Theoretical Foundations of the UML

Lecture 3: Message Sequence Graphs

Joost-Pieter Katoen

Lehrstuhl für Informatik 2
Software Modeling and Verification Group

http://moves.rwth-aachen.de/i2/uml09100/

16. Oktober 2012

RWTH AACHEN UNIVERSITY

Sets of MSCs

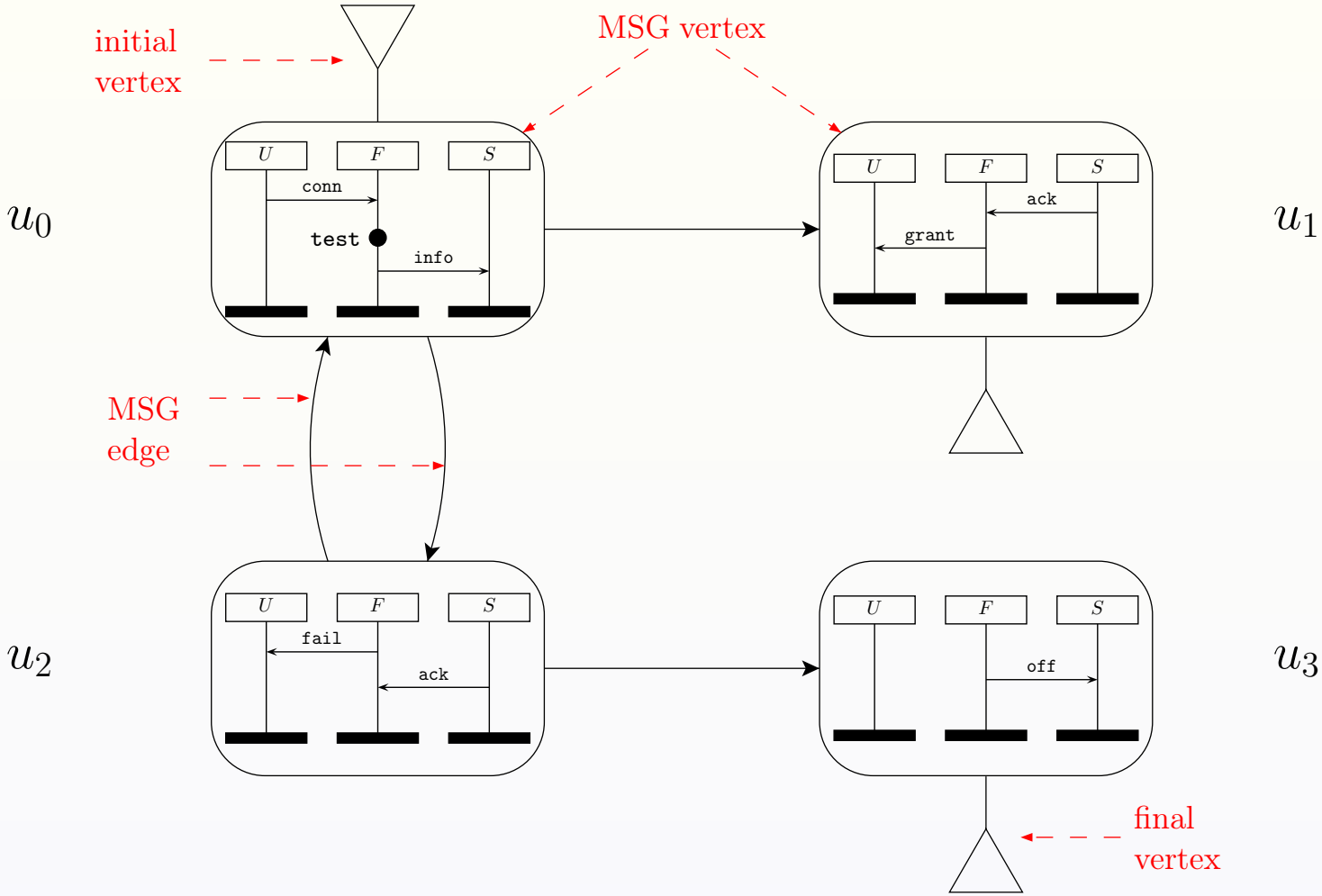
- MSC specifies a single scenario
- Typically: a set of scenarios
- + an ordering relation between them:
  - after scenario 1, scenario 2 occurs
  - after scenario 1, scenario 2 or 3 occurs
  - scenario 1 occurs repeatedly
- Need for: sequential composition (= concatenation), alternative composition, and iteration of MSCs

$\Rightarrow$ This yields Message Sequence Graphs

- Alternatives: ensembles of MSCs, high-level MSCs (MSC'96)

RWTH AACHEN UNIVERSITY

Message Sequence Graphs



$\underbrace{u_0\ u_2\ u_0\ u_1}_{\text{path in graph}}$ yields $\underbrace{\lambda(u_0) \bullet \lambda(u_2) \bullet \lambda(u_0) \bullet \lambda(u_1)}_{\text{MSC of the path}}$

Definition

Let $\mathbb{M}$ be the set of MSCs (up to isomorphism, i.e., event identities).

A Message Sequence Graph (MSG) $G = (V, \rightarrow, v_0, F, \lambda)$ with:

- $(V, \rightarrow)$ is a digraph with finite set V of vertices and $\rightarrow \subseteq V \times V$ a set of edges
- $v_0 \in V$ is the starting (or: initial) vertex
- $F \subseteq V$ is a set of final vertices
- $\lambda : V \rightarrow \mathbb{M}$ associates to each vertex $v \in V$, an MSC $\lambda(v)$

Note:

1. an MSG is an NFA without input alphabet where states are MSCs
2. every MSC is an MSG

RWTH AACHEN UNIVERSITY

Concatenation of MSCs (1)

Let $M_i = (\mathcal{P}_i, E_i, \mathcal{C}_i, l_i, m_i, <_i) \quad i \in \{1, 2\}$
be two MSCs with $E_1 \cap E_2 = \varnothing$

The concatenation of M_1 and M_2 is the MSC
$M_1 \bullet M_2 = (\mathcal{P}, E, \mathcal{C}, l, m, <)$ with:

$$\mathcal{P} = \mathcal{P}_1 \cup \mathcal{P}_2 \qquad E = E_1 \cup E_2 \quad (\text{with } E_? = E_{1,?} \cup E_{2,?} \text{ etc.}) \qquad \mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$$

$$l(e) = \begin{cases} l_1(e) & \text{if} \quad e \in E_1 \\ l_2(e) & \text{if} \quad e \in E_2 \end{cases} \qquad m(e) = \begin{cases} m_1(e) & \text{if} \quad e \in E_1 \\ m_2(e) & \text{if} \quad e \in E_2 \end{cases}$$

$$< = \big(<_1 \cup <_2 \cup \{(e, e') \mid \exists p \in \mathcal{P}.\, e \in E_1 \cap E_p\,,\ e' \in E_2 \cap E_p\} \big)^*$$

RWTH AACHEN UNIVERSITY

Concatenation of MSCs (2)

Note

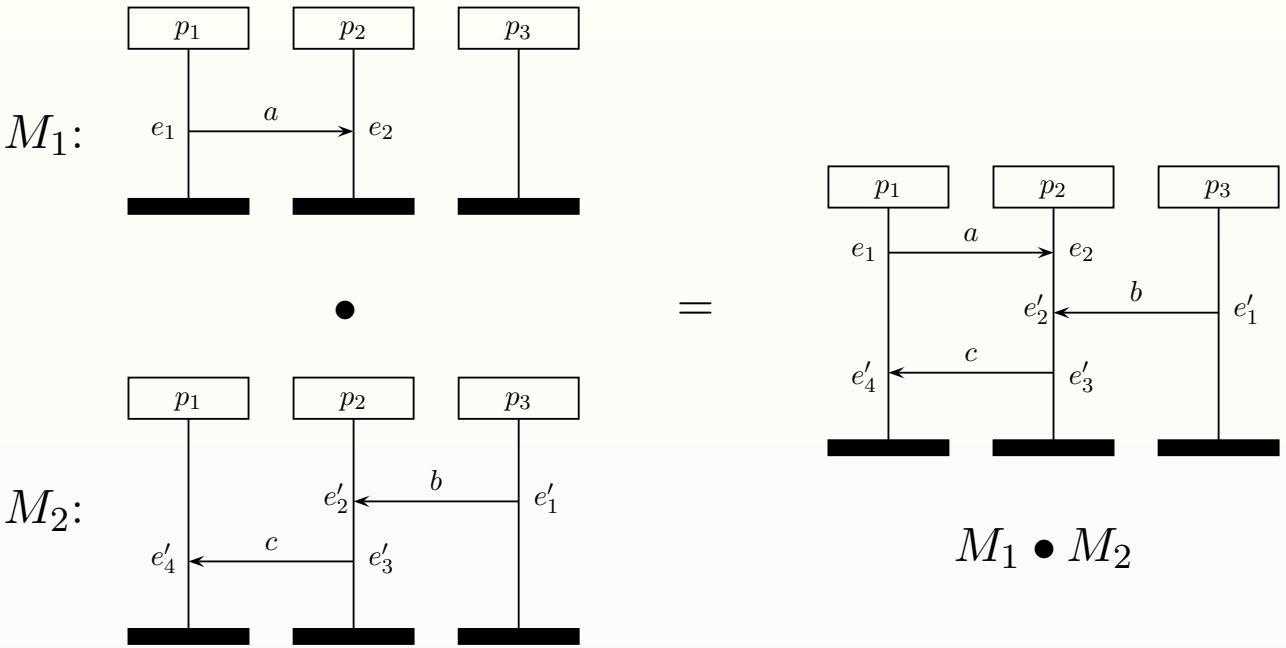
- events are ordered process-wise:

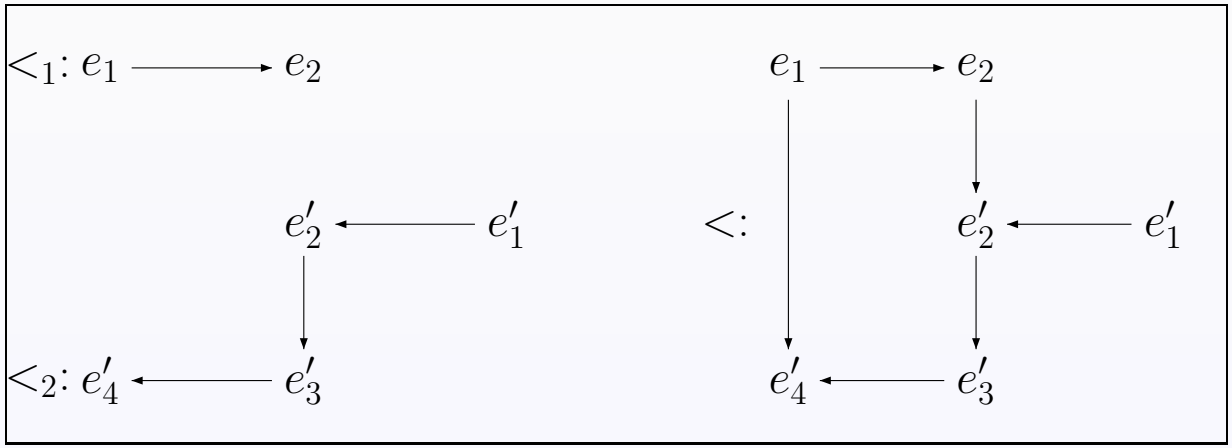
  $$events\ at\ p\ in\ MSC\ M_1\ precede\ events\ at\ p\ in\ MSC\ M_2$$

- thus: some processes may proceed to M_2 before others!
- $\neq$: first complete M_1 then execute M_2

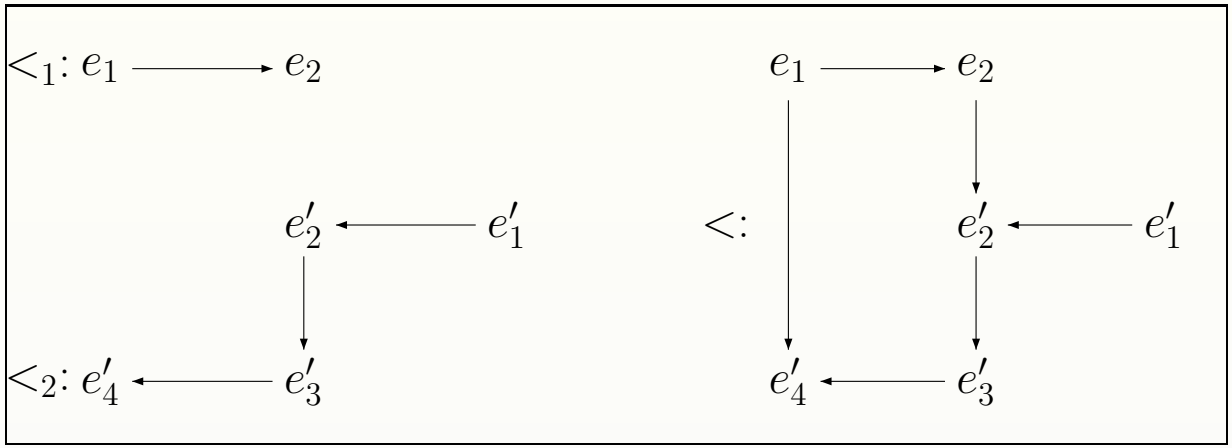
RWTH AACHEN UNIVERSITY

Example (1)


M_1:
p_1
p_2
p_3
e_1
a
e_2
$\bullet$
M_2:
e'_2
b
e'_1
e'_4
c
e'_3
$=$
$M_1 \bullet M_2$


$<_1$: e_1 e_2 e'_2 e'_1 $<_2$: e'_4 e'_3
$<$:

Example (2)



Note:

Events e_1 and e'_1 are not ordered in $M_1 \bullet M_2$

Example:

$e_1 \quad e_2 \quad e'_1 \quad e'_2 \ldots \in Lin(M_1 \bullet M_2)$

$e'_1 \quad e_1 \quad e_2 \quad e'_2 \ldots \in Lin(M_1 \bullet M_2)$

RWTH AACHEN UNIVERSITY

Properties

1. Concatenation is associative:

$$(M_1 \bullet M_2) \bullet M_3 \;=\; M_1 \bullet (M_2 \bullet M_3)$$

2. Concatenation preserves the FIFO property:

$$M_1 \text{ is FIFO } \wedge M_2 \text{ is FIFO} \quad \text{implies} \quad M_1 \bullet M_2 \text{ is FIFO}$$

3. Race-freeness, however, is not preserved

$$M_1 \text{ is race-free } \wedge M_2 \text{ is race-free} \quad \not\Rightarrow \quad M_1 \bullet M_2 \text{ is race-free}$$

RWTH AACHEN UNIVERSITY

Preliminaries

Let $G = (V, \rightarrow, v_0, F, \lambda)$ be an MSG.

Definition

A path π of G is a finite sequence

$$\pi = u_0\ u_1\ \ldots\ u_n \text{ with } u_i \in V\ \ (0 \leq i \leq n) \text{ and } u_i \rightarrow u_{i+1}\ \ (0 \leq i < n)$$

Definition

Path $\pi = u_0\ \ldots\ u_n$ is accepting if: $u_0 = v_0$ and $u_n \in F$.

Definition

The MSC of a path $\pi = u_0\ \ldots\ u_n$ is:

$$M(\pi) = \underbrace{\lambda(u_0)}_{\text{MSC of } u_0} \bullet \underbrace{\lambda(u_1)}_{\text{MSC of } u_1} \bullet \ldots \bullet \underbrace{\lambda(u_n)}_{\text{MSC of } u_n}$$

Language of an MSG

Definition

The (MSC) language of MSG G is defined by:

$$L(G) = \{M(\pi) \mid \pi \text{ is an accepting path of } G\}.$$

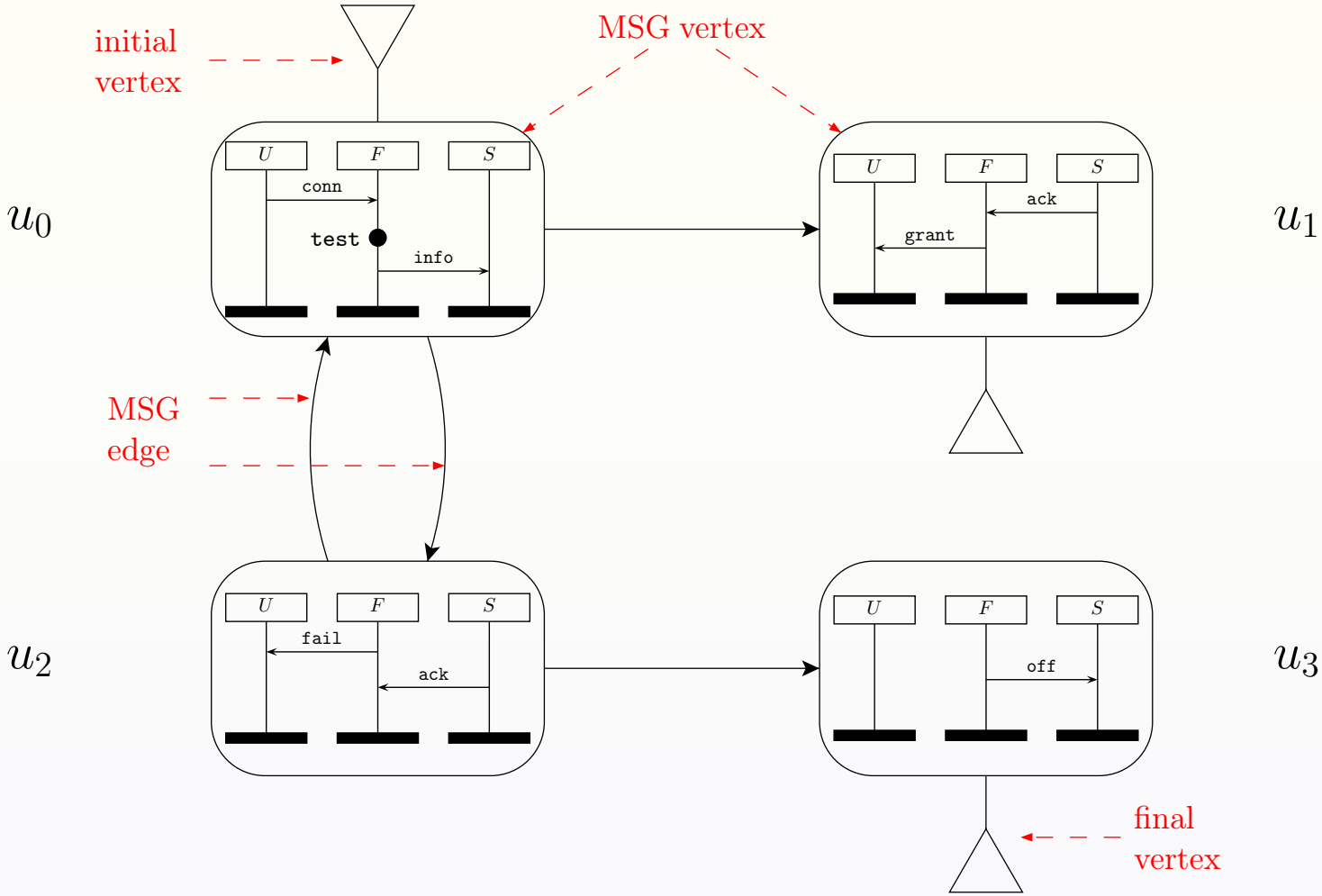
Definition

The word language of MSG G is $Lin(L(G))$ where

$$Lin(\{M_1, \ldots, M_k\}) = \bigcup_{i=1}^{k} Lin(M_i).$$

RWTH AACHEN UNIVERSITY

Example



$u_0\ u_2\ u_0\ u_1$ is accepting; $u_0\ u_2\ u_0\ u_2$ is not accepting

Races in MSGs

Recall: MSC M has a race if $< \not\subseteq \ll^*$
or, equivalently $Lin(E, <) \not\subseteq Lin(E, \ll^*)$
or, equivalently $Lin(E, <) \subset Lin(E, \ll^*)$

Definition

MSG G has a race if $Lin(G, <) \subset Lin(E, \ll^*)$

Theorem ([Muscholl & Peled '99])

The decision problem "MSG G has a race" is undecidable.

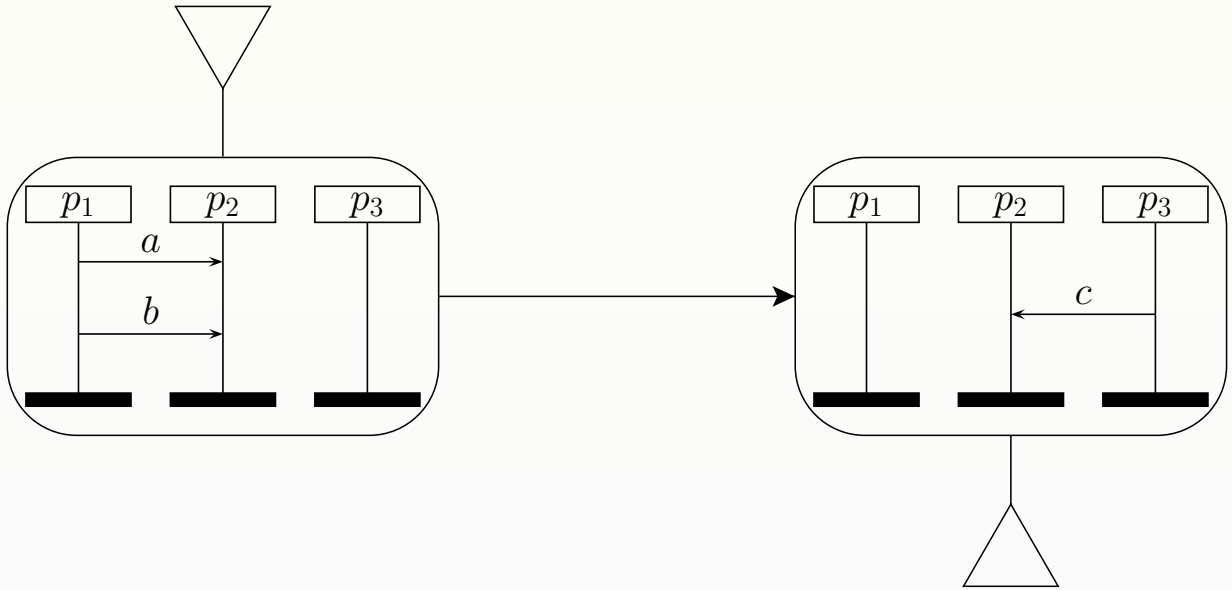
Proof.

by a reduction from Post's Correspondence Problem (PCP). Not easy.
We will see a similar—though simpler—proof later on. □

RWTH AACHEN UNIVERSITY

Example

G:



MSG G has a race.

Expressiveness of MSGs (1)

Fact 1:

The state space of an MSG G may be infinite.

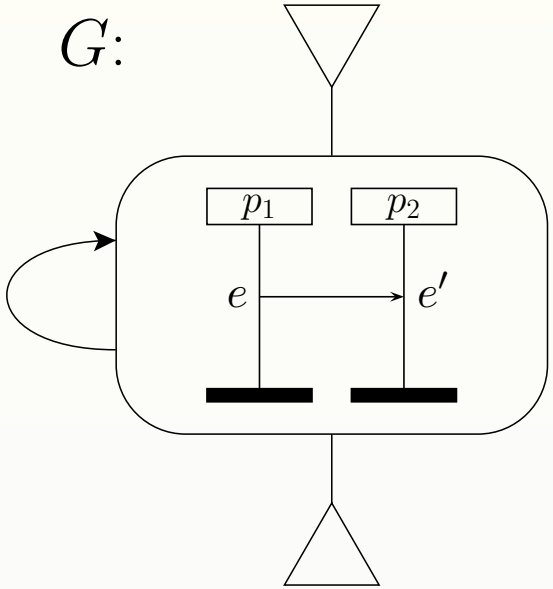
The state of an MSC with event set E is $E' \subseteq E$ such that $e \in E' \wedge e' < e \Longrightarrow e' \in E'$ (i.e., E' is downward-closed wrt. $<$)

The set of states of MSC M is M's state space

The state space of MSG G is the union of the state spaces of M_i for all $M_i \in L(G)$.

RWTH AACHEN UNIVERSITY

Example



MSG G is infinite state

A possible state is $\{e^{(1)}, e^{(2)}, e^{(3)}, \ldots\}$
(where $e^{(i)}$ is the occurrence of e in the i-th iteration)

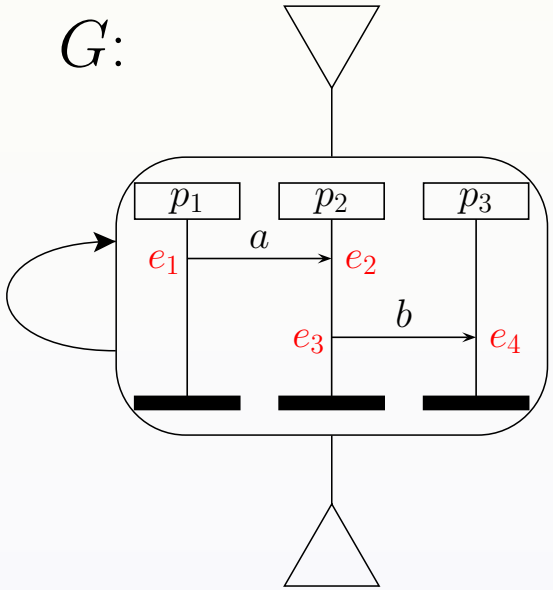
$\Longrightarrow$ system that realizes G requires unbounded communication channel

RWTH AACHEN UNIVERSITY

Expressiveness of MSGs (2)

Fact 2:

The state space of an MSG may not be context-free.



States of G are of the form $\{e_1^k\ e_2^l\ e_3^m\ e_4^n \mid k \geq l \geq m \geq n\}$

This language is not context-free

RWTH AACHEN UNIVERSITY

Expressiveness of MSGs (3)

Fact 3:

The state space of an MSG is context-sensitive.

Let $w, w' \in E^*$, and M an MSC with event set E. Then it holds:

(1) $w\ e\ e'\ w' \in Lin(M)\ ,\quad l(e) = ?(q,p,b)$
$\qquad\qquad\qquad\qquad\qquad\qquad\quad l(e') = !(p,q,a)$

implies $\quad w\ e'\ e\ w' \in Lin(M).$ not the reverse!

(2) $w\ e\ e'\ w' \in Lin(M)\ ,\quad l(e) = !(p,q,a)$ and
$\qquad\qquad\qquad\qquad\qquad\qquad\quad l(e') = ?(q,p,b)$

$$\underbrace{\sum_{m \in \mathcal{C}} |w|_{!(p,q,m)}}_{\text{number of sends from } p \text{ to } q \text{ in } w} > \underbrace{\sum_{m \in \mathcal{C}} |w|_{?(q,p,m)}}_{\text{number of receipts of } q \text{ from } p \text{ in } w}$$

implies $\quad w\ e'\ e\ w' \in Lin(M).$

RWTH AACHEN UNIVERSITY

Expressiveness of MSGs (4)

(3) $w\ e\ e'\ w' \in Lin(M)\ ,\ e \in E_p\ ,\ e' \in E_q\ ,\ p \neq q$
and e, e' do not match like in (1) or (2) (cf. previous slide)

implies $w\ e'\ e\ w' \in Lin(M)$.

Note:

Rule (2) is a context-sensitive rule of form $X\ a\ b\ Y \longrightarrow X\ b\ a\ Y$ as its applicability depends on the number of sends and receipts in the context X.

Note:

The results so far do not imply that any context-sensitive language is MSG-definable.

Context sensitivity (informal argument)

- Take MSG G and use vertex identities as vertex labels.
- $K(G)$ = set of "accepting" vertex sequences (this is regular)
- Replace each vertex v by $Lin(\lambda(v))$
  (interpret sequencing element wise)
- Let the resulting set be $\widetilde{K}(G)$ (this is regular)
- Close $\widetilde{K}(G)$ under the permutation rules (1), (2), (3)
  (cf. previous two slides)

The resulting word language is context-sensitive.

RWTH AACHEN UNIVERSITY