

## 1 Lecture 3: Races

# Theoretical Foundations of the UML

## Lecture 3: Races

Joost-Pieter Katoen

Lehrstuhl für Informatik 2  
Software Modeling and Verification Group

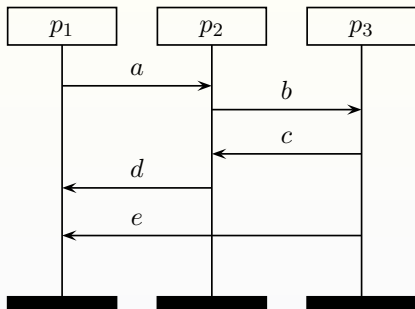
[moves.rwth-aachen.de/teaching/ss-16/theoretical-foundations-of-the-uml/](http://moves.rwth-aachen.de/teaching/ss-16/theoretical-foundations-of-the-uml/)

26. Oktober 2017

# Summary of Lecture #2

- 1 A Message Sequence Chart is a **partial order**
  - between send and receive events
  - totally ordered per process vertical ordering
  - receive events happen after their send events horizontal ordering
  - respecting the first-in first out (FIFO) property
- 2 **Linearizations** are totally ordered extensions of partial orders
  - all linearizations of an MSC are **well-formed**
    - 1 every receive is preceded by a corresponding send
    - 2 respects the FIFO ordering
    - 3 no send events without corresponding receive
- 3 Every well-formed word can be **transformed** into an MSC
  - two linearizations of the same MSC yield **isomorphic** MSCs
- 4 So: there is a **1-to-1 relation** between an MSC and its linearizations

# Example



These pictures are formalized using **partial orders**.

# Message Sequence Chart (MSC) (1)

## Definition

An MSC  $M = (\mathcal{P}, E, \mathcal{C}, l, m, \preceq)$  with:

- $\mathcal{P}$ , a finite set of **processes**  $\{p_1, p_2, \dots, p_n\}$  with  $n > 1$
- $E$ , a finite set of **events**

$$E = \bigsqcup_{p \in \mathcal{P}} E_p = E_? \cup E_!$$

- $\mathcal{C}$ , a finite set of **message contents**
- $l : E \rightarrow Act$ , a **labelling** function defined by:

$$l(e) = \begin{cases} !(p, q, a) & \text{if } e \in E_p \cap E_! \\ ?(p, q, a) & \text{if } e \in E_p \cap E_? \end{cases}, \text{ for } p \neq q \in \mathcal{P}, a \in \mathcal{C}$$

# Message Sequence Chart (MSC) (2)

## Definition

- $m : E_! \rightarrow E_?$  a bijection (“**matching function**”), satisfying:

$$m(e) = e' \wedge l(e) = !(p, q, a) \text{ implies } l(e') = ?(q, p, a) \quad (p \neq q, a \in \mathcal{C})$$

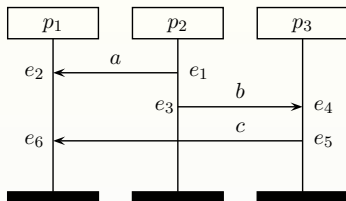
- $\preceq \subseteq E \times E$  is a partial order (“**visual order**”) defined by:

$$\preceq = \left( \underbrace{\bigcup_{p \in \mathcal{P}} <_p}_{<_p \text{ is a total order = "top-to-bottom" order on process } p} \cup \underbrace{\{(e, m(e)) \mid e \in E_!\}}_{\text{communication order } <_c} \right)^*$$

where for relation  $R$ ,  $R^*$  denotes its reflexive and transitive closure.

# Example

# Visual order can be misleading



can  $e_6$  occur  
before  $e_2$ ?

If message  $b$  takes much shorter than message  $a$ ,  
then  $c$  might arrive at  $p_1$  before  $a$ .

In practice,  $e_6$  might occur before  $e_2$ , but  $e_2 <_{p_1} e_6$  and thus  $e_2 \preceq e_6$ .

This is misleading and called a **race**.



# What is a race?

A race condition asserts a particular order of events will occur because of the visual ordering (i.e., the partial order  $\preceq$ ) when, in practice, this order cannot be guaranteed to hold.

Q: When are race conditions possible and how to detect them?

# Causal order

Main principles:

- Send events should happen before their matching receive events
- The ordering of events wrt. sends on same process is unaffected
- Receive events on a process sent from the same process are ordered as their sends

## Definition

For MSC  $M = (\mathcal{P}, E, \mathcal{C}, l, m, \preceq)$ , relation  $\ll \subseteq E \times E$  is defined by:

$$e \ll e' \quad \text{iff} \quad e' = m(e)$$

$$\text{or} \quad e <_p e' \text{ and } E! \cap \{e, e'\} \neq \emptyset$$

$$\text{or} \quad e, e' \in E_p \cap E_q \text{ and } m^{-1}(e) <_q m^{-1}(e')$$

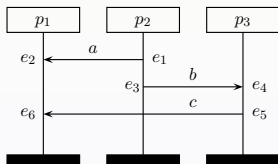
$\ll^*$  is a partial order (referred to as **causal order**) in which events at the same process are not necessarily ordered.

# Causal order: example

## Definition

For MSC  $M = (\mathcal{P}, E, \mathcal{C}, l, m, \preceq)$ , relation  $\ll \subseteq E \times E$  is defined by:

$e \ll e'$  iff  $e' = m(e)$   
or  $e <_p e'$  and  $E_1 \cap \{e, e'\} \neq \emptyset$   
or  $e, e' \in E_p \cap E_q$  and  $m^{-1}(e) <_q m^{-1}(e')$



## Example

$e_1 \ll e_2$ ,  $e_3 \ll e_4$ ,  $e_5 \ll e_6$ ,  $e_1 \ll e_3$ ,  $e_4 \ll e_5$ , **not** ( $e_2 \ll e_6$ )

## Definition

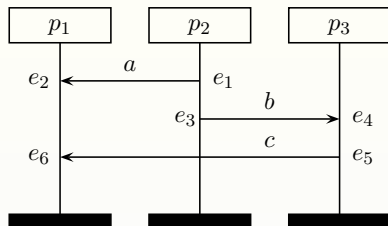
MSC  $M$  contains a **race** if for some  $e, e' \in E?$  and process  $p$ :

$$e <_p e' \text{ but not } (e \ll^* e')$$

where  $\ll^* \subseteq E \times E$  is the reflexive and transitive closure of  $\ll$ .

As relation  $\ll^*$  contains at most all orderings in  $\preceq$ ,  
the MSC  $M$  has a race whenever  $\preceq \not\subseteq \ll^*$ .

# Race: example



## Visual order versus causal order

- 1  $e_1 \preceq e_2$ ,  $e_3 \preceq e_4$ ,  $e_5 \preceq e_6$ ,  $e_1 \preceq e_3$ ,  $e_4 \preceq e_5$ ,  $e_2 \preceq e_6$
- 2  $e_1 \ll e_2$ ,  $e_3 \ll e_4$ ,  $e_5 \ll e_6$ ,  $e_1 \ll e_3$ ,  $e_4 \ll e_5$ , **not** ( $e_2 \ll e_6$ )

As  $\preceq \not\subseteq \ll^*$ , this MSC contains a race.

On the black board.

# Why are races problematic?

Recall: MSC  $M$  has a **race** if  $\preceq \not\subseteq \ll^*$  or equivalently:

$$\exists e, e' \in E? . (e <_p e' \text{ and } e \not\ll^* e')$$

Whenever  $\preceq \not\subseteq \ll^*$ , implementations based on  $<_p$  may cause problems:

- 1 unspecified message reception
  - a process receives a message which by the MSC is not possible
- 2 deadlocks
  - a process blocking on receipt of an unexpected message may block others too
- 3 message loss
  - unexpectedly received messages may be ignored
- 4 exploiting wrong message content

# Checking whether an MSC has a race

- MSC  $M$  has a **race** if  $\preceq \not\subseteq \ll^*$
- How to check whether MSC  $M$  has a race?  
*compute  $\ll^*$  and check whether  $\preceq \subseteq \ll^*$*
- transitive closure  $\ll^*$  is computed using **Floyd-Warshall's** algorithm
  - algorithm for finding shortest paths in a weighted digraph with positive or negative edge weights<sup>1</sup>
  - easily adapted for computing the transitive closure of digraphs
  - worst-case time complexity  $\mathcal{O}(|E|^3)$
  - by using some specifics of MSC, this is reduced to  $\mathcal{O}(|E|^2)$
- So: **race checking** can be done **quadratically** in the number of **events**

---

<sup>1</sup>for digraphs without negative cycles.



## Algorithm

compute  $\ll^*$  and compare with  $\preceq$

Warshall's algorithm

Warshall's algorithm: input:  $R \subseteq X \times X$  where  $X$  is a set  
output:  $R^*$

## Idea:

Consider  $R$  and  $R^*$  as directed graphs

There is an edge  $x \Rightarrow y$  in  $R^*$  iff there is a (possibly empty) sequence

$$x = x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \dots \rightarrow x_n = y \text{ in } R$$

(our setting:  $X = E, R = \ll, R^* = \ll^*$ )

# Warshall's algorithm: preliminaries

- assume: graph vertices are numbered  $\{1, 2, \dots, n\}$  where  $n = |E|$
- for  $j \in \{1, \dots, n+1\}$  define relation  $\xrightarrow{j}$  as follows:  
 $x \xrightarrow{j} y$  iff  $\exists$  path in  $R$  from  $x$  to  $y$  such that all vertices on the path ( $\neq x, y$ ) have a smaller number than  $j$
- Then: (1)  $x \Longrightarrow y$  iff  $x \xrightarrow{n+1} y$   
(2)  $x \xrightarrow{1} y$  iff  $x = y$  or  $x \ll y$   
(3)  $x \xrightarrow{y+1} z$  iff  $x \xrightarrow{y} z$  or  $x \xrightarrow{y} y \xrightarrow{y} z$
- Algorithm: determine the relations  $\xrightarrow{1}, \dots, \xrightarrow{n}, \xrightarrow{n+1}$  iteratively using properties (2) + (3); Result is then given by (1).
- Store  $\xrightarrow{i}$  in a boolean matrix  $C$  of cardinality  $|E| \times |E|$
- Postcondition:  $C[x, y] = \mathbf{true}$  iff  $(x, y) \in R^*$
- Precondition:  $\forall x, y \in X . C[x, y] = \mathbf{false}$

# Warshall's algorithm

```
/* first compute  $x \xrightarrow{1} y$  */
for  $x := 1$  to  $n$  do
  for  $y := 1$  to  $n$  do
     $C[x, y] := (x = y \text{ or } \underbrace{(x, y) \in R}_{x \ll y})$ 
/* loop invariant: after the  $j$ -th iteration of
/* outermost loop it holds:  $C[x, y] = \text{true}$  iff  $x \xrightarrow{j+1} y$  */
for  $y := 1$  to  $n$  do
  for  $x := 1$  to  $n$  do
    if  $C[x, y]$  then
      for  $z := 1$  to  $n$  do
        if  $C[y, z]$  then
           $C[x, z] := \text{true}$ 
```

# Correctness and complexity

## Lemma: correctness

After  $j$  iterations:  $x \xrightarrow{j+1} y$  iff  $C[x, y] = \text{true}$ .

## Proof.

*if*: trivial; *only if*: by induction on  $j$ . □

## Complexity

Worst-case time complexity of Warshall's algorithm :  $\mathcal{O}(n^3)$  with  $n = |X|$

## Proof.

follows from the fact that there is a triple-nested loop of which each loop has at most  $n$  iterations. □

Warshall's algorithm computes  $R^*$  for **every** binary relation  $R \subseteq X \times X$ .

Recall: our interest is in determining  $R^*$  for  $R = \ll$

Using some properties of  $\ll$ , the complexity can be improved.

Exploit that for  $\ll$ :

- ①  $\ll$  is acyclic (as it is a partial order)
- ② the number of **immediate predecessors** of  $e \in E$  under  $\ll$  is at most two

(why?)

Note that  $e$  is an **immediate** predecessor of  $e'$  (under  $\ll$ ) if:

$$e \ll e' \text{ and } \neg(\exists e'' \notin \{e, e'\}. e \ll e'' \wedge e'' \ll e')$$

The main loop of Warshall's algorithm:

```
for  $e := 1$  to  $n$  do
  for  $e' := 1$  to  $n$  do
    if  $C[e', e]$  then
      for  $e'' := 1$  to  $n$  do
        if  $C[e, e'']$  then
           $C[e', e''] := \mathbf{true}$ 
```

The main loop of [Alur et al.'s algorithm](#) for detecting races in MSCs:

```
for  $e := 1$  to  $n$  do
  for  $e' := e - 1$  downto  $1$  do
    if (not  $C[e', e]$  and  $e' \ll e$ ) then
       $C[e', e] := \mathbf{true}$ 
      for  $e'' := 1$  to  $e' - 1$  do
        if  $C[e'', e']$  then
           $C[e'', e] := \mathbf{true}$ 
```

## Theorem

Let  $M$  be an MSC with set  $E$  of events and  $n = |E|$ . Checking whether  $M$  has a race can be done in  $\mathcal{O}(n^2)$ .

## Proof.

Since  $\ll$  is acyclic, we number the events such that the numbering defines a total order that is consistent with visual order  $\preceq$ . This can be done in  $\mathcal{O}(n)$  using a standard topological sort. Then observe that the innermost loop:

for  $e'' := 1$  to  $e' - 1$  do  
    if  $C[e'', e']$  then  $C[e'', e] := \mathbf{true}$

of the triple-nested main loop is executed for  $(e, e')$  only if  $e'$  is an immediate predecessor of  $e$  under  $\ll$ . As for MSCs, an event can have at most two immediate predecessors, the innermost two loop is executed at most  $2 \cdot n$  times in total. This yields a total worst-case time complexity of  $n^2 + 2 \cdot n$ . □