Electronic Communications of the EASST
Volume 29 (2010)

Proceedings of the
Ninth International Workshop on
Graph Transformation and
Visual Modeling Techniques
(GT-VMT 2010)

Decidable Race Condition and Open Coregions in HMSC

Vojtěch Řehák Petr Slovák Jan Strejček Loïc Hélouët

12 pages

Guest Editors: Jochen Küster, Emilio Tuosto
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Decidable Race Condition and Open Coregions in HMSC

Vojtěch Řehák[1][*] Petr Slovák[1][†] Jan Strejček[1][‡] Loïc Hélouët[2]

[1]Faculty of Informatics, Masaryk University, Brno, Czech Republic
[2]INRIA/IRISA, Rennes, France

Abstract: *Message Sequence Charts (MSCs)* is a visual formalism for the description of communication behaviour of distributed systems. An MSC specifies relations between communication events with partial orders. A situation when two visually ordered events may occur in any order during an execution of an MSC is called a *race* and is usually considered as a design error. While there is a quadratic time algorithm detecting races in a finite communication behaviours called *Basic Message Sequence Charts (BMSCs)*, the race detection problem is undecidable for *High-level Message Sequence Charts (HMSCs)*, an MSC formalism describing potentially infinite sets of potentially unbounded behaviours. To improve this negative situation for HMSCs, we introduce two new notions: a new concept of race called trace-race and an extension of the HMSC formalism with open coregions, i.e. coregions that can extend over more than one BMSC. We present three arguments showing benefits of our notions over the standard notions of race and HMSC. First, every trace-race-free HMSC is also race-free. Second, every race-free HMSC can be equivalently expressed as a trace-race-free HMSC with open coregions. Last, the trace-race detection problem for HMSC with open coregions is decidable and PSPACE-complete. Finally, the proposed extension of coregions allows to represent in a visual fashion whether an arbitrary number of racing events in the usual MSC formalism are concurrent or not.

Keywords: HMSC; race condition; trace-race condition; open coregions;

1 Introduction

Message Sequence Chart (MSC) [ITU04] is a popular visual formalism for specification of distributed systems behaviours (e.g. communication protocols or multi-process systems). Its simplicity and intuitiveness come from the fact that MSC describes only exchange of messages between system components, while other aspects of the system (e.g. content of the messages, computation steps) are abstracted away. Even such an incomplete model can indicate serious errors in the designed system. This paper focuses on a common error called *race condition*.

MSCs are based on composition of simple chronograms called *Basic Message Sequence Charts (BMSCs)*. A BMSC consists of a finite number of *processes* and *events*. Processes are represented by vertical lines, and all events executed by some process are located on its lifeline

[*] Partially supported by Czech Science Foundation (GAČR), grants No. 201/08/P459 and No. P202/10/1469.
[†] Partially supported by the research centre "Institute for Theoretical Computer Science (ITI)", project No. 1M0545.
[‡] Partially supported by Czech Science Foundation (GAČR), grant No. 201/08/P375, and Ministry of Education of the Czech Republic, project No. MSM0021622419.

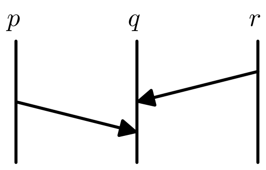

Figure 1: A BMSC containing a race

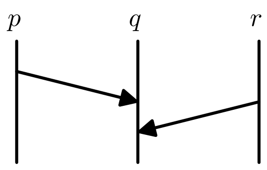

Figure 2: A similar BMSC containing a race

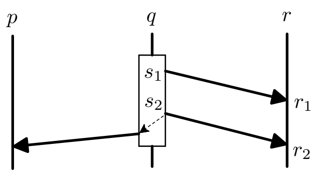

Figure 3: A BMSC containing a race between r_1, r_2

and ordered from top to bottom. Messages are represented by an arrow from a sending event to a receiving event. Total orderings of events on lifelines and messages form a *visual order* $<$, which provides graphically information on the respective ordering of events. However, the visual order can not always be enforced by the architecture of the modelled system. In addition to the visual order, there exists a *causal order* $\ll$, that is weaker than $<$. Intuitively, events e, f are in causal order $e \ll f$, if the BMSC enforces that e always precedes f. There are several definitions of causal order depending on the settings of the modelled system and semantics of the model. For example, if one process sends two messages to another process, the corresponding receive events are causally ordered if and only if the considered message transport protocol has the *FIFO property*: two messages sent from one process to another are always received in the same order. In this paper, we assume that every process has one unbounded buffer for all incoming messages and that the message transport protocol satisfies the FIFO property.

A BMSC contains a *race condition* (or simply *race*) [AHP96] if there are two visually ordered events that are not causally ordered (i.e. they can actually occur in an arbitrary order). For example, Figure 1 depicts that the process q receives a message from r followed by a message from p. As processes and communication in BMSCs are always asynchronous, the messages can be also received in the opposite order as shown in Figure 2. In both figures, the two receive events are in race as they are ordered visually but not causally. Races in BMSC description should be considered as a design error, as they exhibit discrepancies between the intended ordering designed in a BMSC, and the ordering that a real implementation of this BMSC would enforce. Races in a BMSC can be detected in quadratic time [AHP96].

While a BMSC describes only a single and finite communication scenario, its extension called *High-level Message Sequence Chart (HMSC)* [RGG96, AHP96] can describe more complex interactions, with iterations and alternatives between several scenarios. An HMSC is a finite state transition system where each state is labelled by a BMSC or a reference to another HMSC. In the sequel, we will only consider HMSCs labelled by BMSCs. Each run (i.e. a path starting in the initial state and ending in a final state) of an HMSC can be understood as a single BMSC, which is a concatenation of the BMSCs labelling the states along the run. Hence, an HMSC represents a potentially infinite set of BMSCs of unbounded size.

The definition of race was extended to HMSCs in [MP99]. Roughly speaking, an HMSC H has a race if some BMSC represented by H contains a race and H does not represent any BMSC where the two racing events are defined with the opposite visual order. Unfortunately, the problem whether a given HMSC contains a race is undecidable [MP99, ITU04].

In this paper, we propose an alternative definition of race for HMSCs called *trace-race*. Intuitively, an HMSC has a trace-race if some BMSC represented by H contains a race. Clearly, every trace-race-free HMSC is also race-free but not vice versa. To improve the expressive power of trace-race-free HMSCs, we extend the HMSC formalism with *open coregions*. A coregion is a standard part of the MSC formalism that allows some events on the same process in a BMSC to be visually unordered. In particular, coregions can be used to visually order only causally related events (hence making concurrency a visual property). While this application of coregions can remove all races in BMSCs, it is not sufficient for removing all races in HMSCs. An open coregion is basically a coregion spread over several BMSCs. We present a transformation of an arbitrary race-free HMSC into an equivalent trace-race-free HMSC with open coregions, where equivalence means that the two HMSCs have the same linearizations. Finally, we show that the problem whether a given HMSC with open coregions contains a trace-race is decidable and PSPACE-complete. In fact, our algorithm is polynomial for HMSCs with fixed number of processes and gates. For definitions of gates and linearizations see Sections 2 and 3, respectively.

The rest of the paper is organized as follows. Section 2 recalls the definitions of BMSCs, HMSCs, and race condition for BMSCs. The race and trace-race conditions for HMSCs are defined and compared in Section 3. Section 4 is devoted to the translation of race-free HMSCs into equivalent trace-race-free HMSCs. The decidability and complexity of the trace-race detection problem is discussed in Section 5. Section 6 briefly summarizes benefits of the presented notions. Due to the space limitations, we present only crucial lemmata and theorems accompanied by explanations of basic ideas. Proofs with all technical details can be found in [ŘSSH09].

2 Preliminaries

The following definitions omit some features of MSCs given by the ITU standard [ITU04], e.g. atomic actions, labelling of messages with names, timers etc. However, these restrictions are quite common, and our results can be extended to MSCs with atomic actions and message labelling using the technique of [DGH08].

2.1 BMSCs with (open) coregions, gates, and general ordering

The basic concepts of BMSCs are described in Section 1. In the visual representation of a BMSC, processes are depicted as vertical lines and messages are represented by arrows between these lines. Events located on the same process line are visually ordered from top to bottom. A process line may contain segments called *coregions* delimiting subsets of events. Events in a coregion are a priori not in visual order, but they can be visually ordered using a *general ordering* relation. (this relation need not be a partial order). Coregions are visually represented by rectangles and general ordering by dashed arrows between pairs of ordered events (see Figure 3).

In existing MSC formalisms, coregions are limited to finite set of events located in a single BMSC. We extend the definition of BMSCs with *open coregions* and *gates*. These features allow coregions of arbitrary size, spread over several concatenated BMSCs. Gates enable events of different BMSCs to be generally ordered within the final joined coregion. Similar ideas for connecting orders using gates or predicates was already proposed for instance in [Pra86, GH07].

A coregion can be open on top (*top-open coregion*), on bottom (*bottom-open coregion*), or open on both sides. All processes use a common gate name space **G**. For each process p, we define the sets of *top gates* $p.\overline{\mathbf{G}} = \{p.\overline{g} \mid g \in \mathbf{G}\}$ and *bottom gates* $p.\underline{\mathbf{G}} = \{p.\underline{g} \mid g \in \mathbf{G}\}$ located on process p. Given a BMSC with a set of processes $\mathbf{P}$, we set $\mathbf{P}.\overline{\mathbf{G}} = \bigcup_{p\in\mathbf{P}} p.\overline{\mathbf{G}}$ and $\mathbf{P}.\underline{\mathbf{G}} = \bigcup_{p\in\mathbf{P}} p.\underline{\mathbf{G}}$ to be the sets of all top and bottom gates in this BMSC, respectively. We also extend the general ordering to range over both events and gates within an open coregion.

Definition 1 Let $\mathbf{G}$ be a finite gate name space. A BMSC over $\mathbf{G}$ is a tuple $M = (\mathbf{P}, E_S, E_R, P, \{<_p\}_{p\in\mathbf{P}}, \mathbf{M}, \mathbf{C}, \{\prec_C\}_{C\in\mathbf{C}})$ where

- $\mathbf{P}$ is a finite set of processes.
- E_S, E_R are disjoint finite sets of *send* and *receive events*, respectively. We set $E = E_S \cup E_R$.
- $P : E \to \mathbf{P}$ is a mapping that associates each event with a process.
- $<_p$ is a total order on all events on a process p.
- $\mathbf{M} \subseteq (E_S \times E_R)$ is a bijective mapping, relating every send with a unique receive. We assume that a process cannot send a message to itself, i.e. $P(e) \neq P(f)$ whenever $(e, f) \in \mathbf{M}$. For any $(e, f) \in \mathbf{M}$, we use $\mathbf{M}(e)$ to denote the receive event f, and $\mathbf{M}^{-1}(f)$ to denote the send event e.
- $\mathbf{C}$ is a finite set of pairwise disjoint *coregions*, where a coregion $C \in \mathbf{C}$ is a consistent nonempty subset of events and gates of a single process p, i.e.

  - $\emptyset \neq C \subseteq P^{-1}(p) \cup p.\overline{\mathbf{G}} \cup p.\underline{\mathbf{G}}$ for some $p \in \mathbf{P}$
  - if $e <_p d <_p f$ and $e, f \in C$, then $d \in C$.

  A coregion C containing a top gate is called *top-open* and it has to contain all top gates $p.\overline{\mathbf{G}}$ and satisfy that if $e <_p f$ and $f \in C$ then $e \in C$. A coregion C containing a bottom gate is called *bottom-open* and it has to contain all bottom gates $p.\underline{\mathbf{G}}$ and satisfy that if $e <_p f$ and $e \in C$ then $f \in C$. A coregion which is both top-open and bottom-open is called just *open*.
- $\prec_C$ is an acyclic relation called *general ordering* on elements in C such that $\prec_C \subseteq (p.\overline{\mathbf{G}} \cup P^{-1}(p)) \times (p.\underline{\mathbf{G}} \cup P^{-1}(p))$, where p is the process containing the coregion C.

The definition says that a top-open coregion has to contain all top gates. As coregions are pairwise disjoint, there is at most one top-open coregion. Similarly, each BMSC contains at most one bottom-open coregion. Note that we do not impose that coregions contain events. For example, an open coregion covering an inactive process can connect top and bottom gates.

In the visual representation, an open coregion is depicted as a rectangle without the side(s) which are open. Gates are represented by small squares on the corresponding missing side of these rectangles. As gates are always depicted in the same order, their names become redundant (and they are often omitted). For example of BMSCs with open coregions see Figure 4. Recall that dashed arrows represent a general ordering.

2.2 Visual order, causal order and race in BMSCs

Every BMSC induces two preorders on events: visual $<$ and causal $\ll$ ordering. The *visual order* represents the order of events directly described by the BMSC. Loosely speaking, $<$ is the reflexive and transitive closure of total orders $<_p$ of events on each process, excluding the order of events within each coregion, plus general ordering and the order generated by the FIFO property and the fact that every send event precedes the corresponding receive event. The visual order is actually defined over the union of events and gates.

Definition 2 Let $M = (\mathbf{P}, E_S, E_R, P, \{<_p\}_{p\in\mathbf{P}}, \mathbf{M}, \mathbf{C}, \{\prec_C\}_{C\in\mathbf{C}})$ be a BMSC over $\mathbf{G}$. A *visual order* $<$ given by M is the least preorder $< \subseteq (\mathbf{P}.\overline{\mathbf{G}} \cup E) \times (\mathbf{P}.\underline{\mathbf{G}} \cup E)$ such that

(i) $<$ contains the relation $\left(\bigcup_{p\in\mathbf{P}} <_p \setminus \bigcup_{C\in\mathbf{C}} C\times C\right) \cup \left(\bigcup_{C\in\mathbf{C}} \prec_C\right) \cup \mathbf{M}$,

(ii) $<$ respects the FIFO property, i.e. for every $e, f \in E_S$ such that $P(e) = P(f)$ and $P(\mathbf{M}(e)) = P(\mathbf{M}(f))$, it holds that $e < f$ implies $\mathbf{M}(e) < \mathbf{M}(f)$.[1]

One can define a BMSC where $<$ is not a partial order. This situation is clearly a design error and it can be detected by a cycle detection algorithm. In the sequel, we always assume that $<$ is a partial order.

In contrast to the visual order, the *causal order* captures the partial order of events that has to be respected by all executions as it is enforced by the semantics of the design. Hence, the causal order represents the interpretation of a BMSC relevant to its implementation.

Definition 3 Given a BMSC $M = (\mathbf{P}, E_S, E_R, P, \{<_p\}_{p\in\mathbf{P}}, \mathbf{M}, \mathbf{C}, \{\prec_C\}_{C\in\mathbf{C}})$ over $\mathbf{G}$, we define a *causal order* $\ll$ as the least partial order on E such that $e \ll f$, if

- $(e, f) \in \mathscr{M}$, i.e. *send and receive events of each message are ordered*, or
- $P(e) = P(f)$ and $e < f$ and $f \in E_S$, i.e. *any send event is delayed until all previous events took place*, or
- $P(e) = P(f)$ and $\exists e', f' \in E$ such that $e' < f'$, $P(e') = P(f')$, $(e', e) \in \mathscr{M}$ and $(f', f) \in \mathscr{M}$, i.e. *causal order respects the FIFO property.*

Lemma 1 *For every BMSC it holds $\ll \subseteq <$. Further, for each $f \in E_S$ it holds $e < f \iff e \ll f$.*

A race is defined as a difference between visual and causal order on events.

Definition 4 If a BMSC contains some events e, f satisfying $e < f$ and $e \not\ll f$, we say that the BMSC contains a *race (between events e, f)*. Otherwise, the BMSC is called *race-free*.

Theorem 1 ([AHP96]) *The problem whether a given BMSC with n events contains a race is decidable in time $\mathcal{O}(n^2)$.*

Note that [AHP96] deals with BMSCs without any coregions. However, an extension of this theorem to BMSCs with (possibly open) coregions and general ordering is straightforward.

The following lemma says that a BMSC contains a race if and only if it contains a race between two events on the same process.

Lemma 2 *A BMSC contains a race if and only if there are two events e, f such that $e < f$, $e \not\ll f$, and $P(e) = P(f)$.*

[1] The FIFO property is usually not included in the definition of a visual order. However, once we choose the FIFO message passing setting, violating this property should be considered as a design error and it can be easily detected. Hence, we included the property directly in our definition.

2.3 HMSCs

An HMSC is a finite directed graph with an *initial* state and a set of *final* states, where each state is labelled with a BMSC.

Definition 5 An *HMSC* is a tuple $(S, \rightarrow, s_0, S_F, L, \mathscr{L})$ where

- S is a finite set of states, $s_0 \in S$ is an *initial state*, $S_F \subseteq S$ is a set of *final states*,
- $\rightarrow \subseteq S \times S$ is a transition relation,
- $\mathscr{L}$ is a finite set of BMSCs over a common gate name space,
- $L(s) : S \rightarrow \mathscr{L}$ is a mapping assigning to each state a BMSC.

A sequence of states $\sigma = s_1 s_2 \cdots s_k$ is a *path*, if $(s_i, s_{i+1}) \in \rightarrow$ for every $1 \leq i < k$. A path is a *run* if $s_1 = s_0$ and $s_k \in S_F$.

To give a semantics of HMSCs, we need to define a concatenation operation on BMSCs. Intuitively, the concatenation of BMSCs M_1 and M_2 is done by gluing the corresponding process lines together with the BMSC M_2 drawn beneath M_1. If M_1 and M_2 contain bottom-open and top-open coregions on a process p, respectively, then the two coregions are merged and each bottom gate $p.\underline{g}$ of the upper open coregion is joined with the corresponding top gate $p.\overline{g}$ of the lower open coregion. Further, whenever an event e of the upper coregion is generally ordered with a joined gate and this joined gate is generally ordered with an event f of the lower coregion, the events e, f become generally ordered in the newly created coregion. The joined gates are then removed. If M_1 contains a bottom-open coregions on a process p that is not in M_2, then the coregion remains bottom-open. However, if M_1 contains a bottom-open coregion on a process p and M_2 contains the process p without any top-open coregion on it, then the bottom side of the coregion is closed.

Definition 6 Let $M_i = (\mathbf{P}_i, E_{Si}, E_{Ri}, P_i, \{<_{ip}\}_{p \in \mathbf{P}}, \mathbf{M}_i, \mathbf{C}_i, \{\prec_{iC}\}_{C \in \mathbf{C}_i})$ for $i = 1, 2$ be two BMSCs over a common gate name space $\mathbf{G}$ and such that the sets $E_{S1} \cup E_{R1}$ and $E_{S2} \cup E_{R2}$ are disjoint (we can always rename events so that the sets become disjoint). The concatenation of M_1 and M_2 is the BMSC $M_1 \cdot M_2 = (\mathbf{P}_1 \cup \mathbf{P}_2, E_{S1} \cup E_{S2}, E_{R1} \cup E_{R2}, P_1 \cup P_2, \{<_p\}_{p \in \mathbf{P}}, \mathbf{M}_1 \cup \mathbf{M}_2, \mathbf{C}, \{\prec_C\}_{C \in \mathbf{C}})$ where

$$<_p = \begin{cases} <_{1p} \text{ if } p \in \mathbf{P}_1 \setminus \mathbf{P}_2 \\ <_{2p} \text{ if } p \in \mathbf{P}_2 \setminus \mathbf{P}_1 \\ \text{transitive closure of } <_{1p} \cup <_{2p} \cup (P_1^{-1}(p) \times P_2^{-1}(p)) \\ \text{if } p \in \mathbf{P}_1 \cap \mathbf{P}_2 \end{cases}$$

and $\mathbf{C}$ contains all coregions C of the following five kinds:

1. $C \in \mathbf{C}_1$ and C is not bottom-open or C is on a process $p \in \mathbf{P}_1 \setminus \mathbf{P}_2$. We set $\prec_C = \prec_{1C}$.
2. $C \in \mathbf{C}_2$ and C is not top-open or C is on a process $p \in \mathbf{P}_2 \setminus \mathbf{P}_1$. We set $\prec_C = \prec_{2C}$.
3. $C = C_1 \setminus p.\underline{\mathbf{G}}$ for some $C_1 \in \mathbf{C}_1$ such that $p.\underline{\mathbf{G}} \subseteq C_1$ and $p.\overline{\mathbf{G}} \cap C_2 = \emptyset$ for all $C_2 \in \mathbf{C}_2$, i.e. C corresponds to a coregion of $\mathbf{C}_1$ that is bottom-open but there is no matching top-open coregion in $\mathbf{C}_2$ (note that C is closed on the bottom side). We set $\prec_C = \prec_{1C_1} \cap (C \times C)$.

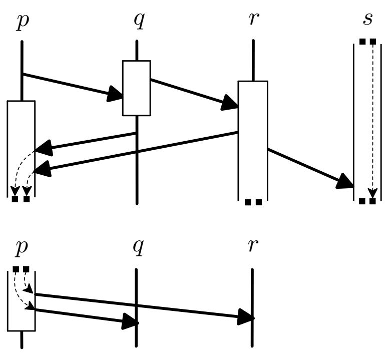

Figure 4: BMSCs M_1 (upper) and M_2

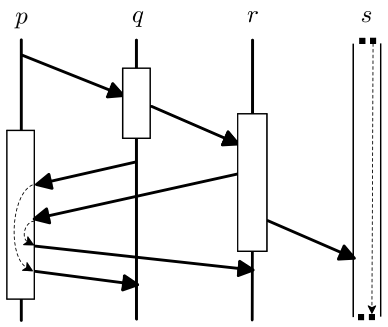

Figure 5: Concatenation $M_1 \cdot M_2$

4. $C = C_2 \setminus p.\overline{\mathbf{G}}$ for some $C_2 \in \mathbf{C}_2$ such that $p.\overline{\mathbf{G}} \subseteq C_2$ and $p.\underline{\mathbf{G}} \cap C_1 = \emptyset$ for all $C_1 \in \mathbf{C}_1$, i.e. C corresponds to a coregion of $\mathbf{C}_2$ that is top-open but there is no matching bottom-open coregion in $\mathbf{C}_1$. We set $\prec_C = \prec_{2C_2} \cap (C \times C)$.
5. $C = (C_1 \setminus p.\underline{\mathbf{G}}) \cup (C_2 \setminus p.\overline{\mathbf{G}})$ for some $C_1 \in \mathbf{C}_1$ and $C_2 \in \mathbf{C}_2$ satisfying $p.\underline{\mathbf{G}} \subseteq C_1$ and $p.\overline{\mathbf{G}} \subseteq C_2$, i.e. C is a bottom-open coregion of $\mathbf{C}_1$ merged with the matching top-open coregion of $\mathbf{C}_2$. We set

$$\begin{aligned}\prec_C = \{(e,f) \mid\ & \big((e,f) \in \prec_{1C_1} \text{ and } f \notin p.\underline{\mathbf{G}}\big) \text{ or } \big((e,f) \in \prec_{2C_2} \text{ and } e \notin p.\overline{\mathbf{G}}\big) \\ & \text{or } \big((e,p.\underline{g}) \in \prec_{1C_1} \text{ and } (p.\overline{g},f) \in \prec_{2C_2} \text{ for some } g \in \mathbf{G}\big)\}.\end{aligned}$$

Note that if visual orders of M_1 and M_2 are partial orders, then the visual order of $M_1 \cdot M_2$ is also a partial order. Figures 4 and 5 provide an example of two BMSCs and their concatenation.

Each path $s_1 s_2 \cdots s_k$ of an HMSC represents a single BMSC given by concatenation of the BMSCs assigned to $s_1, s_2, \ldots, s_k$, i.e. a path $\sigma = s_1 s_2 \cdots s_k$ represents the BMSC $L(\sigma) = L(s_1) \cdot L(s_2) \cdot \ldots \cdot L(s_k)$. Hence, an HMSC represents a set of BMSCs corresponding to its runs. As an HMSC may contain a cycle, the represented set of BMSCs can be infinite and there is no bound on the size (i.e. number of events) of such BMSCs.

3 Race conditions in HMSCs

First we explain the idea of race conditions for a set of BMSCs. Let us consider a system where two processes p and r send a message to a third process q, that receives them in arbitrary order. This behaviour can be specified (even without any coregion) by two BMSCs depicted in Figures 1 and 2. Even if both BMSCs contain a race, the specification given by this pair of BMSCs should be considered as race-free because both permutations of the two receive events on process q allowed by causal ordering are included in the specification.

The race condition for a set of BMSCs can formulated very simply using the following terminology. An *execution induced by a BMSC M* is a totally ordered set $(E, \sqsubset)$, where E is the set of events of M and $\sqsubset$ is a linear extension of the causal order $\ll$ given by M. We say that such an execution $(E, \sqsubset)$ *corresponds to a BMSC M'* if M' has the same set of events and $\sqsubset$ is a linear extension of the visual order $<$ of M'.

Definition 7 We say that a set of BMSCs contains a *race* if there exists an execution induced by some BMSC of the set and not corresponding to any BMSC of the set.

The race condition for HMSCs introduced in [MP99] follows the same principle. Unfortunately, we cannot directly say that an HMSC contains a race if it represents a set of BMSCs containing a race. The problem is that the BMSCs represented by the HMSC are constructed with the concatenation operation during which events can be renamed. Therefore, the events are replaced by *labels* keeping the information about sending and receiving processes. Further, the linearly ordered executions are replaced by words called *linearizations*.

Definition 8 Let $M = (\mathbf{P}, E_S, E_R, P, \{<_p\}_{p\in\mathbf{P}}, \mathbf{M}, \mathbf{C}, \{\prec_C\}_{C\in\mathbf{C}})$ be a BMSC. We define an auxiliary function $label : E \to \{p!q, p?q \mid p,q \in \mathbf{P}\}$ such that

$$label(e) = \begin{cases} p!q & \text{if } e \in E_S,\ p = P(e), \text{ and } q = P(\mathbf{M}(e)) \\ p?q & \text{if } e \in E_R,\ p = P(e), \text{ and } q = P(\mathbf{M}^{-1}(e)). \end{cases}$$

A *linearization* of M w.r.t. a partial order $\sqsubset \in \{<, \ll\}$ is a word $label(e_1)label(e_2)\cdots label(e_n)$ such that $E = \{e_1, e_2, \cdots, e_n\}$ and $e_i \sqsubset e_j$ implies $i < j$. Moreover, we define $Lin_{\sqsubset}(M)$ to be the set of all linearizations of M w.r.t. $\sqsubset$. Finally, we define *linearizations* of an HMSC H w.r.t. $\sqsubset \in \{<, \ll\}$ to be the set

$$Lin_{\sqsubset}(H) = \bigcup_{\sigma \text{ is a run of } H} Lin_{\sqsubset}(L(\sigma)).$$

Intuitively, $Lin_{\ll}(M)$ represents all executions induced by M, while $Lin_{<}(M)$ represents all executions corresponding to M.

Definition 9 ([MP99]) An HMSC H contains a *race* if $Lin_{<}(H) \neq Lin_{\ll}(H)$.

This definition of race has several drawbacks. First of all, the problem whether an HMSC contains a race is undecidable even if we restrict the problem to HMSCs without coregions [MP99, MP00]. Further, as soon as we consider HMSCs with coregions, this notion of race does not tally with the definition of race for BMSCs. For example, the BMSC drawn in Figure 3 contains a race as the messages from q to r can be sent in arbitrary order while the receive events r_1, r_2 are visually ordered. If we look at this BMSC as an HMSC with only one state, then there is no race with respect to Definition 9 as both events r_1, r_2 are represented in linearizations by the same label $r?q$ and therefore the information about their order is lost.

A simple definition of a trace-race follows.

Definition 10 An HMSC H contains a *trace-race* if there is a run σ of H such that the BMSC $L(\sigma)$ contains a race.

If an HMSC H contains no trace-race, then each of its runs σ represents a race-free BMSC $L(\sigma)$. As visual and causal orders of a race-free BMSC coincide, we get that $Lin_{<}(L(\sigma)) = Lin_{\ll}(L(\sigma))$. Hence, every trace-race-free HMSC is also race-free. The inverse implication does not hold. For example, the race-free HMSC of Figure 6 has a trace-race (the HMSC describes the system discussed at the beginning of this section).

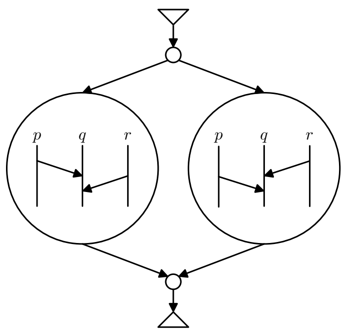

Figure 6: An race-free HMSC with a trace-race

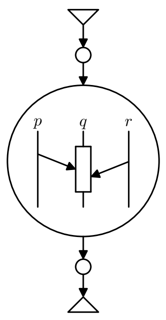

Figure 7: A trace-race-free HMSC

As the definition of trace-race does not replace events by labels, trace-race tallies with the definition of race for BMSCs, i.e. a BMSC has a race if and only if it has a trace-race when seen as a single state HMSC.

It is commonly agreed that designers should avoid races in HMSCs. We provide an intuitive explanation why we think that designers should actually avoid trace-races as well. Let H be a race-free HMSC with a trace-race. As H has a trace-race, there has to be a run σ such that $L(\sigma)$ induces an execution not corresponding to $L(\sigma)$. As H is race-free, this execution corresponds to some BMSC $L(\sigma')$, where $\sigma' \neq \sigma$ is another run of H.[2] Lemma 1 implies that all executions corresponding to a run are also induced by the run. Hence, the execution is in fact induced by (at least) two different runs of the HMSC. This is a potential source of errors as an implementation of this kind of description tends to violate the "write things once" programming principle. Moreover, trace-race-free HMSCs are usually more compact and their use may encourage a cleaner way for designing systems. For example, compare the trace-race-free system depicted on Figure 7, which models the same behaviour as the race-free HMSC of Figure 6.

4 Transformation of HMSCs into trace-race-free HMSCs

We present a transformation of an arbitrary HMSC H into a trace-race-free HMSC H'. The transformation modifies only BMSCs in the states of H. The modified BMSCs have the same processes, events, and causal orders as the original BMSCs, but they induce different visual orders. As the structure of H' remains the same, it has the same set of runs as H. The HMSC is changed in such a way that both visual and causal orders of the BMSC corresponding to a run σ in H' are the same as the causal order of the BMSC corresponding to σ in H. Hence, $Lin_<(H') = Lin_\ll(H') = Lin_\ll(H)$ and H' is trace-race-free. Moreover, if H was race-free (i.e. $Lin_<(H) = Lin_\ll(H)$), then $Lin_<(H) = Lin_\ll(H) = Lin_<(H') = Lin_\ll(H')$ and we say that H and H' are equivalent. The transformation of the original HMSC H proceeds in two steps.

Step 1 We modify each BMSC M in a state of H such that each process is covered with a coregion open on both sides, while BMSCs represented by the resulting HMSC remain the same as those represented by H: the same events on the same processes with the same visual and causal orders. We use general orderings and two fresh gate names pre, suc to induct the same

[2] In fact, the linearization corresponding to the mentioned execution has to be in $Lin_<(L(\sigma'))$.

visual (and hence also causal) orders. The definition of concatenation implies that, on a process p, all events of BMSCs preceding M in some run of H are visually ordered before all events of M (except those in a top-open coregion). This relation is preserved using the gate $p.\overline{pre}$. Similarly, all events of M (except those in a bottom-open coregion) are visually ordered before all events of BMSCs succeeding M in some run of H. This relation is preserved using the gate $p.\underline{suc}$.

More precisely, the general ordering $\prec_{C'}$ of a coregion C' open on both sides and covering a process p is defined as the least relation satisfying the following conditions (where $\mathbf{G}' = \mathbf{G} \cup \{pre, suc\}$ and $<$ refers to the original visual order of M):

- For every $e \in (E \cup p.\overline{\mathbf{G}}), f \in (E \cup p.\underline{\mathbf{G}})$, if $e < f$, then $(e,f) \in \prec_{C'}$.
- For every $e \in E$, if e is not in a top-open coregion in M, then $(p.\overline{pre}, e) \in \prec_{C'}$.
- For every $e \in E$, if e is not in a bottom-open coregion in M, then $(e, p.\underline{suc}) \in \prec_{C'}$.
- $p.\overline{suc} \times (E \cup p.\underline{\mathbf{G}'}) \subseteq \prec_{C'}$.
- $(E \cup p.\overline{\mathbf{G}'}) \times p.\underline{pre} \subseteq \prec_{C'}$.

Step 2 We restrict the general orderings to induce visual orders equivalent to the original causal orders. Due to Definition 3, it is sufficient to generally order pairs (e,f) such that $e < f$ and $f \in E_S$ (where $<$ refers to the original visual order). Formally, we replace every general ordering $\prec_{C'}$ computed in the previous step with $\prec_{C'} \cap ((\mathbf{P}.\overline{\mathbf{G}'} \cup E) \times (\mathbf{P}.\underline{\mathbf{G}'} \cup E_s))$.

5 Trace-race detection problem for HMSCs

This section studies decidability and complexity of the *trace-race detection problem*, i.e. the problem whether a given HMSC (with open coregions) contains a trace-race. We assume that each state of a given HMSC $H = (S, \rightarrow, s_0, S_F, L, \mathbf{L})$ appears on some run of H and it is labelled with a race-free BMSC. Recall that H contains a trace-race if and only if there is a run σ such that the BMSC $L(\sigma)$ contains a race. There is such a run if and only if there is a path $\pi = s_0 s_1 s_2 \dots s_k$ where $L(s_0 \dots s_{k-1})$ is a race-free BMSC containing an event e and $L(s_k)$ is a race-free BMSC containing an event f such that $L(\pi)$ contains a race between e and f. Due to Theorem 1, one can easily check whether a given path $\pi = s_0 s_1 s_2 \dots s_k$ meets these conditions. However, it does not solve the trace-race detection problem as there could be infinitely many paths starting in s_0.

Our detection technique relies on a precise characterization of races appearing in concatenation $M_1 \cdot M_2$ of two race-free BMSCs. For this, we need two new functions returning sets of *joined gates*. Intuitively, a joined gate $p.g$ for a concatenation $M_1 \cdot M_2$ is a reference to a gate that appears as a bottom gate $p.\underline{g}$ in M_1 and is identified with the corresponding top gate $p.\overline{g}$ of M_2 during concatenation. We denote by $p.\mathbf{G}$ and $\mathbf{P}.\mathbf{G}$ the sets all joined gates over gate name space $\mathbf{G}$ and the process p or all processes $\mathbf{P}$, respectively.

Definition 11 Given a BMSC $M = (\mathbf{P}, E_S, E_R, P, \{<_p\}_{p \in \mathbf{P}}, \mathbf{M}, \mathbf{C}, \{\prec_C\}_{C \in \mathbf{C}})$ over gate name space $\mathbf{G}$, we define two functions $\downarrow(), \uparrow() : (E_S \cup E_R) \rightarrow 2^{\mathbf{P}.\mathbf{G}}$ as $\downarrow(e) = \{p.g \in \mathbf{P}.\mathbf{G} \mid e < p.\underline{g}\}$ and $\uparrow(e) = \{p.g \in \mathbf{P}.\mathbf{G} \mid p.\overline{g} < e\}$.

In the context of a concatenation $M_1 \cdot M_2$, $\downarrow(e)$ and $\uparrow(e)$ always refer to the values of these functions in the BMSC M_i (where $i \in \{1,2\}$) originally containing the event e. The characterization of races in $M_1 \cdot M_2$ is formulated in Lemma 3. The lemma assumes that the two race-free BMSCs M_1, M_2 are in the *special form* of Section 4, where all events and gates on each process are covered by a coregion open on both sides. Note that every BMSC can be converted to this form by the first step of the transformation presented in the previous section.

Lemma 3 *Let $M_1 \cdot M_2$ be a concatenation of two race-free BMSCs M_1, M_2 in the special form, e be an event of M_1, and f be an event of M_2 such that $P(e) = P(f) = p$. Then e and f are in race if and only if all the following conditions hold.*

1. *f is a receive event*
2. $\downarrow(e) \cap \uparrow(f) \cap p.\mathbf{G} \neq \emptyset$
3. $\downarrow(e) \cap \uparrow(\mathscr{M}^{-1}(f)) = \emptyset$
4. $label(e) = label(f) \implies \downarrow(\mathscr{M}^{-1}(e)) \cap \uparrow(\mathscr{M}^{-1}(f)) = \emptyset$
5. *$\forall$ receive events f' of M_2 such that $f' < \mathscr{M}^{-1}(f)$:* $label(e) = label(f') \implies \downarrow(\mathscr{M}^{-1}(e)) \cap \uparrow(\mathscr{M}^{-1}(f')) = \emptyset$

The precondition $P(e) = P(f)$ is not a serious restriction thanks to Lemma 2. The characterization says that to decide whether an event e of M_1 is in race with some event of M_2, one needs to know only $label(e)$, $\downarrow(e)$ and, if e is a receive action, then also $\downarrow(\mathscr{M}^{-1}(e))$. Triples $(label(e), \downarrow(e), \downarrow(\mathscr{M}^{-1}(e)))$ for receive events e and $(label(e), \downarrow(e), \emptyset)$ for send events e are called *footprints* of M_1. Note that the number of footprints for a fixed set of processes $\mathbf{P}$ and a gate name space $\mathbf{G}$ is bounded by $2 \cdot |\mathbf{P}|^2 \cdot 2^{|\mathbf{P}| \cdot |\mathbf{G}|} \cdot 2^{|\mathbf{P}| \cdot |\mathbf{G}|}$. Extending function P to labels as $P(p!q) = P(p?q) = p$, Lemma 3 can be reformulated as follows:

Lemma 4 *Let M_1 and M_2 be two race-free BMSCs in the special form. The concatenation $M_1 \cdot M_2$ contains a race if and only if there is a receive event f in M_2 and a footprint (l, F, F') of M_1 such that all the following conditions hold.*

1. $P(l) = P(f) = p$
2. $F \cap \uparrow(f) \cap p.\mathbf{G} \neq \emptyset$
3. $F \cap \uparrow(\mathscr{M}^{-1}(f)) = \emptyset$
4. $l = label(f) \implies F' \cap \uparrow(\mathscr{M}^{-1}(f)) = \emptyset$
5. *$\forall$ receive events f' of M_2 such that $f' < \mathscr{M}^{-1}(f)$:* $l = label(f') \implies F' \cap \uparrow(\mathscr{M}^{-1}(f')) = \emptyset$

Now we return to the observation from the beginning of this section. Let s, s' be two states of the HMSC H such that $(s, s') \in \rightarrow$. If we have the set of footprints of all BMSCs of the form $L(\pi)$ where π is a path leading from s_0 to s, we are able to decide whether any concatenation $L(\pi) \cdot L(s')$ contains a race. Moreover, we can effectively compute the set of footprints of all BMSCs of the form $L(\pi).L(s')$.

Hence, with each state s of the HMSC H we associate the set of all footprints of all BMSCs corresponding to paths starting in s_0 and leading to s. These sets of associated footprints can be easily computed using the fixpoint approach. If no race is detected during the computation, then the HMSC is trace-race-free. The precise algorithm, its complexity analysis, and complexity analysis of the trace-race detection problem can be found in [ŘSSH09].

Theorem 2 *Given an HMSC $H = (S, \rightarrow, s_0, S_f, L, \mathscr{L})$ (with open coregions and gates) over a gate name space $\mathbf{G}$, the problem whether H contains a trace-race is decidable in time $\mathscr{O}(|S|^2 \cdot b^3 \cdot |\mathbf{P}|^2 \cdot 2^{2 \cdot |\mathbf{P}| \cdot (|\mathbf{G}|+2)})$, where b is the size of the largest BMSC in $\mathbf{L}$. Hence, the problem is in* P *if the number of processes and gates is fixed.*

Theorem 3 *The trace-race detection problem is* PSPACE*-complete.*

6 Conclusions

We have introduced two new notions for HMSCs: an extension of the formalism with *open coregions* and a new race condition for HMSCs called *trace-race*. Definitions of race and trace-race directly imply that every trace-race-free HMSC is also race-free. We have shown that every race-free HMSC can be translated into an equivalent trace-race-free HMSC using open coregions, where by equivalence we mean that the two HMSCs represent sets of BMSCs with identical linearizations. Hence, trace-race-free HMSCs with open coregions are as expressive as race-free HMSCs with open coregions (and we conjecture that trace-race-free HMSCs with open coregions are in fact strictly more expressive than race-free HMSCs without open coregions). While the race detection problem is undecidable even for HMSCs without coregions [MP99], we have demonstrated that the trace-race detection problem is decidable (and PSPACE-complete) for HMSCs with open coregions. Therefore, HMSCs with open coregions and the trace-race notion appear as good candidates for tractable analysis of race ambiguities in scenario based designs.

The trace-race detection algorithm is implemented in *Sequence Chart Studio*, a Microsoft Visio add-on available at `http://scstudio.sourceforge.net/`. The studio currently supports HMSCs with closed coregions only (a support of open coregions is planned too).

Acknowledgements: We would like to thank Philippe Darondeau for an important hint.

Bibliography

[AHP96] R. Alur, G. Holzmann, D. Peled. An Analyzer for Message Sequence Charts. In *TACAS'96*. LNCS, pp. 35–48. Springer, 1996.

[DGH08] P. Darondeau, B. Genest, L. Hélouët. Products of Message Sequence Charts. In *FOSSACS 2008*. LNCS 4962, pp. 458–473. Springer, 2008.

[GH07] T. Gazagnaire, L. Hélouët. Event Correlation with Boxed Pomsets. In *FORTE 2007*. LNCS 4574, pp. 160–176. Springer, 2007.

[ITU04] ITU Telecommunication Standardization Sector - Study group 17. ITU Recommandation Z.120, Message Sequence Charts (MSC). 2004.

[MP99] A. Muscholl, D. Peled. Message Sequence Graphs and Decision Problems on Mazurkiewicz Traces. In *MFCS'99*. LNCS 1672, pp. 81–91. Springer, 1999.

[MP00] A. Muscholl, D. Peled. Analyzing Message Sequence Charts. In *SAM 2000: Proceedings of the 2nd Conference on MSC and SDL*. Pp. 3–17. 2000.

[Pra86] V. R. Pratt. Modeling concurrency with partial orders. *International Journal of Parallel Programming* 15(1):33–71, 1986.

[RGG96] E. Rudolph, P. Graubmann, J. Grabowski. Tutorial on Message Sequence Charts. *Computer Networks and ISDN Systems* 28(12):1629–1641, 1996.

[ŘSSH09] V. Řehák, P. Slovák, J. Strejček, L. Hélouët. Decidable Race Condition for HMSC. Technical report FIMU-RS-2009-10, Faculty of Informatics, Masaryk Univ., 2009.