

A Framework for Pathologies of Message Sequence Charts

Haitao Dan^a, Robert M. Hierons^a, Steve Counsell^a

^a*School of Information Systems, Computing and Mathematics, Brunel University, Uxbridge, Middlesex, United Kingdom*

Abstract

Context: It is known that a Message Sequence Chart (MSC) specification can contain different types of pathology. However, definitions of different types of pathology and the problems caused by pathologies are unclear, let alone the relationships between them. In this circumstance, it can be problematic for software engineers to accurately predict the possible problems that may exist in implementations of MSC specifications and to trace back to the design problems in MSC specifications from the observed problems of an implementation.

Objective: We focus on generating a clearer view on MSC pathologies and building formal relationships between pathologies and the problems that they may cause.

Method: By concentrating on the problems caused by pathologies, a categorisation of problems that a distributed system may suffer is first introduced. We investigate the different types of problems and map them to a categories of pathologies. Thus, existing concepts related to pathology are refined and necessary concepts in the pathology framework are identified. Finally, we formally prove the relationships between the concepts in the framework.

Results: A pathology framework is established as desired based on a restriction that considers problematic scenarios with a single undesirable event. In this framework, we define disjoint categories of both pathologies and the problems caused; the identified types of pathology are successfully mapped to the problems that they may cause.

Conclusion: The framework achieved in this paper introduces taxonomies into and clarifies relationships between concepts in research on MSC pathologies. The taxonomies and relationships in the framework can help software engineers to predict problems and verify MSC specifications. The single undesirable event restriction not only enables a categorisation of pathological scenarios, but also has the potential practical benefit that a software engineer can concentrate on key problematic scenarios. This may make it easier to either remove pathologies from an MSC specification \mathcal{M} or test an implementation developed from \mathcal{M} for potential problems resulting from such pathologies.

Keywords: Message sequence charts, Pathology, Race, Non-local choice, Implied scenarios

1. Introduction

Message Sequence Charts (MSCs) are a specification language suitable for describing the behaviour of distributed systems [1]. MSCs have become increasingly popular in the telecommunication and software industries and are widely used for requirements analysis and system design [2, 3, 4, 5, 6].

It is known that an MSC specification \mathcal{M} can contain message interaction patterns that may introduce problems into the systems built from \mathcal{M} . These patterns are referred to as *pathologies* in the MSC research literature [7, 8]. Research has been conducted to investigate the different pathologies in MSCs, such as *race*

[9, 10, 11, 12, 13, 14], *non-local choice* [15, 16, 17], *implied scenarios* [18, 19], *implicit synchronisation* [8], *confluence* [8], *process divergence* [16], *boundedness* [20], *false-underspecification* [7] and *time inconsistency* [9, 21]. Pathologies introduced in the requirement and design stages can cause problems at runtime such as deadlocks, synchronisation problems and communication buffer overflow. Therefore, the analysis of pathologies in MSCs can identify possible defects at early stages of a system development process. For example, it was reported that an MSC static analysis tool specifically designed for detecting race and false-underspecification, MINT, helped Motorola reduce appraisal costs and improve productivity [22] and found errors in approximately one out of five sequence diagrams in an early draft version of MOST [13]. Mooij et al. reported that non-local choices were identified in

Email addresses: `haitao.dan@brunel.ac.uk` (Haitao Dan),
`rob.hierons@brunel.ac.uk` (Robert M. Hierons),
`steve.counsell@brunel.ac.uk` (Steve Counsell)

IEEE Standard 1073.2 [23]. In addition, a well-know MSC analysis tool, LTSA-MSC, has been successfully applied to web service composition [24, 25].

However, there are no formal classifications of pathologies and the relationships between the pathologies and the problems that they caused in the literature. This may bring uncertainties into formal analysis of MSCs. Using our experience as an example, our initial research topic was to tackle problems in distributed testing with MSCs. We found that it was difficult to use existing research results to support our research on distributed testing [26]. For example, from observed problematic test runs, we cannot infer the types of design problem. In addition, we cannot relate an identified controllability problem with an existing type of pathology. Here, a controllability problem occurs in testing from MSCs if the testers may not be able to ensure that the input to the System Under Test (SUT) follows the given test case when applied in a distributed test environment. Examples 2 and 5, in Section 3, further explain controllability problems and how they relate to pathologies.

In this paper, we thus focus on clarifying the relationships between concepts related to pathologies. We focus on MSCs that cause synchronisation problems in part because of our interest in testing and the desire to ensure that the SUT receives the expected sequence of inputs. We argue that race and non-local choice are the two main pathologies that cause synchronisation problems.

A pathology framework shown in Figure 1 is introduced to describe relationships between the two types of pathology and the scenarios with synchronisation problems. In the proposed framework, there are three existing concepts: race, non-local choice and implied scenarios. We also introduce two new concepts: *pathological scenarios* which are possible problematic scenarios that are not described by MSC specifications and *race scenarios* which are a type of pathological scenario caused by race. These two new concepts are circled of the dotted line in Figure 1.

The proposed framework has two layers and relationships are denoted by lines and arrows. At the upper layer, there are two concepts: pathologies and pathological scenarios. Pathologies in an MSC specification \mathcal{M} may lead to pathological scenarios in a system implemented from \mathcal{M} . The “leads to” relationship is represented by an arrow from pathologies to pathological scenarios in Figure 1. At the bottom layer, there are race, non-local choice, race scenarios and implied scenarios. Race and non-local choice are two types of pathology; race scenarios and implied scenarios are two types of pathological scenarios that could result from

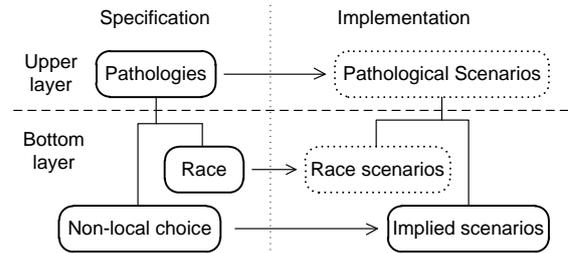


Figure 1: Pathology framework of MSCs

using an MSC specification that has pathologies. The “belongs to” relationship is represented by a line from the upper layer to the bottom layer. All relationships described in the figure are formally proven in the following sections.

Based on the standard semantics given in [1] and the assumption that the underlying communication is asynchronous, non-FIFO and has an individual buffer for each channel, we achieve the following results in this paper:

- Race and non-local choice are the two pathologies that cause scenarios with synchronisation problems. Other identified issues in MSCs can be categorised into semantic errors, inconsistencies and boundedness problems.
- Based on an assumption that we make which we refer to as single undesirable event restriction, we introduce the concept of pathological scenarios. This clarifies the possible problematic scenarios that pathologies may cause. The concept of pathological scenarios reduces the number of problematic scenarios that the software engineer needs to consider. In addition, it enables the categorisation of problematic scenarios caused by pathologies.
- We refine the definition of race in MSC specifications and introduce the concept of race scenarios. We show that races cause race scenarios.
- We show that non-local choices cause implied scenarios and detecting non-local choices and solving implied scenarios are identical problems.
- We prove that race and non-local choice are orthogonal types of pathologies and lead to different type of pathological scenarios.

To conclude, the main contribution of this research is the introduction of an MSC pathology framework. The framework provides a clearer view on difficult issues in MSC research. By understanding such relationships, we

are in a better position to explore the identified pathologies and resultant issues.

The remainder of this paper is organised as follows. In the next section, we give preliminary definitions of MSCs. Section 3 briefly introduces the identified pathologies and the formal definitions of race, non-local choice and implied scenarios. In Section 4, we propose the concepts of pathological scenarios and pathology free and then a taxonomy for pathological scenarios. We then give the relationship between non-local choice and implied scenarios based on our previous work (Section 5) [27]. In Section 6, a new definition of race in MSC specifications is given. We then prove the relationship between race and race scenarios. In Section 7, we compare our race free condition with the traditional one [11] and we discuss how to apply the framework to systems with different types of communication. Finally, Section 8 presents conclusions.

2. Preliminaries

In this section, we give formal definitions of the core concepts of MSCs. For a detailed description of MSCs, the reader is referred to [1].

2.1. bMSCs

The annex B of the MSC standard [28] described two approaches to interpret the meaning of a bMSC: intuitive partial order based semantics and process algebra based operational semantics. The partial order described in a bMSC is also called *visual order* which has been introduced in [9] and this approach was used in most of the research on MSC pathologies [7, 8, 9, 10, 11, 12, 13, 17, 18, 19, 20, 29]. In this paper, we adopt the visual order approach.

Definition 1. (bMSCs) An *bMSC* M is a tuple $\langle E, C, \mathcal{P}, l, msg, < \rangle$ where: E is a set of events, C is a message alphabet and $\mathcal{P} = \bigcup_{i=1}^n P_i$ is a set of processes; E is partitioned into a set S of send events and a set R of receive events, $E = S \cup R$; $l : E \mapsto \mathcal{A}$ is a labelling function and $\mathcal{A} = \mathcal{A}^S \cup \mathcal{A}^R$ where $\mathcal{A}^S = \{send(i, j, a) : 1 \leq i, j \leq n \wedge a \in C\}$ is the set of sending of messages and $\mathcal{A}^R = \{receive(i, j, m) : 1 \leq i, j \leq n \wedge m \in C\}$ is the set of receiving of messages; $send(i, j, m)$ represents the sending of message m from P_i to P_j and $receive(i, j, m)$ the receiving of the corresponding message¹; \mathcal{A}_i represents the set of labels on process P_i ; $msg : S \mapsto R$

¹We will use $!m$ and $?m$ as abbreviations of $send(i, j, m)$ and $receive(i, j, m)$, respectively, where i, j are clear.

is a bijection between send and receive events, matching each send with its corresponding receive; the inverse mapping is $msg^{-1} : R \mapsto S$ between receive and send events; there is a helper mapping $p : E \mapsto [1, n]$ that maps each event $e \in E$ to the index of the process on which e occurs; for each $1 \leq i \leq n$, a total order $<_i$ on the events of process P_i , i.e., on the elements of $p^{-1}(i)$, such that the transitive closure of the relation $< \doteq \bigcup_{1 \leq i \leq n} <_i \cup \{(s, msg(s)) : s \in S\}$ is a partial order on E , namely, visual order ($<^*$). Note that, since $<_i$ is a total order, it is antisymmetric.

In the literature, partial order semantics is often used accompanied with trace semantics [7, 11, 12, 18, 30, 31]. For simplifying trace semantics, two assumptions are generally made. For a message m we restrict $p(!m) \neq p(?m)$ which means that messages sent and received by the same process are not considered. In addition, we assume that the non-degeneracy condition, also called weak-FIFO, is satisfied. Non-degeneracy means that degenerate MSCs are not allowed. An MSC is degenerate if two send events $e1$ and $e2$ exist such that $l(e1) = l(e2)$, $e1 < e2$ and $msg(e2) < msg(e1)$.

The trace semantics can be described as follows. Given a bMSC M , a *word* of M is a string $w = w_1 \cdots w_{|E|}$ over \mathcal{A} if and only if there exists a total order $e_1 \cdots e_{|E|}$ of the events in E such that whenever $e_i <^* e_j$ we have $i < j$, and for $1 \leq i \leq |E|$, $w_i = l(e_i)$. Conversely, a well-formed and complete word uniquely characterises a bMSC. A word is *well-formed* if all of its receive events have earlier matching sends and a word w is *complete* if all send events have matching receives. A bMSC M describes a set of well-formed and complete words and the set of words is the language of the bMSC, denoted as $L(M)$. We use $pref(w)$ and $pref(L)$ to denote the set of prefixes of word w and the set of prefixes of language L , respectively.

2.2. MSC specifications

There are different approaches to organising MSC specifications. The most popular ones used in discussing pathologies are High-level MSCs (HMSCs) [8, 16, 19, 20, 21, 23, 29] and MSC Graph (MSG) [11, 17, 20, 32, 31]. Generally speaking, MSGs are a simpler version of HMSCs. Both are formalised to a graph with nodes labelled by MSCs. The differences are that HMSCs support parallel composition and allow their nodes to be labelled by other HMSCs. In fact, a number of research studies based on HMSCs do not consider parallel composition [16, 19, 20, 21, 23, 29]. The semantics of an HMSC or an MSG depends heavily on how to interpret the concatenation of two bMSCs

where weak sequencing [33] was adopted by most of the research on pathologies [7, 8, 11, 16, 20, 21, 32, 31]. Weak sequencing means that the individual processes of bMSCs are concatenated and there is no synchronisation at concatenation points between bMSCs. Therefore, the additional visual orders in the synthesised bMSC are only formed by the event pairs in which one event from the first bMSC and the other from the second bMSC and both events happen on the same process. Consequently, in a weak sequencing of two bMSCs M_1 and M_2 , events from M_2 may be executed before events from M_1 . The semantics of an HMSC or MSG is given by the weak sequencing of the nodes on the paths of the HMSC/MSG and this leads to a set of bMSCs. We note that the set of bMSCs can be infinite due to the loop structure in the graph. In addition, the language represented by an MSC specification can be non-regular even without considering parallel composition in HMSCs [32, 20, 34].

The other approach to describe an MSC specification is by directly considering a specification as a set of bMSCs [18, 30]. In this paper, we follow this approach. It is clear that an HMSC/MSG defines a set of bMSCs. Therefore, the results in this paper can be applied to MSC specifications described by HMSCs and MSGs. We define the semantics of an MSC specification to be the language of the specification.

Definition 2. (Language of MSC Specifications) For an MSC specification \mathcal{M} , the language of \mathcal{M} , $L(\mathcal{M})$, is the union of the sets of words of all bMSCs in \mathcal{M} , $L(\mathcal{M}) = \bigcup_{M \in \mathcal{M}} L(M)$. $L(\mathcal{M})$ is regarded as the semantics of \mathcal{M} .

Accordingly, we say that bMSC M is a *member bMSC* of \mathcal{M} if $L(M) \subseteq L(\mathcal{M})$.

In a bMSC, the behaviour of a process is modelled by a totally ordered sequence of events corresponding to a single word on labels. For an MSC specification, behaviours of process P_i are captured by the projection of the language $L(\mathcal{M})$ on P_i , denoted as $L(\mathcal{M})|P_i$, namely the process language.

3. Pathologies in MSCs

In this section, we briefly introduce pathologies identified in the literature. We categorise these pathologies into four types. The first three types do not cause synchronisation problems but the last one does. We show that different groups of researchers have considered different aspects of the same pathologies. We then focus on race, non-local choice and implied scenarios, which

are concepts related to synchronisation problems, and give formal definitions to these concepts.

3.1. Categorisation of identified pathologies

Existing research on MSC pathologies has been conducted by different researchers: once a problematic message interaction pattern has been identified, it is tackled. However, the relationships between the types of pathology have not been formally analysed. In fact, the types of pathology previously mentioned in Section 1 can be categorised by the types of problems that they may cause.

Semantic errors: implicit synchronisation and confluence [8]. Both are caused by misunderstandings of the semantics of specific constructs of MSCs and lead to wrong specifications because of the differences between semantics and designers' intention.

Inconsistencies: time inconsistency [9, 21] and false-under specifications [7]. Time inconsistencies are conflicts between the visual orders and time constraints or between different sets of time constraints; false-under specifications are conflicts between visual orders and general orders manually added by designers.

Boundedness problems: divergence [16], model checking problem [20, 34], synthesis problem [19, 34, 35, 36, 37, 30] and non-regularity [32]. Although they are slightly different problems, we group them together since an unbounded communication pattern in a loop structure in HMSCs causes all these problems. An unbounded communication pattern may lead to a situation in which there is no bound on the number of messages that have been sent but not yet received. For the research on divergence, this results in problems at runtime since a network will have a bound on its capacity. From the model checking problem point of view, this communication pattern makes the model checking problem for HMSCs undecidable. The synthesis problem generally is to determine whether an MSC specification implements a state-based model. An unbounded pattern makes this problem undecidable. Finally, it was also shown that an HMSC represents a non-regular language if there is an unbounded communication pattern.

Synchronisation problems: races [9, 10, 11, 12], non-local choices [15, 16, 17] and implied scenarios

[18, 19, 29]. Race and non-local choice lead to undesirable scenarios that violate the order of events described in an MSC specification. It was shown that they are hard to tackle and cause runtime problems in real implementations. For the three issues, the literature contains apparently contradictory statements and some concepts overlap. For example, it has been said that “an implied scenario is due to a non-local choice situation” [29] and also that “non-local choices are implied scenarios; the converse is not the case” [19]. A second example is that, as shown later, an MSC specification that is not race free under the definition given in [11] may cause implied scenarios but not race scenarios. In addition, there are multiple non-local choice definitions and they are not equivalent [15, 16, 27]. Ladkin and Leue first identified non-local choice in [15], but no formal definition was given. The idea of non-local choice was then refined with both semantic and syntactic definitions in [16]. These definitions depend on the explicit branch structure in HMSCs. In [27], non-local choice was defined as a sub-type of choice on a process so that the branch structure in the HMSC is no longer required for the existence of non-local choice in an MSC specification. Therefore, it is a more general formalisation of non-local choice. One way of simplifying the situation is to adopt a unified approach to conducting research on MSC pathologies. This should clarify relationships between concepts in MSC related research such as race, non-local choice and implied scenarios and explain apparently contradictory statements and observations in the literature.

3.2. Concepts related to synchronisation problems

In this section, three complex issues in MSCs are introduced: race, non-local choice and implied scenarios.

Race

Race was originally defined as a discrepancy between two orders: visual order and enforced order [9]. Visual order is defined by the semantics, but enforced order may change over implementations since enforced order depends on causalities described by for example, the MSC specification or the underlying communication system [9, 38].

For the sake of simplicity, race will be investigated based on basic enforced orders. This means that enforced orders are introduced by only two types of causality following the *happened before* relation given

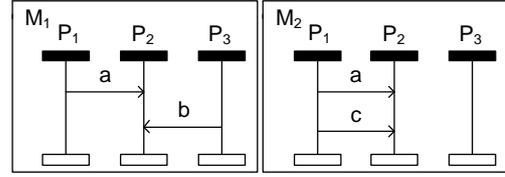


Figure 2: Race

in [39]: the sending of a message always being before the receiving of the same message and a send event on some process always being after the events visually above it on the same process. The second type of causality is based on an assumption that subsystems follow their local behaviour specifications which are process languages in terms of MSC specifications. We will call this the *sensible local behaviour assumption*. The motivation is that MSC specifications concentrate on synchronisation between subsystems. To analyse synchronisation problems, it is better to isolate them from local system errors. Actually, let us assume that communication between processes is asynchronous and non-FIFO and each channel has an independent buffer that is unbounded as in [9, 11, 12], no order will be further enforced or reduced between events in bMSCs by the underlying communication system.

The enforced order of a bMSC is defined as follows:

Definition 3. (Enforced Order) Given a bMSC M with set E of events, the transitive closure of the relation, $\ll \doteq \{(x, y) \in E : y = \text{msg}(x) \vee (y \in S \wedge p(x) = p(y))\}$, is a partial order on E , namely the *enforced order*.

Therefore, race and race free in a bMSC are defined as follows.

Definition 4. (Race in a bMSC) Events e and f from process P in bMSC M are said to be in a *race*, denoted as $[e, f]$, if $e <^* f$ but *not* $e \ll^* f$. We say that M is *race free*, if M contains no races.

Example 1. (Race) Consider MSC M_1 given in Figure 2. Processes P_1 and P_3 should send messages a and b respectively to process P_2 . According to M_1 , message a should arrive before message b . However, it is possible that P_1 and P_3 send their messages as desired but b arrives at P_2 before a in an execution. According to the definition, there is a race $[?a, ?b]$ in bMSC M_1 since $?a <^* ?b$ but $?a \not\ll^* ?b$. Another example is given in M_2 shown in the same figure. In this example, two messages are sent from P_1 . As described in M_2 , a should

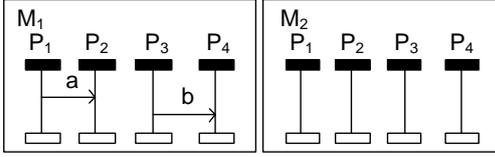


Figure 3: Choice with terminations

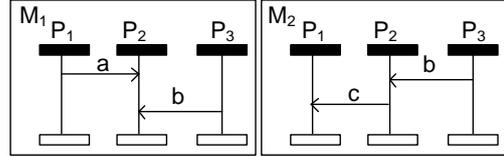


Figure 4: Non-local choice

arrive before c . The problem is that c may overtake a in a real run. There is a race $[?a, ?c]$ since $?a <^* ?c$ but $?a \not\ll^* ?c$ in the second example.

Races may cause different types of problems in a software engineering process. In the context of testing with MSCs, a race may lead to controllability problems of timing [26].

Example 2. (Race and testing) Let us suppose that M_1 in Figure 2 describes a test case where P_1 and P_3 in the bMSC are two testers and P_2 is the SUT. Race $[?a, ?b]$ implies that there are controllability problems for testers P_1 and P_3 since they cannot decide when to send messages a and b to ensure that the two messages arrive at the SUT as shown in M_1 .

Non-local choice

Non-local choice can be described as a choice that depends on information from other processes, but the information is not accessible due to the *local* assumption. Here, the local assumption is that a process is prevented from directly accessing the status of other processes. Generally, a choice is between two event labels. There is a special case that a choice may happen between the termination of a process and an event label.

Example 3. (Choice with a termination of process) Consider MSC specification M_1 with two member MSCs M_1 and M_2 shown in Figure 3. Both processes P_1 and P_3 have a choice as to whether to send a message or terminate the processes directly.

To explicitly describe choices between the termination of a process and other events, we include terminations into the alphabet of MSC languages [27] and so L' is used to denote the MSC language extended with an alphabet of terminations of processes, represented by $\downarrow_{\mathcal{P}} = \{\downarrow_i : P_i \in \mathcal{P}\}$.

In this paper, we define choice on local behaviours. A choice represents the alternative next behaviours that can be chosen for a process after some execution steps if the MSC specification contains multiple member bMSCs. The formal definition of choice is as follows.

Definition 5. (Choice [27]) Given an MSC specification \mathcal{M} , a *choice* on P_i is a triple (w, x, y) , where $w \in \text{pref}(L'(\mathcal{M})|P_i)$, $x, y \in \mathcal{A}_i \cup \downarrow_i$ and $x \neq y$ such that $wx, wy \in \text{pref}(L'(\mathcal{M})|P_i)$.

Non-local choice is thus a subtype of choice. The formal definition of non-local choices based on Definition 5 is as follows.

Definition 6. (Non-local Choice [27]) Given an MSC specification \mathcal{M} , a *non-local choice* is a choice (w, x, y) on P_i , such that there exists a word $v \in \text{pref}(L'(\mathcal{M}))$, where $v|P_i = w$, $vx \in \text{pref}(L'(\mathcal{M}))$, vy is well-formed and $vy \notin \text{pref}(L'(\mathcal{M}))$. We say that \mathcal{M} is *non-local choice free*, if \mathcal{M} contains no non-local choices.

Example 4. (Non-local choice) Consider an MSC specification M_2 with two bMSCs M_1 and M_2 shown in Figure 4. There is a non-local choice $(\varepsilon, ?c, !a)$ on P_1 since we can find $!b?b!c?c \in \text{pref}(L(M_1))$ but $!b?b!c!a \notin \text{pref}(L(M_2))$.

A correctly implemented distributed system is based on properly implemented subsystems. However, the non-local choice pathology implies that it is not sufficient to have correctly implemented subsystems since these can lead to additional behaviours that are not in the specification. In the context of testing from MSCs, non-local choice causes controllability problems of choice [26].

Example 5. (Non-local choice and testing) Let us suppose that MSC specification M_2 describes a test case where P_1 and P_3 are two testers and P_2 is the SUT. Non-local choice $(\varepsilon, ?c, !a)$ leads to a controllability problem regarding whether tester P_1 should send message a or wait for the incoming message c .

Implied scenarios

The concept of implied scenario was defined by Alur et al. [18]. According to [18, 19], the synthesised model from an MSC specification may exhibit additional behaviours which are not described by the original specification even though each process follows its local specification. In addition, an MSC specification \mathcal{M} is said to

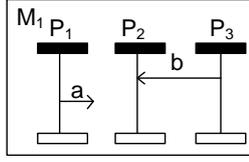


Figure 5: Implied scenario of $\mathcal{M}2$

be *safely realisable* if and only if there exists a synthesised model whose behaviour contains no implied scenarios which may be in deadlock [18].

Example 6. (Implied Scenarios) For example, reconsider MSC specification $\mathcal{M}2$ with the two bMSCs M_1 and M_2 shown in Figure 4, $\mathcal{M}2$ leads to an implied scenario M_1 given in Figure 5, because the sending of a and receiving of b are consistent with the process behaviours, but the overall behaviour violates $\mathcal{M}2$.² Note that M_1 in Figure 5 also represents a word in which the sending of a happens before the receiving of b . In this case, it is still an implied scenario.

The additional behaviours shown in the example are implied scenarios which can be formalised as follows.

Definition 7. (Implied Scenarios [27]) w represents an *implied scenario* of MSC specification \mathcal{M} if w is a well-formed word and for each $w|P_i$ $i \in [n]$, a word $v \in \text{pref}(L'(\mathcal{M}))$ exists such that $w|P_i = v|P_i$, but $w \notin \text{pref}(L'(\mathcal{M}))$.

Definition 7 is from a previous study by the authors [27] and is consistent with the concept of safe realisability of Alur et al. [18, 32]. It is worth noting that the introduction of the termination alphabet helps to capture a special type of implied scenario.

Example 7. (Implied Scenario with termination alphabet) For example, reconsider MSC specification $\mathcal{M}1$ with two bMSCs M_1 and M_2 shown in Figure 3. $\mathcal{M}2$ can then be described as word $\downarrow_1 \downarrow_2 \downarrow_3 \downarrow_4$. M_1 in Figure 6 can be described as $!a?a \downarrow_3 \downarrow_4 \downarrow_1 \downarrow_2$ and is an implied scenario of $\mathcal{M}1$ since each projection of M_1 in Figure 6 on a process follows the corresponding process behaviour of $\mathcal{M}1$.

Definition 7 considers prefixes of $L'(\mathcal{M})$ because the implied scenarios may lead to deadlock [18].

²The half arrows shown in Figure 5 and other figures in this paper represent the sending of messages. We note that this is not a standard construct but it is an ideal way to visualise the prefix of a trace which is often used in the literature.

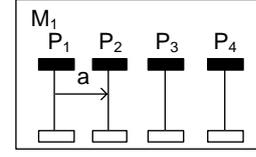


Figure 6: Implied scenario of $\mathcal{M}1$

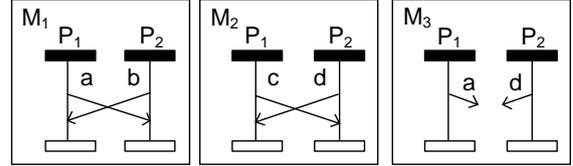


Figure 7: Implied scenario with deadlock

Example 8. (Deadlock) For example, consider MSC specification $\mathcal{M}3$ with member bMSCs M_1 and M_2 shown in Figure 7. They imply a new scenario M_3 in which the two processes start by sending each other conflicting messages leading to the scenario of deadlock.

Definition 7 is different from the implied scenario in [19] in which synchronous communication is applied. The implied scenarios with deadlock given in the example do not happen with synchronous communication.

In addition, it has been shown that checking weak realisability (without considering the possibility of deadlock) is even more difficult than checking safe realisability [18].

4. Pathological scenarios

Pathologies in MSC specifications are described as defects in MSC specifications [7]. In this section, we introduce a new concept: *pathological scenarios*. Pathological scenarios are problematic behaviours of implemented systems caused by ‘defects’ in MSC specifications with the single undesirable event restriction. It is worth noting that the relationship between pathologies and pathological scenarios is similar to the concepts ‘fault’ and ‘failure’ in software testing.

Intuitively, we may think that an observed scenario of an implementation from an MSC specification is problematic if it is not consistent with the specification. Formally, problematic scenarios are defined as follows.

Definition 8. (Problematic scenarios) Let us use a word w and \mathcal{M} to represent the observed scenario and the MSC specification, w is a problematic scenario if $w \notin \text{pref}(L(\mathcal{M}))$. An observed scenario w *violates* specification \mathcal{M} if w is a problematic scenario.

A distributed system may continue to run until all subsystems realise that a problem has occurred. There is a possibility that no subsystem can realise a problem has occurred since the problematic execution can be correct for each subsystem locally. This means that a problematic scenario may contain multiple undesirable events. If these undesirable events are caused by different pathologies and interleave each other, two issues may be introduced into the software engineering process. First, engineers may have to spend more time on tracing back the design problems. Second, taxonomies on both problematic scenarios and pathologies cannot be formally introduced. This may bring complexities into develop a tool to accurately find design problems in specifications or from observed scenarios. Examples are given for further explanations after the introduction of definitions.

A better solution is to consider the prefixes of problematic scenarios which contain only one undesirable event violating the specification. We call the problematic scenarios with this restriction *pathological scenarios*. The restriction is referred to as *single undesirable event restriction*.

The formal definition of a pathological scenario is given as follows.

Definition 9. (Pathological Scenario) wa , where $a = l(f)$ and f is an event on P_i , represents a *pathological scenario* of \mathcal{M} , if $w \in \text{pref}(L(\mathcal{M}))$ such that wa is well-formed and $wa \notin \text{pref}(L(\mathcal{M}))$.

In this definition, wa should be well-formed since a pathological scenario is a problematic scenario. The word representing a problematic scenario should be a possible observation of a real run. Formally, let S be the set of problematic scenarios. The set of pathological scenarios, S' , is thus a subset of S . For each scenario $s \in S$, an $s' \in S'$ can always be found such that $s = s'x$ where x is a word on the given alphabet. Thus, crucially, the restriction to an undesirable event has the property that for a given MSC specification S is empty *if and only if* S' is empty.

In this paper, we investigate pathologies that eventually lead to pathological scenarios. Therefore, we say that an MSC specification is pathology free if an implementation of an MSC specification cannot cause pathological scenarios under the sensible local behaviour assumption. We thus have the following definition.

Definition 10. (Pathology Free) An MSC specification \mathcal{M} is pathology free if for all $w \in \text{pref}(L(\mathcal{M}))$ and $f \in E$ such that word wa where $a = l(f)$ is well-formed

and satisfies the sensible local behaviour assumption, we have that $wa \in \text{pref}(L(\mathcal{M}))$.

This definition states that any possible successive event of word w according to the local behaviour specifications should form a valid prefix of the specification with w . Consequently, we have the following proposition, since pathology free is based on pathological scenarios.

Proposition 1. *Under the the sensible local behaviour assumption, an MSC specification \mathcal{M} is pathology free if and only if \mathcal{M} has no pathological scenarios.*

PROOF. This proposition follows from Definitions 9 and 10. \square

In devising a formal taxonomy on pathological scenarios, the property of event label a in a pathological scenario is important since this is the only difference between a pathological scenario and the valid prefix of a scenario. One obvious property of a is that it may represent the sending or receiving of a message. The other property is that, suppose that a happens on process P_i , it may or may not violate the local behaviour $L(\mathcal{M})|P_i$. Based on the two properties, pathological scenarios can be divided into four categories.

1. $a \in \mathcal{A}^R$ and $wa|P_i \in \text{pref}(L(\mathcal{M})|P_i)$;
2. $a \in \mathcal{A}^S$ and $wa|P_i \in \text{pref}(L(\mathcal{M})|P_i)$;
3. $a \in \mathcal{A}^R$ and $wa|P_i \notin \text{pref}(L(\mathcal{M})|P_i)$;
4. $a \in \mathcal{A}^S$ and $wa|P_i \notin \text{pref}(L(\mathcal{M})|P_i)$.

The fourth category of problematic scenarios are actually caused by implementation problems in subsystems since it requires an unspecified send event. Generally, the sensible local behaviour assumption should be satisfied when discussing coordination between subsystems based on MSC specifications as explained in Section 3.2. With the sensible local behaviour assumption, the fourth category of pathological scenarios is invalid.

Comparing the first two categories of pathological scenarios and Definition 7, we have the following proposition.

Proposition 2. *If w represents a pathological scenario of MSC specification \mathcal{M} and for each projection of w , $w|P_i \in \text{pref}(L(\mathcal{M})|P_i)$ then w represents an implied scenario.*

PROOF. This proposition follows from Definition 7 and Definition 9. \square

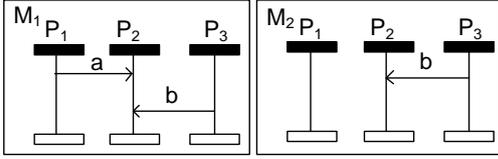


Figure 8: Race scenario

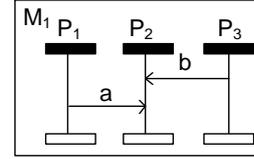


Figure 9: Race scenario or implied scenario

However, by definition, not all implied scenarios are pathological scenarios, since implied scenarios may contain more than one event that violates the specification.

The third category of pathological scenarios says a is a receive event and the local behaviour is violated by a . Actually, this means that a overtakes some events since wa is well-formed. This is consistent with the informal description of the scenarios caused by race in [9]. Here, we give a formal definition of scenarios caused by races.

Definition 11. (Race Scenario) wa , where $a = l(f)$ and f is an event on P_i , represents a *race scenario* of \mathcal{M} if wa is well-formed, $w \in \text{pref}(L(\mathcal{M}))$, $f \in R$ and $wa \mid P_i \notin \text{pref}(L(\mathcal{M})|P_i)$.

Example 9. (Race Scenario) Consider an MSC specification \mathcal{M}_4 with only one bMSC M_1 shown in Figure 8. This MSC specification leads to race scenarios. According to our definition, M_2 given in the same figure is a race scenario of \mathcal{M}_4 since the receiving of b violates the specification.

We note that a race scenario of a member bMSC may be a valid prefix of the specification. More interestingly, a race scenario of a member bMSC can be an implied scenario of the specification.

Example 10. (Race scenario or implied scenario) For MSC specification \mathcal{M}_4 with only one member bMSC M_1 given in Figure 8, M_2 in the same figure is a race scenario caused by the race in M_1 . For an MSC specification \mathcal{M}_2 with two bMSCs M_1 and M_2 shown in Figure 4, although M_1 of \mathcal{M}_2 is the same as M_1 of \mathcal{M}_4 , M_2 in Figure 8 is no longer a pathological scenario of \mathcal{M}_2 in Figure 4, because it is a prefix of M_2 in Figure 4. However, \mathcal{M}_2 is not pathology free since it leads to an implied scenario M_4 in Figure 5.

In the example, M_2 in Figure 8 and its extension M_4 in Figure 5 can be valid scenarios, race scenarios or implied scenarios in terms of different MSC specifications. However, one word cannot represent both a race scenario and an implied scenario of a specific MSC specification. Comparing definitions of pathological scenario,

race scenario and implied scenario, we have the following proposition.

Proposition 3. *The set of implied scenarios IS with only one undesirable event and the race scenarios RS of MSC specification \mathcal{M} describe different behaviours, $L(IS) \cap L(RS) = \emptyset$; the set of pathological scenarios PS of \mathcal{M} equals $RS \cup IS$.*

PROOF. For the first part of the proposition, according to the definition of race scenarios, race scenarios always have a process behaviour on some P_i that is inconsistent with the process language defined by \mathcal{M} . According to the definition of implied scenarios, an implied scenario has all process behaviours conforming to the process languages of specification \mathcal{M} . Consequently, an implied scenario and a race scenario cannot be described by the same word.

The second part of the proposition follows the definition of pathological scenario, race scenario and implied scenario with the sensible local behaviour assumption. \square

Since race scenarios and a taxonomy on pathological scenarios have been introduced based on the single undesirable event restriction, we can give examples to illustrate conversely how the single undesirable event restriction helps to clarify the relationship between pathological, race and implied scenarios.

Example 11. (Single undesirable event restriction and problematic scenario 1) Let us consider problematic scenario M_1 shown in Figure 9. For \mathcal{M}_4 with only M_1 given in Figure 8, this may be considered to be a race scenario if we do not restrict attention to cases involving a single undesirable event; in fact, it is just an extension of a race scenario M_2 in Figure 8 since the receiving of b already violates the behaviour of P_2 .

For \mathcal{M}_2 with two bMSCs given in Figure 4, it is difficult to decide whether M_1 in Figure 9 is a race scenario or an implied scenario if we do not use the single undesirable event restriction. It can be a race scenario, since the behaviour on P_2 violates the specification. However,

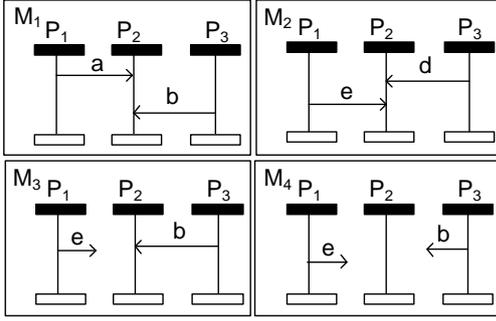


Figure 10: Single undesirable event restriction

it can also be an implied scenario, because M_1 in Figure 9 is a complete version of M_1 in Figure 5 which is already an implied scenario.

Example 12. (Single undesirable event restriction and problematic scenario 2) More interestingly, consider an MSC specification $M5$ with bMSC M_1 and M_2 shown in Figure 10. M_3 in the same figure obviously describes problematic scenarios with multiple undesirable events. Both events $!e$ and $?b$ are undesirable. Since there is no visual order defined between $!e$ and $?b$ in M_3 , they can show up in either way. In other words, both $!b!e$ and $!b?b$ can be prefixes of the scenarios described in M_3 . If the observed scenario is $!b!e$ which can be visualised as bMSC M_4 in the same figure, $!b!e$ is an implied scenario since all its projections satisfy the local behaviour specifications. If the observed scenario is $!b?b$ in which $?b$ comes first as bMSC M_2 given in Figure 8, it is a race scenario since the behaviour on P_2 is not described in specification $M5$.

The last two examples show that if we consider multiple undesirable events in a pathological scenario, multiple types of violations may exist at the same time. Pathological scenarios therefore cannot be uniquely classified.

There is another benefit from introducing the single undesirable event restriction: by considering only one event, (potentially exponentially) fewer scenarios need to be investigated when trying to compute the set of pathological scenarios of an MSC specification.

Example 13. (Problematic scenario explosion) Consider bMSC M_1 in Figure 11 in which there are n messages sent from P_1 to P_2 and n messages sent from P_3 to P_2 . There are $\frac{2n!}{2!(2n-2)!} = n(2n-1)$ races in M_1 but $2n!/2 - 1$ problematic scenarios may be observed if we consider that every scenario contains all the events in M_1 .

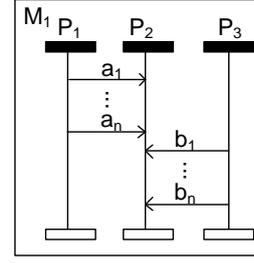


Figure 11: Exponentially many pathological scenarios

If such a restriction is not used then there may be little value in returning the set of problematic scenarios to the software engineer: there may be too many such scenarios and most may be formed from different combinations of a set of potential synchronisation problems. If the number of pathological scenarios is not too large then this set might be used to show what synchronisation problems might occur when developing an implementation from the given MSC specification and it might also be used as the basis of a test suite that aims to determine whether any of these synchronisation problems have occurred in the actual implementation.

5. Non-local choice and implied scenarios

In this section, we explore the relationships between non-local choice and implied scenarios. According to Alur et al. [18], the following two Closure Conditions (CCs) can be used to determine whether an MSC specification contains implied scenarios.³ We note that the two CCs were defined on the alphabet without terminations.

CC 1. For all $w, v \in \text{pref}(L(\mathcal{M}))$, if $w|P_i = v|P_i$ for some process P_i , $wx \in \text{pref}(L(\mathcal{M}))$ and vx is well-formed for some x on P_i , then vx is in $\text{pref}(L(\mathcal{M}))$.

CC 1 considers two possible prefixes of specification \mathcal{M} , w, v . If process P_i cannot distinguish w and v from its local view and x on P_i is a valid continuation of w but not for v , then vx is an implied scenario.

CC 2. Given a well-formed and complete word $w \in \text{pref}(L(\mathcal{M}))$, if for all P_i , $1 \leq i \leq n$, there exists a word $v^i \in L(\mathcal{M})$ such that $w|P_i = v^i|P_i$, then w is in $L(\mathcal{M})$.

CC 2 means that a prefix w of a specification language, $L(\mathcal{M})$, is an implied scenario if w is not a word

³The two closure conditions were named CC2' and CC3' in [18].

of $L(\mathcal{M})$ and for any projection of w , a word of $L(\mathcal{M})$ can be found with an equal projection. It is a complement to CC 1 for capturing a specific type of implied scenarios where some processes of such an implied scenario terminated earlier than expected.

Example 14. (Implied scenarios captured by CC 2)

Consider again the MSC specification $\mathcal{M}1$ with two member bMSCs given in Figure 3. As shown in the example after Definition 7, scenario M_1 from Figure 6 is an implied scenario of $\mathcal{M}1$ and M_1 can be represented by the word $!a?a$. If we do not include symbols representing terminations into the alphabet then M_1 cannot be classified as an implied scenario on the basis of CC1 since $!a?a$ is a prefix of a word in $L(\mathcal{M}1)$. However, $!a?a$ is a prefix of a word in $L(\mathcal{M}1)$ but is not in $L(\mathcal{M}1)$ and so is an implied scenario. This type of implied scenario is captured by condition CC2.

As mentioned in Section 3.1, there are contradictory statements on the relationship between non-local choice and implied scenarios in the literature. The inconsistency between [29] and [19] has been pointed out in [17], but it does not provide a formal analysis of this topic. Finally, in a previous paper [27], we showed that Alur’s CCs for safe realisability captures all non-local choices. This is stated in the following two propositions proven in [27].

Proposition 4. *Given an MSC specification \mathcal{M} , a language $L(\mathcal{M})$ is safely realisable if and only if \mathcal{M} is non-local choice-free.*

Proposition 5. *If an implied scenario can be caused by an MSC specification, then it corresponds to a non-local choice in that specification.*

Based on Propositions 4 and 5, it is easy to see that non-local choices in MSC specifications lead to implied scenarios. This is stated in the following proposition.

Proposition 6. *An MSC specification is non-local choice free if and only if it does not lead to implied scenarios.*

PROOF. This proposition follows Proposition 4 since safely realisable means \mathcal{M} leads to no implied scenario. \square

6. Race and race scenarios

In this section, we discuss the relationship between race and race scenarios in MSC specifications. In Alur

et al. [9], race was defined based on a bMSC. However, there are issues if we reuse the original definition for race in MSC specifications. First, the races of an MSC specification are not the union of races in the member bMSCs. Second, race should be a category of pathologies which, ideally, cause a specific type of pathological scenario. Therefore, to prove the relationship between race and race scenarios, a definition of race in MSC specifications is necessary. We thus first develop a race free condition based on the concept of race scenarios of bMSC. Based on the enhanced condition, new definitions of race and race free in MSC specifications can thus be derived.

6.1. Race in bMSC

In addition to Definition 4, there are two other conditions for race free of bMSC in [9, 10], respectively. They are stated in Propositions 7 and 8 as properties of the definition of race free given in Definition 4. The enforced order \ll^* of bMSC M corresponds to a set of words on the same alphabet as $L(M)$. We denote the language corresponding to \ll^* of M as $L^{en}(M)$.

Proposition 7. *A bMSC M is race free if and only if $L(M) = L^{en}(M)$.*

PROOF. This proposition follows Definition 4. This is because M is race free if and only if M contains no races and this corresponds to the visual order being equal to the enforced order in M . \square

Proposition 8 was given as a definition of race free to show how to detect races in a bMSC using only a part of the bMSC [10].

Proposition 8. *A bMSC M with alphabet \mathcal{A} is race free if and only if for every event x and message m , such that $x, !m, ?m \in E$,*

$$x <^* ?m \Rightarrow (x <^* !m \vee x = !m).^4$$

PROOF. To prove this proposition, we will use Proposition 9.

For the forward direction, we use proof by contradiction. Let us suppose that M is race free but there is an x such that $x <^* ?m$, $x \neq !m$, and $\neg(x <^* !m)$. According to Proposition 9, we have $\neg(x \ll^* !m)$ and so $\neg(x \ll^* ?m)$. This means that $L(M) \neq L^{en}(M)$ and this contradicts the assumption. The forward direction is established.

⁴Here, $!m$ is not an event, but the label of an event. For simplicity, we use labels with events when there is a bijection between the specific sets of labels and events.

For the converse direction, let us suppose that for every event x in M , $x <^* ?m \Rightarrow (x <^* !m \vee x = !m)$. We show that M is race free. Obviously, race is formed by a pair of events where the second one is a receive event. Let x and $?m$ be a such pair of events. $x <^* ?m \Rightarrow (x <^* !m \vee x = !m)$ means that, for any such pair of events, we have $x <^* !m$. According to Proposition 9, we also have $x \ll^* !m$. Therefore, $x \ll^* ?m$ and the converse direction is established. \square

Proposition 9. *Let e, f be two events in a bMSC such that $f \in S$ and $p(e) \neq p(f)$. $e <^* f$ if and only if $e \ll^* f$.*

PROOF. For the forward direction, let us suppose that $p(e) = i$ and $p(f) = j$. The visual order between different processes is only introduced by message event pairs (the sending and receiving of messages) and the orders between sending and receiving of messages are enforced orders. We can always find a set of messages passing the enforced order from P_i to P_j visually after e and before f . Moreover, because f is a send event, according to the definition of enforced order, $e \ll^* f$.

The converse direction follows immediately since the enforced order is a subset of the visual order. \square

Considering the race definition of a bMSC with the framework introduced in Section 4, the following relationship exists between race and race scenario.

Proposition 10. *A bMSC M is race free if and only if it leads to no race scenarios.*

PROOF. For the forward direction, we will use proof by contradiction. Let us assume that M is a race free bMSC. Assume also that we can find a race scenario M' which can be derived from M . According to Definition 11, there is a word wa , where $a = l(f)$ which represents M' , such that f is in some process P_i , $f \in R$, wa is well-formed and $w \in \text{pref}(L(M))$ but $wa|P_i \notin \text{pref}(L(M))|P_i$. This means that the next event on P_i following $w|P_i$ cannot be f . Since w is well-formed, f must happen in M and there is a sequence of events $\alpha \neq \epsilon$ between the last event of $w|P_i$ and f . Let us suppose that the first event in α is e . Obviously, e corresponds to a letter after w . Since wa is well-formed, the letter corresponding to $\text{msg}^{-1}(f)$ is in w . Therefore, $\neg(e <^* \text{msg}^{-1}(f))$. This means that $\neg(e \ll^* f)$ according to Proposition 9. Therefore, $[e, f]$ forms a race in M . This contradicts the assumption that M is race free. The forward direction is established.

For the converse direction, we will also use proof by contradiction. Assume that M does not lead to race

scenarios, but there is a race $[e, f]$ on process P_i of M . We thus construct a new bMSC M'' from M by placing f before e and removing all events x such that $\neg(x <^* f)$. It is clear that for each process P_j such that $j \neq i$, $L(M'')|P_j \in \text{pref}(L(M))|P_j$ and also that $L(M'')|P_i \notin \text{pref}(L(M))|P_i$. M'' is thus a race scenario. This gives a contradiction as required. \square

This proposition means that, to avoid race scenarios, for a bMSC it is sufficient to remove all races. This is the motivation for race detection at the requirements stage. Moreover, it shows that a race scenario of a bMSC can always be constructed from a detected race. The technique for constructing race scenarios is called *Race Scenario Construction* (RSC) in this paper. Let us suppose that $[e, f]$ is a race on process P of bMSC M ; RSC constructs a race scenario M' by placing f before e and removing all events x such that $\neg(x <^* f)$. We call M' the race scenario of $[e, f]$.

Example 15. (RSC) For example, we can construct a race scenario from bMSC M_1 shown in Figure 8 in which $[?a, ?b]$ is a race on P_2 . First, $?b$ is placed before $?a$, and scenario M_1 in Figure 9 is constructed; second, all the events $x : \neg(x <^* ?b)$ in M_1 of Figure 9 are removed; the resultant M_2 in Figure 8 is the race scenario of $[?a, ?b]$.

6.2. Race in MSC specifications

We have seen that in a bMSC, race and race scenarios are related as stated in Proposition 10. If we follow the original definition of race in [9] and define the the union of races in each member bMSC as races of MSC specifications, the relationship between race and race scenarios will not hold in some MSC specifications because not every race in a member bMSC leads to a race scenario.

Example 16. (Race scenarios or not) For example, consider MSC specification M_6 with three member bMSC M_1, M_2 and M_3 shown in Figure 12. M_4 and M_5 are race scenarios of bMSC M_1 , but they are prefixes of member bMSCs M_2 and M_3 . Therefore, M_4 and M_5 are not race scenarios of M_6 . M_6 is a race scenario of M_2 and is not a prefix of any member bMSC, but it is still not a race scenario of M_6 . This is because each projection of M_6 is a prefix of the corresponding process language of M_6 and so M_6 is an implied scenario of M_6 .

This also shows that a race scenario of a member bMSC can be another (partial) member bMSC or an implied scenario of the MSC specification. Formally, this is stated in the following proposition.

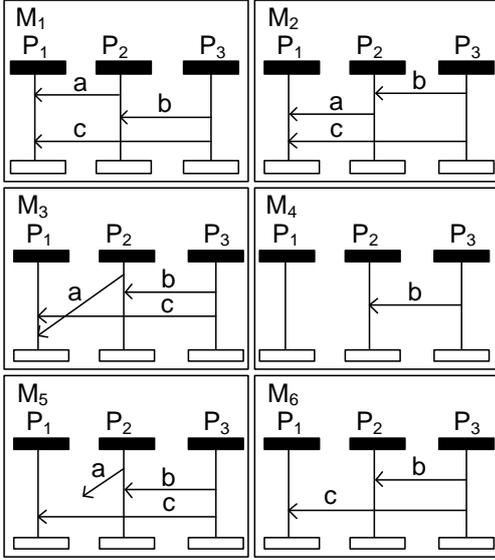


Figure 12: Race scenarios or not

Proposition 11. *If M' is a race scenario of a member bMSC M of specification M , but is not a race scenario of M , then either $L(M') \subseteq \text{pref}(L(M))$ or M' is an implied scenario.*

PROOF. Let us suppose that e, f are two events on process P such that M can be represented by a word $wau'b$, where $a = l(e)$, $b = l(f)$ and $[e, f]$ is race in M ; M' can be represented by word wb and w, u are two words on alphabet \mathcal{A} of specification M . To prove the proposition, if wb is not a race scenario of M , it is sufficient to show that either $wb \in \text{pref}(L(M))$ or is an implied scenario of M . This follows from Definitions 2, 7 and 11. \square

To develop a race definition of MSC specifications that causes pathological scenarios defined in Section 4, it is necessary to rule out races in (partial) member bMSCs which do not lead to race scenarios. Following this idea, we give the definition of race in MSC specifications.

Definition 12. (Race in MSC Specifications) Let us assume that M is an MSC specification, e, f are two events on process P_i . $[e, f]$ is a *race* of M if $[e, f]$ is a race in some member bMSC M of M such that $uau'b \in \text{pref}(L(M)|P_i)$ where $a = l(e)$, $b = l(f)$ and u, u' are two words in alphabet \mathcal{A} ; and $ub \notin \text{pref}(L(M))|P_i$. If this is the case then we say that $[e, f]$ is a race of M based on M . We say that M is *race free*, if M contains no races.

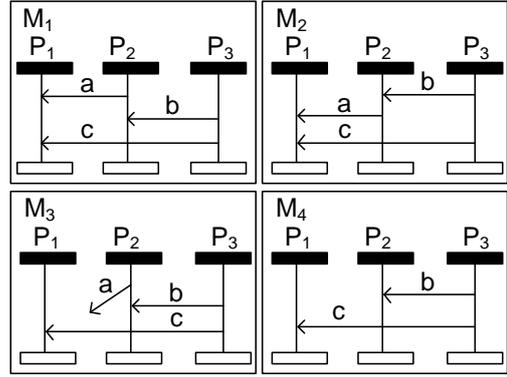


Figure 13: Race scenarios in an MSC specification

Informally, a race in an MSC specification is first a race of a member bMSC and this race leads to race scenarios. In the definition, $uau'b$, where $a = l(e)$, $b = l(f)$, represents a prefix of the projection of M on process P_i . It contains the labels of the two events which form a race $[e, f]$. ub is the projection of the race scenario formed by applying RSC to M .

Example 17. (Race and race scenario in MSC specification) Consider MSC specification $M7$ given in Figure 13 which has M_1 and M_2 as its member bMSCs. M_3 and M_4 are two race scenarios of the individual member bMSCs, respectively. Both event pairs corresponding to event labels $?a$ and $?c$ in M_1 and M_2 form races of $M7$ since M_3 and M_4 are race scenarios of $M7$.

According to Definition 12, let e and f be two events in M_1 , $?a = l(e)$ and $?c = l(f)$. Then $[e, f]$ is a race based on M_1 , i.e., $[e, f]$ is a race of $M7$ and we can demonstrate this by the inspection of M_1 .

In addition, the following relationships hold between race, race free and race scenarios. Proposition 12 states that RSC is also applicable for race of MSC specification.

Proposition 12. *If $[e, f]$ is a race of MSC specification M based on M , then scenario M' generated by RSC from M is a race scenario of M .*

PROOF. We have that race $[e, f]$ is a race on process P based on M . A bMSC M' can be generated by RSC. It is clear that the behaviours of all other processes (excluding P) are consistent with the specification, since they are prefixes of projections of M . According to Definition 12, the projection on P is not in $\text{pref}(L(M)|P)$. Therefore, M' is a race scenario of M . The proposition is established. \square

The motivation for detecting races to avoid problematic scenarios in real runs is stated in the following proposition.

Proposition 13. *An MSC specification \mathcal{M} is race free if and only if \mathcal{M} leads to no race scenarios.*

PROOF. For the forward direction, let us suppose that \mathcal{M} contains no races; we then show that there are no race scenarios in \mathcal{M} . We will use proof by contradiction: assume that there is a race scenario wa where $a = l(f)$ and event f happens on process P_i such that w is a prefix of $L(\mathcal{M})$ but well-formed wa is not a valid prefix. Notice that every member bMSC M of $L(\mathcal{M})$ that contains f and has prefix w can be represented by a well-formed and complete word $wuau'$, where u and u' are two words on alphabet \mathcal{A} . We then have $wuau'|P_i \in L(\mathcal{M})|P_i$. In addition, since $wa \notin \text{pref}(L(\mathcal{M}))$, $wua|P_i \neq wa|P_i$ and so $u|P_i \neq \epsilon$. This means that in M there is an event e on P_i that happens immediately after the event sequence corresponding to $w|P_i$ but before f . Because $\text{msg}^{-1}(f)$ corresponds to a letter in w , we have $\neg(e <^* \text{msg}^{-1}(f))$ and $e \neq \text{msg}^{-1}(f)$. Therefore $[e, f]$ is a race of M according to Proposition 9. Because $wa|P_i \notin \text{pref}(L(\mathcal{M})|P_i)$, $[e, f]$ is a race of \mathcal{M} . This provides a contradiction as required.

For the converse direction, let us suppose that there are no race scenarios in \mathcal{M} ; we then prove that there is no race in \mathcal{M} . We prove this also by contradiction. Let us assume that \mathcal{M} contains no race scenarios but there is a race $[e, f]$ of \mathcal{M} on process P_i in some member bMSC M . According to Proposition 12, a race scenario of \mathcal{M} can always be generated by RSC. This gives a contradiction as required. \square

As stated in Proposition 12, RSC is useful in constructing race scenarios. RSC can also be used to check whether a race of member bMSC is a race of the specification.

Proposition 14. *If, for every race of any member bMSC, the race scenario generated by RSC is not a race scenario of MSC specification \mathcal{M} , then \mathcal{M} is race free.*

PROOF. This follows from Proposition 12. \square

Finally, with the relationship between non-local choice and implied scenarios given in Section 5 and the relationship between race and race scenarios proven, the relationship among race free, non-local choice free and pathology free can be stated in the following proposition.

Proposition 15. *An MSC specification \mathcal{M} is pathology free if and only if \mathcal{M} is race free and non-local choice free.*

PROOF. For the forward direction, let \mathcal{M} be pathology free. \mathcal{M} leads to no pathological scenarios according to Proposition 1. Therefore, \mathcal{M} does not lead to race scenarios and implied scenarios based on Proposition 3. \mathcal{M} thus contains no race and non-local choice according to Propositions 6 and 13. The forward direction is established.

For the converse direction, let \mathcal{M} be race free and non-local choice free. \mathcal{M} thus leads to no race scenarios and implied scenarios according to Propositions 6 and 13. Therefore, \mathcal{M} leads to no pathological scenarios (Proposition 3), and so is pathology free according to Proposition 1. The converse direction is established. \square

7. Discussion

We note that our definition of race free in MSC specifications is different from the definition in [11]. In this section, we provide a comparison between the two definitions. In addition, we briefly discuss how to apply the results in previous sections on systems with different communications using FIFO communication as an example.

7.1. An alternative definition of race free

If we denote the language corresponding to enforced orders of MSC specification \mathcal{M} as $L^{en}(\mathcal{M})$, the definition of race free in [11] is as follows⁵.

Definition 13. (Race Free in MSC Specifications [11]) An MSC specification \mathcal{M} contains races if $L(\mathcal{M}) \subseteq \mathcal{A}^*$ is strictly included in $L^{en}(\mathcal{M}) \subseteq \mathcal{A}^*$.

That is, MSC specification \mathcal{M} is race free if $L(\mathcal{M}) = L^{en}(\mathcal{M})$. Nodes in MSG can only refer to bMSCs. This intuitively appealing definition directly extends Proposition 7 for single bMSC specifications. However, this race free definition is not consistent with race free in Definition 12 for MSC specifications. Let us use an example to discuss both race free definitions.

Example 18. (Race free) Consider an MSC specification $\mathcal{M8}$ with two bMSCs M_1 and M_2 given in Figure 14. With Definition 13, \mathcal{M} is not race free, since $L(\mathcal{M8})$ equals $L(M_1) \cup L(M_2)$ and $L^{en}(\mathcal{M8}) = L^{en}(M_1) =$

⁵This definition is slightly different from the original in [11] which is defined on the traces of the set of events E .

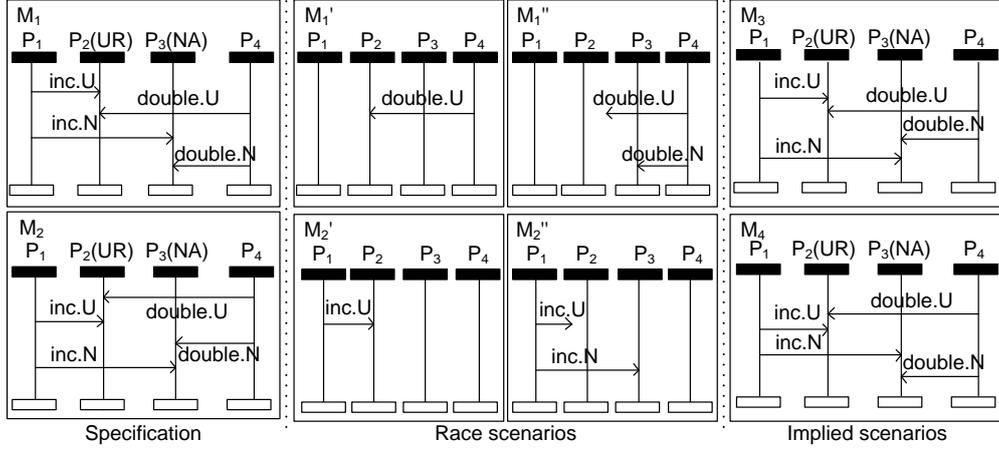


Figure 14: An example of implied scenarios

$L^{en}(M_2) = L(M_1) \cup L(M_2) \cup L(M_3) \cup L(M_4)$, therefore, $L(M_8) \neq L^{en}(M_8)$. This means that there are races in M_8 and the races lead to two problematic scenarios M_3 and M_4 . With Definition 12, M_8 is race free since all scenarios, M_1', M_1'', M_2' and M_2'' , generated by RSC are not race scenarios of M_8 , because the projections of these scenarios are prefixes of the corresponding process languages. In fact, according to Alur et al. [18], M_3 and M_4 are implied scenarios.

The formal relationship between the two race free definitions is stated in Proposition 16.

Proposition 16. *Given an MSC specification M , if M is race free under Definition 12 but $L(M) \neq L^{en}(M)$ then any word $w \in L^{en}(M) \setminus L(M)$ represents an implied scenario of M ; if $L(M) = L^{en}(M)$ then M is race free according to Definition 12.*

PROOF. For the first part, let us suppose that M is race free according to Definition 12, $L(M) \neq L^{en}(M)$, it is sufficient to show that any $w \in L^{en}(M) \setminus L(M)$ represents an implied scenario. Clearly, $w|P_i \in L(M)|P_i$ because M is race free under Definition 12. In addition, w is well-formed but $w \notin L(M)$ since $w \in L^{en}(M) \setminus L(M)$. Therefore, w is an implied scenario according to Definition 7.

For the second part, we will use proof by contradiction. Let us suppose that $L(M) = L^{en}(M)$ and there is a race $[e, f]$ of M based on a member bMSC M . Based on race $[e, f]$, a race scenario M' of M can be constructed by applying RSC. $L(M') \in \text{pref}(L^{en}(M))$ since $\neg(e \ll^* f)$. However, $L(M') \notin \text{pref}(L(M))$ since M' is a race scenario of M . This gives the required contradiction.

The proposition is established. \square

Proposition 16 means that the race free condition in Definition 12 is weaker than that in Definition 13. The MSC specifications that are race free with Definition 12 and not with Definition 13 only cause implied scenarios.

Consequently, we have the following proposition.

Proposition 17. *Let us suppose that M is an MSC specification, if for all $P_i \in \mathcal{P}$, $L^{en}(M)|P_i \subseteq L(M)|P_i$ then M is race free according to Definition 12.*

PROOF. We will use proof by contradiction. Let us suppose that for all $P_i \in \mathcal{P}$ $L^{en}(M)|P_i \subseteq L(M)|P_i$, but there is a race formed by e, f . $[e, f]$ must be a race in some bMSC M such that there is some $uav'b \in L(M)|P_i$, where $a = l(e)$ and $b = l(f)$, for which ub is not a prefix of $L(M)|P_i$.

Since $[e, f]$ is a race in M , $ub \in \text{pref}(L^{en}(M))|P_i$. This contradicts $L^{en}(M)|P_i \subseteq L(M)|P_i$. The proposition is established. \square

This proposition can be used to detect races on a specific process of an MSC specification by comparing the process language and the local language denoted $L^{en}(M)|P_i$. It also suggests that it is possible to cure races in distributed systems by changing the behaviour of local systems following language $L^{en}(M)|P_i$.

Example 19. Let us reconsider process P_2 in specification M_8 given in Figure 14. Obviously, there is no enforced order between events $?inc.U$ and $?double.U$ so the $L^{en}(M_8)|P_2 = L(M_8)|P_2$. Therefore, there is not a race on P_2 in specification M_8 according to Proposition 17.

Let us suppose that the implementation of P_2 only accepts events $?inc.U$, $?double.U$, and in a specific order

as shown in M_1 of Figure 14. Since there is no enforced order between the two events based on M_1 of Figure 14, M_1 is not race free. To solve this problem, one way is to change the implementation of P_2 to allow it to receive the two events in either order, for example adding M_2 in the same figure as a member bMSC. The other way is to introduce additional enforced order between the two events for example using an FIFO communication with a global channel buffer.

7.2. Impact of communications

In this paper, in order to focus on introducing a framework on pathologies in MSCs, we adopt the standard partial order semantics and also use a trace semantics consistent with partial order semantics. With the standard partial order semantics, each message is regarded as two events, messages can overtake previous messages; in addition there are no bounds placed on the number of messages sent by a process. We note that specific MSC specifications may not be implementable with certain types of communication [38]. A type of communication which can implement most of the MSC specifications should be chosen when developing a general framework so that most of the pathologies can be exposed to the analysis. Therefore, similar to much of the work on pathologies [8, 9, 11, 18, 31], we insist that the underlying communications between the subsystems are asynchronous, non-FIFO and that each message channel has an independent buffer, is not order preserving and is unbounded.

It is clear that pathologies and pathological scenarios can be affected by the communication chosen by the implementations. For example, it was shown that race may lead to different results under different types of communication [13]. However, since we place relatively few constraints on the communications, other types of communication may generate fewer pathological scenarios, corresponding to fewer pathologies. This is important, since it means that the pathologies in the specification and the possible pathological scenarios with a system implemented with more restricted types of communication are subsets of the results generated by the configuration adopted in the previous sections. Therefore, the results given in previous sections are adaptable.

In the rest of the section, we extend the results in previous sections to FIFO communications with individual channel buffers to show that our framework is adaptable to other types of communication. For race and race scenarios, the definition of enforced order (Definition 3) needs to be revised to reflect the condition of FIFO communications. In addition to the existing orders in Definition 3, orders between a pair of receive

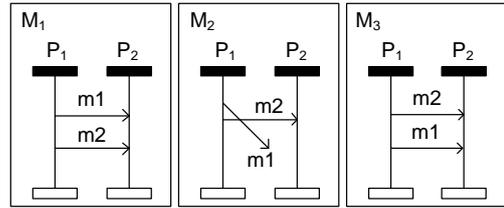


Figure 15: Race and implied scenario with FIFO

events of two messages that are from the same originating process and to the same destination process should be included. Some races with non-FIFO communications may no longer be races with FIFO communications due to the change of enforced order. Consequently, the number of race scenarios may be reduced.

Example 20. (Race with FIFO communication)

Consider an MSC specification \mathcal{M}_9 with only one bMSC M_1 given in Figure 15. M_2 is race scenario of \mathcal{M}_9 since $[?m1, ?m2]$ is a race with non-FIFO communication. However, M_2 is not a race scenario with FIFO communication since $?m1$ and $?m2$ do not form a race with FIFO. This is because we have $?m1 \ll ?m2$ with FIFO.

The situation for non-local choice and implied scenarios is slightly different. It is not necessary to revise definitions and propositions in changing from non-FIFO to FIFO communications. Definitions of non-local choice and implied scenarios are independent of the types of communication. However, the use of FIFO communication leads to a subset of the specified language being possible for the implemented distributed system. Consequently, there are implied scenarios which cannot happen with FIFO communications.

Example 21. (Impossible implied scenarios)

Let us consider MSC specification \mathcal{M}_{10} containing two MSCs, M_1 and M_3 , shown in Figure 15. M_2 is an implied scenario, but M_2 cannot happen with FIFO communications because no message can overtake previous messages that are sent from the same process to the same destination.

8. Conclusions

This paper introduced a framework for MSCs in which pathology and pathological scenarios are regarded as two different concepts. In this framework, pathological scenarios are undesirable scenarios that can arise from synchronisation problems. To reduce the

number of pathological scenarios to be considered, we required that a pathological scenario contained only one event violating the specification. It transpired that this restriction enabled us to categorise pathological scenarios within the framework. It was shown that a famous concept in MSC research, implied scenarios with one undesirable event is a type of pathological scenario. We called the other type of pathological scenario race scenarios. It also transpired that the sets of implied scenarios and race scenarios partition the set of pathological scenarios. This result does not hold if we do not require that pathological scenarios only contain one event that violates the specification.

We concentrated on problems that cause synchronisation issues. We assumed that a process will only send a message when this is consistent with its local specification (the sensible local behaviour assumption) and found that only two pathologies, race and non-local choice, lead to scenarios with synchronisation problems. Moreover, these pathologies are orthogonal. We proved that race and non-local choice cause race scenarios and implied scenarios, respectively. It was also observed that our definition of race in an MSC Specification was different from that traditionally used. We compared the race free definition in [11] with ours. We found that an MSC specification that is race free according to [11] was also race free according to the refined race free definition in this paper, but not vice versa. An MSC specification that is not race free according to [11] can be race free but lead to implied scenarios according to the new definition. Thus, an MSC specification is race free and has no implied scenarios under our definition if and only if it is race free and has no implied scenarios under the definition given in [11]. The main difference between our definition and that in [11] is that our definition leads to the sets of race scenarios and implied scenarios being disjoint.

The framework helped us to clarify some difficult issues and apparently contradictory statements in MSC research. The taxonomy on pathologies and the pathological scenarios and the formal relationships between types of pathologies and pathological scenarios provided an accurate way to address the observed problems and predict the possible problems in a running distributed system. The result implies that each type of pathology leads to a specific type of problem. Conversely, if specific problems were observed, specific pathologies must have been introduced into the specification. In addition, this framework showed that the only causes of synchronisation problems in MSCs are race and non-local choice. As a result, race and non-local choice related problems should be a major focus

of research on MSC pathologies in the future.

The framework developed in this paper has potential to be applied in different research areas. For example, there has been recent interest in using MSCs to define test cases [40, 41]. In our experience, it is difficult to determine the impact of pathologies in testing with MSCs. However, based on the framework, controllability problems of testing with MSCs can be clearly categorised. It has been shown that races cause controllability problems of timing and non-local choices cause controllability problems of choice [26]. In addition, solutions to the controllability problems were developed by adopting corresponding solutions to tackle race and non-local choice pathologies [26].

Our definition of a pathological scenario required that there is only one undesirable event. If multiple undesirable events were allowed then we would obtain (exponentially) more problematic scenarios. There are clear benefits from identifying such a smaller set of scenarios: the software engineer can focus on these when deciding whether the MSC specification should be revised and can also use the pathological scenarios as the basis of validation and verification processes, such as testing to determine whether any of these scenarios are behaviours of the implementation.

We are currently working on finding better algorithms for detecting non-local choices and races in complex MSC specifications. Existing solutions for these problems in the literature are exponential time algorithms [31, 11]. However, these algorithms can be improved from different aspects. There is no algorithm for detecting races based on the refined race definition. This may be a simpler question than detecting traditional races in complex MSC specifications since the refined race definition is orthogonal to non-local choice which has been proved to be difficult to detect. By investigating characteristics of these pathologies, further categorisations for both types of pathology may be identified. Developing algorithms for detecting key subtypes of a pathology might be easier than detecting all pathologies. In addition, since both problems are in EXSPACE, it may be possible to apply heuristic search methods to find most of the pathologies in MSC specifications. Removing most of the pathologies from specifications will lead to more robust implementations of systems. Finally, it could be very interesting to develop a tool based on the proposed framework with single undesirable event restriction. The tool should have the following functions: detecting race and non-local choices in MSC specifications, tracing the pathologies that causes an observed pathological scenario and proposing possible cures to the identified pathologies.

References

- [1] ITU-T, ITU-T Recommendation Z.120 Message Sequence Chart (4 2004).
- [2] S. Mauw, M. Reniers, T. Willemse, Message Sequence Charts in the software engineering process, *Handbook of Software Engineering and Knowledge Engineering 1* (2001) 437–464.
- [3] Ø. Haugen, MSC-2000 interaction diagrams for the new millennium, *Computer Networks* 35 (6) (2001) 721–732.
- [4] SDL Forum Society, <http://www.sdl-forum.org/> (Accessed in 2012).
- [5] I. Krüger, R. Grosu, P. Scholz, M. Broy, From MSCs to statecharts, in: *Proceedings of International Workshop on Distributed and Parallel Embedded Systems*, Schloss Eringerfeld, Germany, 1998, pp. 61–71.
- [6] J. Whittle, R. Kwan, J. Saboo, From scenarios to code: An air traffic control case study, *Software and Systems Modeling* 4 (2005) 71–93.
- [7] P. Baker, P. Bristow, C. Jervis, D. King, R. Thomson, B. Mitchell, S. Burton, Detecting and resolving semantic pathologies in UML sequence diagrams, in: *Proceedings of the 10th European Software Engineering Conference held jointly with the 13th International Symposium on Foundations of Software Engineering*, Lisbon, Portugal, 2005, pp. 50–59.
- [8] L. Hérouët, Some pathological message sequence charts, and how to detect them, in: *Proceedings of the 10th International SDL Forum*, Copenhagen, Denmark, 2001, pp. 348–364.
- [9] R. Alur, G. Holzmann, D. Peled, An analyzer for message sequence charts, *Software Concepts and Tools* 17 (2) (1996) 70–77.
- [10] B. Mitchell, Resolving race conditions in asynchronous partial order scenarios, *IEEE Transactions on Software Engineering* 31 (9) (2005) 767–784.
- [11] A. Muscholl, D. Peled, Message sequence graphs and decision problems on Mazurkiewicz traces, in: *Proceedings of the 24th International Symposium on Mathematical Foundations of Computer Science*, London, UK, 1999, pp. 81–91.
- [12] E. Elkind, B. Genest, D. Peled, Detecting races in ensembles of message sequence charts, in: *Proceedings of 13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Braga, Portugal, 2007, pp. 420–434.
- [13] B. Mitchell, Characterizing communication channel deadlocks in sequence diagrams, *IEEE Transactions on Software Engineering* 34 (3) (2008) 305–320.
- [14] E. Elkind, B. Genest, D. Peled, P. Spoletini, Quantifying the discord: Order discrepancies in message sequence charts, *International Journal of Foundations of Computer Science* 21 (2) (2010) 211–233.
- [15] P. B. Ladkin, S. Leue, Interpreting message flow graphs, *Formal Aspects of Computing* 7 (5) (1995) 473–509.
- [16] H. Ben-Abdallah, S. Leue, Syntactic detection of process divergence and non-local choice in message sequence charts, in: *Proceedings of the 13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Enschede, Netherlands, 1997, pp. 259–274.
- [17] A. J. Mooij, N. Goga, J. Romijn, Non-local choice and beyond: Intricacies of MSC choice nodes, in: M. Cerioli (Ed.), *FASE*, Vol. 3442 of *Lecture Notes in Computer Science*, Springer, 2005, pp. 273–288.
- [18] R. Alur, K. Etessami, M. Yannakakis, Inference of message sequence charts, *IEEE Transactions on Software Engineering* 29 (7) (2003) 623–633.
- [19] S. Uchitel, J. Kramer, J. Magee, Incremental elaboration of scenario-based specifications and behavior models using implied scenarios, *ACM Transactions on Software Engineering and Methodology* 13 (1) (2004) 37–85.
- [20] R. Alur, M. Yannakakis, Model checking of message sequence charts, in: *Proceedings of 10th International Conference on Concurrency Theory*, Eindhoven, Netherlands, 1999, pp. 114–29.
- [21] H. Ben-Abdallah, S. Leue, Timing constraints in message sequence chart specifications, in: *Proceedings of International Conference on Formal Description Techniques: Protocol Specification, Testing and Verification*, Osaka, Japan, 1997, pp. 91–106.
- [22] P. Baker, S. Loh, F. Weil, Model-driven engineering in a large industrial context: motorola case study, in: L. Briand, C. Williams (Eds.), *Model Driven Engineering Languages and Systems*, Vol. 3713 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2005, pp. 476–491.
- [23] A. Mooij, N. Goga, Dealing with non-local choice in IEEE 1073.2s standard for remote control, in: D. Amyot, A. Williams (Eds.), *System Analysis and Modeling*, Vol. 3319 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2005, pp. 257–270.
- [24] S. Uchitel, R. Chatley, J. Kramer, J. Magee, LTSA-MS: tool support for behaviour model elaboration using implied scenarios, in: *Proceedings of the 9th international conference on Tools and algorithms for the construction and analysis of systems*, Springer-Verlag, Berlin, Heidelberg, 2003, pp. 597–601.
- [25] H. Foster, S. Uchitel, J. Magee, J. Kramer, Model-based verification of web service compositions, in: *Proceedings of the 18th IEEE International Conference on Automated Software Engineering*, Montreal Canada, 2003, pp. 152–161.
- [26] H. Dan, R. M. Hierons, Controllability problems in MSC-based testing, *The Computer Journal* (to appear).
- [27] H. Dan, R. M. Hierons, S. Counsell, Non-local choices and implied scenarios, in: *Proceedings of the 8th International Conference on Software Engineering and Formal Methods*, Pisa, Italy, 2010, pp. 53–62.
- [28] ITU-T, ITU-T Recommendation Z.120 Annex B: Formal semantics of message sequence charts (4 1998).
- [29] H. Muccini, Detecting Implied Scenarios Analyzing Non-local Branching Choices, in: *Proceedings of 6th International Conference on Fundamental Approaches to Software Engineering*, Warsaw, Poland, 2003, pp. 372–386.
- [30] B. Bollig, J.-P. Katoen, C. Kern, M. Leucker, Learning communicating automata from MSCs, *IEEE Transactions on Software Engineering* 36 (3) (2010) 390–408.
- [31] R. Alur, K. Etessami, M. Yannakakis, Realizability and verification of MSC graphs, *Theoretical Computer Science* 331 (1) (2005) 97–114.
- [32] J. G. Henriksen, M. Mukund, K. N. Kumar, P. S. Thiagarajan, On message sequence graphs and finitely generated regular MSC languages, in: *Proceedings of the 27th International Colloquium on Automata, Languages and Programming*, London, UK, 2000, pp. 675–686.
- [33] A. Rensink, H. Wehrheim, Weak sequential composition in process algebras, in: B. Jonsson, J. Parrow (Eds.), *CONCUR '94: Concurrency Theory*, Vol. 836 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 1994, pp. 226–241.
- [34] B. Genest, A. Muscholl, H. Seidl, M. Zeitoun, Infinite-state high-level MSCs: Model-checking and realizability, *Journal of Computer and System Sciences* 72 (4) (2006) 617–647.
- [35] H. Kugler, I. Segall, Compositional synthesis of reactive systems from live sequence chart specifications, in: *Proceedings of the 15th International Conference on Tools and Algorithms for the Construction and Analysis of Systems: Held as Part of the Joint European Conferences on Theory and Practice of Soft-*

- ware, ETAPS 2009, TACAS '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 77–91.
- [36] H. Kugler, C. Plock, A. Pnueli, Controller synthesis from LSC requirements, in: Proceedings of the 12th International Conference on Fundamental Approaches to Software Engineering: Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2009, FASE '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 79–93.
- [37] T. Gazagnaire, B. Genest, L. Hélouët, P. Thiagarajan, S. Yang, Causal message sequence charts, *Theoretical Computer Science* 410 (41) (2009) 4094 – 4110.
- [38] A. Engels, S. Mauw, M. Reniers, A hierarchy of communication models for message sequence charts, in: *Science of Computer Programming*, Chapman & Hall, 1997, pp. 75–90.
- [39] L. Lamport, Time, clocks, and the ordering of events in a distributed system, *Communications of the ACM* 21 (7) (1978) 558–565.
- [40] S. Boroday, A. Petrenko, A. Ulrich, Implementing MSC tests with quiescence observation, in: Proceedings of the 21st International Conference on Testing of Software and Communication Systems and 9th International FATES Workshop, Berlin, Heidelberg, 2009, pp. 49–65.
- [41] H. Dan, R. M. Hierons, Conformance testing from Message Sequence Charts, in: Proceedings of the 4th International Conference on Software Testing, Verification and Validation, Berlin, Germany, 2011, pp. 279–288.