $\mathcal{R}_{2}=\llbracket G_{2} \rrbracket L_{\mathrm{B}}=\{\varepsilon\}$. The only B-prefix map candidate is $(\phi, \psi)$ with $\phi=\mathcal{R}_{1} \mapsto \mathcal{R}_{2}$ and $\psi=\mathcal{R}_{1} \mapsto \varepsilon$, hence $\operatorname{div}_{B}^{\phi, \psi}\left(G_{1}, G_{2}\right)=(\{A B ? x\}, \emptyset)$. This reflects the fact that in one branch $B$ receives notification of the choice made by $A$, while in the other branch no information is received.

Example $\mathrm{G}_{\boxed{4 \mathrm{~b}}}$. Here, it is easy to verify that participant A diverges at $(\{\mathrm{AB}!\mathrm{x}\}, \emptyset)$, $B$ at $(\{A B ? x\}, \emptyset), C$ at $(\emptyset,\{C D!x\})$, and $D$ at $(\emptyset,\{C D ? x\})$.

Example $\mathrm{G}_{\sqrt{5}}$. For participant C , the numbers of pomsets in the semantics of the two branches at control point i differ:

$$
\begin{aligned}
& \mathcal{R}_{1}=\left\{[\mathrm{AC} ? \mathrm{~h}],\left[\mathrm{AC} ? \mathrm{y}_{1} \rightarrow \mathrm{AC} ? \mathrm{y}_{2} \rightarrow \mathrm{AC} ? \mathrm{z}_{1}\right]\right\} \\
& \mathcal{R}_{2}=\left\{[\mathrm{AC} ? \mathrm{~h}],\left[\mathrm{AC} ? \mathrm{y}_{1} \rightarrow \mathrm{AC} ? \mathrm{y}_{2} \rightarrow \mathrm{AC} ? \mathrm{z}_{2}\right],\left[\mathrm{AC} ? \mathrm{y}_{1} \rightarrow \mathrm{AC} ? \mathrm{y}_{2} \rightarrow \mathrm{AC} ? \mathrm{z}_{3}\right]\right\}
\end{aligned}
$$

The function $\phi$ can be chosen to

- map the subset $\mathcal{R}_{1 a}=\{[\mathrm{AC} ? \mathrm{~h}]\}$ of $\mathcal{R}_{1}$ to the subset $\{[\mathrm{AC} ? \mathrm{~h}]\}$ of $\mathcal{R}_{2}$ and
- map the subset $\mathcal{R}_{1 b}=\left\{\left[\mathrm{AC}\right.\right.$ ? $\mathrm{y}_{1} \rightarrow \mathrm{AC}$ ? $\mathrm{y}_{2} \rightarrow \mathrm{AC}$ ? $\left.\left.\mathrm{z}_{1}\right]\right\}$ of $\mathcal{R}_{1}$ to the subset $\left\{\left[A C ? y_{1} \rightarrow A C ? y_{2} \rightarrow A C ? z_{2}\right],\left[A C ? y_{1} \rightarrow A C ? y_{2} \rightarrow A C ? z_{3}\right]\right\}$ of $\mathcal{R}_{2}$.
With $\psi=\left\{\begin{array}{l}\mathcal{R}_{1 a} \mapsto[\mathrm{AC} ? \mathrm{~h}] \\ \mathcal{R}_{1 b} \mapsto\left[\mathrm{AC} ? \mathrm{y}_{1} \rightarrow \mathrm{AC} ? \mathrm{y}_{2}\right]\end{array} \quad\right.$ we have that, for participant C , the divergence point is $\operatorname{div}_{C}^{\phi, \psi}\left(G_{1}, G_{2}\right)=\left(\left\{A C ? z_{1}\right\},\left\{A C ? z_{2}, A C ? z_{3}\right\}\right)$. With similar calculations, we can show that $B$ diverges at ( $\{A B ? \mid\},\{A B ? r\}$ ) and that participant $A$ diverges at $\left(\left\{A C!h, A C!y_{1}\right\},\{A B!r\}\right)$. In passing, note that there are other possible choices of prefix maps for $G_{\sqrt[5]{5}}$ not considered here.

We now formalise the concepts of active and passive participants.
Definition 13. A participant $\mathrm{A} \in \mathcal{P}$ is active in $\mathrm{G}_{1}+\mathrm{G}_{2}$ if there exists an A-prefix map $(\phi, \psi)$ of $\mathrm{G}_{1}$ and $\mathrm{G}_{2}$ such that, if $\operatorname{div}_{\mathrm{A}}^{\phi, \psi}\left(\mathrm{G}_{1}, \mathrm{G}_{2}\right)=\left(\tilde{l}_{1}, \tilde{l}_{2}\right)$, then

$$
\tilde{l}_{1} \cup \tilde{l}_{2} \subseteq \mathcal{L}^{!} \quad \tilde{l}_{1} \cap \tilde{l}_{2}=\emptyset \quad \tilde{l}_{1} \neq \emptyset \quad \tilde{l}_{2} \neq \emptyset
$$

Thus, the behaviour of an active participant $A$ in $G_{1}$ and $G_{2}$ must be the same up to the point where it informs the other participants, by sending different messages, which branch it chooses.

Definition 14. A participant $\mathrm{A} \in \mathcal{P}$ is passive in $\mathrm{G}_{1}+\mathrm{G}_{2}$ if there exists an A-prefix map $(\phi, \psi)$ of $\mathrm{G}_{1}$ and $\mathrm{G}_{2}$ such that, if $\operatorname{div}_{\mathrm{A}}^{\phi, \psi}\left(\mathrm{G}_{1}, \mathrm{G}_{2}\right)=\left(\tilde{l}_{1}, \tilde{l}_{2}\right)$, then

- $\tilde{l}_{1} \cup \tilde{l}_{2} \subseteq \mathcal{L}^{?}$ and $\tilde{l}_{1}=\emptyset \Longleftrightarrow \tilde{l}_{2}=\emptyset$
- $\forall \mathcal{R} \in \operatorname{dom} \phi, r \in \mathcal{R}: \tilde{l}_{2} \cap \lambda_{r}\left(\mathcal{E}_{r-\psi(\mathcal{R})}\right)=\emptyset$
- $\forall \mathcal{R} \in \operatorname{cod} \phi, r \in \mathcal{R}: \tilde{l}_{1} \cap \lambda_{r}\left(\mathcal{E}_{r-\psi\left(\phi^{-1}(\mathcal{R})\right)}\right)=\emptyset$

Thus, the behaviour of a passive participant $A$ in $G_{1}$ and $G_{2}$ must be the same up to a point where it receives either of two different messages, each one identifying which branch had been selected. Clearly, A cannot perform outputs at the points of branching.

Interestingly, if one always takes a mapping yielding the empty prefix in the determination of active and passive roles, the definitions above yield exactly the same notions used e.g., in [21, 3, 11].

Lemma 1. Participant A cannot be both passive and active in $\mathrm{G}_{1}+\mathrm{G}_{2}$.
Proof sketch. The proof shows that if a participant is active (passive) for $\phi$ and $\psi$ then for every $\phi^{\prime}$ and $\psi^{\prime}$ the prefixes in the image of $\psi^{\prime}$ are prefixes of the pomsets in the codomain of $\psi$. Details of the proof are reported in 31.

We demonstrate the notion of well-branchedness on the examples of Section 3 . When it exists, the selector of the choice is the active participant that determines the branch to execute. For instance, in the choreography $G_{\sqrt{3 a}}$ participant $A$ is the selector of the choice since it diverges at ( $\{A B!x\},\{A B!y\}$ ). Hence, choreography $G^{3 a}$ is well-branched because the other participant $B$ diverges at $(\{A B ? x\},\{A B ? y\})$ and is therefore passive. Likewise, choreography $G^{3 b}$ is wellbranched since $A$ diverges at $(\{A B!x\},\{A C!y\})$, $B$ diverges at $(\{A B ? x\},\{C B ? w\})$, and $C$ diverges at $(\{B C ? z\},\{A C ? y\})$; thus $A$ is active and the other participants are passive.

Unlike its correspondents in the rest of the literature, our notion of wellbranchedness does not require the selector to exist. For instance, the choreography $G_{\sqrt{3 c}}$ is well-branched even if it has no active participant. Both participants are passive, since for both of them is possible to find a prefix map that leads to divergence at $(\emptyset, \emptyset)$. Another example (usually discharged in the literature by imposing syntactic constraints) is the choreography $G_{\sqrt[3 d]{ }}$. Here, the two branches have the same first interaction. Choosing the prefix $[A B!m]$ for the participant $A$ makes it passive. The participant $C$ diverges at $(\{B C ? x\},\{B C ? y\})$, thus it is passive. Finally, choosing the prefix $A B$ ?m for participant $B$, makes its point of divergence to be $(\{B C!x\},\{B C!y\})$. Thus $B$ is active and the choice well-branched.

An example of non well-branched choreography is $G_{\sqrt{4 a}}$ : the participant $A$ is active (sending either $A B!\times$ or $A C!y$ ). However, $B$ and $C$ are neither passive nor
active. In fact, participant $B$ diverges at $(\{A B ? x\}, \emptyset)$ (there is no other possible point of divergence for $B$ ). Another example of non well-branched choreography is $G_{[4 b}$, since no participant is active or passive.

Choreography $\mathrm{G}_{\sqrt{5}}$ illustrates that our notion of well-branchedness allows to specify complex distributed choices. This example clarifies the need of partitioning the semantics of branches to compute active and passive participants. As shown (page 19), even though the semantics yield different numbers of pomsets in the two branches for participant C , we can choose a C -prefix map $(\phi, \psi)$ such that $\operatorname{div}_{\mathrm{C}}^{\phi, \psi}\left(\mathrm{G}_{1}, \mathrm{G}_{2}\right)=\left(\left\{\mathrm{AC} ? \mathrm{z}_{1}\right\},\left\{\mathrm{AC} ? \mathrm{z}_{2}, \mathrm{AC} ? \mathrm{z}_{3}\right\}\right)$; thus C is passive. Hence, $\mathrm{G}_{5}$ is well-branched since $B$ is passive and that $A$ is the unique active participant. The choreography $\mathrm{G}_{\sqrt{5}}$ has two other interesting well-branched variants. One where the interactions in $G_{2}$ are replaced with

$$
\mathrm{G}_{2}^{\prime}=\mathrm{A} \xrightarrow{\mathrm{r}} \mathrm{~B} ; \mathrm{i}^{\prime \prime}:\left(\mathrm{A} \xrightarrow{\mathrm{~h}} \mathrm{C}+\mathrm{A} \xrightarrow{\mathrm{y}_{1}} \mathrm{C} ; \mathrm{A} \xrightarrow{\mathrm{y}_{2}} \mathrm{C} ; \mathrm{i}^{\prime \prime \prime}:\left(\mathrm{A} \xrightarrow{\mathrm{z}_{2}} \mathrm{C}+\mathrm{A} \xrightarrow{\mathrm{z}_{3}} \mathrm{C}\right)\right)
$$

In $G_{2}^{\prime}$ the interaction $i: A \xrightarrow{r} B$ is factorised before the choice in $i^{\prime \prime}$. The other variant is one where such interaction is executed in parallel to the others in $\mathrm{G}_{2 a}$ or $\mathrm{G}_{2 b}$.

### 4.3. Languages of choreographies

The abstract semantics of a g-choreography is the set of partial orders among the events of the g-choreography. A more concrete semantics, akin to the usual trace-based semantics of global views, can be given by considering the language of a g-choreography. Informally, the language of a g-choreography $G \in \mathcal{G}$ consists of the sequences of words over the alphabet consisting of the communication actions of the events in $G$ that preserve the causal relations of $\llbracket G \rrbracket$, provided that $\llbracket G \rrbracket$ is defined. The language of $G \in \mathcal{G}$ is the set $\mathbb{L}_{\mathrm{G}}$ defined by

$$
\mathbb{L}_{\mathbb{G}}= \begin{cases}\bigcup_{r \in \llbracket \mathbb{G}} \mathbb{L}_{r}, & \text { if } \llbracket \mathbb{G} \rrbracket \text { is defined } \\ \perp, & \text { otherwise }\end{cases}
$$

where the language $\mathbb{L}_{r}$ of a pomset $r=[\mathcal{E}, \leq, \lambda]$ is the set of $\lambda(w)$ such that $w \in \mathcal{E}^{\star}$ and, for any indexes $i$ and $j$ between 0 and the length of $w$ and $i \neq j$,

1. $w[i] \neq w[j]$
2. if $w[i] \leq w[j]$ then $i \leq j$
3. for every $e \in \mathcal{E}$, if $e \leq w[i]$ and $e \neq w[i]$ then there exists $h<i$ such that $w[h]=e$
Clause 1 states that events in the word are not repeated. Clause 2 states that words preserve the causal relations of events. Clause 3 requires that all the predecessors of an event which appears in the word must precede that event in that word. Notice that $\mathbb{L}_{r}$ is prefix-closed (hence $\mathbb{L}_{G}$ is prefix-closed too).

Hereafter, we comment on the languages of the examples in Section 4. The language $\mathbb{L}_{6_{\text {3a) }}}=\{\epsilon, \mathrm{AB}!x, \mathrm{AB}!y, \mathrm{AB}!x \cdot \mathrm{AB} ? x, \mathrm{AB}!y \cdot \mathrm{AB} ? y\}$ is the prefix-closure
of $\{\mathrm{AB}!x \cdot \mathrm{AB} ? x, \mathrm{AB}!y \cdot \mathrm{AB} ? y\}$ (we write _.- for concatenation of words). Similarly, the languages of $\mathbb{L}_{G_{\sqrt{3 b}}}, \mathbb{L}_{G_{\boxed{3 C}}}$, and $\mathbb{L}_{G_{\boxed{3 d}}}$ respectively are the prefix-closure of

$$
\begin{aligned}
& \{\mathrm{AB}!x \cdot \mathrm{AB} ? x \cdot \mathrm{BC}!z \cdot \mathrm{BC} ? z, \mathrm{AC}!y \cdot \mathrm{AC} ? y \cdot \mathrm{CB}!w \cdot \mathrm{CB} ? w\} \\
& \{\mathrm{AB}!\mathrm{m} \cdot \mathrm{AB} ? \mathrm{~m}\} \\
& \{\mathrm{AB}!\mathrm{m} \cdot \mathrm{AB} ? \mathrm{~m} \cdot \mathrm{BC}!x \cdot \mathrm{BC} ? x, \mathrm{AB}!\mathrm{m} \cdot \mathrm{AB} ? \mathrm{~m} \cdot \mathrm{BC}!y \cdot \mathrm{BC} ? y\}
\end{aligned}
$$

The languages $\mathbb{L}_{G_{[4 a}}$ and $\mathbb{L}_{G_{\sqrt[4 b]{ }}}$ are undefined, since the choreographies are not well-branched. The language $\mathbb{L}_{G(5)}$ is bigger and defined in a similar way.

## 5. Projecting on communicating machines

As in [26, 15], we adopt communicating finite state machines (CFSM) as local artefacts. We borrow the definition of CFSMs in [5] adapting it to our context. A CFSM $M=\left(Q, q_{0}, \rightarrow\right)$ is a finite transition system where

- $Q$ is a finite set of states with $q_{0} \in Q$ the initial state, and
- $\rightarrow \subseteq Q \times \mathcal{L} \times Q$; we write $q \xrightarrow{l} q^{\prime}$ for $\left(q, l, q^{\prime}\right) \in \rightarrow$.

A CFSM $M=\left(Q, q_{0}, \rightarrow\right)$ is A-local if $\operatorname{sbj}(l)=\mathrm{A}$ for each $q \xrightarrow{l} q^{\prime}$. Given an A-local CFSM $M_{\mathrm{A}}=\left(Q_{\mathrm{A}}, q_{0 \mathrm{~A}}, \rightarrow_{\mathrm{A}}\right)$ for each $\mathrm{A} \in \mathcal{P}$, the tuple $S=\left(M_{\mathrm{A}}\right)_{\mathrm{A} \in \mathcal{P}}$ is a (communicating) system. For all $\mathrm{A} \neq \mathrm{B} \in \mathcal{P}$, it is assumed that there is an infinite FIFO queue $b_{\mathrm{AB}}$ where $M_{\mathrm{A}}$ puts the message to $M_{\mathrm{B}}$ and from which $M_{\mathrm{B}}$ consumes the messages from $M_{\mathrm{A}}$.

The semantics of communicating systems is defined in terms of transition systems, which keep track of the state of each machine and the content of each queue. Let $S=\left(M_{\mathrm{A}}\right)_{\mathrm{A} \in \mathcal{P}}$ be a communicating system. A configuration of $S$ is a pair $s=\langle\tilde{q} ; \tilde{b}\rangle$ where $\tilde{q}=\left(q_{\mathrm{A}}\right)_{\mathrm{A} \in \mathcal{P}}$ with $q_{\mathrm{A}} \in Q_{\mathrm{A}}$ and $\tilde{b}=\left(b_{\mathrm{AB}}\right)_{\mathrm{AB} \in \mathcal{C}}$ with $b_{\mathrm{AB}} \in \mathcal{M}^{*}$; state $q_{\mathrm{A}}$ keeps track of the state of the machine $M_{\mathrm{A}}$ and buffer $b_{\mathrm{AB}}$ keeps track of the messages sent from A to B . The initial configuration $s_{0}$ is the one where, for all $\mathrm{A} \in \mathcal{P}, q_{\mathrm{A}}$ is the initial state of the corresponding CFSM and all buffers are empty.

A configuration $s^{\prime}=\left\langle\tilde{q}^{\prime} ; \tilde{b}^{\prime}\right\rangle$ is reachable from another configuration $s=$ $\langle\tilde{q} ; \tilde{b}\rangle$ by firing a transition $l$, written $s \stackrel{l}{\Rightarrow} s^{\prime}$ if there is a message $\mathrm{m} \in \mathcal{M}$ such that either (1) or (2) below holds:

1. $l=\mathrm{AB}!\mathrm{m}$ and $q_{\mathrm{A}} \xrightarrow{l} \mathrm{~A} q_{\mathrm{A}}^{\prime}$ and
a. $q_{\mathrm{C}}^{\prime}=q_{\mathrm{C}}$ for all $\mathrm{C} \neq \mathrm{A}$ and
b. $b_{\mathrm{AB}}^{\prime}=b_{\mathrm{AB}} \cdot \mathrm{m}$ and
c. $b_{\mathrm{CD}}^{\prime}=b_{\mathrm{CD}}$ for all (C,D) $\neq$ $(\mathrm{A}, \mathrm{B}) \in \mathcal{C}$
2. $l=\mathrm{AB} ? \mathrm{~m}$ and $q_{\mathrm{B}}{ }^{l} \mathrm{~B} q_{\mathrm{B}}^{\prime}$ and
a. $q_{\mathrm{C}}^{\prime}=q_{\mathrm{C}}$ for all $\mathrm{C} \neq \mathrm{B}$ and
b. $b_{\mathrm{AB}}=m \cdot b_{\mathrm{AB}}^{\prime}$ and
c. $b_{\mathrm{CD}}^{\prime}=b_{\mathrm{CD}}$ for all $(\mathrm{C}, \mathrm{D}) \neq$ $(\mathrm{A}, \mathrm{B}) \in \mathcal{C}$

Condition (1) puts $m$ on channel $A B$, while (2) gets $m$ from channel $A B$. In both cases, any machine or buffer not involved in the transition is left unchanged in the new configuration $s^{\prime}$.

A configuration $s=\langle\tilde{q} ; \tilde{b}\rangle$ is stable if all buffers are empty: $\tilde{b}=\tilde{\varepsilon}$. A configuration $s=\langle\tilde{q} ; \tilde{b}\rangle$ is a deadlock if $s \nRightarrow$ and

- there exists a participant $\mathrm{A} \in \mathcal{P}$ such that $q_{\mathrm{A}} \xrightarrow{\mathrm{AB} ? m} \mathrm{~A}_{\mathrm{A}} q_{\mathrm{A}}^{\prime}$
- or $\tilde{b} \neq \tilde{\varepsilon}$

This definition is adapted from [10] and is meant to capture communications misbehaviour. Observe that, according to this definition, a configuration $s$ where all machines are in a state with no outgoing transitions and all buffers are empty is not a deadlock configuration even though $s \nRightarrow$.

The language of a communicating system $S$ is the set $\mathbb{L}_{S} \in \mathcal{L}^{\star}$ of sequences $l_{0} \ldots l_{n-1}$ such that $s_{0} \stackrel{l_{0}}{\Rightarrow} \ldots \stackrel{l_{n-1}}{\Longrightarrow} s_{n}$. Note that $\mathbb{L}_{S}$ is prefix-closed.

Given two CFSMs $M=\left(Q, q_{0}, \rightarrow\right)$ and $M^{\prime}=\left(Q^{\prime}, q_{0}^{\prime}, \rightarrow^{\prime}\right)$, we use the following notations:

- $M \times M^{\prime}=\left(Q \times Q^{\prime},\left(q_{0}, q_{0}^{\prime}\right), \rightarrow{ }^{\prime \prime}\right)$ is the product of $M$ and $M^{\prime}$ where $\left(\left(q_{1}, q_{1}^{\prime}\right), l,\left(q_{2}, q_{2}^{\prime}\right)\right) \in \rightarrow^{\prime \prime}$ if, and only if,

$$
\left(\left(q_{1}, l, q_{2}\right) \in \rightarrow \quad \text { and } q_{1}^{\prime}=q_{2}^{\prime}\right) \quad \text { or } \quad\left(\left(q_{1}^{\prime}, l, q_{2}^{\prime}\right) \in \rightarrow^{\prime} \quad \text { and } q_{1}=q_{2}\right)
$$

- $\left\{q^{\prime} / q\right\} M$ represents the machine obtained by substituting the state $q$ with the state $q^{\prime}$, provided that $q^{\prime}$ is not in the states of $M$;
- $M \otimes \mathbf{n}$ represents the machine $\left(Q \times \mathbf{n},\left(q_{0}, n\right), \rightarrow \otimes \mathbf{n}\right)$;
- $M \circ M^{\prime}$ represents the machine ( $Q \cup Q^{\prime}, q_{0}, \rightarrow \cup \rightarrow{ }^{\prime}$ );

To define the projection of a g-choreography G to CFSMs we provide a function defined by induction on the syntax of G that returns a triple ( $M, q_{0}, q_{e}$ ) where $M$ is a CFSM, $q_{0}$ is its initial state, and $q_{e}$ is special state of $M$ used to connect it to other machines. We will use $\left(M, q_{0}, q_{e}\right) \otimes \mathbf{n}$ to represent $\left(M \otimes \mathbf{n},\left(q_{0}, n\right),\left(q_{e}, n\right)\right)$. We will use a graphical notation to represent those triples where an arrow without target exiting a state singles out the second element of the pair (so, for ( $M, q_{0}, q_{e}$ ) we will have a dangling arrow leaving $q_{e}$ ); also, an arrow without source entering a state of $M$ singles out the initial state of the machine. Let G be a g-choreography, the function $G \downarrow_{A}$ yields the projection of $G$ over the
participant A as follows:

Figure 9 shows examples of projections for participants A and B.

(a) $A \xrightarrow{x} B$

(c) $\mathrm{A} \xrightarrow{\mathrm{x}} \mathrm{B} \mid \mathrm{A} \xrightarrow{\mathrm{y}} B$

Figure 9: Examples of projections (we use $\bar{q}$ for $(q, 1)$ and $\underline{q}$ for $(q, 2)$ ) for participants A (left) and B (right)

We let $G \downarrow_{A}$ denote the CFSM in the first component of the triplet returned by the projection when the other components (i.e., the states) are not needed. Let $\Delta(M)$ denote the CFSM obtained by determinising $M$ when interpreting them as finite automata on the alphabet $\mathcal{L}$. The following theorem shows that the system made of the projections of a g-choreography $G$ is deadlock-free if $\llbracket \mathrm{G} \rrbracket$ is defined.

Theorem 1 (Progress). Given $G \in \mathcal{G}$ such that $\llbracket \mathrm{G} \rrbracket \neq \perp$, if s is reachable from the initial configuration $s_{0}$ of the communicating $\operatorname{system}\left(\Delta\left(G \downarrow_{\mathrm{A}}\right)\right)_{\mathrm{A} \in \mathcal{P}}$ then $s$ is not a deadlock.

Proof sketch. By structural induction on the syntax of g-choreography. The base cases are straightforward, since the projection of an empty choreography or of a single interaction are deterministic and do not lead to a deadlock by construction. For the inductive steps, we rely on the fact that determinisation of CFSMs preserves the language of the communicating system and does not introduce deadlocks. For sequential and parallel composition, the proof is done by showing that if there is a deadlock in the composed communicating system, then there must be a deadlock in at least one of the constituent systems. This holds straightforwardly for the sequential composition. For the parallel composition, we note that

- in each thread, every output of message, say m, has a corresponding input action in a receiving machine, say $B$;
- the machine $M_{\mathrm{B}}$ of B is the product of the threads on B .

Therefore, the configurations where the message $m$ is sent have to reach a configuration where $B$ has the reception of $m$ enabled (otherwise in one of the threads there would be a deadlock violating the inductive hypothesis). Hence, eventually $m$ will be consumed.

For the non-deterministic composition, we show that if there is a trace in system $S$ made of machines $\left(\Delta\left(\left(\mathrm{G}_{1}+\mathrm{G}_{2}\right) \downarrow_{\mathrm{A}}\right)\right)_{\mathrm{A} \in \mathcal{P}}$, then there must be the same trace in one of the systems made of machines $\left(\Delta\left(\mathrm{G}_{1} \downarrow_{\mathrm{A}}\right)\right)_{\mathrm{A} \in \mathcal{P}}$ or $\left(\Delta\left(\mathrm{G}_{2} \downarrow_{\mathrm{A}}\right)\right)_{\mathrm{A} \in \mathcal{P}}$. This is due to the well-branched condition. If the active participant, say $B$, selects $\mathrm{G}_{i} \downarrow_{\mathrm{B}}$ in the communicating system $S$ then all other participants are forced to follow the same choice. This allows us to build a simulation relation between the communicating system of the non-deterministic choice and the one consisting of the CFSMs $\left(\mathrm{G}_{i} \downarrow_{\mathrm{A}}\right)_{\mathrm{A} \in \mathcal{P}}$.

The following theorem shows that the traces of the system made of the projections of a $g$-choreography $G$ are included in the language of the g-choreography if $\llbracket G \rrbracket$ is defined.

Theorem 2 (Adequacy). If $\mathrm{G} \in \mathcal{G}$ with $\llbracket \mathrm{G} \rrbracket \neq \perp$ and $S=\left(\Delta\left(\mathrm{G} \downarrow_{\mathrm{A}}\right)\right)_{\mathrm{A} \in \mathcal{P}}$ then $\mathbb{L}_{S} \subseteq \mathbb{L}_{\mathrm{G}}$.
Proof sketch. The proof is by structural induction over the syntax of the gchoreographies. The two main proof obligations are to show that $(i)$ the dependencies are preserved in the case of sequential composition and (ii) no additional
communication occurs in the case of parallel composition. For the sequential composition we proceed as follows. By definition, every word $w_{0}$ in $\mathbb{L}_{\mathrm{G}_{;} \mathrm{G}^{\prime}}$ is the shuffling of two words, $w \in \mathbb{L}_{\mathrm{G}}$ and $w^{\prime} \in \mathbb{L}_{\mathrm{G}^{\prime}}$. Additionally, the side condition of the semantics of sequential composition ensures that all the events of $w$ having subject A precede in $w_{0}$ every event of $w^{\prime}$ with subject A. For the second task we rely on the fact that $\llbracket G \rrbracket$ is defined and we follow the same reasoning done for Theorem 1. Details of the proof are reported in 31.

In general, the converse of the inclusion in Theorem 2, that is $\mathbb{L}_{G} \subseteq \mathbb{L}_{S}$, does not hold. The reason is that the semantics of parallel composition of g-choreographies does not assume a FIFO policy on channels. In fact, the communicating system can have less behaviours than the interleaving of the two constituent threads because of the additional dependencies imposed by FIFO channels. For instance, take the g-choreography $G=A \xrightarrow{x} B \mid A \xrightarrow{y} B$; the word $A B!x . A B!y . A B ? y . A B ? x$ is in $\mathbb{L}_{G}$ but it is not in $\mathbb{L}_{\left(\Delta\left(G \downarrow_{A}\right)\right)_{A \in \mathcal{P}}}$.

## 6. An alternative semantics of global graphs

The presentation of the semantics of choreographies in terms of pomsets has the benefits of being elegant and reusing existing theories [29, 16. However, this presentation is not ideal for implementing tools ${ }^{10}$ based on the semantics. Firstly, there is a computational explosion (common to trace-based methods): each choice duplicates events making the resulting number of pomsets exponential in the number of choices in the choreography. Secondly, the pomset semantics is based on equivalence classes, which are notoriously not straightforward to implement. To overcome these limitations, we give an alternative yet equivalent semantics in terms of hypergraphs, which yields compact representations of sets of partial orders.

### 6.1. Preliminaries

Fixed a set $\mathcal{V}$ (of vertices), a (directed) hypergraph on $\mathcal{V}$ is a relation $H \subseteq 2^{\mathcal{V}} \times 2^{\mathcal{V}}$, namely a hyperarc $\left(\tilde{v}, \tilde{v}^{\prime}\right)$ in $H$ relates two sets of vertices, the source $\tilde{v}$ and the target $\tilde{v}^{\prime}$. (To avoid cumbersome parenthesis, singleton sets in hyperarcs are shortened by their element, e.g., we write $\lceil v, \tilde{v}\rangle$ instead of $(\{v\}, \tilde{v} \eta$.) The vertices of our hypergraphs are drawn from the set

$$
\mathcal{V}=(\mathcal{L} \times \mathcal{K}) \cup \mathcal{K} \quad(\text { ranged over by } v)
$$

Vertices in $\mathcal{L} \times \mathcal{K}$ represent communication actions together with the originating control points while those in $\mathcal{K}$ represent "non-observable" actions, like (the execution of) a choice or a merge. In the following, we shorten $(l, i)$ as $l_{[i]}$, extend cp to vertices so that $\operatorname{cp}(v)$ denotes the control point of a vertex $v$, we use

[^8]

Figure 10: Example hypergraphs
$\operatorname{act}\left(l_{[i \mathrm{ij}}\right)=l$ to denote the action at $l_{[\mathrm{ij}}$, and we extend function $\operatorname{sbj}(-)$ to vertices in the obvious way. Take $\operatorname{sbj}(-)$ and $\operatorname{act}(-)$ to be undefined on $\mathcal{K}$ and the subject of a vertex $l_{[\mathrm{j}]}$ to be simply $\operatorname{sbj}(l)$, the subject of its corresponding action $l$.

To systematically identify vertices of hypergraphs corresponding to nonobservable events, we fix a function $\mu: \mathcal{G} \rightarrow(\mathcal{K} \rightarrow \mathcal{K})$ such that, for each g-choreography $\mathrm{G} \in \mathcal{G}, \mu(\mathrm{G})\left(\right.$ written $\left.\mu_{\mathrm{G}}\right)$

- is bijective when restricted to $\operatorname{cp}(\mathrm{G})$ and
- for all $\mathrm{i} \in \mathrm{cp}(\mathrm{G}), \mu_{\mathrm{G}}(\mathrm{i}) \notin \mathrm{cp}(\mathrm{G})$.

Basically, $\mu$ yields a bijective correspondence between (i) branch and merge control points corresponding to choices, (ii) fork and join control points corresponding to parallel compositions, and (iii) complementary send/receive actions. Since we decided to concretely represent control points of $g$-choreographies as positive integers, in the following we assume $\mu_{\mathrm{G}}(\mathrm{i})$ to be the opposite -i , for each $i \in \operatorname{cp}(\mathrm{G})$.

Examples of hypergraphs. In Fig. 10 , the graphs $H_{\sqrt{10 a \mid}}$ and $H_{\sqrt{10 b}}$ contain only
 and $\left.\\left\{A B ? x_{\left[i_{1}\right]}, A B ? y_{\left[i_{2}\right]}\right\},-\mathrm{i}_{3}\right)$. Intuitively, $H_{[10 \mathrm{a}}$ corresponds to the pomset $r_{[6 \mathrm{aa}]}$ of Fig. 6] (establishing a total causal order from the top-most to the bottom-most vertex); graph $H_{\sqrt{10 b}}$ represents a choice at control point $\mathrm{i}_{3}$ between the left and the right branch and corresponds to the set of pomsets $\mathcal{R}_{\text {(6c }}$ of Fig. 6] Finally, graph $H_{\sqrt{100}}$ represents the parallel execution of two threads at the control point $\mathrm{i}_{3}$ and corresponds to the pomset $r_{[6 \mathrm{bb}}$; note that the edge $\left(\mathrm{i}_{3},\left\{\mathrm{AB}!\mathrm{x}_{\left[\mathrm{i}_{1}\right]}, \mathrm{AB}!\mathrm{y}_{\left[\mathrm{i}_{2}\right]}\right\}\right.$ ) of $H_{[100}$ relates the vertex $\mathrm{i}_{3}$ to both $\mathrm{AB}!\mathrm{x}_{\left.\mathrm{i}_{1}\right]}$ and $\mathrm{AB}!\mathrm{y}_{\left[\mathrm{i}_{2}\right]}$. Fig. 11 depicts a further example. Here, both the hypergraph $H_{[119}$ and the set of pomsets $\mathcal{R}_{[11 b]}$ represent the dependencies of the choreography $\mathrm{G}_{[119]}$. Notice that the hypergraph based representation does not grow exponentially when choices are composed sequentially.


Figure 11: A choreography and its semantics represented as set of pomsets and hypergraph

We define the maximal and minimal elements of $H$ respectively as

$$
\begin{aligned}
\max H=\{v \in \mathcal{V} \mid & \left.\nexists\left(\tilde{v}, \tilde{v}^{\prime}\right) \in H: v \in \tilde{v}\right\} \\
\min H=\{v \in \mathcal{V} \mid & \left.\left.\nexists \tilde{v}, \tilde{v}^{\prime}\right) \in H: v \in \tilde{v}^{\prime}\right\}
\end{aligned}
$$

For instance, $H_{\sqrt{10 \mathrm{~b}}}$ and $H_{\sqrt{10 \mathrm{c}}}$ in Fig. 10 respectively have $\min H_{\sqrt{10 \mathrm{~b}}}=\min H_{\sqrt{10 \mathrm{c}}}=$ $\left\{\mathrm{i}_{3}\right\}$ and $\max H_{\sqrt[10 \mathrm{~b}]{ }}=\max H_{\sqrt{10 \mathrm{c}}}=\left\{-\mathrm{i}_{3}\right\}$, while the minimal and maximal elements of $H_{[10 a}$ are $\mathrm{AB}!\mathrm{x}_{\left[\mathrm{i}_{1}\right]}$ and BA ? $\mathrm{y}_{\left[\mathrm{i}_{2}\right]}$ respectively.

We can now define seq $\left(H, H^{\prime}\right)$, the sequential composition of hypergraphs $H$ and $H^{\prime}$ :

$$
\begin{aligned}
\operatorname{seq}\left(H, H^{\prime}\right)=H \cup H^{\prime} \cup\left\{0 v, v^{\prime}| |\right. & \exists\left(\tilde{v}_{1}, \tilde{v}_{2} \downarrow \in H, \ \tilde{v}_{1}^{\prime}, \tilde{v}_{2}^{\prime}\right) \in H^{\prime}: \\
& v \in\left(\tilde{v}_{1} \cup \tilde{v}_{2}\right) \backslash \mathcal{K} \wedge v^{\prime} \in\left(\tilde{v}_{1}^{\prime} \cup \tilde{v}_{2}^{\prime}\right) \backslash \mathcal{K} \wedge \\
& \left.\operatorname{sbj}(v)=\operatorname{sbj}\left(v^{\prime}\right)\right\}
\end{aligned}
$$

As done in Section 4.1 for pomsets, the sequential composition of two hypergraphs $H$ and $H^{\prime}$ adds dependencies between a vertex in $H$ and one in $H^{\prime}$ when they have the same subject.

A hyperedge $\mathrm{h}=\left(\tilde{v}, \tilde{v}^{\prime}\right) \in 2^{\mathcal{V}} \times 2^{\mathcal{V}}$ represents that the causes in $\tilde{v}$ must precede the effects in $\tilde{v}^{\prime}$. Let cs, ef : $2^{\mathcal{V}} \times 2^{\mathcal{V}} \rightarrow 2^{\mathcal{V}}$ be the maps respectively returning the causes (cs) and effects (ef) of a hyperedge, that is: if $\mathrm{h}=\left(\tilde{v}, \tilde{v}^{\prime}\right)$ then $\operatorname{cs}(\mathrm{h})=\tilde{v}$ and ef $(\mathrm{h})=\tilde{v}^{\prime}$. Given $H, H^{\prime} \subseteq 2^{\mathcal{V}} \times 2^{\mathcal{V}}$, define the hypergraph

$$
\left.H \circ H^{\prime}=\left\{0 \operatorname{cs}(\mathrm{~h}), \mathrm{ef}\left(\mathrm{~h}^{\prime}\right)\right) \mid \mathrm{h} \in H, \mathrm{~h}^{\prime} \in H^{\prime}, \mathrm{ef}(\mathrm{~h}) \cap \operatorname{cs}\left(\mathrm{h}^{\prime}\right) \neq \emptyset\right\}
$$

That is, $H \circ H^{\prime}$ is the relational composition of $H$ and $H^{\prime}$. The reflexive-transitive closure $H^{\star}$ of $H$ with respect to the composition relation $\circ$ is defined as expected


Figure 12: Composition of hyperedges
as $H^{\star}=\bigcup_{n \geq 0} \underbrace{H \circ \cdots \circ H}_{n \text {-times }}$. Fig. 12 yields an intuitive representation of how causal relations follow composition: the vertices in $\tilde{v}$ cause all the vertices in $\tilde{v}^{\prime \prime}$ due to the dependency of the vertex $e_{i}^{\prime}$ from the vertices in $\tilde{v}$ and the fact that $e_{i}^{\prime}$ causes each of the vertices in $\tilde{v}^{\prime \prime}$. Formally, the happens-before relation $\widehat{H} \subseteq \mathcal{V} \times \mathcal{V}$ induced by a hypergraph $H$ is

$$
\widehat{H}=\bigcup_{\mid \tilde{v}, \tilde{v}^{\prime} \backslash \in H} \tilde{v} \times \tilde{v}^{\prime}
$$

Basically, $\left\langle v, v^{\prime}\right\rangle \in \widehat{H}$ when $v$ precedes $v^{\prime}$ in $H$, namely $\widehat{H}$ are the causal dependencies among the vertices in $H$.

Let $H$ be an acyclic hypergraph, the happens-before relation of the reflexivetransitive closure of $H$ yields a partial order on the vertices defined as

$$
\sqsubseteq_{H}=\widehat{\left(H^{\star}\right)}
$$

Sometimes it will be convenient to take the intersection of sets of vertices $\tilde{v} \sqcap \tilde{v}^{\prime}$ disregarding their control points: $\tilde{v} \sqcap \tilde{v}^{\prime}=\operatorname{act}(\tilde{v}) \cap \operatorname{act}\left(\tilde{v}^{\prime}\right)$.

### 6.2. Semantics of choreographies

The semantics of a g-choreography is a partial map $\left\{\mathcal{- \}}: \mathcal{G} \rightarrow 2^{\left(2^{\mathcal{V}} \times 2^{\mathcal{V}}\right)}\right.$ defined by induction on the structure of choreographies as per Equations 3/7 below. The base cases and the inductive case for the parallel composition are very simple:

$$
\begin{align*}
\{\mathbf{0}\} & =\emptyset  \tag{3}\\
\{i: A \xrightarrow{m} \mathrm{~B}\} & =\left\{0 \mathrm{AB}!\mathrm{m}_{\mathrm{ij}}, \mathrm{AB} ? \mathrm{~m}_{[\mathrm{ij}} \mathrm{D}\right\}  \tag{4}\\
\left\{\mathrm{i}:\left(\mathrm{G} \mid \mathrm{G}^{\prime}\right)\right\} & = \begin{cases}\{\mathrm{G}\} \cup\left\{\mathrm{G}^{\prime}\right\} \cup H & \text { if }\left(\{\mathrm{G}\} \sqcap\left\{\mathrm{G}^{\prime}\right\}\right) \cap \mathcal{L}^{?}=\emptyset \\
\perp & \text { otherwise }\end{cases} \tag{5}
\end{align*}
$$

where, in (5), $H=\left\{0 \mathrm{i}, \min \{\mathrm{G}\} \cup \min \left\{\mathrm{G}^{\prime}\right\} D,\left(\cap \max \{\mathrm{G}\} \cup \max \left\{\mathrm{G}^{\prime}\right\},-\mathrm{i} \mid\right\}\right.$ and the side condition is analogous to the well-forkedness condition in Section 4.2. Note that the hyperedges in $H$ are not necessary and are added only to make the graphs connected so to simplify the definition of some of the following constructions.

The case of sequential composition requires some auxiliary definitions. We first define the (hyperedges involving) "first" and the "last" communication actions in a hypergraph $H$ :

$$
\begin{aligned}
& \text { fst } \left.H=\left\{0 \tilde{v}, \tilde{v}^{\prime}\right) \in H \mid\left(\tilde{v} \cup \tilde{v}^{\prime}\right) \cap \mathcal{K}=\emptyset \wedge \forall\left(\tilde{v}^{\prime \prime}, \tilde{v}\right) \in H^{\star}: \tilde{v}^{\prime \prime} \backslash \tilde{v} \subseteq \mathcal{K}\right\} \\
& \text { Ist } \left.H=\left\{0 \tilde{v}, \tilde{v}^{\prime}\right) \in H \mid\left(\tilde{v} \cup \tilde{v}^{\prime}\right) \cap \mathcal{K}=\emptyset \wedge \forall\left(\tilde{v}^{\prime}, \tilde{v}^{\prime \prime}\right) \in H^{\star}: \tilde{v}^{\prime \prime} \backslash \tilde{v}^{\prime} \subseteq \mathcal{K}\right\}
\end{aligned}
$$

(note that fst $\left.H=\left(\operatorname{lst}\left(H^{-1}\right)\right)^{-1}\right)$. Also, let $H_{\text {seq }}=\operatorname{seq}\left(H, H^{\prime}\right)$ and $H_{c o m}=$ $\operatorname{cs}($ lst $H) \times \operatorname{ef}\left(\right.$ fst $\left.H^{\prime}\right)$; the predicate $\operatorname{ws}\left(H, H^{\prime}\right)$ is defined as $\widehat{H_{\text {seq }}} \supseteq \widehat{H_{\text {com }}}$, which requires that the sequential composition covers the dependencies between the causes of the last communication actions of $H$ with the effects of the first actions of $H^{\prime}$. This condition corresponds to the predicate $w s\left(r, r^{\prime}\right)$ of Section 4.2.

The semantics of sequential composition can now be defined as follows:

$$
\left\{G ; G^{\prime}\right\}= \begin{cases}\operatorname{seq}\left(\{\mid G\},\left\{G^{\prime}\right\}\right) & \text { if } \operatorname{ws}\left(\{G\},\left\{G^{\prime}\right\}\right)  \tag{6}\\ \perp & \text { otherwise }\end{cases}
$$

Equation 6 establishes happens-before relations as computed by seq( $\{G\},\left\{G^{\prime}\right\}$ ) provided that the well-sequencedness condition holds. Note that well-sequencedness is violated in the example in Fig. 7 e (on page 14).

Similarly to the pomset semantics of Section 4.2 , the semantics of a choice is defined provided that well-branchedness holds; this amounts to saying that (i) there is at most one active participant and (ii) all the other participants are passive. However, the notions of active and passive participants for the hypergraph semantics are defined in a different way; we give the new definitions of active and passive participants in Definitions 15 and 16 below and prove them equivalent to the notions in Definitions 13 and 14 in 31 ..

$$
\left\{i:\left(\mathrm{G}+\mathrm{G}^{\prime}\right)\right\}= \begin{cases}\{\mathrm{G}\} \cup\left\{\mathrm{G}^{\prime}\right\} \cup H & \text { if } \operatorname{wb}\left(\mathrm{G}, \mathrm{G}^{\prime}\right) \text { and }  \tag{7}\\ & H=\{0 \mathrm{i}, \min \{\mathrm{G}\}], 0 \mathrm{i}, \min \left\{\mathrm{G}^{\prime}\right\} D, \\ & \left.\quad 0 \max \{\mathrm{G}\},-\mathrm{i} \mid, 0 \max \left\{\mathrm{G}^{\prime}\right\},-\mathrm{i} D\right\} \\ \perp & \text { otherwise }\end{cases}
$$

The semantics of a choice $\mathrm{i}:\left(\mathrm{G}+\mathrm{G}^{\prime}\right)$ returns the hypergraphs of the branches connected to the control points of the start and merge of the choice. Besides the dependencies induced by $G$ and $G^{\prime},\left\{i:\left(G+G^{\prime}\right)\right\}$ contain those making $i$ (the control point of the branch) precede all minimal vertices of $G$ and $G^{\prime}$; similarly, the maximal vertices of $G$ and $G^{\prime}$ have to precede the conclusion of the choice (marked by the control point -i). This captures the fact that either the actions of the first branch or the actions of the second one should be performed.

Lemma 2. $\forall \mathrm{G} \in \mathcal{G}:\left(\| i, \tilde{v} \mid,\left(\cap i, \tilde{v}^{\prime}\right) \in\{\mid G\} \Longrightarrow \tilde{v} \cap \tilde{v}^{\prime}=\emptyset\right.$.

Proof. The lemma is immediate given that control points cannot have more than one occurrence in a g-choreography G.

Given a hypergraph $H$, we say that $\tilde{v}$ causes $\tilde{v}^{\prime}($ in $H)$ if $\tilde{v} \neq \tilde{v}^{\prime}$ and $\left(\tilde{v}, \tilde{v}^{\prime}\right) \in H^{\star}$.

Lemma 3 (Acyclicity). For all $\mathrm{G} \in \mathcal{G}$ such that $\{\mathrm{G}\}$ is defined, if $\tilde{v}$ causes $\tilde{v}^{\prime}$ in $\{\mathrm{G}\}$ then $\tilde{v}^{\prime}$ does not cause $\tilde{v}$ in $\{\mathrm{G}\}$.

Proof. By induction on the syntax of G. If $G=\mathbf{0}$ then $\{G\}=\emptyset$ and if $G=$ $i: A \xrightarrow{m} B$ then $\{G\}=\left\{0 A B!m_{[i]}, A B ? m_{[i]} D\right\}$ hence the thesis holds vacuously. For the parallel composition $G=\mathrm{i}:\left(\mathrm{G}_{1} \mid \mathrm{G}_{2}\right)$, we have $\{\mathrm{G}\}=\left\{\mathrm{G}_{1}\right\} \cup\left\{\mathrm{G}_{2}\right\} \cup$ $\left\{0 \mathrm{i}, \min \left\{\mathrm{G}_{1}\right\} \cup \min \left\{\mathrm{G}_{2}\right\} \mathrm{D}, 0 \max \left\{\mathrm{G}_{1}\right\} \cup \max \left\{\mathrm{G}_{2}\right\},-\mathrm{i} D\right\}$; by induction we have the thesis since each vertex of $G$ is either a control point j or a pair $(\tilde{l}, \mathrm{j})$ and each $i$ has a unique occurrence in $G$, then the vertices of $\left\{G_{1}\right\}$ and $\left\{G_{2}\right\}$ are disjoint. A similar argument applies in the remaining cases noting that the additional dependencies added in the sequential or non-deterministic composition are all between vertices taken from disjoint sets of vertices.

The notions of well-sequencedness and of well-branchedness (used in the last two clauses in the definition of $\{\}\})$ are very similar to the corresponding ones for pomsets given in Section 4.2, However, they are defined on different structures (indeed, the attentive reader would have noticed the different fonts) and require alternative definitions of active and passive participants, which have now to be adapted to the hypergraphs-based semantics.

The A-only part of a set of vertices $\tilde{v} \in 2^{\mathcal{V}}$ for a participant $A \in \mathcal{P}$ is the set $\tilde{v}_{@ A}$ where the actions of $\tilde{v}$ not having subject A are replaced with the control point of the action; formally

$$
\begin{aligned}
\tilde{v}_{@ A} & =\{v \in \tilde{v} \mid \operatorname{sbj}(v)=\mathrm{A} \vee v \in \mathcal{K}\} \\
& \cup\left\{\operatorname{cp}(v) \mid v \in \tilde{v} \cap\left(\mathcal{L}^{!} \times \mathcal{K}\right) \wedge \operatorname{sbj}(v) \neq \mathrm{A}\right\} \\
& \cup\left\{-\operatorname{cp}(v) \mid v \in \tilde{v} \cap\left(\mathcal{L}^{?} \times \mathcal{K}\right) \wedge \operatorname{sbj}(v) \neq \mathrm{A}\right\}
\end{aligned}
$$

Accordingly, the A-only part of a hypergraph $H$ is defined as

$$
H_{@ A}=\bigcup_{\left\langle\tilde{v}, \tilde{v}^{\prime} D \in H\right.}\left\{0 \tilde{v}_{@ A}, \tilde{v}_{@ A}^{\prime} D\right\}
$$

For instance, the B-only part of $H_{\underline{10 c}}$ (on page 27 ) is

$$
\left\{0 i_{3},\left\{i_{1}, i_{2}\right\} D,\left(i_{1}, A B ? x_{\left[i_{1}\right]}\right),\left(i_{2}, A B ? y_{\left[i_{2}\right]} D, 0\left\{A B ? x_{\left[i_{1}\right]}, A B ? y_{\left[i_{2}\right]}\right\},-i_{3}\right)\right\}
$$

Notice that we use $\operatorname{cp}(v)$ and $-\operatorname{cp}(v)$ for outputs and inputs respectively, so that different vertices not belonging to A remain distinguished and the causality relations are not spoiled.

Given a hypergraph $H, v_{1}, v_{2} \in H$ are independent in $H$ if there are $\mathrm{h} \in H$ and $v_{1}^{\prime}, v_{2}^{\prime} \in \operatorname{ef}(\mathrm{h})$ such that, for each $i, j \in\{1,2\},\left\langle v_{i}^{\prime}, v_{j}\right\rangle \in \widehat{\left(H^{\star}\right)} \Longleftrightarrow i=j$.


Figure 13: Reflectivity

Intuitively, two vertices are independent if they have a common fork ancestor and belong to two different threads. If the first common ancestor is a branch, then the two events are not independent, since they can not both occur in the same execution. For instance, vertices $A B ? x_{\left[i_{1}\right]}$ and $A B ? y_{\left[i_{2}\right]}$ of Fig. 10 b are not independent, because their common ancestor is a branch (i.e. the two events do not belong to the same edge departing from the ancestor and, therefore, are mutually exclusive). Vertices $\mathrm{AB}!\mathrm{x}_{\left[\mathrm{i}_{1}\right]}$ and $\mathrm{AB} ? \mathrm{x}_{\left[\mathrm{i}_{1}\right]}$ of Fig. 10 c are not independent, because they are successors of the same child of the common ancestor (i.e. they belong to the same thread). Finally, vertices $A B ? x_{\left[i_{1}\right]}$ and $A B ? y_{\left[i_{2}\right]}$ of Fig. 10 c are independent. For a participant $\mathrm{A} \in \mathcal{P}$, a set of vertices $\tilde{v} \subseteq H$ is A -uniform in $H$ if $\tilde{v} \cap \mathcal{K}=\emptyset, \operatorname{sbj}(\tilde{v})=\{\mathrm{A}\}, \operatorname{act}(\tilde{v})$ is a singleton, and each $v \neq v^{\prime} \in \tilde{v}$ are not independent and are such that $\left\{\left\langle v, v^{\prime}\right\rangle,\left\langle v^{\prime}, v\right\rangle\right\} \cap \widehat{\left(H^{\star}\right)}=\emptyset$.

The notions of active and passive participants of choices for the hypergraphbased semantics are defined in terms of reflectivity. A partition $V$ of a subset of vertices of $H$ A-reflects a partition $V^{\prime}$ of a subset of vertices of $H^{\prime}$ if there is a bijection $f: V \rightarrow V^{\prime}$ such that the following conditions hold:

- for each $\tilde{v} \in V$ both $\tilde{v}$ and $f(\tilde{v})$ are A-uniform and $\operatorname{act}(\tilde{v})=\operatorname{act}(f(\tilde{v}))$
- $\forall \tilde{v}_{2} \in V, v_{2} \in \tilde{v}_{2},\left\langle v_{1}, v_{2}\right\rangle \in \widehat{H}: \operatorname{sbj}\left(v_{1}\right)=\mathrm{A} \Longrightarrow\left(\exists \tilde{v}_{1} \in V: v_{1} \in \tilde{v}_{1} \wedge \forall v \in\right.$ $\left.\tilde{v}_{2}, v^{\prime} \in \tilde{v}_{1}:\left\langle v, v^{\prime}\right\rangle \notin \widehat{H} \wedge \forall v_{2}^{\prime} \in f\left(\tilde{v}_{2}\right) \exists v_{1}^{\prime} \in f\left(\tilde{v}_{1}\right):\left\langle v_{1}^{\prime}, v_{2}^{\prime}\right\rangle \in \widehat{H^{\prime}}\right)$, and symmetrically
- $\forall \tilde{v}_{2}^{\prime} \in V^{\prime}, v_{2}^{\prime} \in \tilde{v}_{2}^{\prime},\left\langle v_{1}^{\prime}, v_{2}^{\prime}\right\rangle \in \widehat{H^{\prime}}: \operatorname{sbj}\left(v_{1}^{\prime}\right)=\mathrm{A} \Longrightarrow\left(\exists \tilde{v}_{1}^{\prime} \in V^{\prime}: v_{1}^{\prime} \in \tilde{v}_{1}^{\prime} \wedge\right.$ $\left.\forall v \in \tilde{v}_{2}^{\prime}, v^{\prime} \in \tilde{v}_{1}^{\prime}:\left\langle v, v^{\prime}\right\rangle \notin \widehat{H^{\prime}} \wedge \forall v_{2} \in f^{-1}\left(\tilde{v}_{2}^{\prime}\right) \exists v_{1} \in f^{-1}\left(\tilde{v}_{1}^{\prime}\right):\left\langle v_{1}, v_{2}\right\rangle \in \widehat{H}\right)$.

The notion of reflection has an intuitive explanation given in Fig. 13 .
For a participant $\mathrm{A} \in \mathcal{P}$, two g-choreographies $G, \mathrm{G}^{\prime} \in \mathcal{G}$, a partition $V$ of a subset of vertices of $\{\mathrm{G}\}$, and a partition $V^{\prime}$ of a subset of vertices of $\left.\left\{\mathrm{G}^{\prime}\right\}\right\}$ we say that $\left(\tilde{v}_{1}, \tilde{v}_{2}\right)$ is the A-branching pair of $\mathrm{G}+\mathrm{G}^{\prime}$ with respect to $V$ and $V^{\prime}$,
denoted $\left(\tilde{v}_{1}, \tilde{v}_{2}\right)=\operatorname{div}_{\mathrm{A}}^{V, V^{\prime}}\left(\mathrm{G}, \mathrm{G}^{\prime}\right)$, if

$$
V^{\prime} \text { A-reflects } V \text { and }\left\{\begin{array}{l}
\left.\tilde{v}_{1}=\bigcup \operatorname{cs}\left(\operatorname{fst}(\{G\}\}_{@ A}\right)\right) \backslash \bigcup V \\
\text { and } \\
\tilde{v}_{2}=\bigcup \operatorname{cs}\left(\operatorname{fst}\left(\left\{G^{\prime}\right\} @ A\right)\right) \backslash \cup V^{\prime}
\end{array}\right.
$$

The requirement of A-reflectivity is used to identify such common behaviour (i.e. all vertices in $V$ and $V^{\prime}$ ) and to ignore it when checking the behaviour of A in the branches. This has a similar role of prefix-maps of Section 4.2. In fact, by taking the A -only parts of these hypergraphs and selecting their first interactions (that is the A-branching pair $\tilde{v}_{1}, \tilde{v}_{2}$ ) we identify when the behaviour of A in G starts to be different with respect to the behaviour in $\mathrm{G}^{\prime}$.

Definition 15. A participant $\mathrm{A} \in \mathcal{P}$ is active in $\mathrm{G}+\mathrm{G}^{\prime}$ if there are $V$ and $V^{\prime}$ such that $\operatorname{div}_{\mathrm{A}}^{V, V^{\prime}}\left(\mathrm{G}, \mathrm{G}^{\prime}\right)$ is defined and, if $\left(\tilde{v}_{1}, \tilde{v}_{2}\right)=\operatorname{div}_{\mathrm{A}}^{V, V^{\prime}}\left(\mathrm{G}, \mathrm{G}^{\prime}\right)$ then the following holds:

$$
\tilde{v}_{1} \cup \tilde{v}_{2} \subseteq\left(\mathcal{L}^{!} \times \mathcal{K}\right) \quad \tilde{v}_{1} \sqcap \tilde{v}_{2}=\emptyset \quad \tilde{v}_{1} \neq \emptyset \quad \tilde{v}_{2} \neq \emptyset
$$

Given a g-choreography $G \in \mathcal{G}$, the happens-before relation of the reflexivetransitive closure of $\{[G\}$ yields a partial order (by Lemma 3) on the vertices of $G$ defined as

$$
\sqsubseteq_{G}= \begin{cases}\left(\widehat{\{G\}^{\star}}\right), & \text { if }\{G\} \text { is defined } \\ \emptyset, & \text { otherwise }\end{cases}
$$

Observe that $\sqsubseteq_{G}=\sqsubseteq_{\{G\}}$ when $\{G\}$ is defined.
Definition 16. A participant $\mathrm{A} \in \mathcal{P}$ is passive in $\mathrm{G}+\mathrm{G}^{\prime}$ if there are $V$ and $V^{\prime}$ such that $\operatorname{div}_{\mathrm{A}}^{V, V^{\prime}}\left(\mathrm{G}, \mathrm{G}^{\prime}\right)$ is defined and, if $\left(\tilde{v}_{1}, \tilde{v}_{2}\right)=\operatorname{div}_{\mathrm{A}}^{V, V^{\prime}}\left(\mathrm{G}, \mathrm{G}^{\prime}\right)$ then the following holds:

$$
\begin{array}{c|ll}
\tilde{v}_{1} \sqcap\left\{v \in \mathrm{G}^{\prime}\right. & \left.\nexists v^{\prime} \in \tilde{v}_{2}: v \sqsubseteq_{G^{\prime}} v^{\prime}\right\}=\emptyset & \tilde{v}_{1} \cup \tilde{v}_{2} \subseteq\left(\mathcal{L}^{?} \times \mathcal{K}\right) \\
\tilde{v}_{2} \sqcap\{v \in \mathrm{G} & \left.\nexists v^{\prime} \in \tilde{v}_{1}: v \sqsubseteq_{\mathrm{G}} v^{\prime}\right\}=\emptyset & \tilde{v}_{1}=\emptyset \Longleftrightarrow \tilde{v}_{2}=\emptyset
\end{array}
$$

### 6.3. Languages of choreographies from hypergraphs

Informally, the language of a g-choreography $\mathrm{G} \in \mathcal{G}$ consists of the sequences of words made of the communication actions of the vertices in $G$ that preserve the causal relations of $\{G\}$, provided that $\{G\}$ is defined.

Given a g-choreography $\mathrm{G} \in \mathcal{G}$, let $\mathrm{G}^{\oplus}=\{\mathrm{G}\} \cap\left(2^{\mathcal{K}} \times 2^{\mathcal{V}}\right)$ (the hyperedges in G whose source is a control point) and let $D=\bigcup \operatorname{cs}\left(\mathrm{G}^{\oplus}\right)$ be the set of vertices in $\mathrm{G}^{\oplus}$ which do not represent communication events. A map $\rho: D \rightarrow 2^{\mathcal{V}}$ is a resolution of G if $\left(\mathrm{i}, \rho(\mathrm{i}) \downarrow \in \mathrm{G}^{\oplus}(\mathrm{i})\right.$ for each $\mathrm{i} \in D$, where $\mathrm{G}^{\oplus}(\mathrm{i})$ is the set of the outgoing hyperedges of i defined as $\mathrm{G}^{\oplus}(\mathrm{i})=\mathrm{G}^{\oplus} \cap\{0 \mathrm{i}, \tilde{v} \mathrm{D} \mid \tilde{v} \subseteq \mathcal{V}\}$. We will confound $\rho$ with its graph $\{0 \mathrm{i}, \rho(\mathrm{i})| | \mathrm{i} \in D\}$. The partial order corresponding to a resolution is the reflexive and transitive closure of the relation obtained by $(i)$ removing the hyperedges not chosen by the resolution and (ii) removing every
dead vertex (i.e. vertices that are not reachable from the initial vertices after removing the non-selected hyperedges).

We use $\Re_{\mathrm{G}}$ to denote the set of resolutions of G. Intuitively, a resolution fixes a branch for every choice in a g-choreography $G$ and therefore it induces a partial order of the vertices compatible with G and the resolution. Formally, given a resolution $\rho \in \Re_{\mathrm{G}}$ of G , we define the hypergraph $\mathrm{G} ® \rho$ as follows:

$$
\mathrm{G} ® \rho=\operatorname{reachable}\left(\min \{\mathrm{G}\},\{\mathrm{G}\} \backslash\left(\bigcup_{\mathrm{i} \in \operatorname{dom} \rho} \mathrm{G}^{\oplus}(\mathrm{i}) \backslash \rho\right)\right)
$$

where reachable $(\tilde{v}, H)$ is the function that removes every node in the hypergraph $H$ that is not reachable from $\tilde{v}$. The important property of $\mathbf{G} \circledR \rho$ is that resolutions yield consistent choices.

Lemma 4 (Consistency). Given a resolution $\rho \in \Re_{\mathrm{G}}$ of a $g$-choreography $\mathrm{G} \in \mathcal{G}$, if for all $i \neq j \in \operatorname{dom} \rho$

$$
\ i, \rho(i)\rangle \in \mathrm{G} ® \rho \quad \text { and } \quad i \sqsubseteq_{G} j \quad \text { and } \quad \forall v \in \rho(i): v \not \unrhd_{G} j
$$

then $\forall \mathrm{h} \in \mathrm{G} ® \rho: j \notin \mathrm{cs}(\mathrm{h})$.
Proof. Let $\bar{\rho}=\bigcup_{k \in \operatorname{dom} \rho} \mathrm{G}^{\oplus}(\mathrm{k}) \backslash \rho$. Since $\mathrm{i} \sqsubseteq_{\mathrm{G}} \mathrm{j}$, there must be $\left.\ \mathrm{i}, \tilde{v}\right) \in \mathrm{G}^{\oplus}(\mathrm{i})$ such that $v \sqsubseteq{ }_{\mathrm{G}} \mathrm{j}$ for a $v \in \tilde{v}$. Also, it must be $\rho(\mathrm{i}) \neq \tilde{v}$ (otherwise there is a vertex in $\rho(\mathrm{i})$ preceeding j ), therefore $(\mathrm{i}, \tilde{v}) \in \bar{\rho}$ (by Lemma 22). Hence, each vertex in $\tilde{v}$ will not be in $\{\mathrm{G}\} \backslash \bar{\rho} \supseteq \mathrm{G} ® \rho$, which makes j not reachable in $\mathrm{G} ® \rho$ (otherwise we would contradict the hypothesis that $(\mathrm{i}, \rho(\mathrm{i})\rangle \in \mathrm{G} ® \rho$ or that each vertex in $\rho(\mathrm{i})$ does not preceed j$)$. Therefore for any $\mathrm{h} \in \mathrm{G} ® \rho$ we have that $\mathrm{j} \notin \mathrm{cs}(\mathrm{h})$.

Let $\Sigma_{\text {G® } \rho}=\{v \in \mathcal{L} \times \mathcal{K} \mid v$ is a vertex in $\mathrm{G} ® \rho\}$. The language of $\mathrm{G} \in \mathcal{G}$ is undefined when $\{G\}$ is undefined, otherwise it is

$$
\mathbb{L}_{G}=\bigcup_{\rho \in \Re_{G}}\left\{\operatorname{act}(w) \mid w \in\left(\Sigma_{G ® \rho}\right)^{\star} \wedge \gamma(w, \rho)\right\}
$$

where $\gamma(w, \rho)$ holds iff for all $i \neq j$ between 1 and the length of $w$ we have that

1. $w[i] \neq w[j]$, where $w[i]$ stands for the $i$-th vertex in $w$
2. if $w[i] \sqsubseteq \varsigma_{\circledR \rho} \quad w[j]$ and $w[i] \neq w[j]$ then $i<j$
3. if $v \sqsubseteq \epsilon_{\circledast} \rho\left[\begin{array}{l} \\ i]\end{array}\right.$ and $v \neq w[i]$ then there is $h<i$ such that $w[h]=v$

Clause 1 states that vertices in the word are not repeated and, since $w \in\left(\Sigma_{\text {G®P }}\right)^{\star}$, by the consistency lemma (Lemma 4), $w$ is made only of vertices present in the order of the resolution, i.e. $w$ does not mix vertices belonging to different branches. Clause 2 states that words preserve the causal relations of vertices. Clause 3 requires that all the predecessors of an event in the word must precede the event in the word. Notice that $\mathbb{L}_{G}$ is prefix-closed.

Let $\mathrm{G} \in \mathcal{G}$ be a g -choreography such that $\llbracket \mathbb{G} \rrbracket$ and $\{G\}$ are defined. We say that the two semantics are equivalent if the languages of $\llbracket \mathbb{G} \rrbracket$ and $\{G\}$ are equal.

This is demonstrated by showing language equivalence between pomsets and resolutions. Let $r \in \llbracket \mathrm{G} \rrbracket$ be a pomset and $\rho$ be a resolution of G, we say that $r$ is equivalent to $\rho$ when $\rho$ corresponds to $r$ up to $\{G\}$ and each event in $r$ corresponds to some vertex in $\{\mathrm{G}\}$; these correspondences are formalised in 31; here it suffices to give an intuition:

- a vertex $v$ in $\{G\}$ corresponds to $e \in \mathcal{E}_{r}$ for an $r \in \llbracket \mathrm{G} \rrbracket$ (and vice versa) when $\lambda_{r}(e)$ is the label of $v$ and for each $v^{\prime} \sqsubseteq_{\mathrm{G}} v$ on the same branch of $v$ there is an event $e^{\prime} \in \mathcal{E}_{r}$ such that $e^{\prime} \leq_{r} e$ with $e^{\prime}$ corresponding to $v^{\prime}$.
- a resolution $\rho \in \Re_{\mathrm{G}}$ corresponds to $r \in \llbracket G \rrbracket$ when for all vertices $v$ on the branch corresponding to $\rho$ wich are not in $\mathcal{K}$ there is $e \in r$ that corresponds to $v$ (basically, there is a partial order homomorphism between $\rho$ and $r$ ).

To prove the equivalence of our two semantics we must show that there is a bijective correspondence between pomsets in $\llbracket G \rrbracket$ and the set $\Re_{\mathrm{G}}$ of resolutions of $\{G\}$.

Theorem 3 (Correctness and completeness). For each $G \in \mathcal{G},\{G\} \neq \perp$ iff $\llbracket \mathrm{G} \rrbracket \neq \perp$. Moreover, for each $\rho \in \Re_{\mathrm{G}}$ there is a pomset $r \in \llbracket \mathrm{G} \rrbracket$ equivalent to $\rho$ and vice versa.

Details of the proof are reported in 31.

## 7. Conclusions

We introduced an abstract semantics framework of choreographies expressed as global graphs. Our approach is oblivious of the underlying communication semantics and, as discussed below, can be easily adapted to alternative semantics. The semantics of the global artefacts permits to analyse the distributed coordination at the global-level, without the need of examining the local behaviours produced by the projections. Moreover, we can establish precise relations between specifications (choreographies) and their implementations (local artefacts) and we can verify correctness of projections and refinements.

Our framework is more expressive than existing ones; it allows the same participant to operate in both threads of the parallel composition and it does not force passive participants to receive a message signalling the selected choice as first operation in a non-deterministic composition. This is possible due to the well-branched condition. Interestingly, this condition is parametric and depends on the strategy used to find the partitions and the common prefixes required to identify the A-prefix-map. This can range from using always the trivial partitions and the empty prefix (thus enforcing the same syntactical constrains of the existing proposals, some of which have been discussed in Section 22 to finding the partitioning relative to the longest common prefixes. We plan to give a more formal comparison between our semantics and those available in the literature. For this, an interesting approach would be to follow the ideas applied to CCS in [4]. There pomsets where used in combination with proved
transition systems to give an non-interleaving semantics of CCS; basically, given a sequence of transitions $p \xrightarrow{\alpha_{1}} \cdots \xrightarrow{\alpha_{n}} q$ between two CCS processes $p$ and $q$, a pomset $r$ can be derived from a proved transition system so that $r$ represents the equivalence class of traces between $p$ and $q$ "compatible" with traces labelled $\alpha_{1}, \ldots, \alpha_{n}$.

The adequacy of our framework is demonstrated by considering the projections of g-choreographies on communicating machines. With this aim we provide a projection algorithm and we prove its soundness, showing (Theorem 1) that the projection of a sound choreography (i.e. whose semantics is defined) is a deadlock-free system and (Theorem 2) that every execution of the projections is accepted by the choreography.

Theorem 2 manifests the independence of the global semantics from the local one. We regard as a good property of our semantics the fact that global artefacts have "more executions" than the local ones obtained from their projections. Intuitively, this amounts to saying that projections are refinements of the (more abstract) global view. Another advantage is that changing local artefacts does not necessarily require to modify the semantics of the global view. For example, if we consider CFSMs where buffers are used as multisets (instead of as FIFO queues), then all our constructions apply and Theorem 2 can be proved with language equality rather than just inclusion.

Our semantic framework is amenable to variations to consider different semantics at the global level. For instance, an alternative semantics of global views could consider out-of-order outputs; this can be easily formalised by removing the causal dependency between the outputs of two sequential interactions (i.e. the topmost dotted arrows of both Fig. 7a and Fig. 7d are removed). However, soundness of this change depends on the semantics of the local artefacts. In fact, the projections of Fig. 7d can lead to a deadlock if the outputs are interleaved and FIFO CFSM are used as local artefacts, while the interleaved outputs do not cause deadlocks if multiset buffers are used by local artefacts. As another variant one could consider a semantics where a sender has to wait for the receiver to consume the sent messages before proceeding; this is simply attained by adding a causal dependency from the input of B in Fig. 7a to the output from A to $C$ (while removing the dotted relation). We conjecture that this semantics would correspond to the half-duplex semantics of CFSMs. Finally, one can allow sequentially composing choreographies that involve disjoint participants (i.e. allowing the choreography in Fig. 7 e since both $C$ and $D$ do not occur in $A \xrightarrow{x} B$ ).

For simplicity we did not considered recursion/iteration. This can be simply added with a sort of an iterative construct !G where after the execution of the body G, participants "agree" about repeating the loop (unfolding it and generating new communication events) or exiting it. This essentially reduces the problem to a distributed choice, which we solved with the well-branchedness condition. Intuitively, one has to be careful when giving the semantics of (! $G$ ); $\mathrm{G}^{\prime}$. In fact, in order to decide when to exit the body of the iteration G and continue with $\mathrm{G}^{\prime}$ we have to require the existence of an active participant, the well-branchedness of $G+\mathrm{G}^{\prime}$, and the well-sequencedness of $\mathrm{G} ; \mathrm{G}^{\prime}$.

In order to simplify the development of tools based on the semantics we also provide an alternative semantics of choreographies in terms of hypergraphs. Even if its presentation is more cumbersome, it has the benefit that its size does not grow exponentially with the number of choices in the choreography. Moreover, we believe that our semantics could lead to alternative projection algorithms. For instance, we plan to define projections that exploit reflections (which in the hypergraph based semantics resembles the notion of A-prefix-map). This could be better explained by observing what happens when projecting the simple choreography $A \xrightarrow{m} B ; A \xrightarrow{\times} B+A \xrightarrow{m} B ; A \xrightarrow{y} B$, say on participant $B$ (we ignore control points because immaterial). Our algorithm yields the following machine:


However, exploiting the bijection of the reflection, one could directly obtain the machine on the right (avoiding the cost of determinising machines). Note that other projection algorithms capable of handling the example above (as e.g., the one in [26]) also require determinisation, while projections based on types (as e.g., the ones in [21]) are undefined on the previous example because they require prefixes of branches to be pairwise different. Finally, the hypergraph based semantics can be easily extended to graphs with structured loops that are represented as repetitions of g-choreography. This is possible since the semantics side-conditions do not depend on the (possibly infinite) language of the choreography, but rather on the hypergraphs, which are finite.

## References

[1] Charlton Barreto \& et al. (2007): Web Services Business Process Execution Language Version 2.0. https://www.oasis-open.org/committees/ download.php/23964/wsbpel-v2.0-primer.htm.
[2] Davide Basile, Pierpaolo Degano, Gian-Luigi Ferrari \& Emilio Tuosto (2016): Relating two automata-based models of orchestration and choreography. JLAMP 85(3), pp. 425 - 446.
[3] Laura Bocchi, Hernán C. Melgratti \& Emilio Tuosto (2014): Resolving Non-determinism in Choreographies. In: ESOP, pp. 493-512.
[4] Gérard Boudol \& Ilaria Castellani (1988): Permutation of transitions: an event structure semantics for CCS and SCCS. In J.W. de Bakker, W.-P.
de Roever \& G. Rozenberg, editors: Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency, Lecture Notes in Computer Science 354, Springer-Verlag, pp. 411-427.
[5] Daniel Brand \& Pitro Zafiropulo (1983): On Communicating Finite-State Machines. Journal of the ACM 30(2), pp. 323-342.
[6] Mario Bravetti \& Gianluigi Zavattaro (2007): Towards a Unifying Theory for Choreography Conformance and Contract Compliance. In: Proceedings of the 6th International Conference on Software Composition, SC'07, SpringerVerlag, pp. 34-50.
[7] Marco Carbone, Kohei Honda \& Nobuko Yoshida (2007): A Calculus of Global Interaction based on Session Types. Electronic Notes in Theoretical Computer Science 171(3), pp. 127 - 151.
[8] Marco Carbone \& Fabrizio Montesi (2013): Deadlock-freedom-by-design: multiparty asynchronous global programming. In: POPL13, pp. 263-274.
[9] Giuseppe Castagna, Mariangiola Dezani-Ciancaglini \& Luca Padovani (2012): On Global Types and Multi-Party Session. LMCS 8(1).
[10] Gérard Cécé \& Alain Finkel (2005): Verification of programs with half-duplex communication. I\&C 202(2), pp. 166-190.
[11] Mario Coppo, Mariangiola Dezani-Ciancaglini, Nobuko Yoshida \& Luca Padovani (2016): Global progress for dynamically interleaved multiparty sessions. Mathematical Structures in Computer Science 26(2), pp. 238-302.
[12] Mila Dalla Preda, Maurizio Gabbrielli, Saverio Giallorenzo, Ivan Lanese \& Mauro Jacopo (2015): Dynamic Choreographies - Safe Runtime Updates of Distributed Applications. In: COORDINATION 2015, pp. 67-82.
[13] Mila Dalla Preda, Maurizio Gabbrielli, Saverio Giallorenzo, Ivan Lanese \& Mauro Jacopo (2017): Dynamic Choreographies - Safe Runtime Updates of Distributed Applications. Logical methods in Computer Science 13(2).
[14] Pierpaolo Degano \& Ugo Montanari (1987): A Model for Distributed Systems Based on Graph Rewriting. Journal of the ACM 34(2), pp. 411-449.
[15] Pierre-Malo Deniélou \& Nobuko Yoshida (2012): Multiparty Session Types Meet Communicating Automata. In: ESOP, pp. 194-213.
[16] Haim Gaifman \& Vaughan R Pratt (1987): Partial order models of concurrency and the computation of functions. In: LICS, pp. 72-85.
[17] Object Management Group: Business Process Model and Notation. http: //www.bpmn.org.
[18] Roberto Guanciale \& Emilio Tuosto (2016): An Abstract Semantics of the Global View of Choreographies. In: Proceedings 9th Interaction and Concurrency Experience, ICE 2016, Heraklion, Greece, 8-9 June 2016., pp. 67-82.
[19] Kohei Honda, Nobuko Yoshida \& Marco Carbone (2008): Multiparty asynchronous session types. In George C. Necula \& Philip Wadler, editors: Proceedings of the 35th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2008, San Francisco, California, USA, January 7-12, 2008, ACM, pp. 273-284, doi 10.1145/1328438.1328472. Available at http://doi.acm.org/10.1145/1328438.1328472.
[20] Kohei Honda, Nobuko Yoshida \& Marco Carbone (2008): Multiparty asynchronous session types. In: POPL, pp. 273-284.
[21] Kohei Honda, Nobuko Yoshida \& Marco Carbone (2016): Multiparty Asynchronous Session Types. Journal of the ACM 63(1), pp. 9:1-9:67. Extended version of a paper presented at POPL08.
[22] Joost-Pieter Katoen \& Lennard Lambert (1998): Pomsets for message sequence charts. Formale Beschreibungstechniken für Verteilte Systeme, pp. 197-208.
[23] Nickolas Kavantzas, Davide Burdett, Gregory Ritzinger, Tony Fletcher \& Yves Lafon (2004): Web Services Choreography Description Language Version 1.0. http://www.w3.org/TR/2004/WD-ws-cdl-10-20041217.
[24] Ivan Lanese, Claudio Guidi, Fabrizio Montesi \& Gianluigi Zavattaro (2008): Bridging the Gap Between Interaction- and Process-Oriented Choreographies. In: Proceedings of the 2008 Sixth IEEE International Conference on Software Engineering and Formal Methods, SEFM '08, IEEE Computer Society, pp. 323-332.
[25] Julien Lange \& Emilio Tuosto: ChorGram. https://bitbucket.org/ emlio_tuosto/chorgram/wiki/Home
[26] Julien Lange, Emilio Tuosto \& Nobuko Yoshida (2015): From Communicating Machines to Graphical Choreographies. In: POPL15, pp. 221-232.
[27] Julien Lange, Emilio Tuosto \& Nobuko Yoshida (2017): A tool for choreography-based analysis of message-passing software. ACM. To appear. Available at http://www.cs.le.ac.uk/~et52/chorgram_betty_ch.pdf.
[28] James Lewis \& Martin Fowler (2014): Microservices: a definition of this new architectural term. http://martinfowler.com/articles/microservices. html.
[29] Vaughan Pratt (1986): Modeling concurrency with partial orders. International Journal of Parallel Programming 15(1), pp. 33-71.
[30] Zongyan Qiu, Xiangpeng Zhao, Chao Cai \& Hongli Yang (2007): Towards the Theoretical Foundation of Choreography. In: WWW07, ACM, pp. 973-982.
[31] Emilio Tuosto \& Roberto Guanciale (2017): Semantics of global view of choreographies: proofs. Technical Report. Available at http://www.cs.le ac.uk/people/et52/jlamp-with-proofs.pdf.


[^0]:    The authors are grateful to the reviewers for the helpful comments and to Ivan Lanese for useful comments.
    *Principal Corresponding author
    ${ }^{* *}$ Corresponding author

[^1]:    ${ }^{1}$ We will use the term 'artefact' when referring to actual specifications embodying the global/local views. Such embodiments may assume various forms: types [21], programs [13], graphs and automata [26, 15], executable models [23, 1], etc. Typically, the literature uses the (overloaded) word 'model' to refer to this flora of embodiments. We prefer the word 'artefact' because it allows us to refer to different contexts and different abstraction levels without attaching yet another meaning to 'model'.

[^2]:    ${ }^{2}$ The term "message" could be misleading, but we adopt it because it is widespread in the literature.

[^3]:    ${ }^{3}$ We mention a tiny portion of the literature in way of example; no claim of exhaustiveness.

[^4]:    ${ }^{4}$ We are currently working on the definition and implementation of algorithms to establish equivalences of global views based on the abstract semantics presented in this paper.

[^5]:    ${ }^{5}$ We depart from the usual notation $\mathrm{A} \rightarrow \mathrm{B}: \mathrm{m}$ to a have a more lightweight syntax.
    ${ }^{6}$ At the cost of adding technical complexity, one can automatically assign a unique identifier to such control points.
    ${ }^{7}$ Cycles are not considered for simplicity and can be easily added.

[^6]:    ${ }^{8}$ We indexed our examples with the numbering of the figure they are in; therefore, we will hereafter avoid cross-referencing the figures.

[^7]:    ${ }^{9}$ Namely, families of disjoint subsets covering $\llbracket G_{1} \rrbracket l_{A}$ and $\llbracket G_{2} \rrbracket l_{A}$, respectively.

[^8]:    ${ }^{10}$ We are currently developing the ChorGram 25, 27 tool chain. In particular, we are extending the toolkit with algorithms based on the semantics presented in this section. However, the development is in too a preliminary phase to be in the scope of this paper.

