Checking MSC graphs with Petri nets

Rémi MORIN

Laboratoire d’Informatique Fondamentale de Marseille
Each send corresponds to a receive, and vice versa.
MSCs are regarded as labeled partial orders

In this talk, we do not distinguish message types.

As usual, we assume **FIFO communication**.
Consider a basic MSC $M = (E, \preceq, \xi)$.

Let $\text{Past} \subseteq E$ be a prefix consisting of send events.

Let $\text{Future} \subseteq E$ be a suffix consisting of receive events.

Then $(E \setminus (\text{Past} \cup \text{Future}), \preceq, \xi)$ is a (compositional) MSC.
Each accepted path of this MSG corresponds to a basic MSC.
The sliding window protocol as a non-safe MSG
Protocol with a bounded counter

\[
x = 0 \\
x \leq 10 \\
x \geq 1 \\
x = 0
\]
Background

Composition of compositional MSCs
Emptiness is undecidable
MSGs without matching arrow
The subclass of linear MSGs
Conclusion
Product of two compositional MSCs

The events of $A$ precede the events of $B$ for each process. The $n$-th send in $A$ matches the $n$-th receive for each channel.
Some products are obviously undefined (1/2)
Some products are obviously undefined (2/2)

MSC A

MSC B

Undefined A \cdot B
This product is not associative

\[ MSC \ A \cdot MSC \ B \neq MSC \ (A \cdot A) \cdot B \]

\[ MSC \ (A \cdot (A \cdot B)) \]

Undefined
A prefix MSC is an MSC without unmatched receive.

Prefix MSC

It corresponds intuitively to an execution starting from the configuration where all channels are empty: There is no pending message initially.
Some restrictions (1/2)

A must be a prefix MSC; otherwise $A \cdot B$ is undefined.

This restriction originates with [Gunter et al, TACAS 2001].
Some restrictions (2/2)

A · B must be a prefix MSC; otherwise A · B is undefined.

This restriction originates with [Gunter et al, TACAS 2001], too.
A path $v_0, ..., v_n$ from the initial vertex $v_0 = v_{ini}$ is valid if the product $\text{msc}(v_0) \cdot ... \cdot \text{msc}(v_n)$ is defined.

The language $\mathcal{L}(G)$ collects the product MSCs of all valid paths that reach the final vertex $v_{fin}$.

What is $\mathcal{L}(G)$ in this case?
✔ Background
✔ Composition of compositional MSCs
🛡 Emptiness is undecidable
MSGs without matching arrow
The subclass of linear MSGs
Conclusion
A 2-counter machine is an abstract machine made of

- two unbounded counters $c_1$ and $c_2$ that can hold a non-negative integer (the memory)
- a sequence of labeled instructions (the program).

The allowed instructions are

- "$c_1++$" or "$c_2++$" increments the value of the counter;
- "$if \ c_1 = 0 \ goto \ l' \ else \ c_1--" transfers control to the instruction labeled by $l'$ if $c_1$ equals zero, and otherwise decrements $c_1$ and continues with the next instruction.
- "$if \ c_2 = 0 \ goto \ l' \ else \ c_2--"
2-counter machine [Minsky, 1967]

The initial value of both counters is 0.

A 2-counter machine is deterministic: It will

- either reach the last instruction of its program and halt after a **finite** number of steps
- or perform an **infinite** computation.

It is undecidable whether a given 2-counter machine halts.

**Theorem**

The emptiness problem $\mathcal{L}(G) = \emptyset$ is undecidable.
We build an MSG $G$ over three processes: $i$, $c_1$ and $c_2$ such that the three next properties are equivalent:

(i) $\mathcal{L}(G) \neq \emptyset$;

(ii) Some valid path reaches the final node;

(iii) The 2-counter machine halts.

Each valid path of $G$ corresponds to a computation of the machine, and vice versa.

The value of $c_1$ (resp. $c_2$) is encoded by the number of pending messages in the channel from $i$ to $c_1$ (resp. $c_2$).
The labeled instruction "l : c₁++" is encoded by a node l labeled by the MSC

For each prefix MSC M, the product $M \cdot msc(l)$ is defined.
The labeled instruction “l: if \( c_1 = 0 \) goto l’ else \( c_1-- \)” is encoded by three nodes l, l₀ and l₊:

For each prefix MSC \( M \), one and only one of the two products \( M \cdot msc(l₀) \) and \( M \cdot msc(l₊) \) is defined.
1. An MSG is **deadlock-free** if each valid path can be completed into a valid accepted one. 

   **Deadlock-freeness is undecidable**

   because $G$ is deadlock-free iff the machine halts.

2. The use of a channel in a valid (resp. accepted) path is undecidable.

3. Boundedness is also undecidable.
Background

Composition of compositional MSCs

Emptiness is undecidable

MSGs without matching arrow

The subclass of linear MSGs

Conclusion
An MSG is **without matching arrow** if its nodes are labeled by MSCs with no complete message exchange, i.e. **all events are unmatched**.

**Theorem**

The emptiness problem for MSGs without matching arrow is equivalent to the covering problem for Petri nets.
The valid paths of $G$ coincide with the firing sequences of $N$.

Some valid path reaches the final vertex $v_{\text{fin}}$ if and only if some firing sequence puts a token in the place $v_{\text{fin}}$. 
Let $G = (V, \rightarrow, v_{ini}, v_{fin})$ be an MSG over the set of channels $C$.
Let $P = V \cup C$ be the set of places. We put $m_{ini} = \{v_{ini}\}$.

Each arc $v_1 \rightarrow v_2$ in $G$ corresponds to a rule $r_{v_1 \rightarrow v_2}$ such that

\[
\begin{align*}
(1) \quad \bullet r_{v_1 \rightarrow v_2}(v) &= \begin{cases} 
1 & \text{if } v = v_1 \\
0 & \text{otherwise}
\end{cases} \\
\text{and } r_{v_1 \rightarrow v_2 \bullet}(v) &= \begin{cases} 
1 & \text{if } v = v_2 \\
0 & \text{otherwise}
\end{cases}
\end{align*}
\]

\[
(2) \quad \bullet r_{v_1 \rightarrow v_2}(c) = \bullet \text{msc}(v_2)(c), \text{ i.e. the number of unmatched receives from } c \text{ in } \text{msc}(v_2)
\]

\[
(3) \quad r_{v_1 \rightarrow v_2 \bullet}(c) = \text{msc}(v_2)\bullet(c), \text{ i.e. the number of unmatched sends to } c \text{ in } \text{msc}(v_2)
\]
Let \( N \) be a Petri net.
Let \( m_{\text{ini}}, m_{\text{fin}} \) be two markings.

We can assume that:

- \( m_{\text{ini}} \) contains a single token in the place \( p_{\text{ini}} \);
- \( m_{\text{fin}} \) contains a single token in the place \( p_{\text{fin}} \).

We build the MSG \( G \) with

- \( I = \{i\} \cup P \)
  Tokens in \( p \) are represented by messages from \( i \) to \( p \).
- \( V = \{v_{\text{ini}}, v_{\text{fin}}, v'_{\text{ini}}, v'_{\text{fin}}, v''\} \cup R \)
  The 3 nodes \( v_{\text{ini}}, v_{\text{fin}} \) and \( v'' \) are labeled by the empty MSC.
For each transition rule $r = (r, r')$, the node $r$ is labeled by the MSC $\text{msc}(r)$ and connected to $v''$ with two arcs:

- from $v''$ to $v_r$
- from $v_r$ to $v''$

```
msc(r : 2p_1 → p_1 + p_2 + p_3)
```

```
\begin{array}{c}
i \\
p_1 \\
p_2 \\
p_3
\end{array}
```
✔ Background
✔ Composition of compositional MSCs
✔ Emptiness is undecidable
✔ MSGs without matching arrow
_caption_right

The subclass of linear MSGs

Conclusion
MSGs without matching arrows and realizable MSGs are linear. Moreover the sliding window protocol is linear, too.
Linear MSGs

Definition

An MSG $G$ is linear if its valid paths coincide with the firing sequences of the corresponding Petri net.

Remark

The emptiness problem for linear MSGs is EXPSPACE-complete.

Theorem

Checking the linearity of a given MSG is equivalent to Covering.
\( p_{\text{fin}} \) is covered from \( p_{\text{ini}} \) if and only if \( G \) is not linear.
Characterization of undefined products

Let $A$ and $B$ be two (compositional) MSCs.

**Proposition** [Gunter et al, TACAS 2001]

The product $A \cdot B$ is defined if and only if the three next conditions are satisfied:

1. $\cdot A = 0$, i.e. $A$ is a prefix MSC

2. $A^\cdot \geq \cdot B$, i.e. for each channel, the number of unmatched receives in $B$ is at most equal to the number of unmatched sends in $A$.

3. For each channel $c$, if $B$ contains a matching arrow in $c$ then $A^\cdot (c) = \cdot B(c)$, i.e. the number of unmatched receives in $B$ is equal to the number of unmatched sends in $A$. 
An MSG $G$ is linear if and only if

for each vertex $v$

for each vertex $v'$ with $v' \rightarrow v$ in $G$

for each valid path from $v_{ini}$ to $v'$ with product $M$

the product $M \cdot msc(v)$ is defined if and only if $M^\bullet \geq msc(v)$. 
An MSG $G$ is linear if and only if

for each vertex $v$

for each channel $c$ with a matching arrow in $\text{msc}(v)$

for each vertex $v'$ with $v' \rightarrow v$ in $G$

for each valid path from $v_{\text{ini}}$ to $v'$ with product $M$

If $M^* \geq \cdot \text{msc}(v)$ then $M^*(c) = \cdot \text{msc}(v)(c)$. 
An MSG $G$ is linear if and only if

for each vertex $v$

for each channel $c$ with a matching arrow in $\text{msc}(v)$

$L(G_{v,c}) = \emptyset$. 
Background
Composition of compositional MSCs
Emptiness is undecidable
MSGs without matching arrow
The subclass of linear MSGs

Conclusion
Overview

- MSGs
  - Realizable MSGs
    - Safe MSGs
      - Basic MSGs
    - Linear time
  - Linear MSGs
    - MSGs without m.a.
      - Atomic MSGs
    - Linear time
  - Undecidable
    - EXPSPACE-complete
  - EXPSPACE-complete
  - Linear time
  - EXPSPACE-complete
  - EXPSPACE-complete
  - EXPSPACE-complete
  - Linear time
Linear MSGs

- **Linearity is difficult to check** but some syntactic or behavioural restrictions guarantee linearity.
  1. Unmatched events are forbidden for channels with a matching arrow;
  2. Each channel with a matching arrow is "safe".

- Emptiness, boundedness and other **reachability properties** are decidable for linear MSGs and undecidable in general.
Linear MSGs with counters

- **Counters** can be added to the model with no difficulty.
- A linear MSG can be more difficult to check than a possible equivalent safe one, up to unfolding. But
  - MSGs with counters can be exponentially more concise.
  - MSGs with counters are easier to understand in practice.
• Each linear MSG can be transformed into an equivalent atomic one:

⇒ Model checking bounded linear MSGs against MSO formulae is decidable;

⇒ Prefix-reachability properties, such as universal boundedness, can be reduced to Petri nets, too;

using results from Avellaneda’s thesis.

• Discrete timers can be handled easily with atomic MSGs (and hence with linear MSGs).