# Theoretical Foundations of the UML

## Lecture 5+6: Compositional Message Sequence Graphs

Joost-Pieter Katoen

Lehrstuhl für Informatik 2
Software Modeling and Verification Group

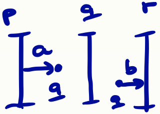
moves.rwth-aachen.de/teaching/ss-20/fuml/

May 4, 2020

# Outline

1. A non-decomposable MSC

motivation




2. Compositional Message Sequence Charts
3. Compositional Message Sequence Graphs
4. Safe Compositional Message Sequence Graphs
5. Existence of Safe Paths
6. Universality of Safe Paths

two decision poblems

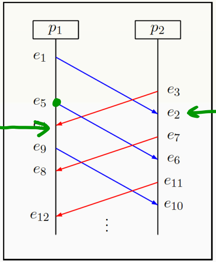
undecidable

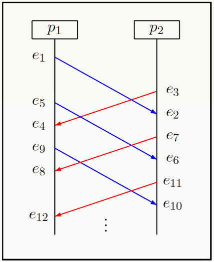
decidable

# Overview

1. A non-decomposable MSC
2. Compositional Message Sequence Charts
3. Compositional Message Sequence Graphs
4. Safe Compositional Message Sequence Graphs
5. Existence of Safe Paths
6. Universality of Safe Paths

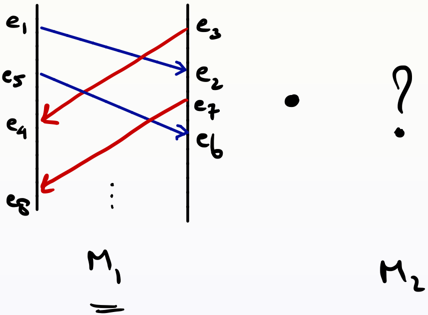
# An MSC that cannot be decomposed [Yannakakis 1999]

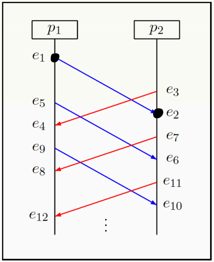





This MSC cannot be decomposed as

$$M_1 \bullet M_2 \bullet \ldots \bullet M_n \quad \text{for } n > 1$$





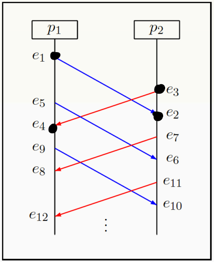

This MSC cannot be decomposed as

$$M_1 \bullet M_2 \bullet \ldots \bullet M_n \quad \text{for } n > 1$$

This can be seen as follows:

- $e_1$ and $e_2 = m(e_1)$ must both belong to $M_1$

# An MSC that cannot be decomposed [Yannakakis 1999]



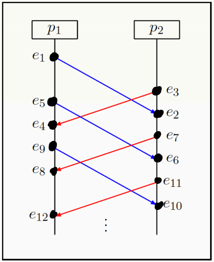

This MSC cannot be decomposed as

$$M_1 \bullet M_2 \bullet \ldots \bullet M_n \quad \text{for } n > 1$$

This can be seen as follows:

- $e_1$ and $e_2 = m(e_1)$ must both belong to $M_1$
- $e_3 \preceq e_2$ and $e_1 \preceq e_4$ thus
  $e_3, e_4 \notin M_j$ , for $j < 1$ and $j > 1$
  $\Longrightarrow e_3, e_4$ must belong to $M_1$

# An MSC that cannot be decomposed [Yannakakis 1999]



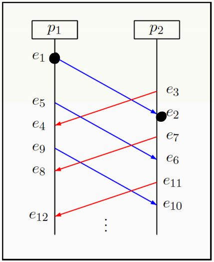

This MSC cannot be decomposed as

$$M_1 \bullet M_2 \bullet \ldots \bullet M_n \quad \text{for } n > 1$$

This can be seen as follows:

- $e_1$ and $e_2 = m(e_1)$ must both belong to $M_1$
- $e_3 \preceq e_2$ and $e_1 \preceq e_4$ thus
  $e_3, e_4 \notin M_j$ , for $j < 1$ and $j > 1$
  $\Longrightarrow e_3, e_4$ must belong to $M_1$
- by similar reasoning: $e_5, e_6 \in M_1$ etc.

# An MSC that cannot be decomposed [Yannakakis 1999]




This MSC cannot be decomposed as

MSG $\quad \boxed{M_1} \bullet \boxed{M_2} \bullet \ldots \bullet \boxed{M_n}$ for $n > 1$

This can be seen as follows:

- $\underline{e_1}$ and $e_2 = m(e_1)$ must both belong to $M_1$
- $e_3 \preceq e_2$ and $e_1 \preceq e_4$ thus
  $e_3, e_4 \notin M_j$ , for $j < 1$ and $j > 1$
  $\Longrightarrow e_3, e_4$ must belong to $M_1$
- by similar reasoning: $e_5, e_6 \in M_1$ etc.

**Problem:**

Compulsory matching between send and receive events in the same MSG vertex (i.e., send $e$ and receive $m(e)$ must belong to <u>the same MSC</u>).

# Overview


1 A non-decomposable MSC

2 Compositional Message Sequence Charts

3 Compositional Message Sequence Graphs

4 Safe Compositional Message Sequence Graphs

5 Existence of Safe Paths

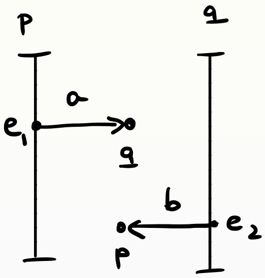
6 Universality of Safe Paths


compulsory matching between send and receive events within the same MSC is relaxed

## Compositional MSCs [Gunter, Muscholl, Peled 2001]

Solution: drop restriction that $e$ and $\underbrace{m(e)}$ belong to the same MSC
(= allow for incomplete message transfer)

one MSC M



# Compositional MSCs [Gunter, Muscholl, Peled 2001]

Solution: drop restriction that $e$ and $m(e)$ belong to the same MSC
(= allow for incomplete message transfer)

## Definition (Compositional MSC)

$M = (\mathcal{P}, E, \mathcal{C}, l, m, \preceq)$ is a compositional MSC (CMSC, for short)
where $\mathcal{P}, E, \mathcal{C}$ and $l$ are defined as before, and

processes

events

message content

event labeling

$l : E \longrightarrow \begin{array}{l} !(p,q,m) \\ ?(q,p,m) \end{array}$

Solution: drop restriction that $e$ and $m(e)$ belong to the same MSC (= allow for incomplete message transfer)

## Definition (Compositional MSC)

$M = (\mathcal{P}, E, \mathcal{C}, l, m, \preceq)$ is a compositional MSC (CMSC, for short) where $\mathcal{P}, E, \mathcal{C}$ and $l$ are defined as before, and

- $m : E_! \to E_?$ is a partial, injective function such that (as before):

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

in MSCs, it is a bijection

in MSCs it is a total function

injective $e_1, e_2 \in E_!$ :

$e_1 \neq e_2 \Rightarrow m(e_1) \neq m(e_2)$

Solution: drop restriction that $e$ and $m(e)$ belong to the same MSC (= allow for incomplete message transfer)

### Definition (Compositional MSC)

$M = (\mathcal{P}, E, \mathcal{C}, l, m, \preceq)$ is a compositional MSC (CMSC, for short) where $\mathcal{P}, E, \mathcal{C}$ and $l$ are defined as before, and

- $m : E_! \to E_?$ is a partial, injective function such that (as before):

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

- $$\preceq \; = \Big( \underbrace{\bigcup_{p \in \mathcal{P}} <_p}_{\text{vertical ordering}} \; \cup \; \underbrace{\{(e, m(e)) \mid e \in \underbrace{\underbrace{dom(m)}_{\text{domain of } m}}_{\text{“}m(e)\text{ is defined”}} \}}_{\text{horizontal ordering}} \Big)^*$$

# Compositional MSCs [Gunter, Muscholl, Peled 2001]

Solution: drop restriction that $e$ and $m(e)$ belong to the same MSC (= allow for incomplete message transfer)

## Definition (Compositional MSC)

$M = (\mathcal{P}, E, \mathcal{C}, l, m, \preceq)$ is a compositional MSC (CMSC, for short) where $\mathcal{P}, E, \mathcal{C}$ and $l$ are defined as before, and

- $m : E_! \rightarrow E_?$ is a partial, injective function such that (as before):

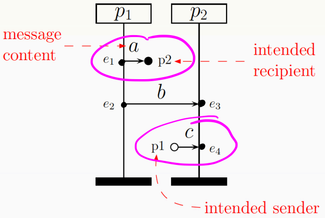
$$m(e) = e' \wedge l(e) = !(p, q, a) \quad \text{implies} \quad l(e') = ?(q, p, a)$$

- $\preceq \; = \Big( \bigcup_{p \in \mathcal{P}} <_p \quad \cup \quad \underbrace{\{(e, m(e)) \mid e \in \underbrace{dom(m)}_{\text{domain of } m}\}}_{\text{"}m(e)\text{ is defined"}} \Big)^*$

## Note:

An MSC is a CMSC where $m$ is total and bijective.

# CMSC example



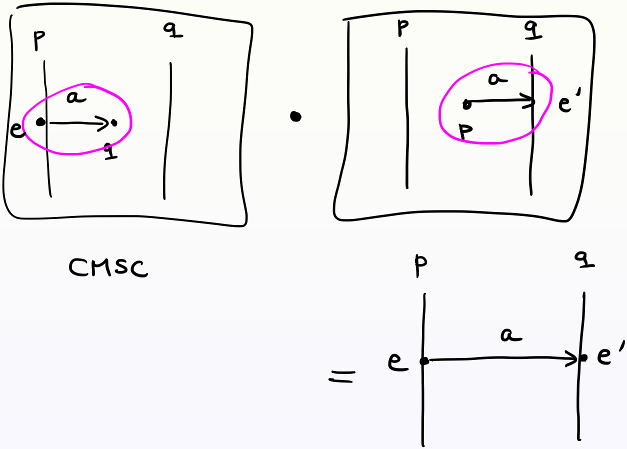

- $m(e_2) = e_3$
- $e_1 \notin dom(m)$
- $e_4 \notin rng(m)$ — $\neg \exists e.\ m(e) = e_4$

$E_! = \{e_1, e_2\}$

$E_? = \{e_3, e_4\}$

the set of compositional MSCs

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




idea

# Concatenation of CMSCs (1)

Let $M_i = (\mathcal{P}_i, E_i, \mathcal{C}_i, l_i, m_i, \preceq_i) \in \mathbb{CM} \qquad i \in \{1, 2\}$
be CMSCs with $E_1 \cap E_2 = \varnothing$

The concatenation of CMSCs $M_1$ and $M_2$ is the CMSC
$M_1 \bullet M_2 = (\mathcal{P}_1 \cup \mathcal{P}_2,\ E,\ \mathcal{C}_1 \cup \mathcal{C}_2, l, m, \preceq)$ with:

# Concatenation of CMSCs (1)

Let $M_i = (\mathcal{P}_i, E_i, \mathcal{C}_i, l_i, m_i, \preceq_i) \in \mathbb{CM} \qquad i \in \{1, 2\}$
be CMSCs with $E_1 \cap E_2 = \varnothing$

The concatenation of CMSCs $M_1$ and $M_2$ is the CMSC
$M_1 \bullet M_2 = (\underline{\mathcal{P}_1 \cup \mathcal{P}_2,\ E,\ \mathcal{C}_1 \cup \mathcal{C}_2, l,} m, \preceq)$ with:

- $E = E_1 \cup E_2$
- $l(e) = \underline{l_1(e)}$ if $\underline{e \in E_1}$ , $\underline{l_2(e)}$ otherwise

all the same as for concatenation of MSCs.

# Concatenation of CMSCs (1)

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

The concatenation of CMSCs $M_1$ and $M_2$ is the CMSC
$M_1 \bullet M_2 = (\mathcal{P}_1 \cup \mathcal{P}_2,\ E,\ \mathcal{C}_1 \cup \mathcal{C}_2, l, m, \preceq)$ with:

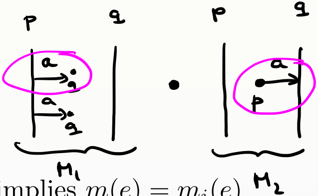
- $E = E_1 \cup E_2$
- $l(e) = l_1(e)$ if $e \in E_1$ , $l_2(e)$ otherwise
- $m(e) = E_! \to E_?$ satisfies:
  1. $m$ extends $m_1$ and $m_2$, i.e., $e \in dom(m_i)$ implies $m(e) = m_i(e)$

for events in $E_i$ for which $m_i$ is defined
the matching event remains the same

# Concatenation of CMSCs (1)

Let $M_i = (\mathcal{P}_i, E_i, \mathcal{C}_i, l_i, m_i, \preceq_i) \in \mathbb{CM} \qquad i \in \{1, 2\}$
be CMSCs with $E_1 \cap E_2 = \varnothing$

The concatenation of CMSCs $M_1$ and $M_2$ is the CMSC
$M_1 \bullet M_2 = (\mathcal{P}_1 \cup \mathcal{P}_2,\ E,\ \mathcal{C}_1 \cup \mathcal{C}_2, l, m, \preceq)$ with:



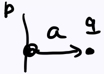

- $E = E_1 \cup E_2$
- $l(e) = l_1(e)$ if $e \in E_1$ , $l_2(e)$ otherwise
- $m(e) = E_! \to E_?$ satisfies:
  1. $m$ extends $m_1$ and $m_2$, i.e., $e \in dom(m_i)$ implies $m(e) = m_i(e)$
  2. $m$ matches unmatched send events in $M_1$ with unmatched receive events in $M_2$ according to order on process (matching from top to bottom)

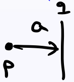
# Concatenation of CMSCs (1)

Let $M_i = (\mathcal{P}_i, E_i, \mathcal{C}_i, l_i, m_i, \preceq_i) \in \mathbb{CM} \qquad i \in \{1, 2\}$
be CMSCs with $E_1 \cap E_2 = \varnothing$

The concatenation of CMSCs $M_1$ and $M_2$ is the CMSC
$M_1 \bullet M_2 = (\mathcal{P}_1 \cup \mathcal{P}_2,\ E,\ \mathcal{C}_1 \cup \mathcal{C}_2, l, m, \preceq)$ with:

- $E = E_1 \cup E_2$
- $l(e) = l_1(e)$ if $e \in E_1$ , $l_2(e)$ otherwise
- $m(e) = E_! \rightarrow E_?$ satisfies:
  1. $m$ extends $m_1$ and $m_2$, i.e., $e \in dom(m_i)$ implies $m(e) = m_i(e)$
  2. $m$ matches unmatched send events in $M_1$ with unmatched receive events in $M_2$ according to order on process (matching from top to bottom)
     the $k$-th unmatched send in $M_1$ is matched with
     the $k$-th unmatched receive in $M_2$ (of the same "type")




same message content
sender corresponds to receiver

# Concatenation of CMSCs (1)

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

The concatenation of CMSCs $M_1$ and $M_2$ is the CMSC
$M_1 \bullet M_2 = (\mathcal{P}_1 \cup \mathcal{P}_2,\ E,\ \mathcal{C}_1 \cup \mathcal{C}_2, l, m, \preceq)$ with:

- $E = E_1 \cup E_2$
- $l(e) = l_1(e)$ if $e \in E_1$ , $l_2(e)$ otherwise
- $m(e) = E_! \rightarrow E_?$ satisfies:
  1. $m$ extends $m_1$ and $m_2$, i.e., $e \in dom(m_i)$ implies $m(e) = m_i(e)$
  2. $m$ matches unmatched send events in $M_1$ with unmatched receive events in $M_2$ according to order on process (matching from top to bottom)
     the $k$-th unmatched send in $M_1$ is matched with
     the $k$-th unmatched receive in $M_2$ (of the same "type")
  3. $M_1 \bullet M_2$ is FIFO (when restricted to matched events)

# Concatenation of CMSCs (2)

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

The concatenation of CMSCs $M_1$ and $M_2$ is the CMSC
$M_1 \bullet M_2 = (\mathcal{P}_1 \cup \mathcal{P}_2, E_1 \cup E_2, \mathcal{C}_1 \cup \mathcal{C}_2, l, m, \preceq)$ with:

# Concatenation of CMSCs (2)

Let $M_i = (\mathcal{P}_i, E_i, \mathcal{C}_i, l_i, m_i, \preceq_i) \in \mathbb{CM} \qquad i \in \{1, 2\}$
be CMSCs with $E_1 \cap E_2 = \varnothing$

The concatenation of CMSCs $M_1$ and $M_2$ is the CMSC
$M_1 \bullet M_2 = (\mathcal{P}_1 \cup \mathcal{P}_2, E_1 \cup E_2, \mathcal{C}_1 \cup \mathcal{C}_2, l, m, \preceq)$ with:

- $l$ and $m$ are defined as on the previous slide
- $\preceq$ is the reflexive and transitive closure of:

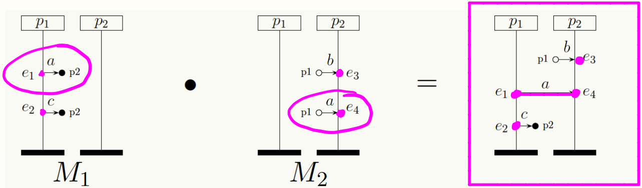
$$\underbrace{\left( \bigcup_{p \in \mathcal{P}} <_{p,1} \cup <_{p,2} \right)}_{\text{process order}} \cup \{(e, e') \mid e \in E_1 \cap E_p,\ e' \in E_2 \cap E_p\}$$
$$\cup\ \{\underbrace{(e, m(e)) \mid e \in dom(m)}\}$$

process wise: all events at process $p$ in $M_2$ happen after all event at $p$ in $M_1$.

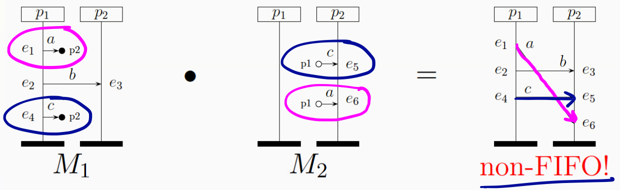
# Examples

# Examples

(1)

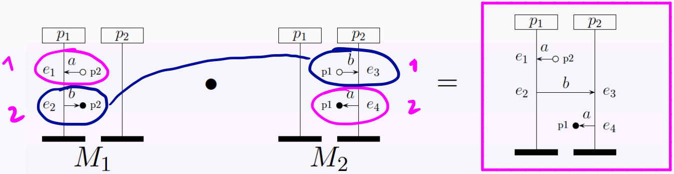



(2)

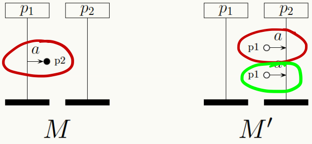



for this $M_1$ and $M_2$, $M_1 \bullet M_2$ is not defined

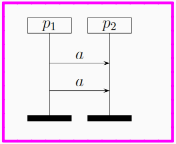
(3)



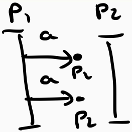
# Associativity



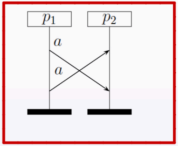

① $(M \bullet M) \bullet M'$:




$M \bullet M$




② $M \bullet (M \bullet M')$:

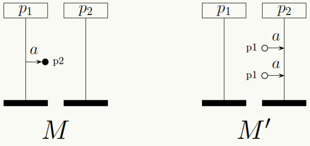



$\Longrightarrow$ this is non-FIFO (and thus undefined)

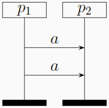
For MSCs, $\bullet$ is associative $(M_1 \bullet M_2) \bullet M_3 = M_1 \bullet (M_2 \bullet M_3)$

no longer for cMSCs

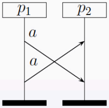
# Associativity




$(M \bullet M) \bullet M'$:




$M \bullet (M \bullet M')$:




$\Longrightarrow$ this is non-FIFO
(and thus undefined)

**Note:**

Concatenation of CMSCs is <u>not</u> associative.

# Overview


1. A non-decomposable MSC
2. Compositional Message Sequence Charts
3. Compositional Message Sequence Graphs
4. Safe Compositional Message Sequence Graphs
5. Existence of Safe Paths
6. Universality of Safe Paths

## Compositional MSG

graph

finite set of vertices

initial vertex $v_0 \in V$

Let $\mathbb{CM}$ be the set of all CMSCs.

### Definition (Compositional MSG)

A compositional MSG (CMSG) $G = (V, \rightarrow, v_0, F, \lambda)$ with $\lambda : V \rightarrow \mathbb{CM}$, where $V, \rightarrow, v_0,$ and $F$ as for MSGs.

final vertices

$\subseteq V \times V$

The difference with an MSG is that the vertices in a CMSG are labeled with compositional MSCs (rather than "real" MSCs).

# Paths

# Paths

Let $G = (V, \rightarrow, v_0, F, \lambda)$ be a CMSG.

# Paths

Let $G = (V, \rightarrow, v_0, F, \lambda)$ be a CMSG.

## Definition (Path in a CMSG)

A path $\pi$ of $G$ is a finite sequence

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

# Paths

Let $G = (V, \rightarrow, v_0, F, \lambda)$ be a CMSG.

## Definition (Path in a CMSG)

A path $\pi$ of $G$ is a finite sequence

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

## Definition (Accepting path of a CMSG)

Path $\pi = u_0\ \ldots\ u_n$ is accepting if: $\underbrace{u_0 = v_0}_{\text{start in initial vertex}}$ and $\underbrace{u_n \in F}_{\text{end in accepting vertex}}$.

# Paths

Let $G = (V, \rightarrow, v_0, F, \lambda)$ be a CMSG.

## Definition (Path in a CMSG)

A path $\pi$ 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 (Accepting path of a CMSG)

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

## Definition (CMSC of a path)

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

$$M(\pi) \; = \; (\ldots(\lambda(u_0) \bullet \lambda(u_1)) \bullet \lambda(u_2) \ldots) \bullet \lambda(u_n)$$

where CMSC concatenation is left associative.

# The MSC language of a CMSG

## Definition (Language of a CMSG)

The (MSC) language of CMSG $G$ is defined by:

$$L(G) = \{ \underbrace{M(\pi) \in \mathbb{M}}_{\text{only "real" MSCs}} \mid \underbrace{\pi \text{ is an accepting path of } G}\}.$$

set of MSCs

CMSC of path $\pi$

$$\boxed{L(G) \subseteq \mathbb{M}}$$

not : $L(G) \subseteq \mathbb{CM} \setminus \mathbb{M}$

# The MSC language of a CMSG

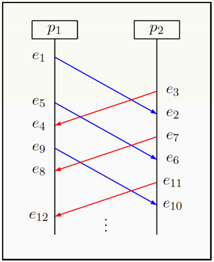
## Definition (Language of a CMSG)

The (MSC) language of CMSG $G$ is defined by:

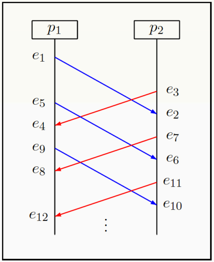
$$L(G) = \{ \underbrace{M(\pi) \in \mathbb{M}}_{\text{only "real" MSCs}} \mid \pi \text{ is an accepting path of } G \}.$$

Note: Accepting paths that give rise to an CMSC (which is not an MSC) are not part of $L(G)$.

# Yannakakis' example as compositional MSG



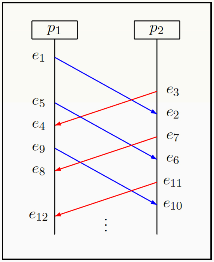
# Yannakakis' example as compositional MSG




This MSC cannot be modeled for $n > 1$ by:

$$M = \underbrace{M_1 \bullet M_2 \bullet \ldots \bullet M_n}_{} \quad \text{with} \quad M_i \in \mathbb{M}$$

# Yannakakis' example as compositional MSG

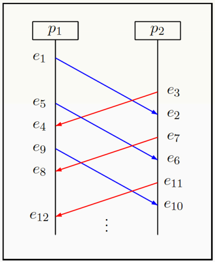



This MSC cannot be modeled for $n > 1$ by:

$$M = M_1 \bullet M_2 \bullet \ldots \bullet M_n \quad \text{with} \quad M_i \in \mathbb{M}$$

Thus it cannot be modeled by a MSG.

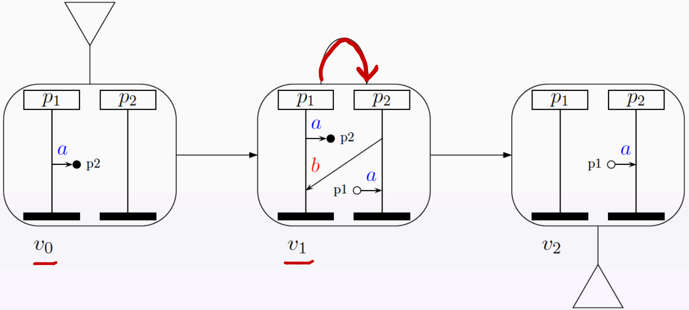
# Yannakakis' example as compositional MSG




This MSC cannot be modeled for $n > 1$ by:

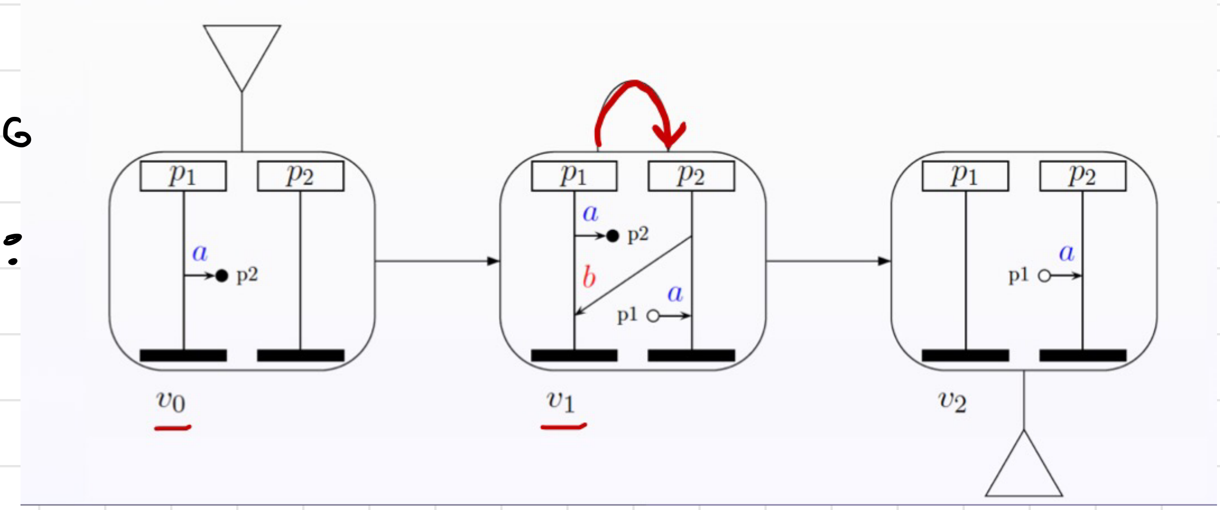
$$M = M_1 \bullet M_2 \bullet \ldots \bullet M_n \quad \text{with} \quad M_i \in \mathbb{M}$$

Thus it cannot be modeled by a MSG.
But it can be modeled as compositional MSG:

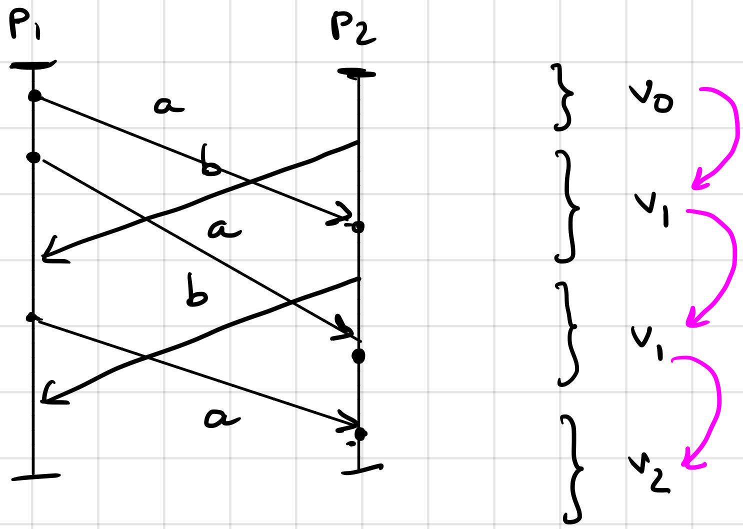


CMSG

$\mathcal{G}$ :




MSC $M_1 \in L(\mathcal{G})$




safe

Every accepting path $\pi$ for $G$ : $M(\pi)$ is an MSC

$\rightarrow M(\pi) \in L(\mathcal{G})$

CMSG $\mathcal{G}$ is called <u>safe</u>

# Overview

1 A non-decomposable MSC

2 Compositional Message Sequence Charts

3 Compositional Message Sequence Graphs

4 Safe Compositional Message Sequence Graphs

5 Existence of Safe Paths

6 Universality of Safe Paths

# Safe paths and CMSGs

# Safe paths and CMSGs

MSC

**Definition (Safe path)**

Path $\pi$ of CMSG $G$ is safe whenever $\underbrace{M(\pi)}_{\text{in the CMSC of } \pi} \in \mathbb{M}$.

# Safe paths and CMSGs

**Definition (Safe path)**

Path $\pi$ of CMSG $G$ is safe whenever $M(\pi) \in \mathbb{M}$.

**Definition (Safe CMSG)**

CMSG $G$ is safe if for every accepting path $\pi$ of $G$, $M(\pi)$ is an MSC.

# Safe paths and CMSGs

**Definition (Safe path)**

Path $\pi$ of CMSG $G$ is safe whenever $M(\pi) \in \mathbb{M}$.

**Definition (Safe CMSG)**

CMSG $G$ is safe if for every accepting path $\pi$ of $G$, $M(\pi)$ is an MSC.

**So:**

CMSG $G$ is safe if on any of its accepting paths there are no unmatched sends and receipts, i.e., if any of its accepting paths is indeed an MSC.

# Overview


1. A non-decomposable MSC
2. Compositional Message Sequence Charts
3. Compositional Message Sequence Graphs
4. Safe Compositional Message Sequence Graphs
5. Existence of Safe Paths
6. Universality of Safe Paths

# Existence of a safe accepting path

**Theorem: undecidability of existence of a safe path**

The decision problem "does CMSG $G$ have at least one safe, accepting path?" is undecidable.

# Existence of a safe accepting path

**Theorem: undecidability of existence of a safe path**

The decision problem "does CMSG $G$ have at least one safe, accepting path?" is undecidable.

**Proof.**

By a reduction from Post's Correspondence Problem (PCP).

… black board … $\square$

# Existence of a safe accepting path

**Theorem: undecidability of existence of a safe path**

The decision problem "does CMSG $G$ have at least one safe, accepting path?" is undecidable.

**Proof.**

By a reduction from Post's Correspondence Problem (PCP).

... black board ... □

The complement decision problem "does CMSG $G$ have no safe, accepting path?" is undecidable too.

Claim: the decision problem "does CMSG $\mathcal{G}$ have at least one safe, accepting path?" is undecidable.

Proof by a reduction from the PCP problem.

Proof idea: instance of PCP $(U, W) \longmapsto$ instance CMSG $G_{U,W}$

$$U = \{u_1, \ldots, u_n\} \qquad u_i \in \Sigma^*$$

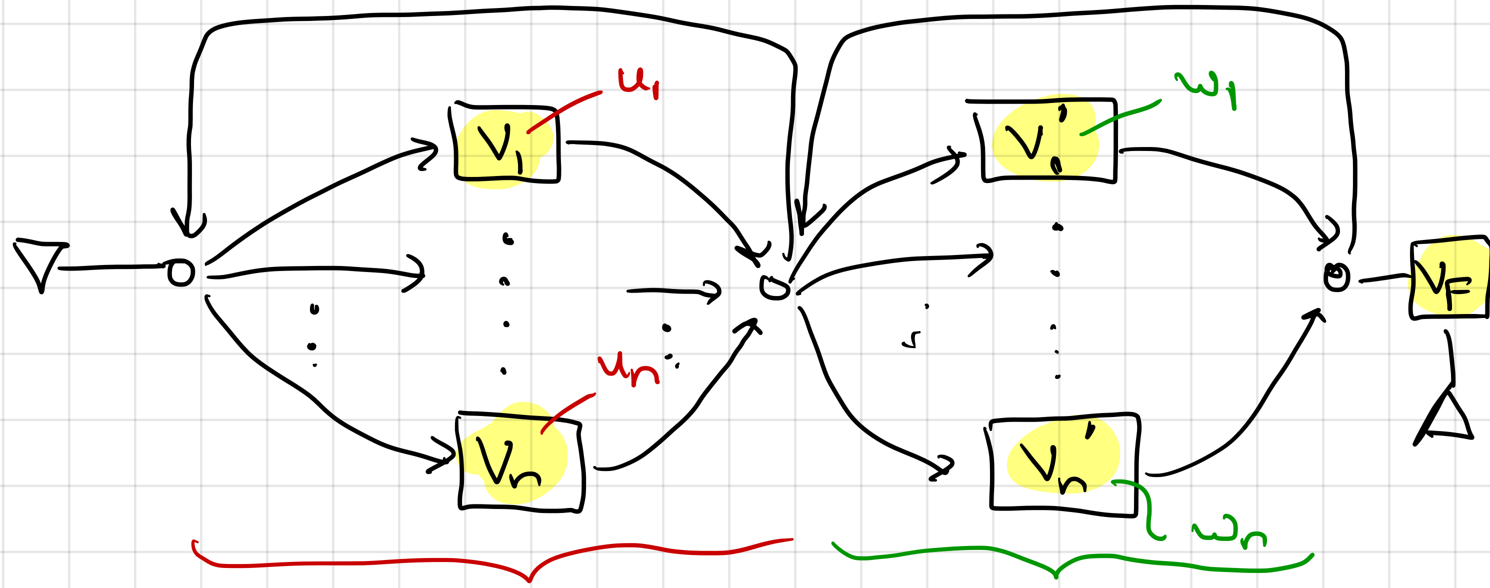
$$W = \{w_1, \ldots, w_n\} \qquad w_i \in \Sigma^*$$

such that $(U, W)$ has a solution if and only if CMSG $\mathcal{G}_{U,W}$ has a safe, accepting path.

$i_1, \ldots, i_k \qquad i_j \in [1..n]$

such that $u_{i_1} u_{i_2} \cdots u_{i_k} = w_{i_1} w_{i_2} \cdots w_{i_k}$

How does the CMSG $G_{U,W}$ look like?



Components of CMSG $\mathcal{G}_{u,w}$:

$P = \{p_1, p_2, p_3, p_4\}$ processes

$C = \Sigma \uplus \{end\} \uplus \underbrace{\{1, \dots, n\}}_{\text{indices } i_j}$

$V = \{v_1, \dots, v_n\} \cup \{v_1', \dots, v_n'\} \cup \{v_F\}$

$F = \{v_F\}$

$\lambda(v_i)$ = CMSG corresponding to the word $u_i$

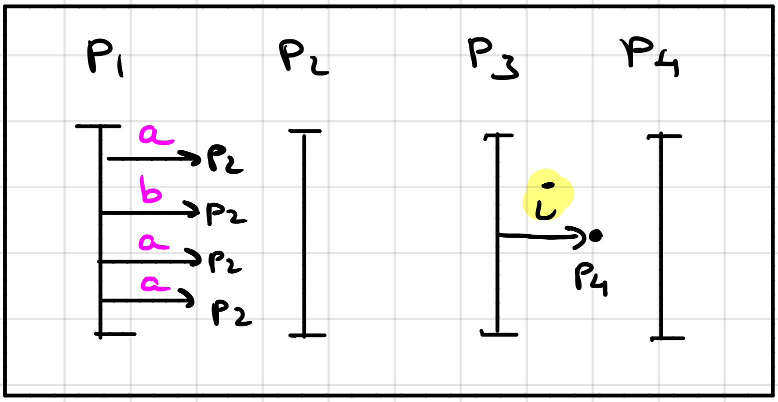
$\lambda(v_i')$ = " " " " $w_i$

How do the vertices $v_i$, $v_i'$ and $v_F$ look like?

By example. let $\Sigma = \{a, b\}$, $u_i = \underline{abaa}$,
$w_i = \underline{ba}$

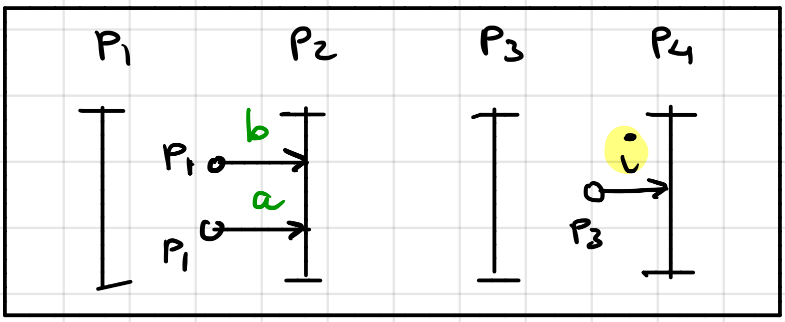
Then:

$\lambda(v_i) =$



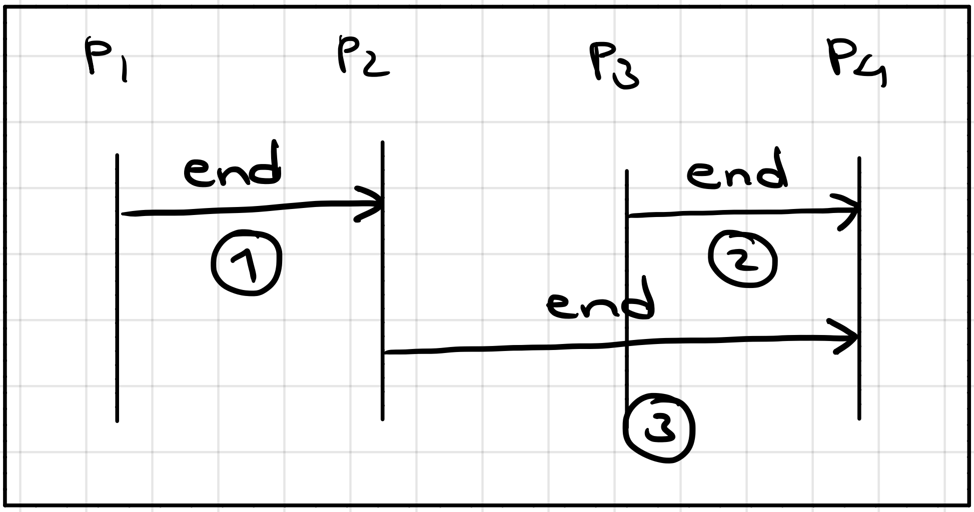

"send the word $u_i$ and index $i$"

$\lambda(v_i') =$




"receive the word $w_i$ plus index $i$"

$\lambda(v_F)$




1 indicates that process $p_1$ has sent all its messages to $p_2$ and if 1 is received by $p_2$, all messages of $p_1$ have been received by $p_2$.

2 similar as 1 but now for the "index" messages that are exchanged between $p_3$ & $p_4$.

3 indicates that both "phases" 1 and 2 have finished.

It remains to prove that the reduction:

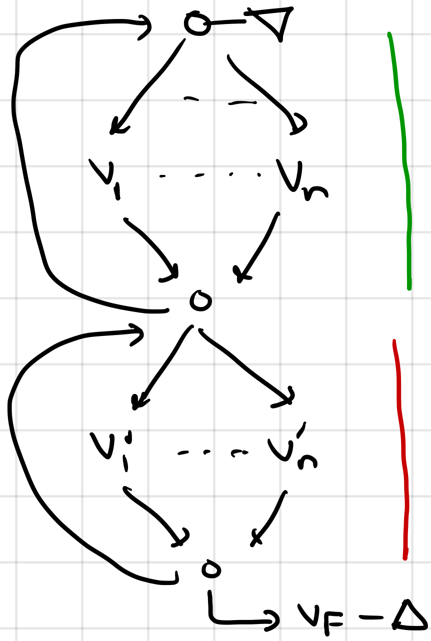
$$\text{PCP instance } (u,w) \longmapsto \text{CMSG } \mathcal{G}_{u,w}$$

is correct. That is, our proof obligation is:

$$\boxed{(u,w) \text{ has a solution iff } \mathcal{G}_{u,w} \text{ has a safe, accepting path}}$$

<u>Proof:</u> "$\Rightarrow$" let index sequence $i_1, \dots, i_k$ be a solution of PCP instance $(u,w)$. Then there is an accepting path in $\mathcal{G}_{u,w}$:

$$\pi = \underbrace{\overbrace{v_{i_1} \dots v_{i_k}}^{\text{"}u\text{"}}}_{\substack{\text{traverse the} \\ \text{"}v_i\text{" vertices} \\ \text{according to} \\ i_1 \dots i_k}} \; \underbrace{\overbrace{v'_{i_1} \dots v'_{i_k}}^{\text{"}w\text{"}}}_{\substack{\text{traverse the} \\ \text{"}v'_i\text{" vertices} \\ \text{according to} \\ i_1 \dots i_k}} \; v_F$$




As $i_1, \dots, i_k$ is a solution to $(u,w)$, and by construction of the CMSG $\mathcal{G}_{u,w}$ it follows that: $M(\pi) =$

$$(((\lambda(v_{i_1}) \cdot \dots) \cdot \lambda(v_{i_k})) \cdot \lambda(v'_{i_1}) \cdot \dots \cdot \lambda(v'_{i_k}) \cdot \lambda(v_F))$$

(left-associated bracketing) is an <u>MSC</u>.

Thus $\pi$ is safe and $\pi$ is accepting.

"$\Leftarrow$" let $\pi$ be a safe, accepting path in $G_{u,w}$

Assume:

$$\pi = \underbrace{v_{i_1} \dots v_{i_m}}_{m \text{ steps}} \; \underbrace{v'_{j_1} \, v'_{j_2} \dots v'_{j_k}}_{k \text{ steps}} \cdot v_F$$

with $i_1, \dots, i_m \in \{1, \dots, n\}$ and $j_1, \dots, j_k \in \{1, \dots, n\}$.

Since $\pi$ is safe and ends in vertex $v_F$, it follows:

1. as $?(p_4, p_3, end)$ occurs in $v_F$, all unmatched sends by $p_3$ in subpath $v_{i_1} \dots v_{i_m}$ are matched by corresponding receive events by $p_4$ in the subpath $v'_{j_1} \dots v'_{j_k}$. As in each vertex $v_{i_\ell}$ <u>one</u> message is sent from $p_3$ and in $v'_{j_\ell}$ <u>one</u> message is received by $p_4$, it follows that <u>$m = k$</u>

2. As $\pi$ is safe, it follows that

$$\underbrace{v_{i_1} \dots v_{i_m}}\; \underbrace{v'_{j_1} \dots v'_{j_m}} \text{ is safe and FIFO.}$$

Thus all "index" messages $i_1, \dots, i_m$ sent by $p_3$ are received by $p_4$ in the same order.

Thus $i_1 = j_1,\ i_2 = j_2,\ \dots,\ i_m = j_m$

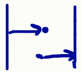
So $\pi = \underbrace{v_{i_1} \cdots v_{i_m}}\ \underbrace{v'_{i_1} \cdots v'_{i_m}}\ v_F$ is safe + accepting

As $\pi$ is completed by $?(p_4, p_2, end)$ and $!(p_2, p_4, end)$ after $?(p_2, p_1, end)$, it follows that once $p_4$ has received all "index" messages, $p_2$ has received all messages sent by $p_1$ in $\underbrace{v_{i_1} \cdots v_{i_m}}$

Process $p_1$ has sent $u_{i_1} \cdots u_{i_m}$ (to $p_2$),
process $p_2$ has received $w_{i_1} \cdots w_{i_m}$
Since $\pi$ is safe, it follows $u_{i_1} \cdots u_{i_m} = w_{i_1} \cdots w_{i_m}$
Thus: $i_1 \cdots i_m$ is a solution to the PCP instance $(u, w)$ $\boxtimes$

# Overview

1. A non-decomposable MSC ✓ Yannakakis
2. Compositional Message Sequence Charts ✓



3. Compositional Message Sequence Graphs ✓ safe path
4. Safe Compositional Message Sequence Graphs ✓
5. Existence of Safe Paths undecidable
6. **Universality of Safe Paths** — does a CMSG only have safe paths?