Theoretical Foundations of the UML

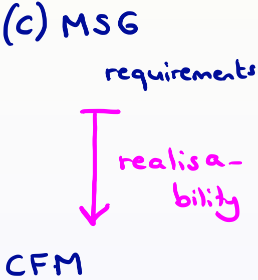
Lecture 09: Realisability

Joost-Pieter Katoen

Lehrstuhl für Informatik 2
Software Modeling and Verification Group

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

May 18, 2020



Outline

1. Introduction
2. Properties of CFMs
   - Deterministic CFMs
   - Deadlock-free CFMs
   - Synchronisation messages add expressiveness
3. Realisability
4. Inference of MSCs
5. Characterisation and complexity of realisability by "weak" CFMs

no synchronisation messages

Overview

Motivation

Practical use of MSCs and CFMs

- MSCs and MSGs are used by software engineers to capture requirements.
- These are the expected behaviours of the distributed system under design.
- Distributed systems can be viewed as a collection of communicating automata.

Central problem

Can we synthesize, preferably in an automated manner, a CFM whose behaviours are precisely the behaviours of the MSCs (or MSG)?

This is known as the realisability problem.

A set of MSCs

e.g. described by a MSG

? realisability poblem

CFM

requirements

syntesis

model of a concrete distributed system

"implementation"

From requirements to implementation

Realisability problem

INPUT: a set of MSCs

OUTPUT: a CFM $\mathcal{A}$ such that $L(\mathcal{A})$ equals the set of input MSCs.

Questions:

1. Is this possible? (That is, is this decidable?)
2. If so, how complex is it to obtain such CFM?
3. If so, how do such algorithms work?

Problem variants (1)

Realisability problem

INPUT: a set of MSCs

OUTPUT: a CFM $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A})$ equals the set of input MSCs.

Different forms of requirements

- Consider finite sets of MSCs, given as an enumerated set.

$\{M_1, M_2, \ldots, M_k\}$

Problem variants (1)

Realisability problem

INPUT: a set of MSCs

OUTPUT: a CFM $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A})$ equals the set of input MSCs.

Different forms of requirements

- Consider finite sets of MSCs, given as an enumerated set.
- Consider MSGs, that may describe an infinite set of MSCs.

Problem variants (1)

Realisability problem

INPUT: a set of MSCs

OUTPUT: a CFM $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A})$ equals the set of input MSCs.

Different forms of requirements

- Consider finite sets of MSCs, given as an enumerated set.
- Consider MSGs, that may describe an infinite set of MSCs.
- Consider MSCs whose set of linearisations is a regular word language.

→ the linearisations of the input MSCs can be described as finite-state automata

Problem variants (1)

Realisability problem

INPUT: a set of MSCs

OUTPUT: a CFM $\mathcal{A}$ such that $\mathcal{L}(\mathcal{A})$ equals the set of input MSCs.

Different forms of requirements

- Consider finite sets of MSCs, given as an enumerated set.
- Consider MSGs, that may describe an infinite set of MSCs.
- Consider MSCs whose set of linearisations is a regular word language.
- Consider MSGs that are "non-local" choice.

Problem variants (2)

Realisability problem

INPUT: a set of MSCs

OUTPUT: a CFM $\mathcal{A}$ such that $L(\mathcal{A})$ equals the set of input MSCs.

Different system models

- Consider CFMs without synchronisation messages.
- Allow CFMs that may deadlock. Possibly, a realisation deadlocks.
- Forbid CFMs that deadlock. No realisation will ever deadlock.
- Consider CFMs that are deterministic.
- Consider CFMs that are bounded.
- ......

Today's lecture

Today's setting

Realisation of a finite set of MSCs by a CFM without synchronisation messages, a simpler acceptance condition, and that may possibly deadlock.

Stated differently:
Realisation of a finite set of well-formed words (= language) by a CFM without synchronisation messages and that may possibly deadlock.

Results:

1. Weak CFMs (no syncs, product acceptance) are weaker than CFMs.
2. Conditions for realisability of a finite set of MSCs by a weak CFM.
3. Checking realisability for such sets is co-NP complete. — tomorrow

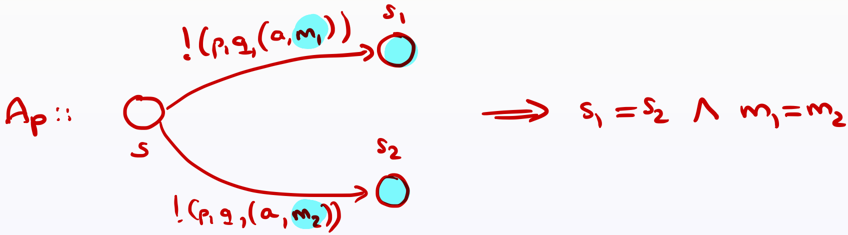
Overview

Determinism

Definition (Deterministic CFM)

A CFM $\mathcal{A}$ is *deterministic* if for all $p \in \mathcal{P}$, the transition relation Δ_p satisfies the following two conditions:

1. $(s, !(p, q, (a, m_1)), s_1) \in \Delta_p$ and $(s, !(p, q, (a, m_2)), s_2) \in \Delta_p$ implies $m_1 = m_2$ and $s_1 = s_2$
2. $(s, ?(p, q, (a, m)), s_1) \in \Delta_p$ and $(s, ?(p, q, (a, m)), s_2) \in \Delta_p$ implies $s_1 = s_2$



Determinism

Definition (Deterministic CFM)

A CFM $\mathcal{A}$ is *deterministic* if for all $p \in \mathcal{P}$, the transition relation Δ_p satisfies the following two conditions:

1. $(s, !(p, q, (a, m_1)), s_1) \in \Delta_p$ and $(s, !(p, q, (a, m_2)), s_2) \in \Delta_p$ implies $m_1 = m_2$ and $s_1 = s_2$
2. $(s, ?(p, q, (a, m)), s_1) \in \Delta_p$ and $(s, ?(p, q, (a, m)), s_2) \in \Delta_p$ implies $s_1 = s_2$

Note:

From a given state, process p may have the possibility of sending messages to more than one process.

Determinism

Definition (Deterministic CFM)

A CFM $\mathcal{A}$ is *deterministic* if for all $p \in \mathcal{P}$, the transition relation Δ_p satisfies the following two conditions:

1. $(s, !(p, q, (a, m_1)), s_1) \in \Delta_p$ and $(s, !(p, q, (a, m_2)), s_2) \in \Delta_p$ implies $m_1 = m_2$ and $s_1 = s_2$
2. $(s, ?(p, q, (a, m)), s_1) \in \Delta_p$ and $(s, ?(p, q, (a, m)), s_2) \in \Delta_p$ implies $s_1 = s_2$

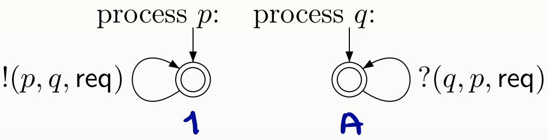
Note:

From a given state, process p may have the possibility of sending messages to more than one process.

Example:

Example CFM (1) and (2) are deterministic, while (3) is not.

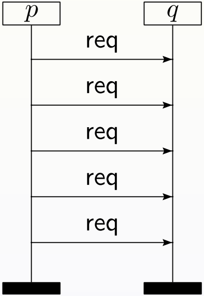
Example (1)



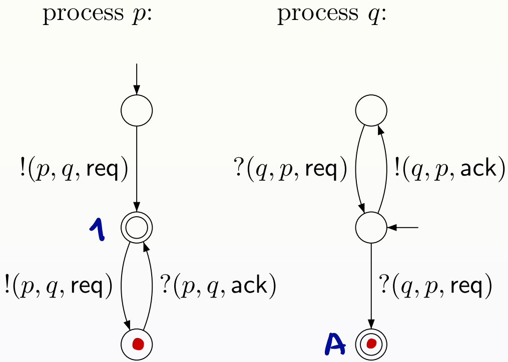
weak CFM

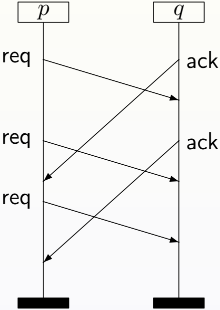
$F = \{(1, A)\}$

$= F_p \times F_q$



Example (2)





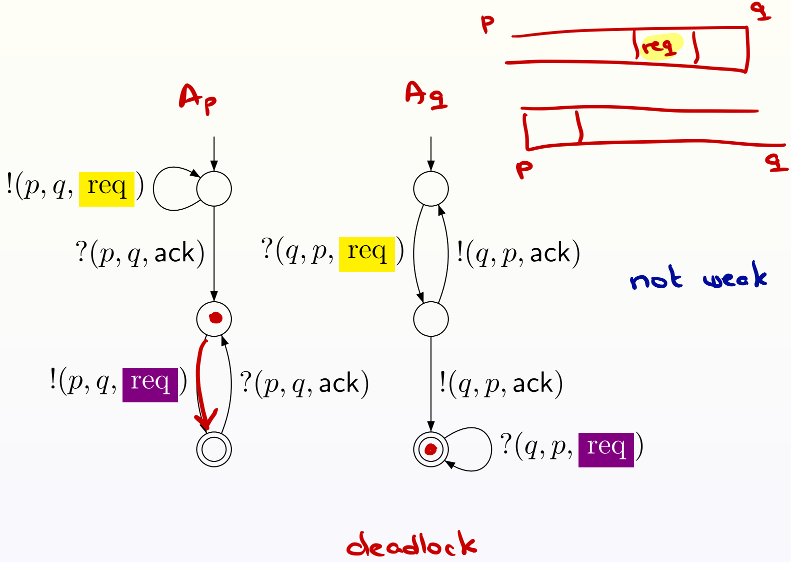
deadlock

weak $|D| = 1$

$F = \prod_{p \in P} F_p$

$F = \{(1, A)\}$

Example (3)



A_p
A_q
$!(p, q, \text{req})$
$?(p, q, \text{ack})$
$!(p, q, \text{req})$
$?(p, q, \text{ack})$
$?(q, p, \text{req})$
$!(q, p, \text{ack})$
$!(q, p, \text{ack})$
$?(q, p, \text{req})$
p
q
req
p
q
not weak
deadlock

Deadlock-freeness

Definition (Deadlock-free CFM)

A CFM $\mathcal{A}$ is *deadlock-free* if, for all $w \in Act^*$ and all runs γ of $\mathcal{A}$ on w, there exist $w' \in Act^*$ and run γ' in $\mathcal{A}$ such that $\gamma \cdot \gamma'$ is an accepting run of $\mathcal{A}$ on $w \cdot w'$.

Example:

Example CFM (1) is deadlock-free, while (2) and (3) are not.

Theorem: [Genest et. al, 2006]

For any $\exists B$-bounded CFM $\mathcal{A}$, the decision problem "is $\mathcal{A}$ deadlock-free?" is decidable (and is PSPACE-complete).

Weak CFMs

Definition (Weak CFM)

A CFM is called *weak* if $|\mathbb{D}| = 1$ and $F = \prod_p F_p$.

one sync message = no sync messages

if each of the local automata are in a final local state

Weak CFMs

Definition (Weak CFM)

A CFM is called *weak* if $|\mathbb{D}| = 1$ and $F = \prod_p F_p$.

Example (1) and (2) are weak CFMs. Example (3) is not.

Q: Are CFMs more expressive than weak CFMs? That is, do there exist languages (over linearizations or, equivalently, MSCs) that can be generated by CFMs but **not** by weak CFMs? Yes.

CFM vs. weak CFM

Theorem:

Weak CFMs are strictly less expressive than CFMs.

CFM vs. weak CFM

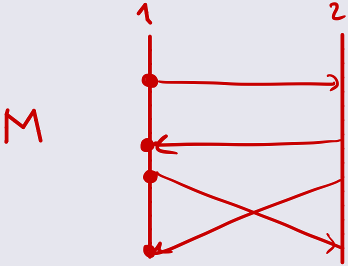
Theorem:

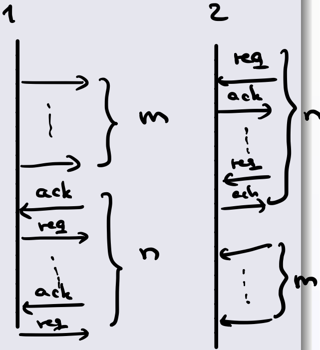
Weak CFMs are strictly less expressive than CFMs.

Proof.

For $m, n \geqslant 1$, let $M(m,n) \in \mathbb{M}$ over $\mathcal{P} = \{1,2\}$ and $\mathcal{C} = \{\text{req}, \text{ack}\}$ be:

- $M \upharpoonright 1 = (!(1,2,\text{req}))^m \; (?(1,2,\text{ack}) \; !(1,2,\text{req}))^n$
- $M \upharpoonright 2 = (?(2,1,\text{req}) \; !(2,1,\text{ack}))^n \; (?(2,1,\text{req}))^m$





CFM vs. weak CFM

Theorem:

Weak CFMs are strictly less expressive than CFMs.

Proof.

For $m, n \geqslant 1$, let $M(m,n) \in \mathbb{M}$ over $\mathcal{P} = \{1,2\}$ and $\mathcal{C} = \{\text{req}, \text{ack}\}$ be:

- $M \upharpoonright 1 = (!(1,2,\text{req}))^m \ (?(1,2,\text{ack}) \ !(1,2,\text{req}))^n$
- $M \upharpoonright 2 = (?(2,1,\text{req}) \ !(2,1,\text{ack}))^n \ (?(2,1,\text{req}))^m$

Claim: there is no weak CFM over $\mathcal{P} = \{1,2\}$ and $\mathcal{C} = \{\text{req}, \text{ack}\}$ whose language is $L = \{M(n,n) \mid n > 0\}$.

$\longrightarrow$ we have seen a CFM A with L(A) = L

m = n

CFM vs. weak CFM

Theorem:

Weak CFMs are strictly less expressive than CFMs.

Proof.

For $m, n \geqslant 1$, let $M(m,n) \in \mathbb{M}$ over $\mathcal{P} = \{1,2\}$ and $\mathcal{C} = \{\text{req}, \text{ack}\}$ be:

- $M \upharpoonright 1 = (!(1,2,\text{req}))^m \ (?(1,2,\text{ack}) \ !(1,2,\text{req}))^n$
- $M \upharpoonright 2 = (?(2,1,\text{req}) \ !(2,1,\text{ack}))^n \ (?(2,1,\text{req}))^m$

Claim: there is no weak CFM over $\mathcal{P} = \{1,2\}$ and $\mathcal{C} = \{\text{req}, \text{ack}\}$ whose language is $L = \{M(n,n) \mid n > 0\}$. By contraposition. Suppose there is a weak CFM $\mathcal{A} = ((\mathcal{A}_1, \mathcal{A}_2), s_{init}, F)$ with $L(\mathcal{A}) = L$. For any $n > 0$, there is an accepting run of $\mathcal{A}$ on $M(n,n)$. If n is sufficiently large, then

- $\mathcal{A}_1$ visits a cycle of length $i > 0$ to read the first n letters of $M(n,n) \upharpoonright 1$
- $\mathcal{A}_2$ visits a cycle of length $j > 0$ to read the last n letters of $M(n,n) \upharpoonright 2$

Then there is an accepting run of $\mathcal{A}$ on $M(n + (i \cdot j), n) \notin L$. Contradiction. □

CFM vs. weak CFM

Theorem:

Weak CFMs are strictly less expressive than CFMs.

Intuition proof

If $\mathcal{A}_1$ traverses a cycle of size i at least once to "generate" $(!(1, 2, \text{req}))^n$, then it can autonomously traverse this cycle more often and thus "pump" to an expression of the form $(!(1, 2, \text{req}))^{n \cdot i}$.

Similar reasoning applies to automaton $\mathcal{A}_2$ for the last n letters of the input word $M \upharpoonright 2$. Suppose its cycle is of size j.

Now if $\mathcal{A}_1$ traverses its cycle of size i, j times, and $\mathcal{A}_2$ traverses its cycle of size j, i times, then the number of requests sent by process 1 matches the number of receipts by process 2.

But this yields a word in $M(n + (i \cdot j), n)$ that is not in L.

Overview

What is realisability?

Definition (Realisability)

1. MSC M is <u>realisable</u> whenever $\{M\} = \mathcal{L}(\mathcal{A})$ for some CFM $\mathcal{A}$.

requirement

realisation/implementation

What is realisability?

Definition (Realisability)

1. MSC M is realisable whenever $\{M\} = \mathcal{L}(\mathcal{A})$ for some CFM $\mathcal{A}$.
2. A finite set $\{M_1, \ldots, M_n\}$ of MSCs is realisable whenever $\{M_1, \ldots, M_n\} = \mathcal{L}(\mathcal{A})$ for some CFM $\mathcal{A}$.

A realises $\{M_1, \ldots, M_n\}$

What is realisability?

Definition (Realisability)

1. MSC M is realisable whenever $\{M\} = \mathcal{L}(\mathcal{A})$ for some CFM $\mathcal{A}$.
2. A finite set $\{M_1, \ldots, M_n\}$ of MSCs is realisable whenever $\{M_1, \ldots, M_n\} = \mathcal{L}(\mathcal{A})$ for some CFM $\mathcal{A}$.
3. MSG G is realisable whenever $\mathcal{L}(G) = \mathcal{L}(\mathcal{A})$ for some CFM $\mathcal{A}$.

implementati

requirements

set of MSCs accepted by G

CFM A realises MSG G

What is realisability?

Definition (Realisability)

1. MSC M is realisable whenever $\{M\} = \mathcal{L}(\mathcal{A})$ for some CFM $\mathcal{A}$.
2. A finite set $\{M_1, \ldots, M_n\}$ of MSCs is realisable whenever $\{M_1, \ldots, M_n\} = \mathcal{L}(\mathcal{A})$ for some CFM $\mathcal{A}$.
3. MSG G is realisable whenever $\mathcal{L}(G) = \mathcal{L}(\mathcal{A})$ for some CFM $\mathcal{A}$.

Equivalently

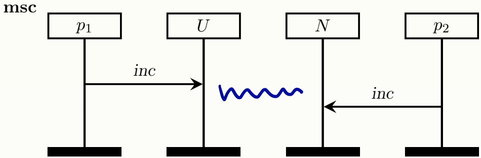
1. MSC M is realisable whenever $Lin(M) = Lin(\mathcal{A})$ for some CFM $\mathcal{A}$.
2. Set $\{M_1, \ldots, M_n\}$ of MSCs is realisable whenever $\bigcup_{i=1}^{n} Lin(M_i) = Lin(\mathcal{A})$ for some CFM $\mathcal{A}$.
3. MSG G is realisable whenever $Lin(G) = Lin(\mathcal{A})$ for some CFM $\mathcal{A}$.

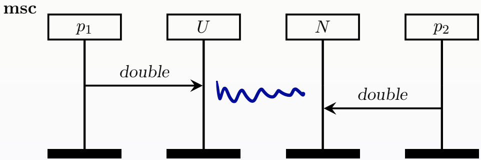
We will consider realisability using its characterisation by linearisations.

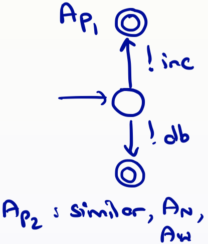
Overview

Two example MSCs

Consider the MSCs M_{inc} (top) and M_{db} (bottom):



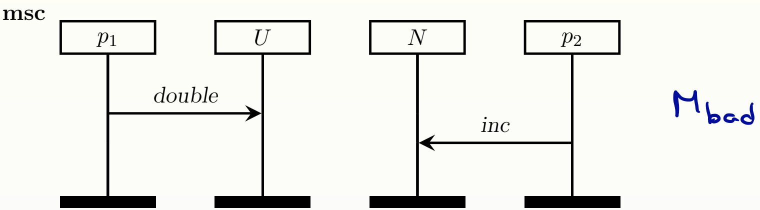




Intuition

In M_{inc}, the volume of U (uranium) and N (nitric acid) is increased by one unit; in M_{db} both volumes are doubled. For safety reasons, it is essential that both ingredients are increased by the same amount!

A third, inferred fatal scenario



So:

The set $\{ M_{inc}, M_{db} \}$ is not realisable, as any CFM that realises this set also realises the inferred MSC M_{bad} above.

Note that:

MSCs M_{inc} or M_{db} alone do not imply M_{bad}. Together they do.

Inference

Definition (Inference)

The set L of MSCs is said to infer MSC $M \notin L$ if and only if:

$$\text{for any CFM } \mathcal{A}.\ (L \subseteq \mathcal{L}(\mathcal{A}) \text{ implies } M \in \mathcal{L}(\mathcal{A}))\,.$$

What we will show later on:

The set L of MSCs is realisable iff L contains all MSCs that it infers.

Intuition

A realisable set of MSCs contains all its implied scenarios.

For computational purposes, an alternative characterisation is required.

Projection (1)

Definition (MSC projection)

For MSC M and process p let $M \upharpoonright p$, the projection of M on process p, be the ordered sequence of actions occurring at process p in M.

Projection (1)

Definition (MSC projection)

For MSC M and process p let $M \upharpoonright p$, the projection of M on process p, be the ordered sequence of actions occurring at process p in M.

Lemma

An MSC M over the processes $\mathcal{P} = \{ p_1, \ldots, p_n \}$ is uniquely determined by the projections $M \upharpoonright p_i$ for $0 < i \leqslant n$.

exercise: $M \upharpoonright p_1, M \upharpoonright p_2, \ldots, M \upharpoonright p_n \longmapsto$ MSC M

$\longleftarrow\!\dashv$

Projection (2)

Definition (Word projection)

For word $w \in Act^*$ and process p, the projection of w on process p, denoted $w \restriction p$, is defined by:

$$
\begin{array}{rcl}
\epsilon \restriction p & = & \epsilon \\
(!(r,q,a) \cdot w) \restriction p & = & \begin{cases} !(r,q,a) \cdot (w \restriction p) & \text{if } r = p \\ w \restriction p & \text{otherwise} \end{cases}
\end{array}
$$

and similarly for receive actions.

Example

$w =$

$!(1,2,\text{req})!(1,2,\text{req})?(2,1,\text{req})!(2,1,\text{ack})?(2,1,\text{req})!(2,1,\text{ack})?(1,2,\text{ack})!(1,2,\text{req})$

$w \restriction 1 = !(1,2,\text{req})!(1,2,\text{req})?(1,2,\text{ack})!(1,2,\text{req})$

$w \restriction 2 = ?(2,1,\text{req})!(2,1,\text{ack})?(2,1,\text{req})!(2,1,\text{ack})$

Projection (3)

Definition (Word projection)

For word $w \in Act^*$ and process p, the projection of w on process p, denoted $w \upharpoonright p$, is defined by:

$$\begin{array}{rcl} \epsilon \upharpoonright p & = & \epsilon \\ (!(r,q,a) \cdot w) \upharpoonright p & = & \begin{cases} !(r,q,a) \cdot (w \upharpoonright p) & \text{if } r = p \\ w \upharpoonright p & \text{otherwise} \end{cases} \end{array}$$

and similarly for receive actions.

Projection (3)

Definition (Word projection)

For word $w \in Act^*$ and process p, the projection of w on process p, denoted $w \restriction p$, is defined by:

$$
\begin{aligned}
\epsilon \restriction p &= \epsilon \\
(!(r,q,a)\cdot w) \restriction p &= \begin{cases} !(r,q,a)\cdot(w \restriction p) & \text{if } r = p \\ w \restriction p & \text{otherwise} \end{cases}
\end{aligned}
$$

and similarly for receive actions.

Lemma

A well-formed word w over Act^* given as projections $w \restriction p_1, \ldots, w \restriction p_n$ uniquely characterises an MSC $M(w)$ over $\mathcal{P} = \{ p_1, \ldots, p_n \}$.

$w \restriction p_1, \ldots, w \restriction p_n \longrightarrow w \longrightarrow M(w)$

Closure

Definition (Inference relation)

For well-formed[a] $L \subseteq Act^*$, and well-formed word $w \in Act^*$, let:

$$L \models w \quad \text{iff} \quad (\forall p \in \mathcal{P}. \exists v \in L. w \upharpoonright p = v \upharpoonright p)$$

[a] Language L is called well-formed iff all its words are well-formed.

$L = \{ v_1, \ldots, v_k \}$

$w \upharpoonright p = v_i \upharpoonright p$

$v_i \in Act^*$

$w \in Act^*$

$p: \quad v_i \upharpoonright p = w \upharpoonright p$

$q: \quad v_j \upharpoonright q = w \upharpoonright q$

$p \neq q$

Closure

Definition (Inference relation)

For well-formed[a] $L \subseteq Act^*$, and well-formed word $w \in Act^*$, let:

$$L \models w \quad \text{iff} \quad (\forall p \in \mathcal{P}.\, \exists v \in L.\, w\restriction p = v \restriction p)$$

[a] Language L is called well-formed iff all its words are well-formed.

Definition (Closure under $\models$)

Language L is closed under $\models$ whenever $L \models w$ implies $w \in L$.

it is impossible to infer a word $w \notin L$.

set of MSCs

w = MSC

Closure

Definition (Inference relation)

For well-formed[a] $L \subseteq Act^*$, and well-formed word $w \in Act^*$, let:

$$L \models w \quad \text{iff} \quad (\forall p \in \mathcal{P}. \exists v \in L. w \upharpoonright p = v \upharpoonright p)$$

[a] Language L is called well-formed iff all its words are well-formed.

Definition (Closure under $\models$)

Language L is closed under $\models$ whenever $L \models w$ implies $w \in L$.

Intuition

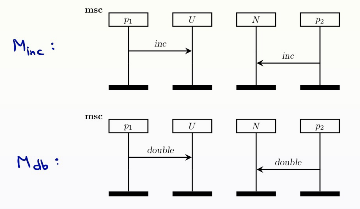
The closure condition says that the set of MSCs (or, equivalently, well-formed words) can be obtained from the projections of the MSCs in L onto individual processes.

Closure: example

Language L is closed under $\models$ whenever $L \models w$ implies $w \in L$.

Example

$L = Lin(\{M_{inc}, M_{db}\})$ is not closed under $\models$.



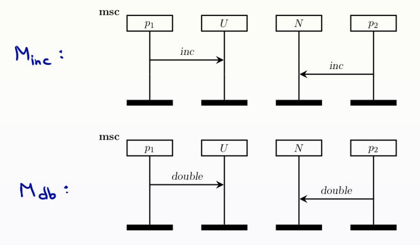
Closure: example

Language L is closed under $\models$ whenever $L \models w$ implies $w \in L$.

Example

$L = Lin(\{M_{inc}, M_{db}\})$ is not closed under $\models$. This is shown as follows:

$$w \;=\; !(p_1, U, double)?(U, p_1, double)!(p_2, N, inc)?(N, p_2, inc) \notin L$$



Closure: example

Language L is closed under $\models$ whenever $L \models w$ implies $w \in L$.

Example

$L = Lin(\{M_{inc}, M_{db}\})$ is not closed under $\models$. This is shown as follows:

$$w = !(p_1, U, double)?(U, p_1, double)!(p_2, N, inc)?(N, p_2, inc) \notin L$$

But: $L \models w$ since

$\forall p.\ \exists v.\ v \upharpoonright p = w \upharpoonright p$

Closure: example

Language L is closed under $\models$ whenever $L \models w$ implies $w \in L$.

Example

$L = Lin(\{M_{inc}, M_{db}\})$ is not closed under $\models$. This is shown as follows:

$$w = !(p_1, U, double)?(U, p_1, double)!(p_2, N, inc)?(N, p_2, inc) \notin L$$

But: $L \models w$ since

- for process p_1, there is $u \in L$ with $w \upharpoonright p_1 = !(p_1, U, double) = u \upharpoonright p_1$, and

$u \in Lin(M_{inc})$

Closure: example

Language L is closed under $\models$ whenever $L \models w$ implies $w \in L$.

Example

$L = Lin(\{M_{inc}, M_{db}\})$ is not closed under $\models$. This is shown as follows:

$$w = !(p_1, U, double)?(U, p_1, double)!(p_2, N, inc)?(N, p_2, inc) \notin L$$

But: $L \models w$ since

- for process p_1, there is $u \in L$ with $w \upharpoonright p_1 = !(p_1, U, double) = u \upharpoonright p_1$, and
- for process p_2, there is $v \in L$ with $w \upharpoonright p_2 = !(p_2, N, inc) = v \upharpoonright p_2$, and

Lin(M_{inc})

Closure: example

Language L is closed under $\models$ whenever $L \models w$ implies $w \in L$.

Example

$L = Lin(\{M_{inc}, M_{db}\})$ is not closed under $\models$. This is shown as follows:

$$w = !(p_1, U, double)?(U, p_1, double)!(p_2, N, inc)?(N, p_2, inc) \notin L$$

But: $L \models w$ since

- for process p_1, there is $u \in L$ with $w \upharpoonright p_1 = !(p_1, U, double) = u \upharpoonright p_1$, and
- for process p_2, there is $v \in L$ with $w \upharpoonright p_2 = !(p_2, N, inc) = v \upharpoonright p_2$, and
- for process U, there is $u \in L$ with $w \upharpoonright U = ?(U, p_1, double) = u \upharpoonright U$, and

Closure: example

Language L is closed under $\models$ whenever $L \models w$ implies $w \in L$.

Example

$L = Lin(\{M_{inc}, M_{db}\})$ is not closed under $\models$. This is shown as follows:

$$w = !(p_1, U, double)?(U, p_1, double)!(p_2, N, inc)?(N, p_2, inc) \notin L$$

But $L \models w$ since

- for process p_1, there is $u \in L$ with $w \upharpoonright p_1 = !(p_1, U, double) = u \upharpoonright p_1$, and
- for process p_2, there is $v \in L$ with $w \upharpoonright p_2 = !(p_2, N, inc) = v \upharpoonright p_2$, and
- for process U, there is $u \in L$ with $w \upharpoonright U = ?(U, p_1, double) = u \upharpoonright U$, and
- for process N, there is $v \in L$ with $w \upharpoonright N = ?(N, p_2, inc) = v \upharpoonright N$.

$\in Lin(M_{inc})$

Weak CFMs

Definition (Recall: weak CFM)

CFM $\mathcal{A}$ is weak if $|\mathbb{D}| = 1$ and $F = \prod_p F_p$.

Intuition

A weak CFM can be considered as CFM without synchronisation messages. (Therefore, the component $\mathbb{D}$ may be omitted.) For simplicity, today we address realisability with the aim of using weak CFMs as implementation. Recall: weak CFMs are strictly less expressive than CFMs.

Realisability by a weak CFM

A finite set $\{M_1, \ldots, M_n\}$ of MSCs is realisable (by a weak CFM) whenever $\{ M_1, \ldots, M_n\} = L(\mathcal{A})$ for some weak CFM $\mathcal{A}$.

Weak CFMs are closed under $\models$

Lemma:

For any weak CFM $\mathcal{A}$, $Lin(\mathcal{A})$ is closed under $\models$.

Weak CFMs are closed under $\models$

Lemma:

For any weak CFM $\mathcal{A}$, $Lin(\mathcal{A})$ is closed under $\models$.

Proof.

Let $\mathcal{A}$ be a weak CFM. Since $\mathcal{A}$ is a CFM, any $w \in Lin(\mathcal{A})$ is well-formed.
Let $w \in Act^*$ be well-formed and assume $Lin(\mathcal{A}) \models w$.
To show that $Lin(\mathcal{A})$ is closed under $\models$, we prove that $w \in Lin(\mathcal{A})$.
By definition of $\models$, for any process p there is $v^p \in Lin(\mathcal{A})$ with $v^p \upharpoonright p = w \upharpoonright p$.
Let π be an accepting run of $\mathcal{A}$ on v^p and let run $\pi \upharpoonright p$ visit only states of $\mathcal{A}_p$ while taking only transitions in Δ_p. Then, $\pi \upharpoonright p$ is an accepting run of "local" automaton $\mathcal{A}_p$ on the word $v^p \upharpoonright p = w \upharpoonright p$.
In absence of synchronisation messages, the "local" accepting runs $\pi \upharpoonright p$ for all processes p together can be combined to obtain an accepting run of $\mathcal{A}$ on w.
Thus, $w \in Lin(\mathcal{A})$. □

Overview

Characterisation of realisability

Theorem: [Alur et al., 2001]

Finite $L \subseteq Act^*$ is realisable (by a weak CFM) iff L is closed under $\models$.

finite set of MSCs

set of MSCs is closed under $\models$.

Characterisation of realisability

Theorem: [Alur et al., 2001]

Finite $L \subseteq Act^*$ is realisable (by a weak CFM) iff L is closed under $\models$.

Proof.

On the black board. □

Theorem Finite $L \subseteq Act^*$ is realisable (by a weak CFM) if and only if L is closed under $\models$.

Proof: "$\Longrightarrow$". Assume L is realisable. Thus, there is a weak CFM $\mathcal{A}$ such that $L = Lin(\mathcal{A})$. As $Lin(\mathcal{A})$ only contains linearisations, and every linearisation is well-formed, each word in L is well-formed. Let $w \in Act^*$, w is well-formed, and assume $L \models w$. By definition of $\models$, for every process p, $\exists v^p \in L$ with $v^p \upharpoonright p = w \upharpoonright p$.

We show $w \in L$. (Then it follows that L is closed under $\models$.)

Let π be an accepting run of CFM $\mathcal{A}$ on v^p. (Such run does exist, otherwise v^p does not belong to L). Transitions along $\pi_p = \pi \upharpoonright p$ corresponds to the "local" transitions of $\mathcal{A}_p$. It follows from $v^p \in L$ that π_p is an accepting run of $\mathcal{A}_p$, on the word $v^p \upharpoonright p = w \upharpoonright p$. This applies to all processes $\{p_1, \ldots, p_n\}$ of the CFM. The local accepting runs $\pi \upharpoonright p_1, \ldots, \pi \upharpoonright p_n$ can be combined uniquely to obtain a run π^w of CFM $\mathcal{A}$ on w. π^w is accepting, because of the weak acceptance criterion. Thus $w \in L$.

"$\Leftarrow$". Assume L is closed under $\models$. As $\models$ is only defined for well-formed words, each word in L is well-formed. Moreover, by definition of closure under $\models$, $L \models w$ implies $w \in L$, for each well-formed $w \in Act^*$. To prove: <u>L is realisable.</u>

Let A_p be an automaton over Act_p accepting

$$L_p = \{ w\restriction_p \mid w \in L \}$$

A_p thus accepts all projections to process p of words in L. Let weak CFM $A = ((A_p)_{p \in P}, s_{init}, F)$ with $F = \prod_{p \in P} F_p$. Then: A realises L, i.e. $Lin(A) = L$.

"$\supseteq$": Let $w \in L$. By construction of the CFM A, $Lin(A_p) = L_p$. But then $w \in Lin(A)$

"$\subseteq$": let $w \in Lin(A)$. Then $w\restriction_p \in Lin(A_p)$ for each p. By def. of $\models$, $L \models w$. Since L is closed under $\models$, it follows $w \in L$ $\boxtimes$

Characterisation of realisability

Theorem: [Alur et al., 2001]

Finite $L \subseteq Act^*$ is realisable (by a weak CFM) iff L is closed under $\models$.

Proof.

On the black board. □

Corollary

The finite set of MSCs $\{M_1, \ldots, M_n\}$ is realisable (by a weak CFM) iff $\bigcup_{i=1}^{n} Lin(M_i)$ is closed under $\models$.

Characterisation of realisability

Theorem

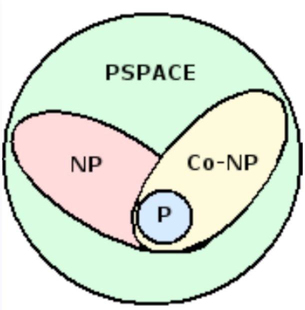
For any well-formed $L \subseteq Act^*$:

L is regular and closed under $\models$
if and only if
$L = Lin(\mathcal{A})$ for some $\forall$-bounded weak CFM $\mathcal{A}$.

Complexity of realisability

Let co-NP be the class of all decision problems C with $\overline{C}$, the complement of C, is in NP.

A problem C is co-NP complete if it is in co-NP, and it is co-NP hard, i.e., each for any co-NP problem there is a polynomial reduction to C.



Complexity of realisability (by a weak CFM)

Theorem: [Alur et al., 2001]

The decision problem "is a given finite set of MSCs realisable by a weak CFM?" is decidable and is co-NP complete.

Complexity of realisability (by a weak CFM)

Theorem: [Alur et al., 2001]

The decision problem "is a given finite set of MSCs realisable by a weak CFM?" is decidable and is co-NP complete.

Proof.

1. Membership in co-NP is proven by showing that its complement is in NP. This is rather standard.
2. The co-NP hardness proof is based on a polynomial reduction of the join dependency problem to the above realisability problem. (Details on the black board.)

□