Causal Message Sequence Charts

Thomas Gazagnaire [1]
Citrix Systems R&D Ltd., UK

Blaise Genest
IRISA/CNRS, France

Loïc Hélouët
IRISA/INRIA, France

P.S. Thiagarajan
School of Computing, National University of Singapore

Shaofa Yang [1]
UNU-IIST, Macao.

Abstract

Scenario languages based on Message Sequence Charts (MSCs) have been widely studied in the last decade. The high expressive power of MSCs renders many basic problems concerning these languages undecidable. However, several of these problems are decidable for languages that possess a behavioral property called "existentially bounded". Unfortunately, collections of scenarios outside this class are frequently exhibited by systems such as sliding window protocols. We propose here an extension of MSCs called causal Message Sequence Charts and a natural mechanism for defining languages of causal MSCs called causal HMSCs (CaHMSCs). These languages preserve decidable properties without requiring existential bounds. Further, they can model collections of scenarios generated by sliding window protocols. We establish here the basic theory of CaHMSCs as well as the expressive power and complexity of decision procedures for various subclasses of CaHMSCs. We also illustrate the modeling power of our formalism with the help of a realistic example based on the TCP sliding window feature.

1 Introduction

Formal modeling and analysis of the behavior of communicating systems helps discover design mistakes at early stages of conception. This domain is well studied; for a typical study, the reader is referred to [18]. Formal models of behaviors are becoming more frequent, at least in terms of documenting

Email addresses: thomas.gazagnaire@citrix.com (Thomas Gazagnaire), blaise.genest@irisa.fr (Blaise Genest), lhelouet@irisa.fr (Loïc Hélouët), thiagu@comp.nus.edu.sg (P.S. Thiagarajan), ysf@iist.unu.edu (Shaofa Yang).

[1] Work done while this author was at IRISA/INRIA, France.

parts of a system using standard notations such as UML. However, UML-like notations are not always equipped with a clear semantics and when this is the case, the proposed formalism often lacks the expressive power needed to model all the relevant behaviors of the designed system. A frequent problem with communication protocols, for instance, is that the state space of the designed system can be infinite. One then must choose between an abstraction to maintain the description as a finite state representation or to explicitly model the status the communication channels using communicating finite state machines or Petri nets but in doing so loose decidability of most of logical properties (see [5,10] for instance). Formal modeling of distributed systems is then a tradeoff between expressiveness of a model and the ability to perform behavioral analysis effectively. The main technical theme of the present paper can be viewed as exploiting one such trade-off in the setting of scenario-based specifications.

Turning now to scenarios, models of infinite state communicating systems such as π-Calculus or communicating finite state machines focus on a process-based descriptions of behaviors. The specific patterns of interactions between the processes that can arise are implicit and need to be extracted by systematically applying the operational semantics of the model to a specific system description. *Scenarios* offer a different modeling paradigm, by focusing on compositions of simple diagrams describing a single finite stretch of interactions of several agents via the exchange of messages. The challenge in studying them does not reside in the the individual scenarios diagrams, which are often simple easy-to-understand use cases of the system. Rather, it lies in the fact that a rich class of scenarios can result through the composition of the basic scenarios as permitted by the protocol/system that is being modeled. Since the basic scenarios can be formally viewed as partial orders, studying scenarios-based specifications of communicating systems amounts to analyzing the composition of partial orders; an idea proposed decades ago by [22,9,26].

Scenarios are often based on formal objects called Message Sequence Charts (MSCs) and consequently MSCs have attracted considerable interest in the last decade [25,24,3,17,14,23,16]. The attractiveness of this notation derives from two major characteristics. Firstly, MSCs have a simple and appealing graphical representation based on just a few concepts: processes (lifelines), messages and internal actions. The graphical notation for MSCs represents the time evolution of processes by vertical lines, atomic actions by squares containing action names and located on a process line, and message exchanges by arrows from the sending process to the receiving one. See for instance the two MSCs shown in Figure 1. Secondly, from a mathematical standpoint, scenario languages can be generated by finite-state automata over an alphabet of MSCs. These automata are called High-level Message Sequence Charts (HMSCs) [19]. HMSCs can in turn be specified using a graphical notation: they are graphs where each node are either a start symbol (a downward pointing

triangle), an end symbol (an upward pointing triangle), a connection point (a circle), or a reference to another MSC or HMSC (A reference name enclosed in a rectangle) [30]. Behaviors defined by HMSCs are simply concatenations of MSC references along paths from a start to an end symbol (concatenation will be defined formally in section 2). For example, the MSC M shown in Figure 2 is a member of the MSC language generated by the HMSC of Figure 1 while the MSC N shown in Figure 2 is not.

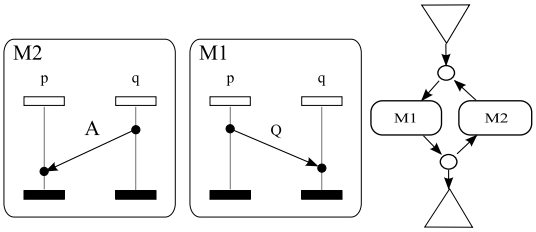

Fig. 1. An HMSC over two MSCs

HMSCs are very expressive and hence a number of basic problems associated with them cannot be solved effectively. For instance, it is undecidable whether two HMSCs generate the same collection of MSCs [25] or whether an HMSC generates a regular MSC language; an MSC language is regular if the collection of all the linearizations of all the MSCs in the language is a regular string language in the usual sense. Consequently, subclasses of HMSCs have been identified [24,3,14] and studied.

On the other hand, a basic limitation of HMSCs is that their MSC languages are finitely generated. More precisely, each MSC in the language can be defined as the sequential composition of elements chosen from a fixed finite set of MSCs [23]. However, the behaviors of many protocols constitute MSC languages that are *not* finitely generated. This occurs for example with scenarios generated by the alternating bit protocol. Such protocols can induce a collection of braids like N in Figure 2 which cannot be finitely generated.

One way to handle this is to work with the so called *compositional* HMSCs [15] in which message emissions and receptions are decoupled in individual MSCs but matched up at the time of composition, so as to yield an MSC. Compositional HMSCs are however notationally awkward and do not possess the visual appeal of HMSCs. Further, the general class of compositional HMSC languages embeds the full expressive power of communicating automata [5] and consequently inherits all their undecidability results.

This paper proposes a new approach to increase the modeling power of HMSCs in a tractable manner. We first extend the notion of an MSC to a *causal* MSC in which the events belonging to each lifeline (process), instead of being linearly ordered, are allowed to be partially ordered. In order to define a composition operation for causal MSCs, we assume a suitable Mazurkiewicz trace alphabet [9] for each lifeline. This leads to the notion of causal HMSCs.

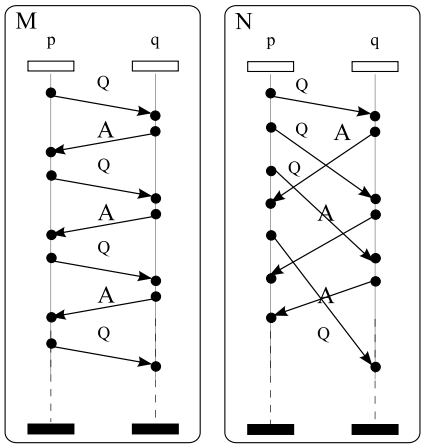

Fig. 2. Two MSCs M and N

Our goal is to develop the basic theory of causal HMSCs. One could hope to exploit the property called *existential boundedness* [13] for this purpose since it leads to a powerful proof technique for establishing decidability results for HMSCs. Informally, this property states that there is a uniform upper bound K such that for every MSC in the language *there exists* an execution along which -from start to finish- all channels remain K-bounded. Unfortunately, a causal HMSC (i.e. the MSC language associated with a causal HMSC) is *a priori* not existentially bounded. Hence the proof method cited above cannot be used to obtain the desired decidability results. Instead, we need to generalize the methods of [24] and of [14] in a non-trivial way.

Our first major result is to formulate natural -and decidable- structural conditions and to show that causal HMSCs satisfying these conditions generate MSC languages that are regular. Our second major result is that the inclusion problem for causal HMSCs (i.e. given two causal HMSCs, whether the MSC language defined by the first one is included in the MSC language of the other) is decidable for causal HMSCs using the same Mazurkiewicz trace alphabets, provided at least one of them has the structural property known as *globally-cooperative*. Furthermore, we prove that the restriction that the two causal HMSCs have identical Mazurkiewicz trace alphabets associated with them is necessary. These results constitute a non-trivial extension for causal HMSCs of comparable results on HMSCs [24,3,16] and [14]. In addition, we identify the property called "window-bounded" which appears to be an important ingredient of the "braid"-like MSC languages generated by many protocols. Basically, this property bounds the number of messages a process p can send to a process q before having received an acknowledgment to the earliest message. We show it is decidable if a causal HMSC generates a window-bounded MSC language. Finally, we compare the expressive power of languages based on causal HMSCs with other known HMSC-based language and give a detailed example based on the TCP protocol to illustrate the modeling potential of causal HMSCs.

This paper is an extended version of the work presented in [12], and contains several important changes, improvements and new examples. Specifically, the definition of s-regularity and of global-cooperativity has been weakened for causal HMSCs. As a result, causal HMSCs which were not s-regular according to [12] are deemed to be s-regular under the weakened definition.

In the next section we introduce causal MSCs and causal HMSCs. We also define the means for associating an ordinary MSC language with a causal HMSC. In the subsequent section we develop the basic theory of causal HMSCs. To this end, we identify the subclasses of s-regular (syntactically regular) and globally-cooperative causal HMSCs and develop our decidability results. In section 4, we identify the "window-bounded" property, and show that one can decide if a causal HMSC generates a window-bounded MSC language. In section 5 we compare the expressive power of languages based on causal HMSCs with other known HMSC-based language classes. Finally, in section 6, we give a detailed example based on the TCP protocol to illustrate the modeling potential of causal HMSCs. Due to lack of space, some proofs are omitted or only sketched, but can be found in an extended version, available at `www.irisa.fr/distribcom/Personal_Pages/helouet/Papers/GGHTY-CMSC-Long.pdf`.

2 MSCs, causal MSCs and causal HMSCs

Through the rest of the paper, we fix a finite set $\mathcal{P}$ of process names with $|\mathcal{P}| > 1$. For convenience, we let p, q range over $\mathcal{P}$ and we will not mention that $p \in \mathcal{P}$ when there is no confusion. We also fix finite nonempty sets Msg, Act of message types and internal action names respectively. We define the alphabets $\Sigma_! = \{p!q(m) \mid p, q \in \mathcal{P}, p \neq q, m \in Msg\}$, $\Sigma_? = \{p?q(m) \mid p, q \in \mathcal{P}, p \neq q, m \in Msg\}$, and $\Sigma_{act} = \{p(a) \mid p \in \mathcal{P}, a \in Act\}$. The letter $p!q(m)$ means the sending of a message with content m from p to q; $p?q(m)$ the reception of a message of content m at p from q; and $p(a)$ the execution of an internal action a by process p. Let $\Sigma = \Sigma_! \cup \Sigma_? \cup \Sigma_{act}$. We define the *location* of a letter α in Σ, denoted $loc(\alpha)$, by $loc(p!q(m)) = p = loc(p?q(m)) = loc(p(a))$. For each process p in $\mathcal{P}$, we set $\Sigma_p = \{\alpha \in \Sigma \mid loc(\alpha) = p\}$.

Definition 1 *A* causal MSC *over* $(\mathcal{P}, \Sigma)$ *is a structure* $B = (E, \lambda, \{\sqsubseteq_p\}, \ll)$, *where* E *is a finite nonempty set of* events, $\lambda : E \to \Sigma$ *is a labelling function, and the following conditions hold:*

- *For each process* p, $\sqsubseteq_p \subseteq E_p \times E_p$ *is a* partial order, *where* $E_p = \{e \in E \mid \lambda(e) \in \Sigma_p\}$.
- $\ll \subseteq E_! \times E_?$ *is a bijection, where* $E_! = \{e \in E \mid \lambda(e) \in \Sigma_!\}$ *and* $E_? = \{e \in E \mid \lambda(e) \in \Sigma_?\}$. *For each* $e \ll e'$, $\lambda(e) = p!q(m)$ *iff* $\lambda(e') = q?p(m)$.
- *The transitive closure* $\leq$ *of the relation* $\Big(\bigcup_{p \in \mathcal{P}} \sqsubseteq_p\Big) \cup \ll$ *is a partial order.*

For each p, the relation $\sqsubseteq_p$ dictates the "causal" order in which events of E_p may be executed. We let $\widehat{\sqsubseteq}_p \subseteq E_p \times E_p$ denote the least relation such that $\sqsubseteq_p$ is the reflexive and transitive closure of $\widehat{\sqsubseteq}_p$ ($\widehat{\sqsubseteq}_p$ is the Hasse diagram of $\sqsubseteq_p$). The relation $\ll$ identifies pairs of message-emission and message-reception events. The *size* of a causal MSC B is denoted by $|B|$ and is simply the cardinal of its set of events $|E|$. We say that the causal MSC B is *weak-FIFO* iff for any $e \ll f$, $e' \ll f'$ such that $\lambda(e) = \lambda(e') = p!q(m)$ (and thus $\lambda(f) = \lambda(f') = q?p(m)$), we have either $e \sqsubseteq_p e'$ and $f \sqsubseteq_q f'$; or $e' \sqsubseteq_p e$ and $f' \sqsubseteq_q f$. In weak-FIFO [2] scenarios, messages of the same content between two given processes cannot overtake. Note however that messages of different kind between two processes can overtake. Note that we do not demand *a priori* that a causal MSC must be weak-FIFO. Testing (weak) fifoness of a causal MSC of size b can be done in at most $\mathcal{O}(\frac{b^2}{8} - \frac{b}{4})$, by checking in the worst case for all pairs of messages $e \ll f$ and $e' \ll f'$ in the MSC that $e \sqsubseteq_p e'$ for some p implies $f \sqsubseteq_q f'$ for some q. We will say that two causal MSCs are *equal*, and write $B_1 = B_2$ when they are defined over the same sets of processes and are isomorphic, that is there exists a bijection from E_1 to E_2 that preserves labeling, $\{\sqsubseteq_p\}$ and $\ll$.

A *linearization* of B is a word $a_1 a_2 \dots a_\ell$ over Σ such that $E = \{e_1, \dots, e_\ell\}$ with $\lambda(e_i) = a_i$ for each i; and $e_i \leq e_j$ implies $i \leq j$ for any i, j. We let $Lin(B)$ denote the set of linearizations of B. Clearly, $Lin(B)$ is nonempty. We set $Alph(B) = \{\lambda(e) \mid e \in E\}$, and $Alph_p(B) = Alph(B) \cap \Sigma_p$ for each p.

The leftmost part of Figure 3 depicts a causal MSC M. The graphical representation of causal MSCs is a light adaptation of the graphical notation for MSCs. In this diagram, we enclose events of each process p in a vertical box and show the partial order $\sqsubseteq_p$ in the standard way. In case $\sqsubseteq_p$ is a total order, we place events of p along a vertical line with the minimum events at the top and omit the box. In particular, in M, the two events on p are not ordered (i.e. $\widehat{\sqsubseteq}_p$ is empty) and $\sqsubseteq_q$ is a total order. Members of $\ll$ are indicated by horizontal or downward-sloping arrows labelled with the transmitted message. Both words $p!q(Q).q!p(A).q?p(Q).p?q(A)$ and $q!p(A).p?q(A).p!q(Q).q?p(Q)$ are linearizations of M.

An *MSC* $B = (E, \lambda, \{\sqsubseteq_p\}, \ll)$ is defined in the same way as a causal MSC except that every $\sqsubseteq_p$ is required to be a *total order*. In an MSC B, the relation $\sqsubseteq_p$ must be interpreted as the visually observed order of events in one sequential execution of p. Let $B' = (E', \lambda', \{\sqsubseteq'_p\}, \ll')$ be a causal MSC. Then we say the MSC B is a *visual extension* of B' if $E' = E$, $\lambda' = \lambda$, $\sqsubseteq'_p \subseteq \sqsubseteq_p$ and $\ll' = \ll$. We let $Vis(B')$ denote the set of visual extensions of B'. Note

[2] A different notion called strong FIFO that does not allow overtakings of messages between two given processes of different content is also frequently used in the MSC literature.

that linearizations alone are not sufficient in general to characterize a single visual extension or causal MSCs. Consider for instance the set of linearizations $\{p!q(m).p!q(m).p!q(m).q?p(m).q?p(m).q?p(m)\}$. This language corresponds to a MSC that contains three messages m from process p to process q, and such that the reception of the last message must occur before the two other receptions, but in which the first and second message may overtake or not.

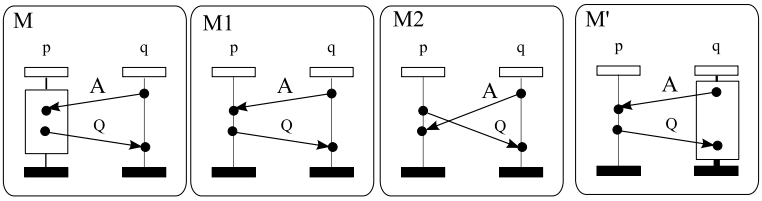

Fig. 3. Two causal MSC M, M' and the visual extensions $M1, M2$ of M.

In Figure 3, $Vis(M)$ consists of MSCs $M1, M2$. The initial idea of visual ordering comes from [2], that notices that depending on the interpretation of an MSC, for example when a lifeline describes a physical entity in a network, imposing an ordering on message receptions is not possible. Hence, [2] distinguishes two orderings on MSCs: a visual order (that is the usual order used for MSCs), that comes from the relative order of events along an instance line, and a causal order, that is weaker, and does not impose any ordering among consecutive receptions.

Note that for a given causal MSC B, we can not always compute an MSC in $Vis(B)$ by replacing $\{\sqsubseteq_p\}_{p\in\mathcal{P}}$ by $\{<_p\}_{p\in\mathcal{P}}$, where $<_p$ is a linear extension of $\sqsubseteq_p$. Indeed, we can not chose separately any linear extension for each process $p \in \mathcal{P}$, as the transitive closure of $\Big(\bigcup_{p\in\mathcal{P}} <_p\Big) \cup \ll$ must remain a partial order. Consider for instance the causal MSC M' in Figure 3: one can not chose as linear extension $q!p(A) <_q q?p(Q)$, as the transitive closure of $(<_p \cup <_q \cup \ll)$ would not be a partial order. Hence, the only visual extension of M' is the MSC M_1.

We next define a concatenation operation for causal MSCs. Unlike for usual MSCs, events of a same process need not be dependent. To express whether there should be a dependency or not, for each process p in $\mathcal{P}$, we fix a concurrent alphabet (Mazurkiewicz trace alphabet [9]) (Σ_p, I_p) for each process $p \in \mathcal{P}$, where $I_p \subseteq \Sigma_p \times \Sigma_p$ is a symmetric and irreflexive relation called the *independence relation* over the alphabet of actions Σ_p. We denote the *dependence relation* $(\Sigma_p \times \Sigma_p) - I_p$ by D_p. These relations are fixed *for the rest of the paper* (unless explicitly stated otherwise). Following the usual definitions of Mazurkiewicz traces, for each (Σ_p, I_p), the associated trace equivalence relation $\sim_p$ over $\Sigma_p^\star$ is the least equivalence relation such that, for any u, v in $\Sigma_p^\star$ and α, β in Σ_p, $\alpha\ I_p\ \beta$ implies $u\alpha\beta v \sim_p u\beta\alpha v$. Equivalence classes of $\sim_p$ are called *traces*. For u in $\Sigma_p^\star$, we let $[u]_p$ denote the trace containing u.

Let $B = (E, \lambda, \{\sqsubseteq_p\}, \ll)$ be a causal MSC. We say $\sqsubseteq_p$ *respects* the trace alphabet (Σ_p, I_p) iff for any $e, e' \in E_p$, the following hold:

(i) $\lambda(e)\ D_p\ \lambda(e')$ implies $e\ \sqsubseteq_p\ e'$ or $e'\ \sqsubseteq_p\ e$
(ii) $e\ \widehat{\sqsubseteq}_p\ e'$ implies $\lambda(e)\ D_p\ \lambda(e')$

A causal MSC B is said to respect the trace alphabets $\{(\Sigma_p, I_p)\}_{p\in\mathcal{P}}$ iff $\sqsubseteq_p$ respects (Σ_p, I_p) for every p. In order to gain modelling power, we have allowed each $\sqsubseteq_p$ to be *any* partial order, not necessarily respecting (Σ_p, I_p). We can now define the concatenation operation of causal MSCs using the trace alphabets $\{(\Sigma_p, I_p)\}_{p\in\mathcal{P}}$.

Definition 2 *Let $B = (E, \lambda, \{\sqsubseteq_p\}, \ll)$ and $B' = (E', \lambda', \{\sqsubseteq'_p\}, \ll')$ be causal MSCs. We define the* concatenation *of B with B', denoted by $B \circledcirc B'$, as the causal MSC $B'' = (E'', \lambda'', \{\sqsubseteq''_p\}, \ll'')$ where:*

- *E'' is the disjoint union of E and E'. λ'' is given by: $\lambda''(e) = \lambda(e)$ if $e \in E$, $\lambda''(e) = \lambda'(e)$ if $e \in E'$ and $\ll'' = \ll \cup \ll'$.*
- *For each p, $\sqsubseteq''_p$ is the transitive closure of*

$$\sqsubseteq_p \bigcup \sqsubseteq'_p \bigcup \{(e, e') \in E_p \times E'_p \mid \lambda(e)\ D_p\ \lambda'(e')\}$$

Clearly, $\circledcirc$ is a well-defined and associative operation. Note that in case B and B' are MSCs and $D_p = \Sigma_p \times \Sigma_p$ for every p, then the result of $B \circledcirc B'$ is the asynchronous concatenation (also called weak sequential composition) of B with B' [27], which we denote by $B \circ B'$. Note that when B_1 and B_2 are weak-FIFO causal MSCs, then their concatenation is also weak-FIFO. This property comes from the irreflexive nature of the independence relations. This remark also holds for the concatenation of MSCs. We also remark that the concatenation of causal MSCs is different from the concatenation of traces. The concatenation of trace $[u]_p$ with $[v]_p$ is the trace $[uv]_p$. However, a causal MSC B needs not respect $\{(\Sigma_p, I_p)\}_{p\in\mathcal{P}}$. Consequently, for a process p, the projection of $Lin(B)$ on $Alph_p(B)$ is *not* necessarily a trace.

Figure 4 shows an example of sequential composition of two causal MSCs B_1 and B_2, with the dependency relations D_p and D_q being the symmetric and reflexive closure of $\Big\{\Big(p!q(m), p!q(n)\Big), \Big(p!q(n), p?q(u)\Big)\Big\}$ and $\Big\{\Big(q?p(n), q!p(v)\Big)\Big\}$ respectively. The resulting causal MSC $B_1 \circledcirc B_2$ is obtained by copying the order contained ib B_2 below B_1, and then adding dependencies between events located on the same processes according to the dependency alphabet. For instance, on process p, the emission of message n and the reception of message u are ordered in $B_1 \circledcirc B_2$, because $(p!q(n), p?q(u)) \in D_p$. Conversely, as $(p!q(m), p?q(u)) \notin D_p$, the sending of message m and the reception of message u are onordered in $B_1 \circledcirc B_2$. Note that although the dependence relation D_p contains the pair $\Big(p!q(m), p!q(n)\Big)$, sendings of messages m and n by process p in B_1 can be unordered, and remain unordered after composition.

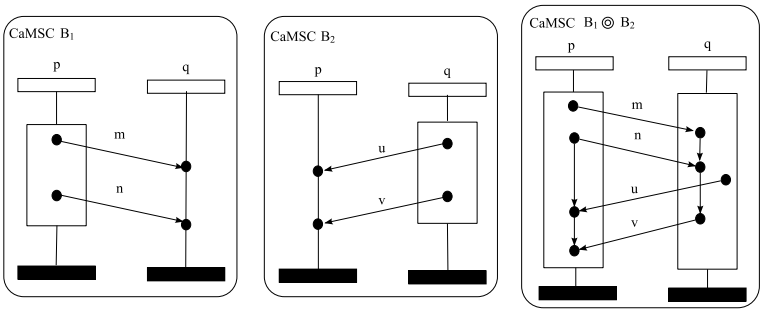

Fig. 4. Concatenation example.

An usual way to extend the composition mechanism is to use an automaton labelled by basic scenarios, to produce scenario languages. These automata are called High-level Message Sequence Charts (or HMSCs for short)[30,28] when MSCs are concatenated, and a similar construct exists for compositional Message Sequence Charts [15,13,8]. We can now define causal HMSCs. Recall that we have fixed a set $\mathcal{P}$ of process names and a family $\{(\Sigma_p, I_p)\}_{p\in\mathcal{P}}$ of Mazurkiewicz trace alphabets.

Definition 3 *A* causal HMSC *over* $(\mathcal{P}, \{(\Sigma_p, I_p)\}_{p\in\mathcal{P}})$ *is a structure* $H = (N, N_{in}, \mathcal{B}, \longrightarrow, N_{fi})$ *where* N *is a finite nonempty set of* nodes, $N_{in} \subseteq N$ *the nonempty set of initial nodes,* $\mathcal{B}$ *a finite nonempty set of causal MSCs,* $\longrightarrow \subseteq N \times \mathcal{B} \times N$ *the transition relation, and* $N_{fi} \subseteq N$ *the nonempty set of final nodes.*

A *path* in the causal HMSC H is a sequence $\rho = n_0 \xrightarrow{B_1} n_1 \xrightarrow{B_2} \cdots \xrightarrow{B_\ell} n_\ell$. If $n_0 = n_\ell$, then we say ρ is a cycle. The path ρ is *accepting* iff $n_0 \in N_{in}$ and $n_\ell \in N_{fi}$. The causal MSC generated by ρ, denoted $\circledcirc(\rho)$, is $B_1 \circledcirc B_2 \circledcirc \cdots \circledcirc B_\ell$. We let $cMSC(H)$ denote the set of causal MSCs generated by accepting paths of H. Intuitively, $cMSC(H)$ is the set of causal MSCs that can be obtained by glueing causal MSCs one after another along paths of H. We also set $Vis(H) = \bigcup\{Vis(M) \mid M \in cMSC(H)\}$ and $Lin(H) = \bigcup\{Lin(M) \mid M \in cMSC(H)\}$. Obviously, $Lin(H)$ is also equal to $\bigcup\{Lin(M) \mid M \in Vis(H)\}$. We shall refer to $cMSC(H)$, $Vis(H)$, $Lin(H)$, respectively, as the causal language, visual language and linearization language of H. Note that HMSCs are sometimes formalized as MSC graphs, where MSCs label states rather than transitions. However, this does not change the expressive power of the models.

An *HMSC* $H = (N, N_{in}, \mathcal{B}, \longrightarrow, N_{fi})$ is defined in the same way as a causal HMSC except that $\mathcal{B}$ is a finite set of MSCs, and that the concatenation operation used to produce MSC languages is the weak sequential composition $\circ$. HMSCs can then be considered as causal HMSCs labelled with MSCs, and equipped with empty independence relation (for every $p \in \mathcal{P}, I_p = \emptyset$). Hence, a path ρ of H generates an MSC by concatenating the MSCs along ρ with operation $\circ$. We let $Vis(H)$ denote the set of MSCs generated by accepting paths of H with $\circ$, and call $Vis(H)$ the visual language of H. Recall that an MSC language (i.e. a collection of MSCs) L is *finitely generated* [23] iff

there exists a finite set X of MSCs satisfying the condition: for each MSC B in L, there exist $B_1, \ldots, B_\ell$ in X such that $B = B_1 \circ \cdots \circ B_\ell$. Many protocols exhibit scenario collections that are *not* finitely generated. For example, sliding window protocols can generate arbitrarily large MSCs repeating the communication behavior shown in MSC N of Figure 2. One basic limitation of HMSCs is that their visual languages are *finitely generated*. In contrast, the visual language of a causal HMSC is *not* necessarily finitely generated. Consider for instance the HMSC H in Figure 1, and the independence relations given by: $I_p = \{((p!q(Q), p?q(A)), (p?q(A), p!q(Q)))\}$ and $I_q = \emptyset$. $M1$ and $M2$ can be seen as causal MSCs, and H as a causal HMSC over $(\mathcal{P} = \{p, q\}, \{(\Sigma_p, I_p)(\Sigma_q, I_q)\})$. Clearly, $Vis(H)$ is not finitely generated, as it contains infinitely many MSCs similar to N of Figure 2. Throughout the paper, we will use the following standardized graphical convention to depict (causal) HMSCs: nodes are represented by circles, initial nodes are nodes connected to a downward pointing triangle, final nodes are nodes connected to an upward pointing triangle, and a transition $t = (n, B, n')$ is represented by an arrow decorated by a rectangle containing the (causal) MSC B.

3 Regularity and Language Inclusion for causal HMSCs

3.1 Semantics for causal HMSCs

As things stand, a causal HMSC H defines three syntactically different languages, namely its linearization language $Lin(H)$, its visual (MSC) language $Vis(H)$ and its causal MSC language $cMSC(H)$. The next proposition shows that they are also semantically different in general. It also identifies the restrictions under which they match semantically.

Proposition 1 *Let H, H' be two causal HMSCs over the* same *family of trace alphabets $\{(\Sigma_p, I_p)\}_{p\in\mathcal{P}}$. Consider the following six hypotheses:*

(i)	$cMSC(H) = cMSC(H')$	*(i)'*	$cMSC(H) \cap cMSC(H') \neq \emptyset$
(ii)	$Vis(H) = Vis(H')$	*(ii)'*	$Vis(H) \cap Vis(H') \neq \emptyset$
(iii)	$Lin(H) = Lin(H')$	*(iii)'*	$Lin(H) \cap Lin(H') \neq \emptyset$

Then we have:

- *$(i) \Rightarrow (ii)$, $(i)' \Rightarrow (ii)'$, $(ii) \Rightarrow (iii)$ and $(ii)' \Rightarrow (iii)'$ but the converses do not hold in general.*
- *If every causal MSC labelling transitions of H and H' respects $\{(\Sigma_p, I_p)\}_{p\in\mathcal{P}}$, then $(i) \Leftrightarrow (ii)$ and $(i)' \Leftrightarrow (ii)'$.*
- *If every causal MSC labelling transitions of H and H' is weak-FIFO, then $(ii) \Leftrightarrow (iii)$ and $(ii)' \Leftrightarrow (iii)'$.*

For most purposes, the relevant semantics for a causal HMSC seems to be its visual language. However in the following we focus first in section 3.2 on the linearization language properties of causal HMSCs. Then, we focus in

section 3.3 on the causal language properties. Using the above proposition 1, it is then straightforward to translate these properties to the visual language of causal HMSCs, when the right hypothesis apply.

3.2 Regular sets of linearizations

It is undecidable in general whether an HMSC has a regular linearization language [24]. In the literature, a subclass of HMSCs called regular [24] (or bounded [3]) HMSCs, has been identified. In this paper, to avoid overloading "regular", we shall refer to this syntactic property as "s-regular". The importance of this property lies in the fact that linearization language of every s-regular HMSC is regular. Furthermore, one can effectively decide whether an HMSC is s-regular. Our goal is to extend these results to causal HMSCs.

The key notion of characterizing s-regular HMSCs is that of the communication graph of an MSC. The communication graph captures intuitively the structure of information exchanges among processes in an MSC. Given an MSC M, its communication graph is a directed graph that has processes of M as vertices, and contains an edge from p to q if p sends a message to q somewhere in M. Given an HMSC H, we shall say that M is a cycle-MSC of H if there is a cycle in H such that M is obtained by concatenating the MSCs encountered along this cycle. H is said to be s-regular iff the communication graph of every cycle-MSC of H is strongly connected. H is said to be globally-cooperative [23] in case the communication graph of every cycle MSC of H is weakly connected.

We can define a similar notion for causal MSCs. As processes do not necessarily impose an ordering on events, it is natural to focus on the associated Mazurkiewicz alphabet. Thus the communication graph is defined w.r.t. the dependency relations $\{D_p\}_{p\in\mathcal{P}}$ used for the concatenation while the dependencies among letters of the same process are disregarded.

Definition 4 *Let $B = (E, \lambda, \{\sqsubseteq_p\}, \ll)$ be a causal MSC. The* communication graph *of B is denoted by CG_B, and is the directed graph $(Q, \rightsquigarrow)$, where $Q = \lambda(E)$ and $\rightsquigarrow \subseteq Q \times Q$ is given by: $(x, y) \in \rightsquigarrow$ iff*

- *$x = p!q(m)$ and $y = q?p(m)$ for some $p, q \in \mathcal{P}$ and $m \in Msg$, or*
- *xD_py.*

The example of figure 5-a) shows the communication graph for the causal MSC $B_1 \circledcirc B_2$ in Figure 4. For instance, there are arrows between $q?p(n)$ and $q!p(v)$ since $q?p(n)\ D_q\ q!p(v)$. However, there is no arrow between $q?p(m)$ and $q?p(n)$, even though some events of $B_1 \circledcirc B_2$ labeled by $q?p(m)$ and $q?p(n)$ are dependent. Note that for a pair of causal MSCs B, B', the communication graph $CG_{B\circledcirc B'} = (Q, \rightsquigarrow)$ can be computed from the communication graphs $CG_B = (Q_B, \rightsquigarrow_B)$ and $CG_{B'} = (Q_{B'}, \rightsquigarrow_{B'})$ as follows: $Q = Q_B \cup Q_{B'}$ and

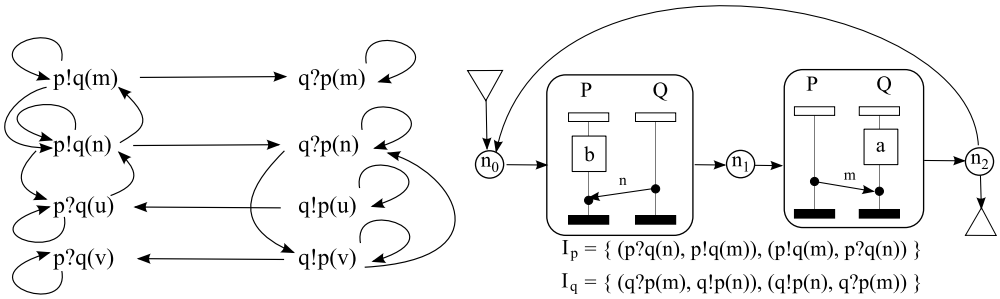

Fig. 5. a) Communication graph for causal MSC $B_1 \circledcirc B_2$ of Figure 4 b) A non finitely generated s-regular causal HMSC

$\rightsquigarrow = \rightsquigarrow_B \cup \rightsquigarrow_{B'} \cup \left(Q^2 \cap \bigcup_{p\in\mathcal{P}} D_p\right)$. Hence, for a fixed set of independence relations, if a causal MSC B is obtained by sequential composition, that is $B = B_1 \circledcirc B_2 \circledcirc \cdots \circledcirc B_k$, then the communication graph of B does not depend on the respective ordering of $B_1, \cdots B_k$, nor on the number of occurrences of each B_i. Hence, for any permutation f on $1..k$ and any $B' = B_{f(1)} \circledcirc B_{f(2)} \circledcirc \cdots \circledcirc B_{f(k)}$, we have that $CG_B = CG_{B'}$.

In the sequel, we will say that the causal MSC B is *tight* iff its communication graph CG_B is weakly connected. We say that B is *rigid* iff its communication graph is strongly connected. We will focus here on rigidity and study the notion of tightness in section 3.3.

Definition 5 *Let $H = (N, N_{in}, \mathcal{B}, N_{fi}, \longrightarrow)$ be a causal HMSC. We say that H is* s-regular *(resp.* globally-cooperative*) iff for every cycle ρ in H, the causal MSC $\circledcirc(\rho)$ is* rigid *(resp.* tight*).*

It is easy to see that the simple protocol modeled by the causal HMSC of Figure 5-b) is s-regular, since the only elementary cycle is labeled by two local events a, b, one message from p to q and one message from q to p. The communication graph associated to this cycle is strongly connected. Note that the visual language of this causal HMSC is not finitely generated, as messages m and n can cross between two occurrences of a and b.

There can be infinitely many cycles in H, hence Definition 5 does not give automatically an algorithm to check whether a causal HMSC is s-regular or globally-cooperative. However, we can use the following equivalent definition to obtain an algorithm: H is s-regular (resp. globally-cooperative) iff for every strongly connected subgraph G of H with $\{B_1, \ldots, B_\ell\}$ being the set of causal MSCs appearing in G, we have $B_1 \circledcirc \ldots \circledcirc B_\ell$ is rigid (resp. tight). As already discussed, the rigidity of $B_1 \circledcirc \ldots \circledcirc B_\ell$ does not depend on the order in which $B_1, \ldots, B_\ell$ are listed. This leads to a co-NP-complete algorithm to test whether a causal HMSC is s-regular.

Theorem 1 *Let $H = (N, N_{in}, \mathcal{B}, \longrightarrow, N_{fi})$ be a causal HMSC. Testing whether H is s-regular (respectively globally-cooperative) can be done in time $\mathcal{O}((|N|^2 +$*

$|\Sigma|^2) \cdot 2^{|\mathcal{B}|})$. *Furthermore these problems are co-NP complete.*

Proof Sketch: we reuse the ideas in the proofs of [24,14], that is guess a subset $X = \{B_1, \cdots B_k\} \subseteq \mathcal{B}$ of causal MSCs and check that the communication graph of $B_1 \circledcirc \cdots \circledcirc B_k$ is not strongly connected. From this guess, it is clear that the algorithm is co-NP. For hardness, we reuse the proof [24], that reduces staisfiability of a boolean formula in conjunctive normal form to connectedness of loops in HMSCs (there is a mapping from assignments of variables to paths in a HMSC built from the formula, and a non satisfying assignment means that the corresponding loop is connected). □

Theorem 2 *Let $H = (N, N_{in}, \mathcal{B}, N_{fi}, \longrightarrow)$ be a s-regular causal HMSC. Then $Lin(H)$ is a regular subset of $\Sigma^\star$, i.e. we can build an automaton $\mathcal{A}_H$ over Σ that recognizes $Lin(H)$. Furthermore, the number of states of $\mathcal{A}_H$ is at most in $\Big(|N|^2 \cdot 2^{|\Sigma|} \cdot (\Sigma+1)^{K \cdot M} \cdot 2^{f(K \cdot M)}\Big)^K$, where $K = |N| \cdot |\Sigma| \cdot 2^{|\mathcal{B}|}$, $M = \max\{|B| \mid B \in \mathcal{B}\}$ (recall that $|B|$ denotes the size of the causal MSC B), and the function f is given by $f(n) = \frac{1}{4}n^2 + \frac{3}{2}n + \mathcal{O}(\log_2 n)$.*

In [21], the regularity of linearization languages of s-regular HMSC was proved by using an encoding into connected traces and building a finite state automaton which recognizes such connected traces. In our case, finding such embedding into Mazurkiewicz traces seems impossible due to the fact that causal MSCs need not be FIFO (we can find two messages $e \ll f$ and $e' \ll f'$ such that $e \sqsubseteq e'$ and $f' \sqsubseteq f$). Instead, we shall use techniques from the proof of regularity of trace closures of loop-connected automata from [9,24].

The rest of this subsection is devoted to the proof of Theorem 2. We fix a s-regular causal HMSC H as in the theorem, and show the construction of the finite state automaton $\mathcal{A}_H$ over Σ which accepts $Lin(H)$. First, we establish some technical results.

Lemma 1 *Let $\rho = \theta_1 \dots \theta_2 \dots \theta_{|\Sigma|}$ be a path of H, where for each $i = 1 \dots |\Sigma|$, the subpath $\theta_i = n_{i,0} \xrightarrow{B_{i,1}} n_{i,1} \dots n_{i,\ell_i-1} \xrightarrow{B_{i,\ell_i}} n_{i,0}$ is a cycle (these cycles need not be contiguous). Suppose further that the sets $\widehat{\mathcal{B}}_i = \{B_{i,1}, \dots, B_{i,\ell_i}\}$, $i = 1, \dots, |\Sigma|$, are equal. Let e be an event in $\circledcirc(\theta_1)$ and e' an event in $\circledcirc(\theta_{|\Sigma|})$. Let $\circledcirc(\rho) = (E, \lambda, \{\sqsubseteq_p\}, \ll)$. Then we have $e \leq e'$.*

Proof Sketch: As events with same labels are necessarily ordered, and communication graphs do not depend on the order of MSCs, each iteration of a loop creates a causal dependency from an even e of this loop to all events of the next loop labelled by a successor of $\lambda(e)$ in the communication graph. □

Let $\rho = n_0 \xrightarrow{B_1} \cdots \xrightarrow{B_\ell} n_\ell$ be a path in H, where $B_i = (E_i, \lambda_i, \{\sqsubseteq_p^i\}, \ll_i)$ for $i = 1, \dots, \ell$. Let $\circledcirc(\rho) = (E, \lambda, \{\sqsubseteq_p\}, \ll, \leq)$. A *configuration* of ρ is a $\leq$-closed subset of E. Let C be a configuration of ρ. A *C-subpath* of ρ is a

maximal subpath $\varrho = n_u \xrightarrow{B_{u+1}} \ldots \xrightarrow{B_{u'}} n_{u'}$, such that $C \cap E_i \neq \emptyset$ for each $i = u, \ldots, u'$. For such a *C-subpath* ϱ, we define its *C-residue* to be the set $(E_{u+1} \cup E_{u+2} \cup \cdots \cup E_{u'}) - C$. Figure 6 illustrates these notions for a path $\rho = n_0 \xrightarrow{B_1} n_1 \xrightarrow{B_2} n_2 \xrightarrow{B_3} n_3 \xrightarrow{B_4} n_4 \xrightarrow{B_5} n_5 \xrightarrow{B_6} n_6 \xrightarrow{B_7} n_7$. Each causal MSC is represented by a rectangle. Events in the configuration C are indicated by small filled circles, events not in C but in the C-residues are indicated by small blank circles, and events that are not in C nor in its residues are indicated by blank squares. Note that the configuration contains only events from B_1, B_3, B_4 and B_5. The two *C-subpaths* identified on Figure 6 are the sequences of transitions $\rho_1 = n_0 \xrightarrow{B_1} n_1$ and $\rho_2 = n_2 \xrightarrow{B_3} n_3 \xrightarrow{B_4} n_4 \xrightarrow{B_5} n_5$ that provide the events appearing in C. One can also notice from this example that *C-subpaths* do not depend on the length of the considered path, and that the suffix of each path that does not contain an event in C can be ignored.

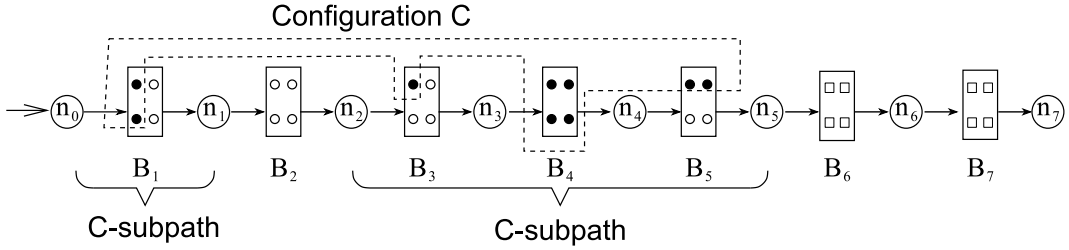

Fig. 6. An example of path, a configuration C, and its C-subpaths.

Lemma 2 *Let ρ be a path in H and C be a configuration of ρ. Then,*

(i) The number of C-subpaths of ρ is at most $K_{subpath} = |N| \cdot |\Sigma| \cdot 2^{|\mathcal{B}|}$.
(ii) Let ϱ be a C-subpath of ρ. Then the number of events in the C-residue of ϱ is at most $K_{residue} = |N| \cdot |\Sigma| \cdot 2^{|\mathcal{B}|} \cdot \max\{|B| \mid B \in \mathcal{B}\}$.

Proof Sketch: If $K_{subpath}$ is greater than the given bound, then it allows for the repetition of several subpaths ending on the same node of the HMSC, and hence for the repetition of several cycles over identical sets of MSCs. After a certain number of repetitions, it becomes impossible to build a $\leq$-closed configuration, which is required but in contradiction with lemma 1. □

We are now ready to define the finite state automaton $\mathcal{A}_H = (S, S_{in}, \Sigma, S_{fi}, \Rightarrow)$ which accepts $Lin(H)$. As usual, S will be the set of states, $S_{in} \subseteq S$ the initial states, $\Longrightarrow \subseteq S \times \Sigma \times S$ the transition relation, and $S_{fi} \subseteq S$ the final states. Fix $K_{subpath}$, $K_{residue}$ to be the constants defined in Lemma 2. If $B = (E, \lambda, \{\sqsubseteq_p\}, \ll)$ is a causal MSC and E' a subset of E, then we define the restriction of B to E' to be the causal MSC $B' = (E', \lambda', \{\sqsubseteq'_p\}, \ll')$ as follows. As expected, λ' is the restriction of λ to E'; for each p, $\sqsubseteq'_p = \sqsubseteq_p \cap E' \times E'$ and $\ll' = \ll \cap E' \times E'$. Note that we are slightly abusing the definition of causal MSCs, as in restrictions, messages can be incomplete ($e \in E'$ and $f \ll eore \ll f$ does not imply $f \in E'$). However, as the restrictions of causal MSCs will be performed on subsets E' that are closed by $\leq$ ($e \in E'$ and $e \leq f$ implies $f \in E'$), restrictions will only contain unmatched receptions, that can

be considered as atomic actions. Hence, the concatenation defined for causal MSCs also hold for their restrictions in what follows.

Intuitively, for a word σ in $\Sigma^\star$, $\mathcal{A}_H$ guesses an accepting path ρ of H and checks whether σ is in $Lin(\circledcirc(\rho))$. After reading a prefix σ' of σ, $\mathcal{A}_H$ memorizes a sequence of subpaths from which σ' was "linearized" (i.e. the C-subpath of a path ρ such that C is a configuration reached after reading σ' and $\circledcirc(\rho)$ contains C). With Lemma 2, it will become clear later that at any time, we should remember at most $K_{subpath}$ such subpaths. Moreover, for each subpath, we need to know only a *bounded* amount of information, which will be stored in a data structure called "segment".

A causal MSC $B = (E, \lambda, \{\sqsubseteq_p\}, \ll)$ is of size lower than K if $|E| \leq K$. A *segment* is a tuple (n, Γ, W, n'), where $n, n' \in N$, Γ is a nonempty subset of Σ, and W is either a non-empty causal MSC of size lower than $K_{residue}$, or the special symbol $\perp$. The state set S of $\mathcal{A}_H$ is the collection of finite sequences $\theta_1\theta_2 \ldots \theta_\ell$, $0 \leq \ell \leq K_{subpath}$, where each θ_i is a segment. Intuitively, a segment (n, Γ, W, n') keeps track of a subpath ϱ of H which starts at n and ends at n'. Γ is the collection of letters of events in $\circledcirc(\varrho)$ that have been "linearized". Finally, W is the restriction of $\circledcirc(\varrho)$ to the set of events in $\circledcirc(\varrho)$ that are not yet linearized. In case all events in $\circledcirc(\varrho)$ have been linearized, we set $W = \perp$. For convenience, we extend the operator $\circledcirc$ by: $W \circledcirc \perp = \perp \circledcirc W = W$ for any causal MSC W; and $\perp \circledcirc \perp = \perp$.

We define $\mathcal{A}_H = (S, S_{in}, \Sigma, S_{fi}, \Longrightarrow)$ as follows:

- As mentioned above, S is the collection of finite sequence of at most $K_{subpath}$ segments.
- The initial state set is $S_{in} = \{\varepsilon\}$, where ε is the null sequence.
- A state is final iff it consists of a single segment $\theta = (n, \Gamma, \perp, n')$ such that $n \in N_{in}$ and $n' \in N_{fi}$ (and Γ is any nonempty subset of Σ).
- The transition relation $\Longrightarrow$ of $\mathcal{A}_H$ is the least set satisfying the following conditions.
  —**Condition** (i):
  Suppose $n \xrightarrow{B} n'$ where $B = (E, \lambda, \{\sqsubseteq_p\}, \ll, \leq)$. Let e be a minimal event in B (with respect to $\leq$) and let $a = \lambda(e)$. Let $\theta = (n, \Gamma, W, n')$ where $\Gamma = \{a\}$. Let $R = E - \{e\}$. If R is nonempty, then W is the restriction of B to R; otherwise we set $W = \perp$. Suppose $s = \theta_1 \ldots \theta_k\theta_{k+1} \ldots \theta_\ell$ is a state in S where $\theta_i = (n_i, \Gamma_i, W_i, n_i')$ for each i. Suppose further that, e is a minimal event in $W_1 \circledcirc W_2 \circledcirc \ldots \circledcirc W_k \circledcirc W$.
  · ("create a new segment") Let $\hat{s} = \theta_1 \ldots \theta_k\theta\theta_{k+1} \ldots \theta_\ell$. If $\hat{s}$ is in S, then $s \overset{a}{\Longrightarrow} \hat{s}$. In particular, for the initial state ε, we have $\varepsilon \overset{a}{\Longrightarrow} \theta$.
  · ("add to the beginning of a segment") Suppose $n' = n_{k+1}$. Let $\hat{\theta} = (n, \Gamma \cup \Gamma_{k+1}, \widehat{W}, n'_{k+1})$, where $\widehat{W} = W \circledcirc W_{k+1}$. Let $\hat{s} = \theta_1 \ldots \theta_k\hat{\theta}\theta_{k+2} \ldots \theta_\ell$. If $\hat{s}$ is in S, then $s \overset{a}{\Longrightarrow} \hat{s}$.

· ("append to the end of a segment") Suppose $n = n'_k$. Let $\hat{\theta} = (n_k, \Gamma_k \cup \Gamma, \widehat{W}, n')$, where $\widehat{W} = W_k \circledcirc W$. Let $\hat{s} = \theta_1 \ldots \theta_{k-1}\hat{\theta}\theta_{k+1} \ldots \theta_\ell$. If $\hat{s}$ is in S, then $s \stackrel{a}{\Longrightarrow} \hat{s}$.

· ("glue two segments") Suppose $n = n'_k$ and $n' = n_{k+1}$. Let $\hat{\theta} = (n_k, \Gamma_k \cup \Gamma \cup \Gamma_{k+1}, \widehat{W}, n'_{k+1})$, where $\widehat{W} = W_k \circledcirc W \circledcirc W_{k+1}$. Let $\hat{s}$ be $\theta_1 \ldots \theta_{k-1}\hat{\theta}\theta_{k+2} \ldots \theta_\ell$. If $\hat{s}$ is in S, then $s \stackrel{a}{\Longrightarrow} \hat{s}$.

—**Condition** (ii):

Suppose $s = \theta_1 \ldots \theta_k\theta_{k+1} \ldots \theta_\ell$ is a state in S where $\theta_i = (n_i, \Gamma_i, W_i, n'_i)$ for $i = 1, 2, \ldots, \ell$. Suppose $W_k \neq \perp$. Let $W_k = (R_k, \eta_k, \{\sqsubseteq_p^k\}, \ll_k, \leq_k)$ and e a minimal event in W_k. Suppose further that e is a minimal event in $W_1 \circledcirc W_2 \circledcirc \ldots \circledcirc W_k$.

Let $\hat{\theta} = (n_k, \Gamma_k \cup \{a\}, \widehat{W}, n'_k)$, where $\widehat{W}$ is defined as follows. Let $\widehat{R} = R_k - \{e\}$. If $\widehat{R}$ is nonempty, then $\widehat{W}$ is the restriction of W to $\widehat{R}$; otherwise, set $\widehat{W} = \perp$. Let $\hat{s} = \theta_1 \ldots \theta_{k-1}\hat{\theta}\theta_{k+1} \ldots \theta_\ell$. Then we have $s \stackrel{a}{\Longrightarrow} \hat{s}$, where $a = \eta_k(e)$. (Note that $\hat{s}$ is guaranteed to be in S.)

We have now completed the construction of $\mathcal{A}_H$. It remains to show that $\mathcal{A}_H$ recognizes $Lin(H)$.

Lemma 3 *Let $\sigma \in \Sigma^\star$. Then σ is accepted by $\mathcal{A}_H$ iff σ is in $Lin(H)$.*

Proof: Let $\sigma = a_1a_2 \ldots a_k$. Suppose σ is in $Lin(H)$. Let $\rho = n_0 \xrightarrow{B_1} \ldots \xrightarrow{B_\ell} n_\ell$ be an accepting path in H such that σ is a linearization of $\circledcirc(\rho)$. Hence we may suppose that $\circledcirc(\rho) = (E, \lambda, \{\sqsubseteq_p\}, \ll, \leq)$ where $E = \{e_1, e_2, \ldots, e_k\}$ and $\lambda(e_i) = a_i$ for $i = 1, \ldots, k$. And $e_i \leq e_j$ implies $i \leq j$ for any i, j in $\{1, \ldots, k\}$. Consider the configurations $C_i = \{e_1, e_2, \ldots, e_i\}$ for $i = 1, \ldots, k$. For each C_i, we can associate a state s_i in $\mathcal{A}_H$ as follows. Consider a fixed C_i. Let $\rho = \ldots \varrho_1 \ldots \varrho_2 \ldots \varrho_h \ldots$ where ϱ_1, ϱ_2, …, ϱ_h are the C_i-subpaths of ρ. Then we set $s_i = \theta_1 \ldots \theta_h$ where $\theta_j = (n_j, \Gamma_j, W_j, n'_j)$ with n_j being the starting node of ϱ_j, and Γ_j the collection of all $\lambda(e)$ for all events e that are in both $\circledcirc(\varrho_j)$ and C_i. Let R_j be the C_i-residue of ϱ_j. If R_j is nonempty, W_j is the causal MSC $(R_j, \eta_j, \{\sqsubseteq_p^j\}, \ll_j, \leq_j)$ where η_j is the restriction of λ to R_j; $\sqsubseteq_p^j$ is the restriction of $\sqsubseteq_p$ to those events in R_j that belong to process p, for each p; and $\ll_j$ the restriction of $\ll$ to R_j. If R_j is empty, then set $W_j = \perp$. Finally, n'_j is the ending node of ϱ_j.

Now it is routine (though tedious) to verify that $\varepsilon \stackrel{a_1}{\Longrightarrow} s_1 \ldots s_{k-1} \stackrel{a_k}{\Longrightarrow} s_k$ is an accepting run of $\mathcal{A}_H$. Conversely, given an accepting run of $\mathcal{A}_H$ over σ, it is straightforward to build a corresponding accepting path of H. □

With Lemma 3, we establish Theorem 2. As for complexity, the number of states in $\mathcal{A}_H$ is at most $(N_{seg})^{K_{subpath}}$, where N_{seg} is the maximal number of segments. Now, N_{seg} is $|N|^2 \cdot 2^{|\Sigma|} \cdot N_{res}$, where N_{res} is the possible number of residues. Recall that a residue is of size at most $K_{residue}$. According to Kleitman & Rotschild [20], the number of partial orders of size k is in $2^{f(k)}$ where $f(k) = \frac{1}{4}k^2 + \frac{3}{2}k + \mathcal{O}(\log_2(k))$. It follows that the number of $|\Sigma|$-labeled

posets of size k is in $2^{f(k)} \cdot |\Sigma|^k$. All residues of size up to k can be encoded as a labeled poset of size k with useless events, labelled by a specific label $\sharp$. Hence the number of residues N_{res} is lower than $2^{f(K_{residue})} \cdot (|\Sigma| + 1)^{K_{residue}}$. Combining the above calculations then establishes the bound in theorem 2 on the number of states of $\mathcal{A}_H$.

3.3 Inclusion and Intersection of causal HMSC Languages

It is known that problems of inclusion, equality and non-emptiness of the intersection of the MSC languages associated with HMSCs are all undecidable [24]. Clearly, these undecidability results also apply to the causal languages of causal HMSCs. As in the case of HMSCs, theses problems are decidable for s-regular causal HMSCs since their linearization languages are regular.

It is natural to ask whether we can still obtain positive results for these problems beyond the subclass of s-regular causal HMSCs. In the setting of HMSCs, one can show that for all $K \geq 0$, the set of K-bounded linearizations of any globally-cooperative HMSC is regular. Moreover, for a suitable choice of K, the set of K-bounded linearizations is sufficient for effective verification [13]. Unfortunately, this result uses Kuske's encoding [21] into traces that is based on the existence of an (existential) bound on communication channels. Consequently, this technique does not apply to globally-cooperative causal HMSCs, as the visual language of a causal HMSC needs not be existentially bounded. For instance, consider the causal HMSC H of Figure 7. It is globally-cooperative and its visual language contains the MSCs of the form shown in the right part of Figure 7, which contain an arbitrary number of messages from p to r, that have to be sent before a message m from p to q is sent. In order to receive the first message from p to r, the message from p to q and the message from q to r have to be sent and received in every MSC of $Vis(H)$. Hence every message from p to r has to be sent before receiving the first message from p to r, in every MSC of $Vis(H)$, which means that there H is not existentially bounded.

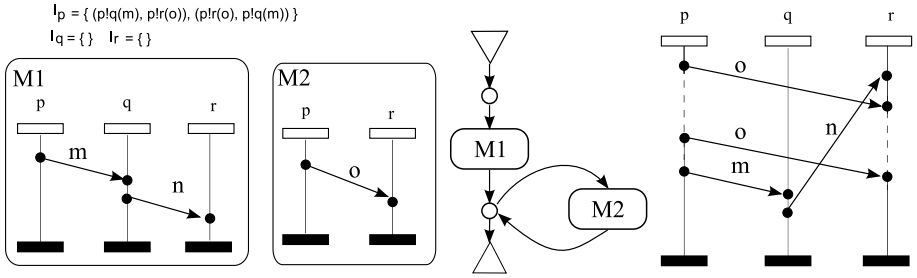

Fig. 7. A globally-cooperative causal HMSC that is not existentially bounded

We shall instead adapt the notion of atoms [1,17] and the techniques from [14].

Definition 6 *A causal MSC B is a* basic part *(w.r.t. the trace alphabets $\{(\Sigma_p, I_p)\}_{p\in\mathcal{P}}$) if there do not exist causal MSCs B_1, B_2 such that $B = B_1 \circledcirc B_2$.*

Note that we require that the set of events of a causal MSC is not empty.

Now for a causal MSC B, we define a *decomposition* of B to be a sequence $B_1 \cdots B_\ell$ of basic parts such that $B = B_1 \circledcirc \cdots \circledcirc B_\ell$. For a set $\mathcal{B}$ of basic parts, we associate a trace alphabet $(\mathcal{B}, I_\mathcal{B})$ (w.r.t. the trace alphabets $\{(\Sigma_p, I_p)\}_{p\in\mathcal{P}}$) where $I_\mathcal{B}$ is given by: $B\ I_\mathcal{B}\ B'$ iff for every p, for every $\alpha \in Alph_p(B)$, for every $\alpha' \in Alph_p(B')$, it is the case that $\alpha\ I_p\ \alpha'$. We let $\sim_\mathcal{B}$ be the corresponding trace equivalence relation and denote the trace containing a sequence $u = B_1.\ldots.B_\ell$ in $\mathcal{B}^\star$ by $[u]_\mathcal{B}$ (or simply $[u]$). For a language $L \subseteq \mathcal{B}^\star$, we define its trace closure $[L]_\mathcal{B} = \bigcup_{u\in L} [u]_\mathcal{B}$. We begin by proving that the decomposition of any causal MSC B is unique up to commutation. More precisely,

Proposition 2 *Let $B_1 \ldots B_k$ be a decomposition of a causal MSC B. Then the set of decompositions of B is $[B_1 \ldots B_k]$.*

Proof: It is clear that every word in $[B_1 \ldots B_k]$ is a decomposition of B. For the converse, let us suppose that there exists a decomposition $B = W_1 \circledcirc W_2 \circledcirc \cdots \circledcirc W_q$ such that $W_1 \ldots W_q \notin [B_1 \ldots B_k]$. It means that there exists a permutation ϕ and an index i with $W_j = B_{\phi(j)}$ for all $j < i$, $B_{\phi(1)} \cdots B_{\phi(k)} \in [B_1 \ldots B_k]$, and $W' = W_i \cap B_{\phi(i)} \neq \emptyset$ and $W'' = W_i \setminus B_{\phi(i)} \neq \emptyset$. It suffices now to prove that W' and W'' are causal MSCs and that $W_i = W' \circledcirc W''$ to get a contradiction with the fact that W_i is a basic part. By definition, the restriction of $\ll_{W_i}$ to W' is still a bijection (a send in W' matches a receive in W'). It implies that the restriction of $\ll_{W_i}$ to W'' is a bijection too. Both W' and W'' are thus causal MSCs. The fact that $W_i = W' \circledcirc W''$ comes from the definition of $\circledcirc$ and from the fact that $W' \subseteq B_{\phi(i)}$ and $W'' \subseteq B_{\phi(i+1)} \cdots B_{\phi(k)}$. □

It is thus easy to compute the (finite) set of basic parts of a causal MSC B, denoted $Basic(B)$, since it suffices to find one of its decompositions.

Proposition 3 *For a given causal MSC B, we can effectively decompose B in time $O(|B|^2)$.*

Proof Sketch: We can reuse the quadratic techniques of [17]. □

In view of Proposition 3, we assume through the rest of this section that every transition of a causal HMSC H is labelled by a basic part. This obviously incurs no loss of generality, since we can simply decompose each causal MSC in H into basic parts and decompose any transition of H into a sequence of transitions labeled by these basic parts. Given a causal HMSC H, we let $Basic(H)$ be the set of basic parts labelling transitions of H. Trivially, a causal MSC is uniquely defined by its basic part decomposition. Then instead of the visual language we can use the *basic part language* of H, denoted by $BP(H) = \{B_1 \ldots B_\ell \in Basic(H)^\star \mid B_1 \circledcirc \ldots \circledcirc B_\ell \in cMSC(H)\}$. Notice that $BP(H) = [BP(H)]$ by Proposition 2, that is, $BP(H)$ is closed by commutation. We can also view H as a finite state automaton over the alphabet $Basic(H)$, and denote by $\mathcal{L}_{Basic}(H) = \{B_1 \cdots B_\ell \in Basic(H)^\star \mid n_0 \xrightarrow{B_1} n_1 \cdots \xrightarrow{B_\ell} n_\ell$ is an accepting path of H$\}$ its associated (regular) language. We

now relate $BP(H)$ and $\mathcal{L}_{Basic}(H)$.

Proposition 4 *Let H be a causal HMSC. Then $BP(H) = [\mathcal{L}_{Basic}(H)]$.*

Proof: First, let us take a word w in $[\mathcal{L}_{Basic}(H)]$. Thus $w = B_1 \dots B_k \sim B_{i_1} \dots B_{i_k}$ such that $B_{i_1} \dots B_{i_k} \in \mathcal{L}_{Basic}(H)$. As $\mathcal{L}_{Basic}(H) \subseteq BP(H)$ and $B_1 \circledcirc \dots \circledcirc B_k = B_{i_1} \circledcirc \dots \circledcirc B_{i_k}$ we conclude that $[\mathcal{L}_{Basic}(H)] \subseteq BP(H)$. Second, let us take a word w in $BP(H)$. Let us note $\circledcirc(w)$ its corresponding causal MSC, i.e. for $w = B_1 \dots B_k$, $\circledcirc(w) = B_1 \circledcirc \dots \circledcirc B_k$. Then this word is generated by an accepting path $\rho = n_0 \xrightarrow{P_1} n_1 \dots \xrightarrow{P_l} n_l$ of H such that $\circledcirc(w) = P_1 \circledcirc \dots \circledcirc P_l$. By proposition 2, we know that any other decomposition of $\circledcirc(w)$ belongs to $[B_1 \dots B_k]$, and in particular, the one we choose. Thus we obtain that $BP(H) \subseteq [\mathcal{L}_{Basic}(H)]$. □

Assuming we know how to compute the trace closure of the regular language $\mathcal{L}_{Basic}(H)$, we can obtain $BP(H)$ with the help of Proposition 4. In general, we cannot effectively compute this language. However if H is globally-cooperative, then $[\mathcal{L}_{Basic}(H)]$ is regular and a finite state automaton recognizing $[\mathcal{L}_{Basic}(H)]$ can be effectively constructed [9,24]. Considering globally-cooperative causal HMSCs as finite state automata over basic parts, we can apply [24] to obtain the following decidability and complexity results:

Theorem 3 *Let H, H' be causal HMSCs over the* same *family of trace alphabets $\{(\Sigma_p, I_p)\}_{p \in \mathcal{P}}$. Suppose H' is globally-cooperative. Then we can build a finite state automaton $\mathcal{A}'$ over $Basic(H')$ such that $\mathcal{L}_{Basic}(A') = [\mathcal{L}_{Basic}(H')]$. Moreover, $\mathcal{A}'$ has at most $2^{O(n \cdot b)}$ states, where n is the number of nodes in H and b is the number of basic parts in $Basic(H)$. Consequently, the following problems are decidable:*

(i) Is $cMSC(H) \subseteq cMSC(H')$?
(ii) Is $cMSC(H) \cap cMSC(H') = \emptyset$?

Furthermore, the complexity of (i) is PSPACE-complete and that of (ii) is EXPSPACE-complete.

The above theorem shows that we can model-check a causal HMSC against a globally-cooperative causal HMSC specification. Note that we can only apply Theorem 3 to two causal HMSCs over the *same* family of trace alphabets. If the causal HMSCs H, H' in theorem 3 satisfy the additional condition that every causal MSC labeling the transitions of H and H' respects $\{(\Sigma_p, I_p)\}_{p \in \mathcal{P}}$, then we can compare the visual languages $Vis(H)$ and $Vis(H')$, thanks to Proposition 1.

On the other hand, if the independence relations are different, the atoms of H and H' are unrelated. Theorem 4 proves that comparing the MSC languages of two globally-cooperative causal HMSCs H, H' using different independence relations is actually undecidable. The only way to compare them is then to compare their linearization languages. Consequently, we would need to work

with s-regular causal HMSCs.

Theorem 4 *Let G, H be globally-cooperative causal HMSCs with respectively families of trace alphabets $\{(\Sigma_p, I_p)\}_{p\in\mathcal{P}}$ and $\{(\Sigma_p, J_p)\}_{p\in\mathcal{P}}$, where for each p, I_p and J_p are allowed to differ. Then it is undecidable to determine whether $Vis(G) \cap Vis(H) = \emptyset$.*

Proof Sketch: We can encode a PCP with two HMSCs $H1$ and $H2$, defined over 3 processes P1,P2,P3. $H1$ represents the words (v_i, w_i) in the PCP with causal HMSCs, with loops of the form $(Vi.Wi)$, and lets everything commute on $P2, P3$. $H2$ forces all receptions of messages from V_i's to be followed by their counterpart in W_i's, and does not allow commutations in $V \times V$ nor in $W \times W$. □

4 Window-bounded causal HMSCs

One of the chief attractions of causal MSCs is that they enable the specification of behaviors containing braids of arbitrary size such as those generated by sliding windows protocols. Very often, sliding windows protocols appear in a situation where two processes p and q exchange bidirectional data. Messages from p to q are of course used to transfer information, but also to acknowledge messages from q to p. If we abstract the type of messages exchanged, these protocols can be seen as a series of query messages from p to q and answer messages from q to p. Implementing a sliding window means that a process may send several queries in advance without needing to wait for an answer to each query before sending the next query. Very often, these mechanisms tolerate losses, i.e. the information sent is stored locally, and can be retransmitted if needed (as in the alternating bit protocol). To avoid memory leaks, the number of messages that can be sent in advance is often bounded by some integer k, that is called the size of the sliding window. Note however that for scenario languages defined using causal HMSCs, such window sizes do not always exist. This is the case for example for the causal HMSC depicted in Figure 1 with independence relations $I_p = \{((p!q(Q), p?q(A)), (p?q(A), p!q(Q)))\}$ and $I_q = \{((q?p(Q), q!p(A)), (q!p(A), q?p(Q))\}$. The language generated by this causal HMSC contains scenarios where an arbitrary number of messages from p to q can cross an arbitrary number of messages from q to p. A question that naturally arises is to know if the number of messages crossings is bounded by some constant in all the executions of a protocol specified by a causal HMSC. In what follows, we define these crossings, and show that their boundedness is a decidable problem.

Definition 7 *Let $M = (E, \lambda, \{\sqsubseteq_p\}, \ll)$ be an MSC. For a message (e, f) in M, that is, $(e, f) \in \ll$, we define the* window *of (e, f), denoted $W_M(e, f)$, as the set of messages $\{e' \ll f' \mid loc(\lambda(e')) = loc(\lambda(f))$ and $loc(\lambda(f')) = loc(\lambda(e))$ and $e \leq f'$ and $e' \leq f\}$.*

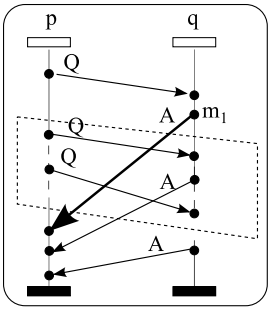

Fig. 8. Window of message m_1

We say that a causal HMSC H is *K-window-bounded* iff for every $M \in Vis(H)$ and for every message (e, f) of M, it is the case that $|W_M(e, f)| \leq K$. H is said to be window-bounded iff H is K-window-bounded for some K.

Figure 8 illustrates notion of window, where the window of the message m_1 (the first answer from q to p) is symbolized by the area delimited by dotted lines. It consists of all but the first message Q from p to q. Clearly, the causal HMSC H of Figure 1 is not window-bounded. We now describe an algorithm to effectively check whether a causal HMSC is window bounded. It builds a finite state automaton whose states remember the labels of events that must appear in the *future* of messages (respectively in the *past*) in any MSC of $Vis(H)$.

Formally, for a causal MSC $B = (E, \lambda, \{\sqsubseteq_p\}, \ll)$ and $(e, f) \in \ll$ a message of B, we define the future and past of (e, f) in B as follows:

$$Future_B(e, f) = \{a \in \Sigma \mid \exists x \in E, f \leq x \wedge \lambda(x) = a\}$$

$$Past_B(e, f) = \{a \in \Sigma \mid \exists x \in E, x \leq e \wedge \lambda(x) = a\}$$

Notice that for a message $m = (e, f)$, we always have $\lambda(e) \in Past_B(m)$ and $\lambda(f) \in Future_B(m)$. For instance, in Figure 8, $Past_B(m_1) = \{p!q(Q), q?p(Q), q!p(A)\}$. Intuitively, if a letter of the form $p!q(Q)$ is in $Future_B(m)$ of a message (e, f) in a causal MSC B, then any message Q from p to q that appears in a MSC B' that is appended to B *is* in the causal future of (e, f). Hence, it can not appear in the window of (e, f). Note that a symmetric property holds for the past of (e, f). Then from the definitions, we obviously obtain the following proposition.

Proposition 5 *Let $B = (E, \lambda, \{\sqsubseteq_p\}, \ll)$ and $B' = (E', \lambda', \{\sqsubseteq'_p\}, \ll')$ be two causal MSCs, and let $m \in \ll$ be a message of B. Then,*

$$-Future_{B \circledcirc B'}(m) = Future_B(m) \cup \{a' \in \Sigma \mid \exists x, y \in E'$$
$$\exists a \in Future_B(m) \text{ s.t. } \lambda(y) = a' \wedge x \leq' y \wedge a \; D_{loc(a)} \; \lambda(x)\}$$

$$-Past_{B' \circledcirc B}(m) = Past_B(m) \cup \{a' \in \Sigma \mid \exists x, y \in E'$$
$$\exists a \in Past_B(m) \text{ s.t. } \lambda(y) = a' \wedge y \leq' x \wedge a \; D_{loc(a)} \; \lambda(x)\}$$

Let B be a causal MSC, and let m be a message of B. Clearly, if a message n appears both in its past and future sets of m ($p!q(n) \in PastB(m)$ and $p!q(n) \in FutureB(m)$), then the window of message m contains a bounded number of occurrences of messages n in any concatenation $B' \circledcirc B \circledcirc B''$. Let $H = (N, N_{in}, \mathcal{B}, \longrightarrow, N_{fi})$ be a causal HMSC. Consider a path ρ of H with $\circledcirc(\rho) = B_1 \circledcirc \cdots \circledcirc B_\ell$ and a message m in B_1. First, the sequence of sets $Future_{B_1}(m)$, $Future_{B_1 \circledcirc B_2}(m)$, ..., $Future_{B_1 \circledcirc \cdots \circledcirc B_\ell}(m)$ is non-decreasing. Using proposition 5, these sets can be computed on the fly and with a finite number of states. Similar arguments hold for the past sets. Now consider a message (e, f) in a causal MSC B labelling some transition t of H. With the above observation on *Future* and *Past*, we can show that, if there is a bound $K_{(e,f)}$ such that the window of a message (e, f) in the causal MSC generated by any path containing t is bounded by $K_{(e,f)}$, then $K_{(e,f)}$ is at most $b|N||\Sigma|$ where $b = \max\{|B| \mid B \in \mathcal{B}\}$. Further, we can effectively determine whether such a bound $K_{(e,f)}$ exists by constructing a finite state transition system whose states memorize the future and past of (e, f). Thus we have the following:

Theorem 5 *Let $H = (N, N_{in}, \mathcal{B}, \longrightarrow, N_{fi})$ be a causal HMSC. Then we have:*

(i) If H is window-bounded, then H is K-window-bounded, where K is at most $b \cdot |N| \cdot |\Sigma|$.

(ii) Further, we can effectively determine whether H is window-bounded in time $O(s \cdot |N|^2 \cdot 2^{|\Sigma|})$, where s is the sum of the sizes of causal MSCs in $\mathcal{B}$.

Proof Sketch: We build two automata that behave as the original HMSCs, then chose nondeterministically a message m, and then memorize the Future or Past of the message. Future and Past are increasing, and bounded. Hence these automata are finite. Then, if one of these automata contains a cycle that allows for the repetition of a message m' that is not in the Future (resp. past) of m, then the window of m is not bounded. □

5 Relationship with Other Scenario Models

We compare here the expressive power of causal HMSCs with two other well-studied scenario languages, namely HMSCs and compositional HMSCs. For comparison, we will only consider weak-FIFO scenario languages, that is HMSCs that are labelled by weak FIFO MSCs and causal HMSCs that are labelled by causal MSCs which visual extensions are weak-FIFO. Indeed, non weak-FIFO scenarios can be seen as a little degenerate descriptions, as they can be differentiated by their visual languages, but not by their linearization languages. Hence, within this weak-FIFO setting, the comparisons established in this section holds both for visual languages and linearization languages.

The first topic of comparison is causal HMSCs themselves. A previous definition [12] of s-regular and globally-cooperative causal HMSCs required that for every $p \in \mathcal{P}$, and for every B labeling a cycle of a causal HMSC, $Alph_p(B)$ was D_p-connected (i.e. the undirected graph $(Alph_p(B), D_p)$ is connected). It is not necessary in the definition of this paper: two letters from the same pro-

cess can be connected through communication via another process, and not directly on the same process.

An important question is the class of Communicating Finite State Machine (CFM for short) corresponding to HMSC languages. It has been shown in [16] that regular (compositional) HMSCs corresponds to universally bounded CFMs. The natural model to compare causal HMSCs and CFMs could be asynchronous cellular automata with type [4], also called mixed machine in [11], which allows communication both through Mazurkiewicz traces and messages. It has been shown in [11] that using the same s-regular definition as in this paper, universally bounded mixed model and s-regular causal (compositional) HMSCs coincide. It is easy to see that this is not the case with the old definition of [12]. Consider the following example: a causal HMSC composed of a single loop labelled by a causal MSC M_1 that contains four unordered messages: m, o from process p to process q, and n, g from process q to process p. The dependence relation D_p is defined as the reflexive and symmetric closure of $\{(p!q(m), p?q(g)); (p?q(n), p!q(o))\}$ and $D_q = \Sigma_q \times \Sigma_q$. This example and the communication graph associated to its single loop are represented in Figure 9. Clearly, this causal HMSC is not s-regular nor even globally-cooperative with the definition of [12] (Σ_p is not D_p-connected), but it is s-regular with the definition of this paper. Also, there is no globally-cooperative causal HMSC in the terms of [12] recognizing the same language.

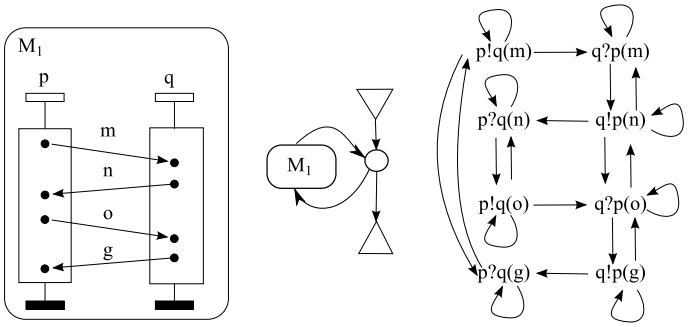

Fig. 9. A s-regular causal HMSC with the communication graph of its cycle

Then, we consider HMSCs. The standardized language allows for partial odering along instance line, by using coregions (a subset of events located on the same process) and general ordering (that defines more or less a partial order) on events located in these coregions. However, outside coregion, events are supposed totally ordered. Hence, coregions relax this total ordering only on a finite set of events contained in a single MSC, while the commutation mechanism provided by causal HMSCs allows for concurrency in sets of events of arbitrary size. Two important strict HMSC subclasses are (i) *s-regular* (also called regular in [24] and bounded in [3]) HMSCs which ensure that the linearizations form a regular set and (ii) *globally-cooperative* HMSCs [14], which ensure that for a suitable choice of K, the set of K-bounded linearizations form a regular representative set. By definition, causal HMSCs, s-regular causal HMSCs and globally-cooperative causal HMSCs extend respectively HMSCs, s-regular

HMSCs and globally-cooperative HMSCs.

Figure 7 shows a globally-cooperative causal HMSC which is not in the subclass of s-regular causal HMSCs. Thus, s-regular causal HMSCs form a strict subclass of globally-cooperative causal HMSCs. Trivially, globally-cooperative causal HMSCs are a strict subclass of causal HMSCs. Figure 5-b) displays a s-regular causal HMSC whose visual language is not finitely generated. It follows that (s-regular/globally-cooperative) causal HMSCs are strictly more expressive than (s-regular/globally-cooperative) HMSCs.

Another extension of HMSCs is *compositional* HMSCs (or CHMSCs for short), first introduced by [15]. CHMSCs generalize HMSCs by allowing dangling message-sending and message-reception events, i.e. the message pairing relation $\ll$ in a compositional MSC is only a partial non-surjective mapping contained in $E_! \times E_?$. The concatenation of two compositional MSCs $M \circ M'$ performs the instance-wise concatenation as for MSCs, and computes a new message pairing relation $\ll''$ defined over $(E_! \cup E'_!) \times (E_? \cup E'_?)$ extending $\ll \cup \ll'$, and preserving the FIFO ordering of messages with similar content (actually, in the definition of [15], there is no channel content). We refer here to the definition of [8], where compositional HMSCs generate weak-FIFO MSCs. Note that so far, compositional HMSCs do not allow for non-weak-FIFO description, but that a small variant of the language could easily be defined to allow this kind of description (as long as non FIFOness remains inside a node of the CHMSC). Note also that dangling messages were also proposed in [27,19], but that thay are interpreted as atomic actions.

A CHMSC H generates a set of MSCs, denoted $Vis(H)$ by abuse of notation, obtained by concatenation of compositional MSCs along a path of the graph. With this definition, some path of a CHMSC may not generate any correct MSC. Moreover, a path of a CHMSC generates at most one MSC. The class of CHMSC for which each path generates exactly one MSC is called *safe* CHMSC [15,13], still a strict extension over HMSCs. S-regular and globally-cooperative HMSCs have also their strict extensions in terms of safe CHMSCs, namely as s-regular CHMSC and globally-cooperative CHMSCs.

Let us now compare causal HMSCs and CHMSCs. It is not hard to build a regular compositional HMSC which MSC language can not be defined with a causal HMSC. An example is a CHMSC H that generates the visual language $Vis(H) = \{M_i \mid i = 0, 1, \ldots\}$, where each M_i consists of an emission of a message m from p to r, then a sequence of i blocks of three messages: a message n from p to q followed by a message o from q to r then a message s from r to p. And at last the reception of message m on r. This MSC language is represented in Figure 10-a). This visual language can be easily defined with a CHMSC, by separating emission and reception of m and iterating a MSC containing messages n, o, s an arbitrary number of times. Clearly, this $Vis(H)$

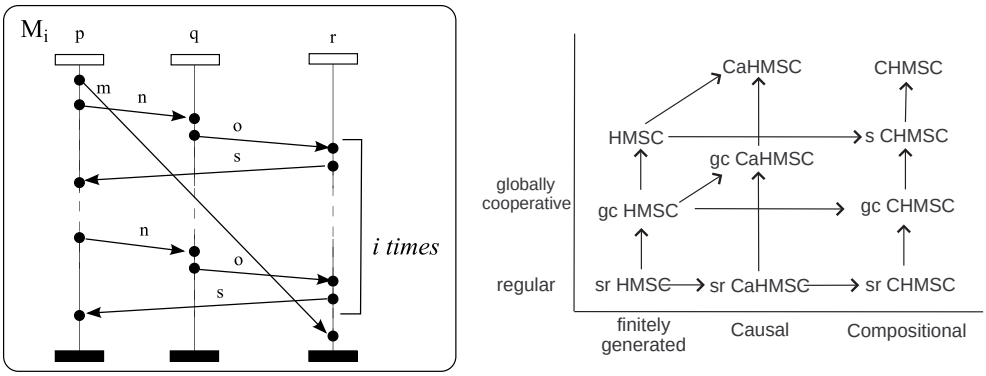

Fig. 10. a) A regular (but not finitely generated) set of MSCs b) Comparison of Scenario languages

is not finitely generated, and it is not either the visual language of a causal HMSC. Assume for contradiction, that there exists a causal HMSC G with $Vis(G) = Vis(H)$. Let k be the number of messages of the biggest causal MSC which labels a transition of G. We know that M_{k+1} is in $Vis(G)$, hence $M_{k+1} \in Vis(\circledcirc(\rho))$ for some accepting path ρ of G. Let $N_1, \ldots, N_\ell$ be causal MSCs along ρ, where $\ell \geq 2$ because of the size k. It also means that there exist $N'_1 \in Vis(N_1)$, ..., $N'_\ell \in Vis(N_\ell)$ such that $N'_1 \circ \cdots \circ N'_m \in Vis(G)$. Thus, $N_1 \circ \cdots \circ N_\ell = M_j$ for some j, a contradiction since M_j is a basic part (i.e. cannot be the concatenation of two MSCs). That is (s-regular) compositional HMSCs are not included into causal HMSCs. On the other hand, s-regular causal HMSCs have a regular set of linearizations (Theorem 2). Also by the results in [16], it is immediate that the class of visual languages of s-regular compositional HMSCs captures all the MSC languages that have a regular set of linearizations. Hence the class of s-regular causal HMSCs is included into the class of s-regular compositional HMSCs. Last, we already know with Figure 7 that globally-cooperative causal HMSCs are not necessarily existentially bounded, hence they are not included into safe compositional HMSC.

The relationships among these scenario models are summarized by Figure 10-b), where arrows denote *strict* inclusion of visual languages. Two classes are incomparable if they are not connected by a transitive sequence of arrows. We use the abbreviation *sr* for s-regular, *gc* for globally-cooperative, *s* for safe, *CaHMSC* for causal HMSCs and *CHMSC* for compositional HMSCs.

In this paper, we mainly considered scenario languages and compared them with respect to their expressive power. There are numerous specification languages, ranging from finite state automata, Petri nets, process algebra, to communicating finite state machines. Causal MSCs and all these models are uncomparable in general. For instance, one can not implement any bounded protocol (hence described with an automaton) using causal HMSCs. We already know that the linearization languages of some (causal) HMSCs can not be expressed with finite automata nor Petri Nets. However, one can always find a Petri Net which language contains all linearizations of a HMSC [6]. This

question remains open for causal HMSCs. Similarly, [13] has proved that communicating finite state machines and globally cooperative compositional MSCs have the same expressive power. Similar question is open for causal HMSCs, but as already mentionned, asynchronous cellular automata with type [4] seem to be an adequate implementaion model. From a more practical point of view, causal HMSCs satisfy an essential need that was not satisfied by classical HMSCs, i.e. the possibility to define TCP-like protocols. As mentioned in introduction, choice of a formal model is often a tradeof between expressive power and decidability. With respect to these criteria, causal HMSCs are very satisfactory: it is an expressive non-interleaved model, that can specify braid-like protocol while keeping some interesting properties decidable. Beyond expressive power, specification languages based on composition of partial orders also have known drawbacks, as model checking of some temporal logics is undecidable without syntactical restrictions [24].

6 A Detailed Example

We consider the TCP sliding window mechanism to show the usefulness and the expressive power of causal HMSCs. The *Transmission Control Protocol* (TCP) is one of the core protocols of Internet. Using TCP, applications on networked hosts can create point-to-point connections to one another, over which they can exchange data in packets. The protocol guarantees reliable and in-order delivery of data from sender to receiver. TCP also distinguishes data for multiple connections by concurrent applications (e.g. Web server and e-mail server) running on the same host.

We first explain the relevant aspects of the TCP protocol [29]. For reasons of simplicity and readability, we shall abstract away some of the technical aspects of the protocol when constructing a model. The classical TCP is divided into 3 parts. The first one is *connection establishment*. The procedure to establish connections uses a synchronize (*syn*) packet and involves an exchange of three messages. This exchange is called a three-way handshake [7]. Once a connection is established it can be used to carry data in both directions, that is, the connection is "full duplex". This connection phase can be modeled using MSCs, as shown in Figure 11. In this example, MSC $Connect12$ describes the case when process 1 initiates a connection, and MSC $Connect21$ the case when process 2 initiates the connection. When a process executes an event labeled by *start*, it is ready to begin the data transfer phase of TCP.

The second phase of the TCP protocol is *data transfer*. TCP is able to transfer a continuous stream of bytes in each direction between its users by packaging some number of bytes into segments for transmission through the Internet system. TCP uses sequence numbering in order to recover from data that is damaged, lost, duplicated, or delivered out of order by the Internet communication system. This is achieved by assigning a sequence number to each

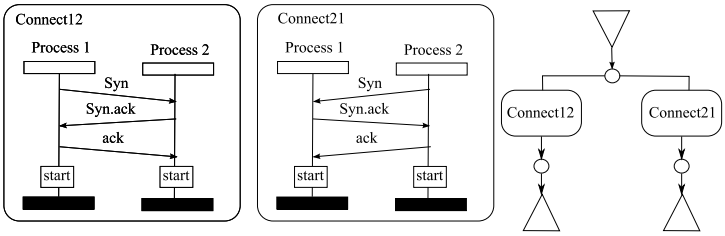

Fig. 11. Connection establishment between process 1 and process 2.

segment transmitted, and requiring a positive acknowledgment (*ack*) from the receiving *tcp*. Actually, the sequence number of *ack* sent by process p is the sequence number of the next *tcp* packet that p expects. Figure 12 shows a causal HMSC modeling a bi-directional data transfer, where sequence numbers are abstracted.

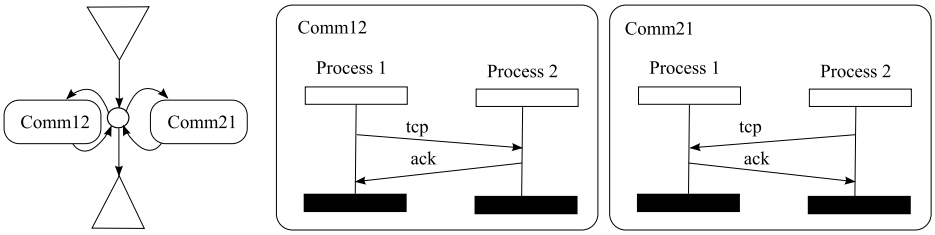

Fig. 12. Data transfer between process 1 and process 2.

The last phase of the TCP protocol is *connection termination*. The connection termination phase uses a four-way handshake. Each side of the connection terminates the session independently. When an endpoint wishes to stop its half of the connection, it transmits a fin packet, which the other end acknowledges with an *ack*. Therefore, a typical tear-down requires a pair of fin and *ack* segments from each *tcp* endpoint. The four-way handshake is modeled on figure 13: an *end* event is seen on process p when no more *tcp* packets are sent from p. In MSC $Fin1$, process 1 stops first, and process 2 can continue to send *tcp* packets, then process 2 stops. In MSC $Fin12$, process 1 and process 2 stop at the same time. In MSC $Fin2$ process 2 initiates the termination of the communication, and then process 1 stops.

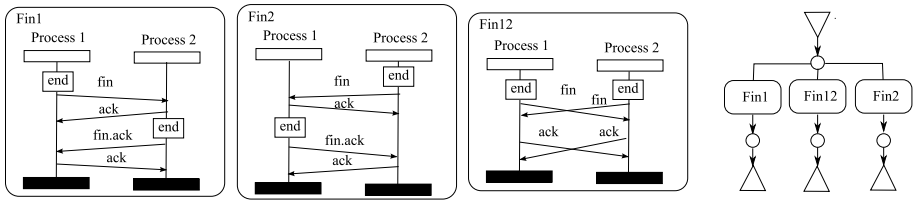

Fig. 13. The TCP connection termination.

A connection can be "half-open" when one side has terminated its connection, but not the other. The side that has terminated can no longer send data using this connection, but the other side can. Finally, it is possible for both hosts to send fin simultaneously. In this case, both sides just have to send *ack* packets to terminate the TCP connection. This can be considered as a 2-way handshake since the fin/ack sequence is done in parallel in both directions.

Automata of Figure 11, Figure 12 and Figure 13 model the 3 phases of TCP protocol. A complete description of the TCP protocol with MSCs can be obtained as a composition of these tree models, just by performing a classical sequential composition of the automata, as shown on Figure 14.

So far, we have proposed scenario descriptions of the TCP protocol, and provided the HMSCs describing the typical executions of TCP, but we did not define the commutation relation over events of the protocol that allows for the interleaving of different phases of the protocol. Let us define the local dependency relations D_p for process p in $\{Process1, Process2\}$. If we chose the normal weak concatenation, as defined in HMSCs, i.e. $D_p = \Sigma_p \times \Sigma_p$, we obtain a synchronized execution of data transfer phase: every process send only one *tcp* packet and waits for the corresponding *ack* packet. Middle part of figure 14 describes this kind of execution, for the (causal) HMSC depicted in the left part of the figure. Note however that in an implementation of the TCP protocol, data transfer from the two sites can be performed in parallel. Hence, the classical sequential composition of HMSCs does not suffice to model interesting behaviors of TCP. Moreover, processes can send *tcp* packets without waiting for acknowledgments. Thus, events occurring between $p(start)$ and $p(end)$ can occur in any order in a visual extension, i.e. $I_p = \{p!q(tcp), p?q(ack), p?q(tcp), p!q(ack), p?q(fin)\}^2 - \{(a, a) \in \Sigma_p^2\}$ for each process p in $\{Process1, Process2\}$ and $q = \{Process1, Process2\} \setminus \{p\}$. An execution of this causal HMSC is shown on the right part of figure 14.

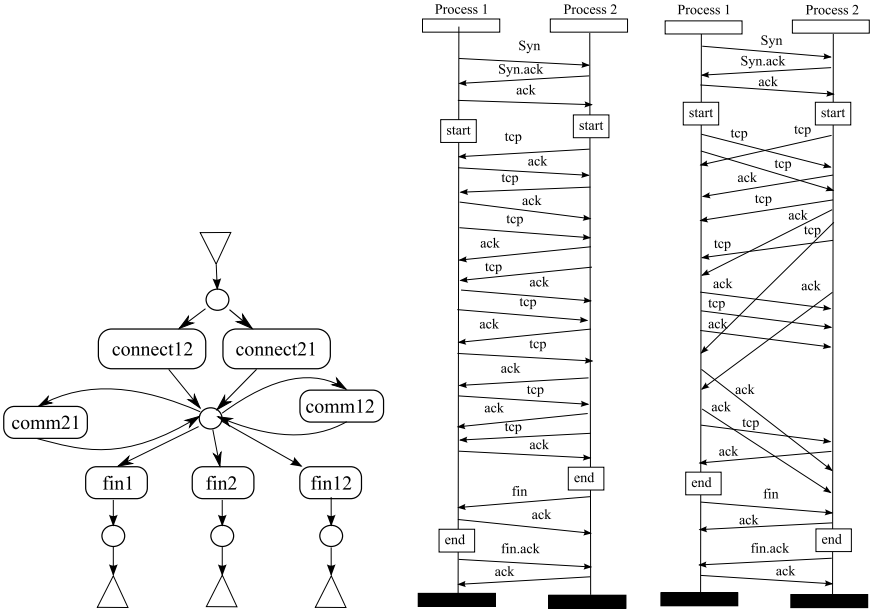

Fig. 14. A (causal) HMSC model of the TCP protocol, and two executions of the TCP example, with the HMSC semantics (in the middle) and with the causal HMSC semantics (on the right).

Note that the causal HMSC model of TCP in Figure 14 is not s-regular, and its linearization language is not regular either: this means this protocol cannot

be modeled by classical communicating finite state machines. Furthermore, this model is not window bounded, as an infinite number of *ack* messages can cross *tcp* ones. Finally, this causal HMSC H is not globally-cooperative, as the cycle labeled by $Comm12.Comm21$ does not have a weakly connected communication graph. However, it is possible [11] to show that the language of H viewed as an automaton on basic parts is closed by commutation, that is $\mathcal{L}_{Basic}(H) = [\mathcal{L}_{Basic}(H)]$. In this case, we can apply directly the second part of Theorem 3 with $\mathcal{A}'$ equal to H'. Consequently, this protocol can be model-checked against any property described by a causal HMSC.

Consider for instance the causal HMSC of Figure 15. Let us call H_{bad} the HMSC H of Figure 14 in which the data transfer part (the two cycles over $Comm21$ and $Comm12$) is replaced by the erroneous data protocol in Figure 15. Then, if $cMSC(H) \cap cMSC(H_{bad}) \neq \emptyset$, we have proved that there are (undesired) behaviors of our protocol where tcp messages are not acknowledged. In a similar way, we can easily check that no message is sent after the connection has been closed.

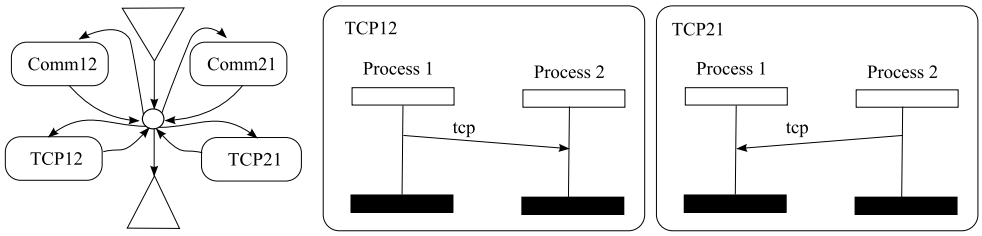

Fig. 15. Unacknowledged TCP packets

7 Conclusion

We have defined an extension of HMSC called causal HMSC that allows the definition of braids, such as those appearing in sliding window protocols. We have also identified in this setting, many subclasses of scenarios that were defined for HMSCs which have decidable verification problems. An interesting class that emerges is globally-cooperative causal HMSCs. This class is incomparable with safe compositional HMSCs because the former can generate scenario collections that are not existentially bounded. Yet, decidability results of verification can be obtained for this class.

An interesting open problem is deciding whether the visual language of a causal HMSC is finitely generated. Yet another interesting issue is to consider the class of causal HMSCs whose visual languages are window-bounded. The set of behaviors generated by these causal HMSCs seems to exhibit a kind of regularity that could be exploited. Finally, designing suitable machine models (along the lines of Communicating Finite Automata [5]) is also an important future line of research.

References

[1] M. Ahuja, A.D. Kshemkalyani, and T. Carlson. A Basic Unit of Computation in Distributed Sytems. In *Proc. of ICDS'90*, pages 12–19, 1990.

[2] R. Alur, G.J. Holzmann, and D. Peled. An analyzer for Message Sequence Charts. In *Proc. of TACAS'96*, number 1055 in LNCS, pages 35–48, 1996.

[3] R. Alur and M. Yannakakis. Model checking of Message sequence Charts. In *Proc. of CONCUR'99*, number 1664 in LNCS, pages 114–129. Springer, 1999.

[4] B. Bollig. On the Expressiveness of Asynchronous Cellular Automata. In *FCT*, pages 528–539, 2005.

[5] D. Brand and P. Zafiropoulo. On Communicating Finite State Machines. Technical Report RZ1053, IBM Zurich Research Lab, 1981.

[6] Benoît Caillaud, Philippe Darondeau, Loïc Hélouët, and Gilles Lesventes. Hmscs as partial specifications ... with pns as completions. In *MOVEP*, pages 125–152, 2000.

[7] Y. Dalal and C. Sunshine. Connection Management in Transport Protocols. *Computer Networks*, 2(6):454–473, 1978.

[8] P. Darondeau, B. Genest, and L. Hélouet. Products of message sequence charts. In *FOSSACS 2008*, number 4962 in LNCS, pages 459–474, 2008.

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

[10] Javier Esparza and Mogens Nielsen. Decidability issues for petri nets - a survey. *Bulletin of the EATCS*, 52:244–262, 1994.

[11] T. Gazagnaire. *Scenario Languages: Using Partial Orders to Model, Verify and Supervize Distributed and Concurrent Systems*. PhD thesis, Univ. Rennes 1, 2008.

[12] T. Gazagnaire, B. Genest, L. Hélouët, P.S. Thiagarajan, and S. Yang. Causal Message Sequence Charts. In *CONCUR*, pages 166–180, 2007.

[13] B. Genest, D. Kuske, and A. Muscholl. A Kleene Theorem and Model Checking for a Class of Communicating Automata. *Information and Computation.*, 204(6):920–956, 2006.

[14] B. Genest, A. Muscholl, H. Seidl, and M. Zeitoun. Infinite-state High-level MSCs: Model-checking and Realizability. *Journal of Computer and System Sciences*, 72(4):617–647, 2006.

[15] E. Gunter, A. Muscholl, and D. Peled. Compositional Message Sequence Charts. In *Proc. of TACAS'01*, number 2031 in LNCS. Springer, 2001.

[16] J.G. Henriksen, M. Mukund, K. Narayan Kumar, M. Sohoni, and P.S. Thiagarajan. A Theory of Regular MSC Languages. *Information and Computation*, 202(1):1–38, 2005.

[17] L. Hélouët and P. Le Maigat. Decomposition of Message Sequence Charts. In *Proc. of SAM'00*, 2000.

[18] G.J. Holzmann. *Design And Validation Of Computer Protocols*. Prentice Hall, 1991.

[19] ITU-TS. *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)*. ITU-TS, 2004.

[20] D.J Kleitman and B.L Rotschild. Asymptotic enumeration of partial orders on a finite set. *Transactions of the American Mathematical Society*, (205):205–220, 1975.

[21] D. Kuske. Regular sets of infinite message sequence charts. *Information and Computation*, 187(1):80–109, 2003.

[22] A. Mazurkiewicz. *Concurrent Program Schemes and Their Interpretation.* Aarhus University, Comp. Science Depart., DAIMI PB-78, 1977.

[23] R. Morin. Recognizable sets of Message Sequence Charts. In *Proc. of STACS'02*, number 2285 in LNCS, pages 523–534. Springer, 2002.

[24] A. Muscholl and D. Peled. Message Sequence Graphs and Decision Problems on Mazurkiewicz Traces. In *Proc. of MFCS'99*, number 1672 in LNCS, pages 81–91. Springer, 1999.

[25] A. Muscholl, D. Peled, and Z. Su. Deciding Properties for Message Sequence Charts. In *Proc. of FoSSaCS'98*, number 1378 in LNCS, pages 226–242. Springer, 1998.

[26] V. Pratt. Modeling Concurrency with Partial Orders. *International Journal of Parallel Programming*, 15(1), 1986.

[27] M. Reniers. *Message Sequence Chart: Syntax and Semantics*. PhD thesis, Eindhoven University of Technology, 1999.

[28] M. Reniers and S. Mauw. High-level Message Sequence Charts. In A. Cavalli and A. Sarma, editors, *SDL97: Time for Testing - SDL, MSC and Trends*, Proc. of the 8th SDL Forum, pages 291–306, Evry, France, Septembre 1997.

[29] RFC793. Transmission control protocol, 1981. DARPA Internet Program Protocol Specification.

[30] E. Rudolph, P. Graubman, and J. Grabowski. Tutorial On Message Sequence Charts. *Computer Networks and ISDN Systems*, 28:1629–1641, 1996.