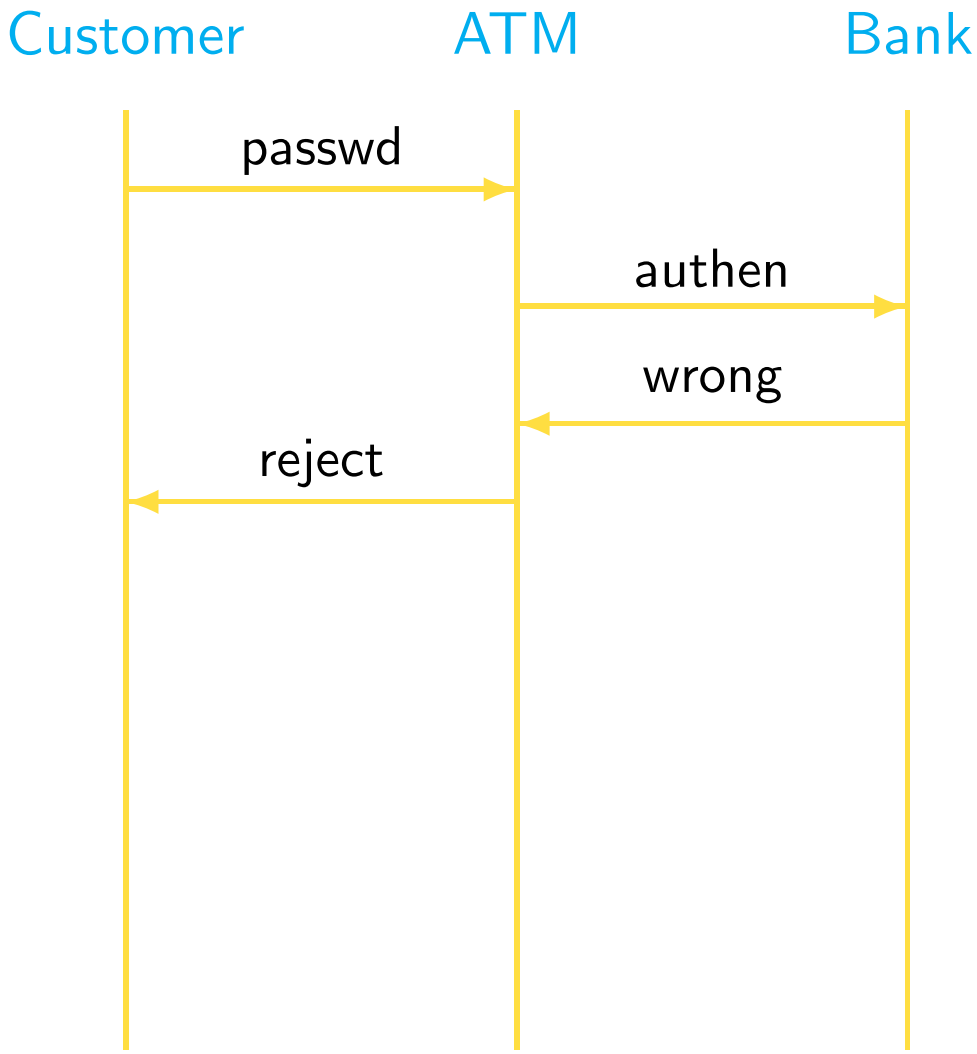
The Theory of Message Sequence Charts

K Narayan Kumar

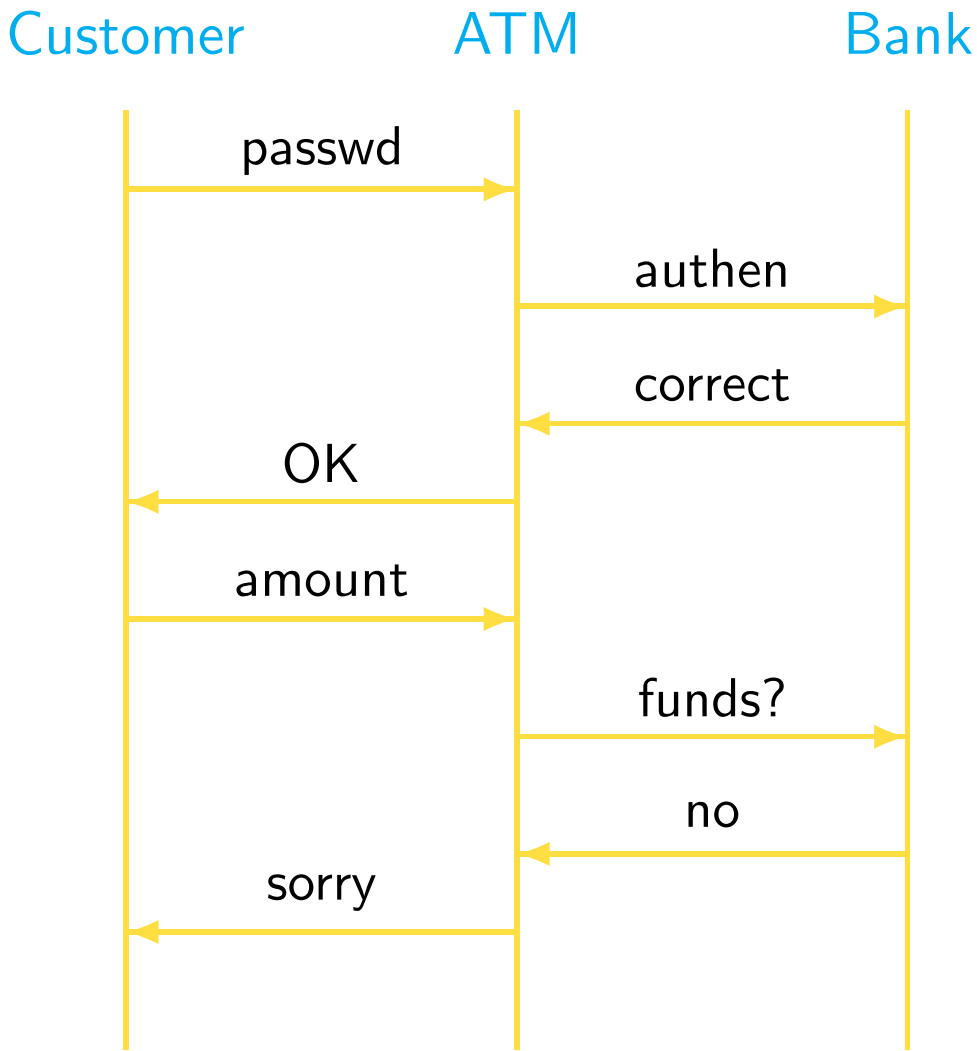
Chennai Mathematical Institute
http://www.cmi.ac.in/~kumar

TIFR, Mumbai, 28 April, 2009

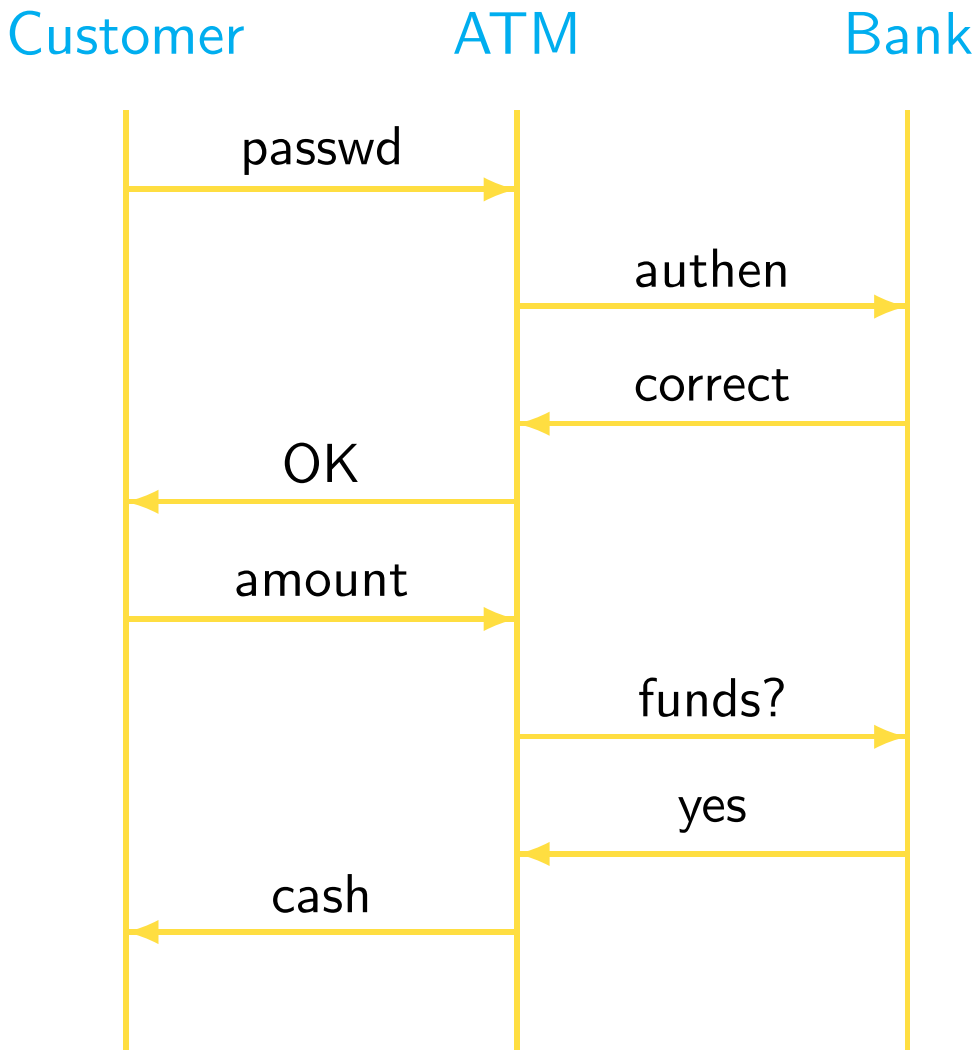
An ATM



An ATM

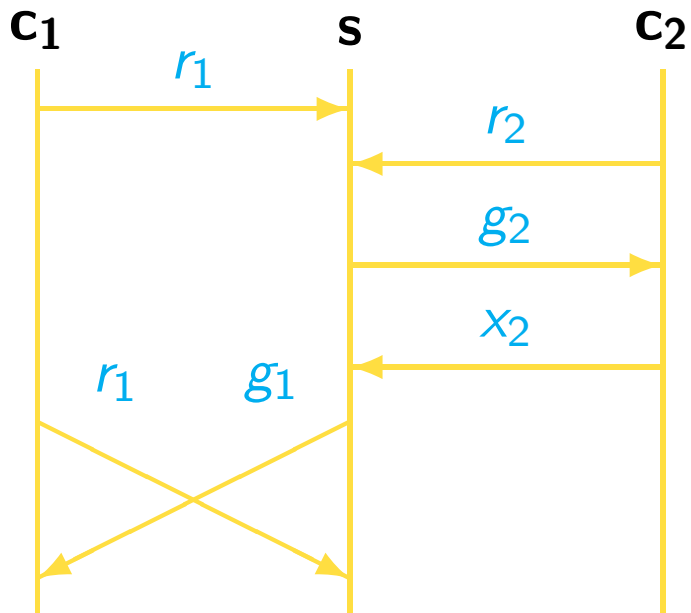

Customer
ATM
Bank
passwd
authen
correct
OK
amount
funds?
no
sorry

An ATM


Customer
ATM
Bank
passwd
authen
correct
OK
amount
funds?
yes
cash

Message Sequence Charts

Two clients and a server



Overview

Message Sequence Charts

Overview

Message Sequence Charts

- Visual formalism for specifying scenarios.

Overview

Message Sequence Charts

- Visual formalism for specifying scenarios.
- Part of the UML Standard

Overview

Message Sequence Charts

- Visual formalism for specifying scenarios.
- Part of the UML Standard
- Used quite extensively, for instance in the telecom industry.

Overview

Message Sequence Charts

- Visual formalism for specifying scenarios.
- Part of the UML Standard
- Used quite extensively, for instance in the telecom industry.

- Formalize these pictures as ...

Overview

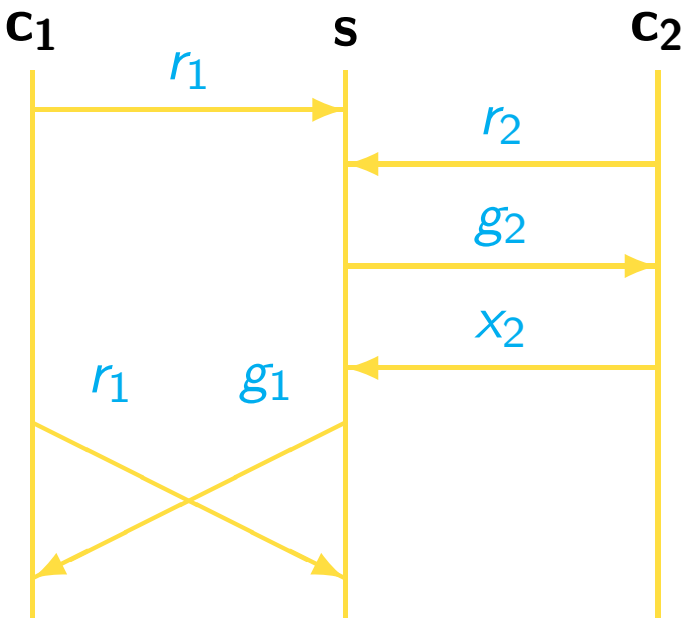
Message Sequence Charts

- Visual formalism for specifying scenarios.
- Part of the UML Standard
- Used quite extensively, for instance in the telecom industry.

- Formalize these pictures as ...
- Formulate natural (and relevant) questions ...

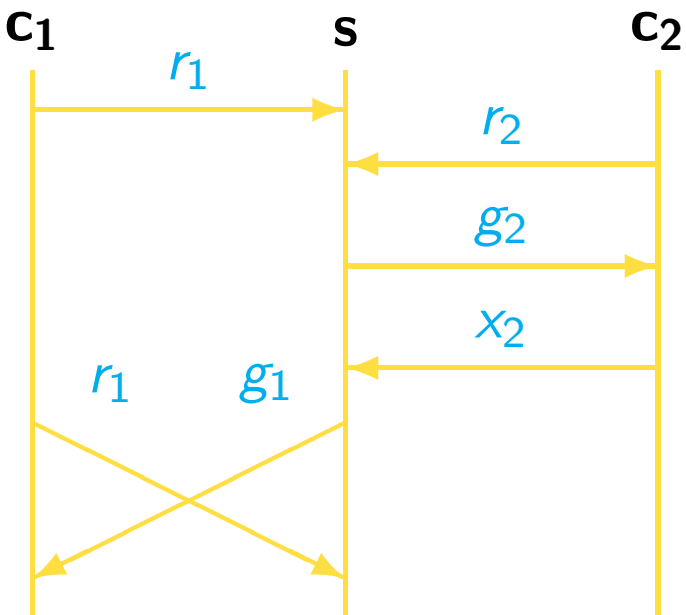
MSCs

Two clients and a server

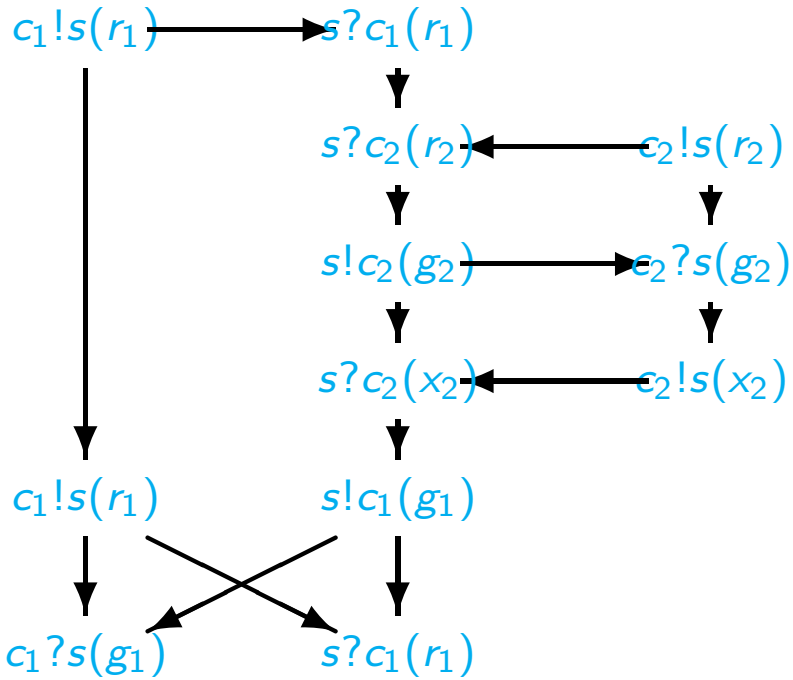


MSCs as Labelled Partial Orders

Two clients and a server,

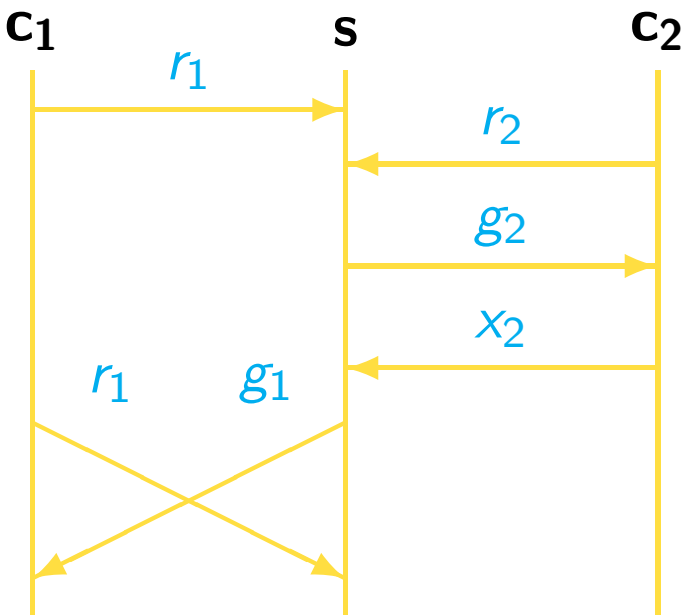


and a partial order representation

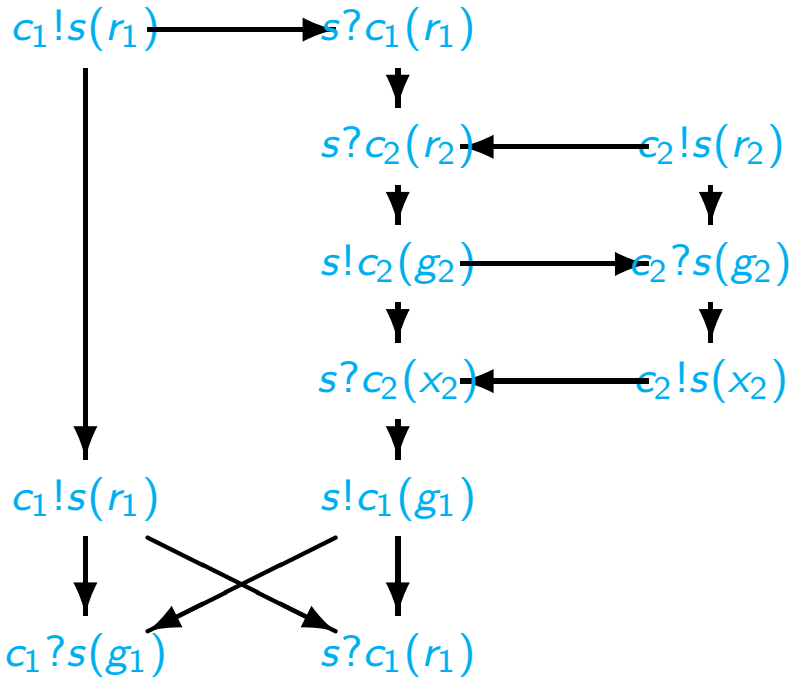


MSCs as Labelled Partial Orders

Two clients and a server,



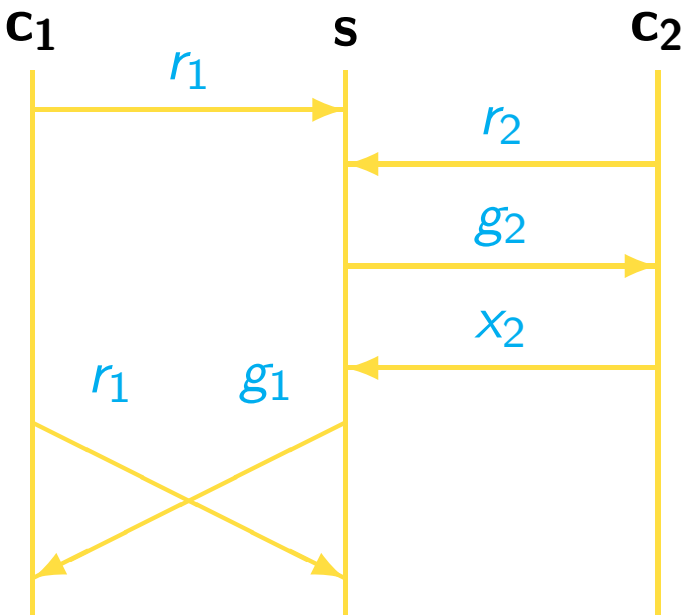
and a partial order representation



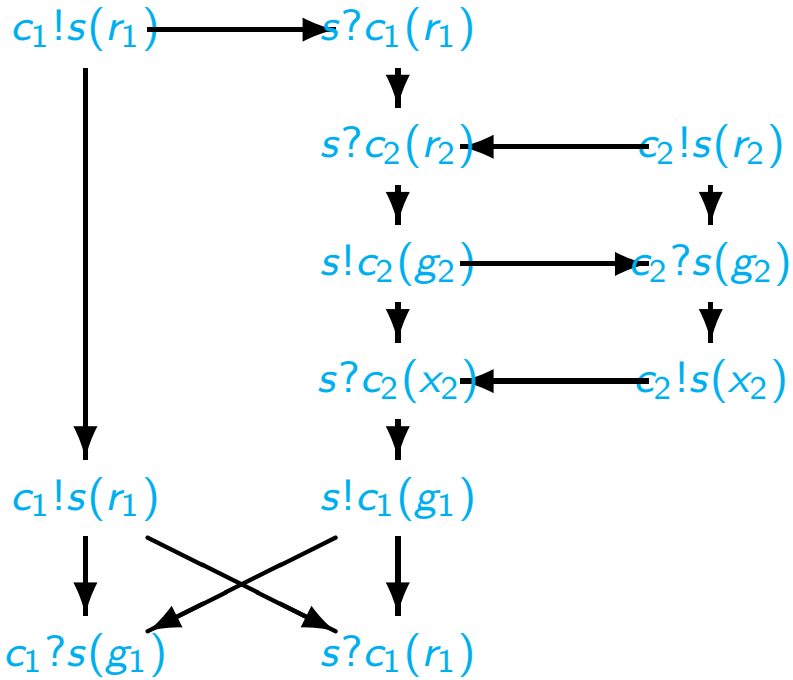
- Assume that all the channels are FIFO.

MSCs as Labelled Partial Orders

Two clients and a server,

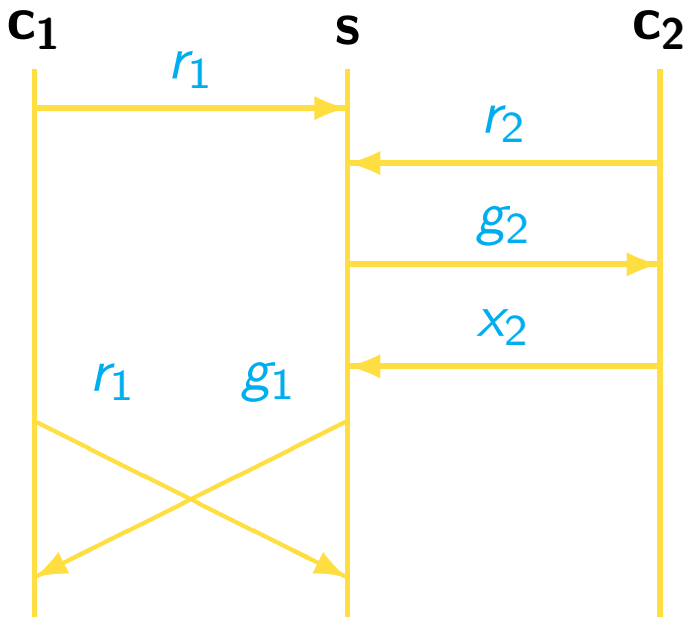


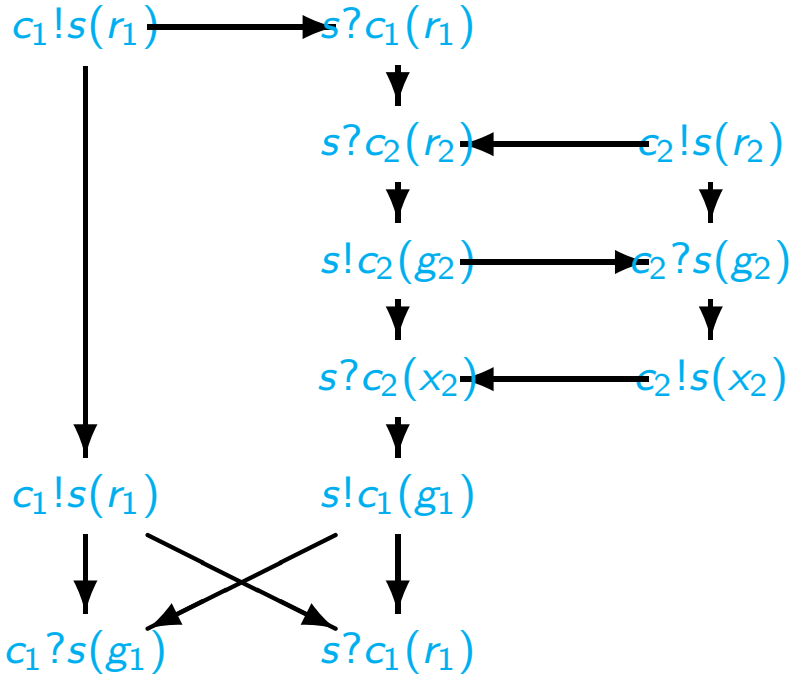
and a partial order representation



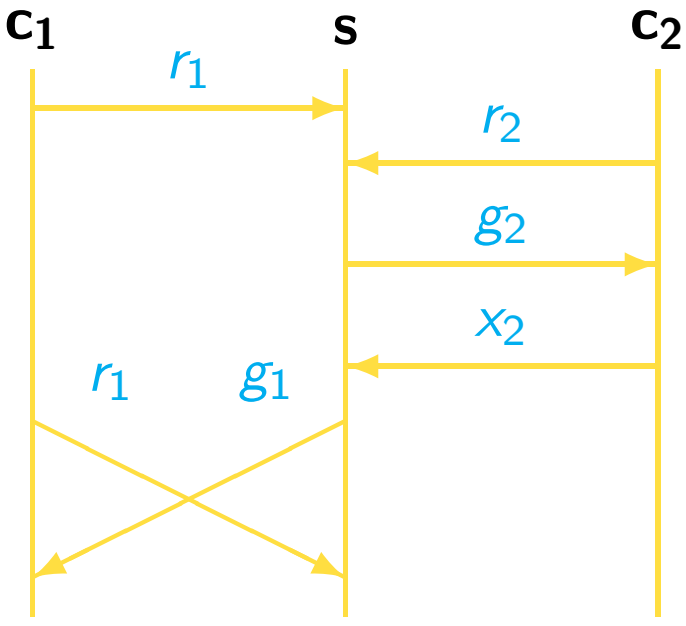
- Assume that all the channels are FIFO.

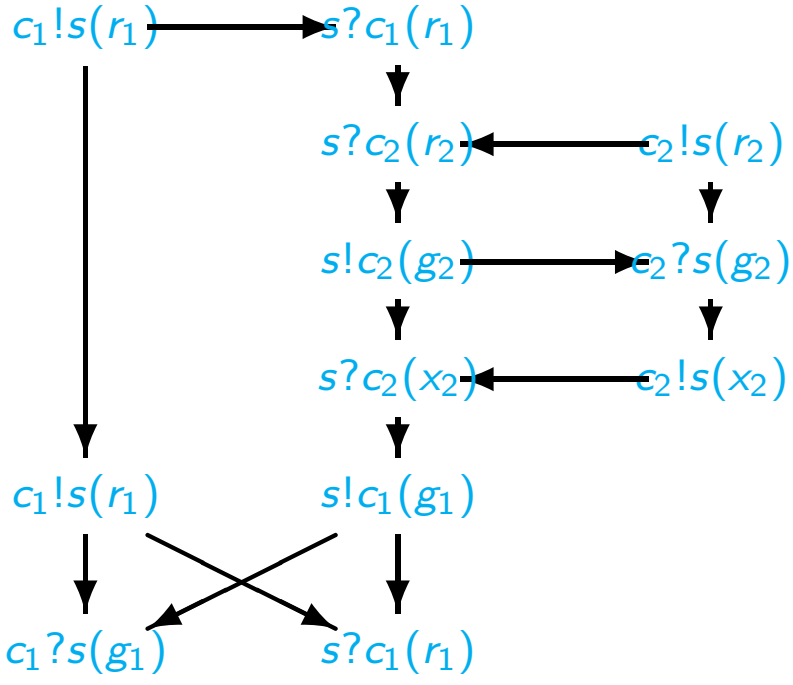
Linearizations of an MSC





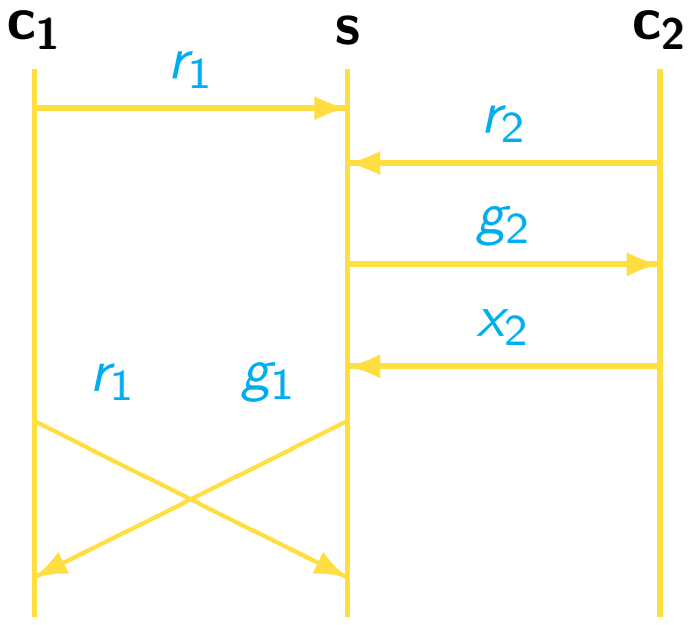
Linearizations of an MSC

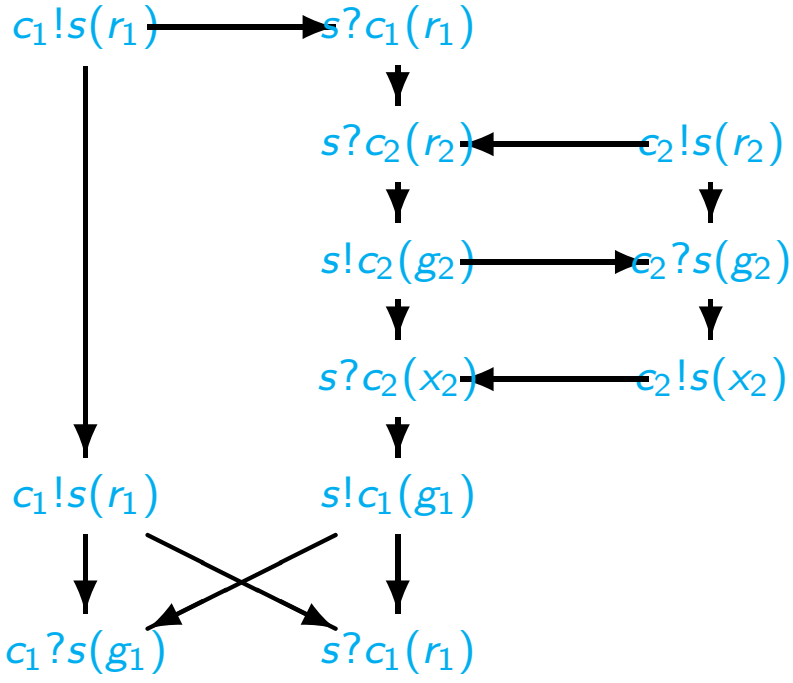




$c_2!s(r_2)\ c_1!s(r_1)\ c_1!s(r_1)\ s?c_1(r_1)\ s?c_2(r_2)\ s!c_2(g_2)\ c_2?s(g_2)\ c_2!s(x_2)$
$s?c_2(x_2)\ s!c_1(g_1)\ c_1?s(g_1)\ s?c_1(r_1)$

Linearizations of an MSC





$c_1!s(r_1)\ c_2!s(r_2)\ s?c_1(r_1)\ s?c_2(r_2)\ s!c_2(g_2)\ c_2?s(g_2)\ c_2!s(x_2)$
$s?c_2(x_2)\ s!c_1(g_1)\ c_1!s(r_1)\ c_1?s(g_1)\ s?c_1(r_1)$

Properties of Linearizations

Let w be a linearization of an MSC. Then,

Properties of Linearizations

Let w be a linearization of an MSC. Then,

- A message is received only after it is sent.

$$\forall \mathbf{x} \leq \mathbf{w}.\forall \mathbf{p}, \mathbf{q}. \quad \#_{\mathbf{p!q}}\mathbf{x} \geq \#_{\mathbf{q?p}}\mathbf{x}$$

Properties of Linearizations

Let w be a linearization of an MSC. Then,

- A message is received only after it is sent.

$$\forall \mathbf{x} \leq \mathbf{w}.\forall \mathbf{p}, \mathbf{q}.\ \ \#_{\mathbf{p!q}}\mathbf{x} \geq \#_{\mathbf{q?p}}\mathbf{x}$$

- All sent messages are received.

$$\forall \mathbf{p}, \mathbf{q}.\ \ \#_{\mathbf{p!q}}\mathbf{w} = \#_{\mathbf{q?p}}\mathbf{w}$$

Properties of Linearizations

Let w be a linearization of an MSC. Then,

- A message is received only after it is sent.

$$\forall \mathbf{x} \leq \mathbf{w}.\forall \mathbf{p}, \mathbf{q}.\ \ \#_{\mathbf{p!q}}\mathbf{x} \geq \#_{\mathbf{q?p}}\mathbf{x}$$

- All sent messages are received.

$$\forall \mathbf{p}, \mathbf{q}.\ \ \#_{\mathbf{p!q}}\mathbf{w} = \#_{\mathbf{q?p}}\mathbf{w}$$

As a matter of fact, under the FIFO assumption, any word satisfying these two properties is the linearization of a unique MSC.

Linearizations to MSCs

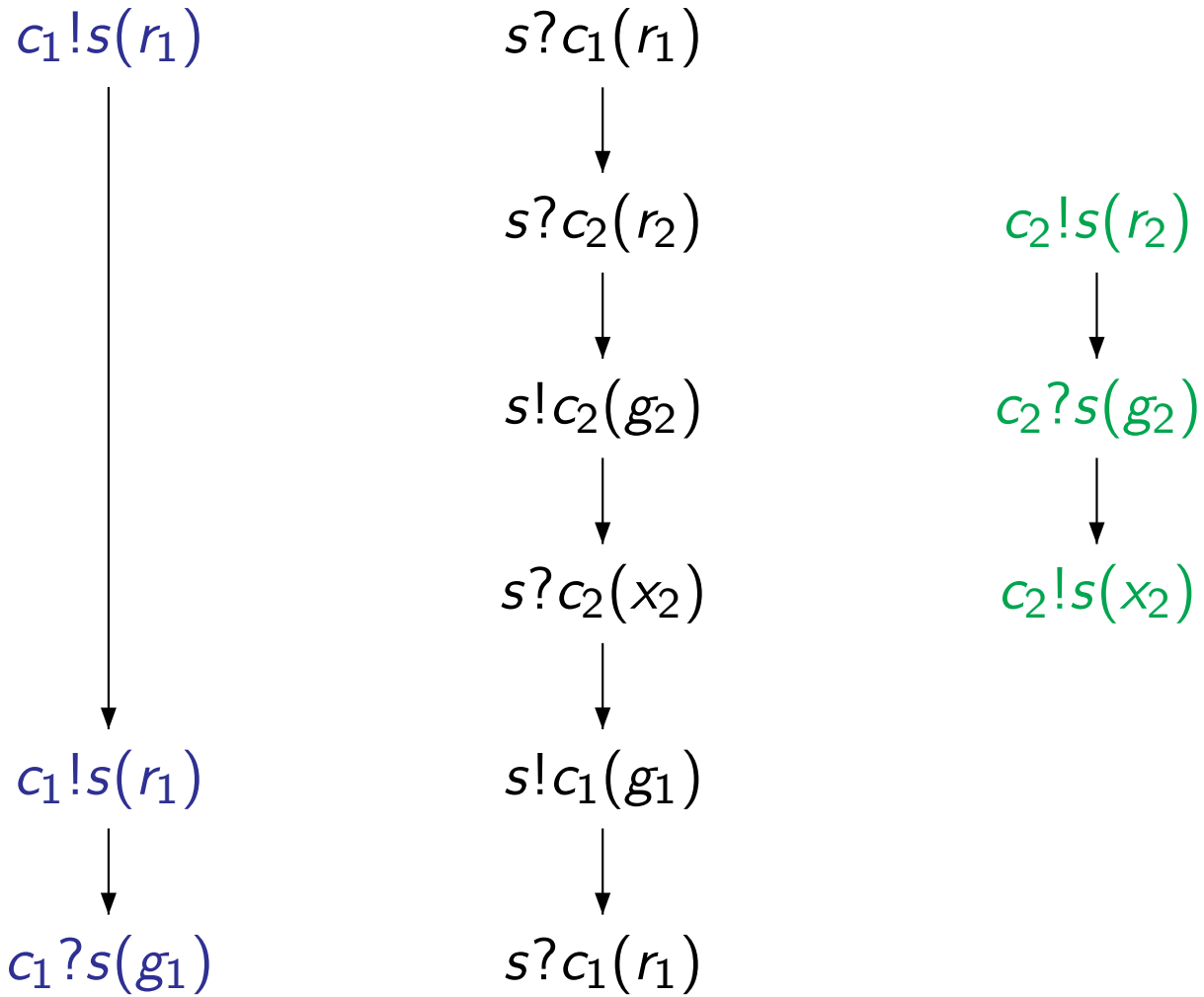
$c_2!s(r_2)$ $c_1!s(r_1)$ $c_1!s(r_1)$ $s?c_1(r_1)$ $s?c_2(r_2)$ $s!c_2(g_2)$ $c_2?s(g_2)$ $c_2!s(x_2)$
$s?c_2(x_2)$ $s!c_1(g_1)$ $c_1?s(g_1)$ $s?c_1(r_1)$

Linearizations to MSCs

$c_2!s(r_2)$ $c_1!s(r_1)$ $c_1!s(r_1)$ $s?c_1(r_1)$ $s?c_2(r_2)$ $s!c_2(g_2)$ $c_2?s(g_2)$ $c_2!s(x_2)$
$s?c_2(x_2)$ $s!c_1(g_1)$ $c_1?s(g_1)$ $s?c_1(r_1)$

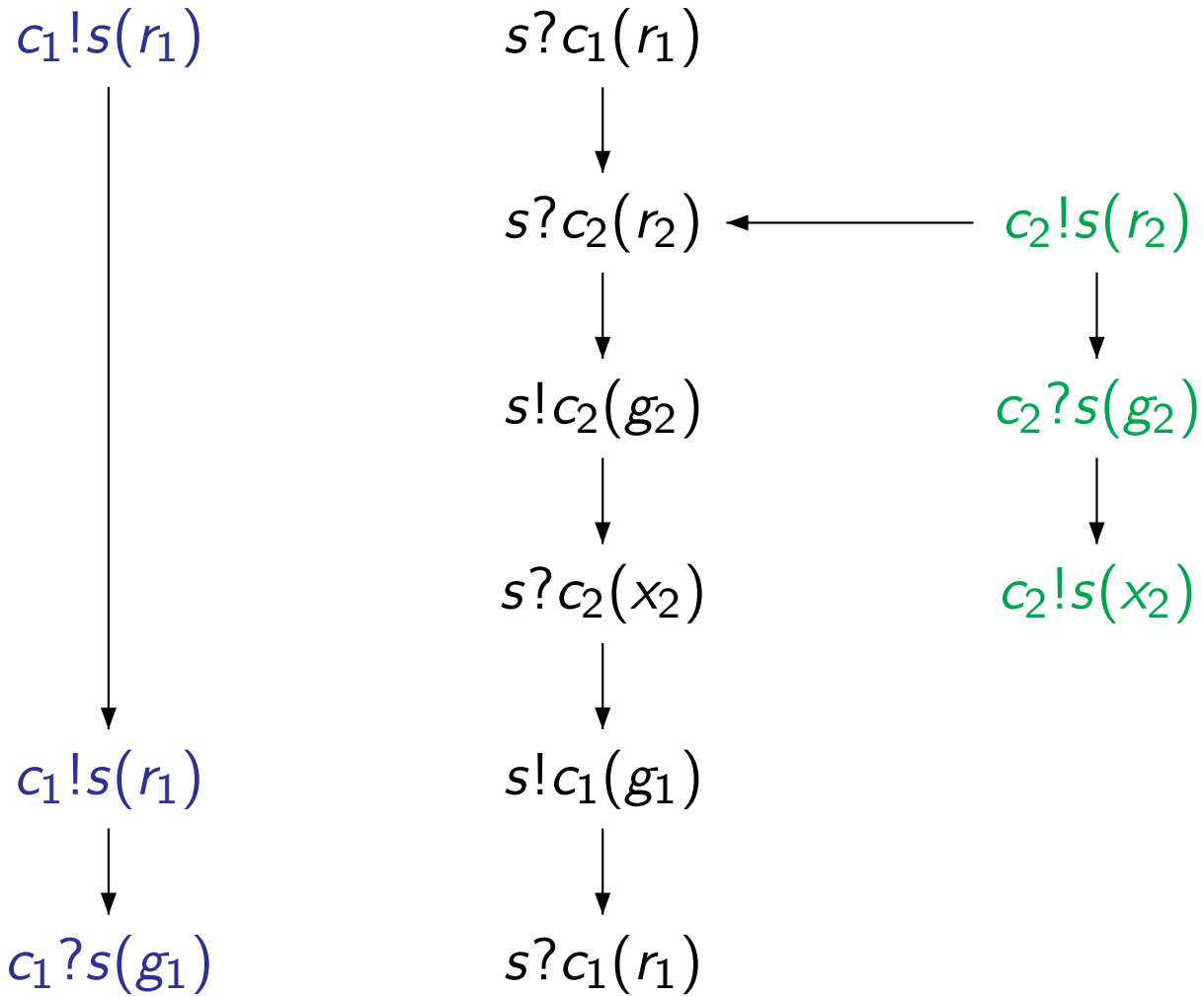
Linearizations to MSCs

$c_2!s(r_2)$ $c_1!s(r_1)$ $c_1!s(r_1)$ $s?c_1(r_1)$ $s?c_2(r_2)$ $s!c_2(g_2)$ $c_2?s(g_2)$ $c_2!s(x_2)$
$s?c_2(x_2)$ $s!c_1(g_1)$ $c_1?s(g_1)$ $s?c_1(r_1)$



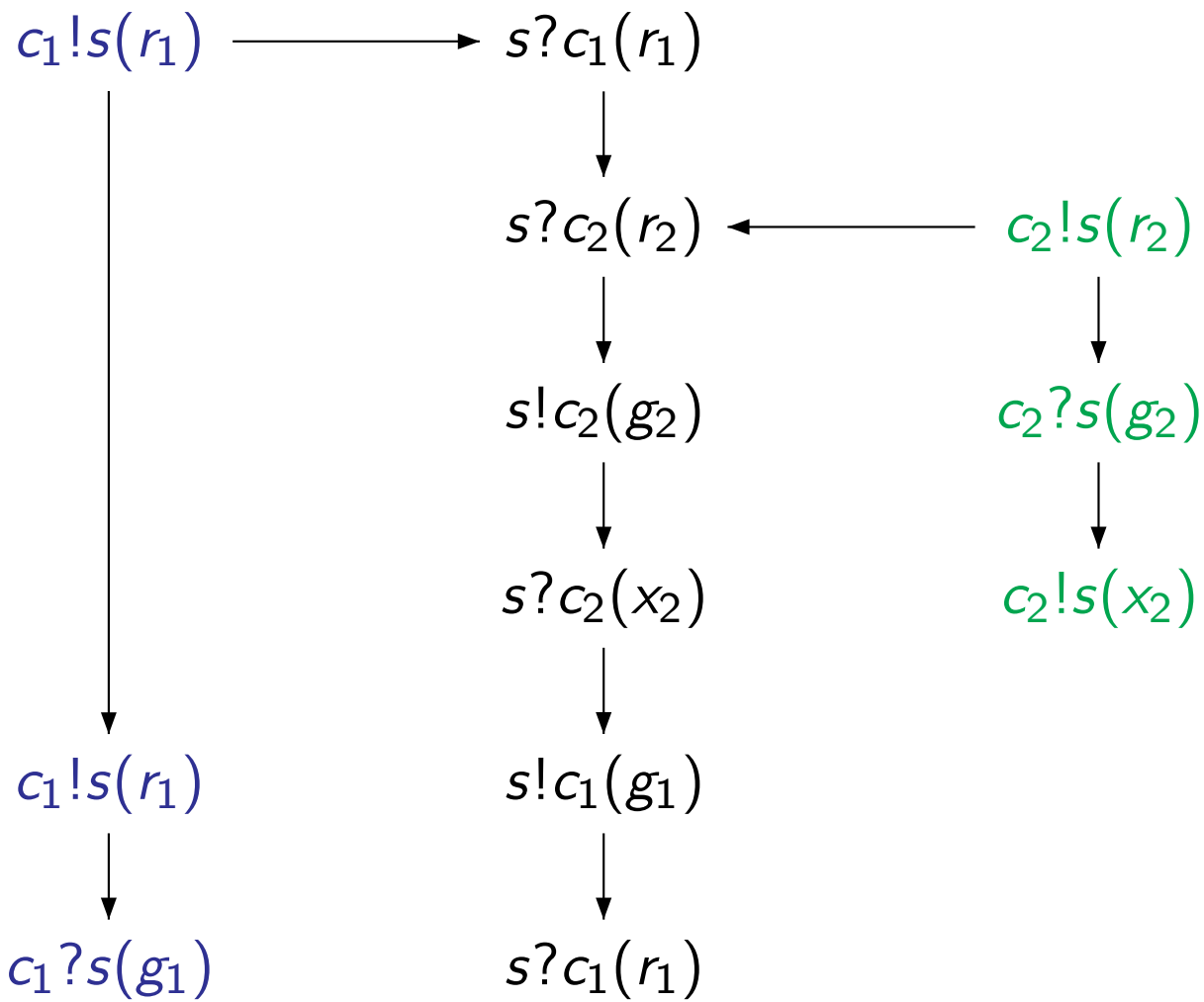
Linearizations to MSCs

$c_2!s(r_2)$ $c_1!s(r_1)$ $c_1!s(r_1)$ $s?c_1(r_1)$ $s?c_2(r_2)$ $s!c_2(g_2)$ $c_2?s(g_2)$ $c_2!s(x_2)$
$s?c_2(x_2)$ $s!c_1(g_1)$ $c_1?s(g_1)$ $s?c_1(r_1)$



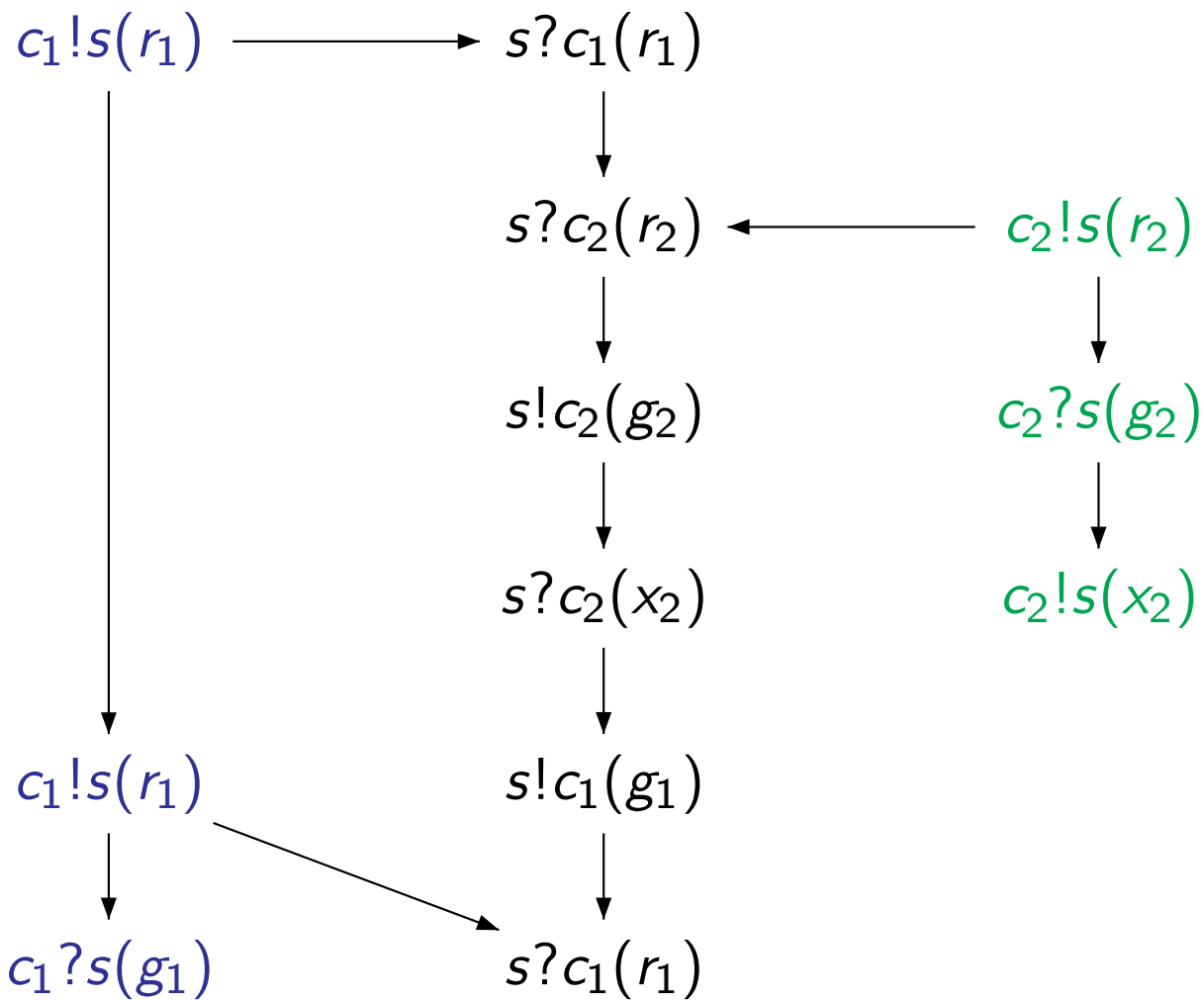
Linearizations to MSCs

$c_2!s(r_2)$ $c_1!s(r_1)$ $c_1!s(r_1)$ $s?c_1(r_1)$ $s?c_2(r_2)$ $s!c_2(g_2)$ $c_2?s(g_2)$ $c_2!s(x_2)$
$s?c_2(x_2)$ $s!c_1(g_1)$ $c_1?s(g_1)$ $s?c_1(r_1)$



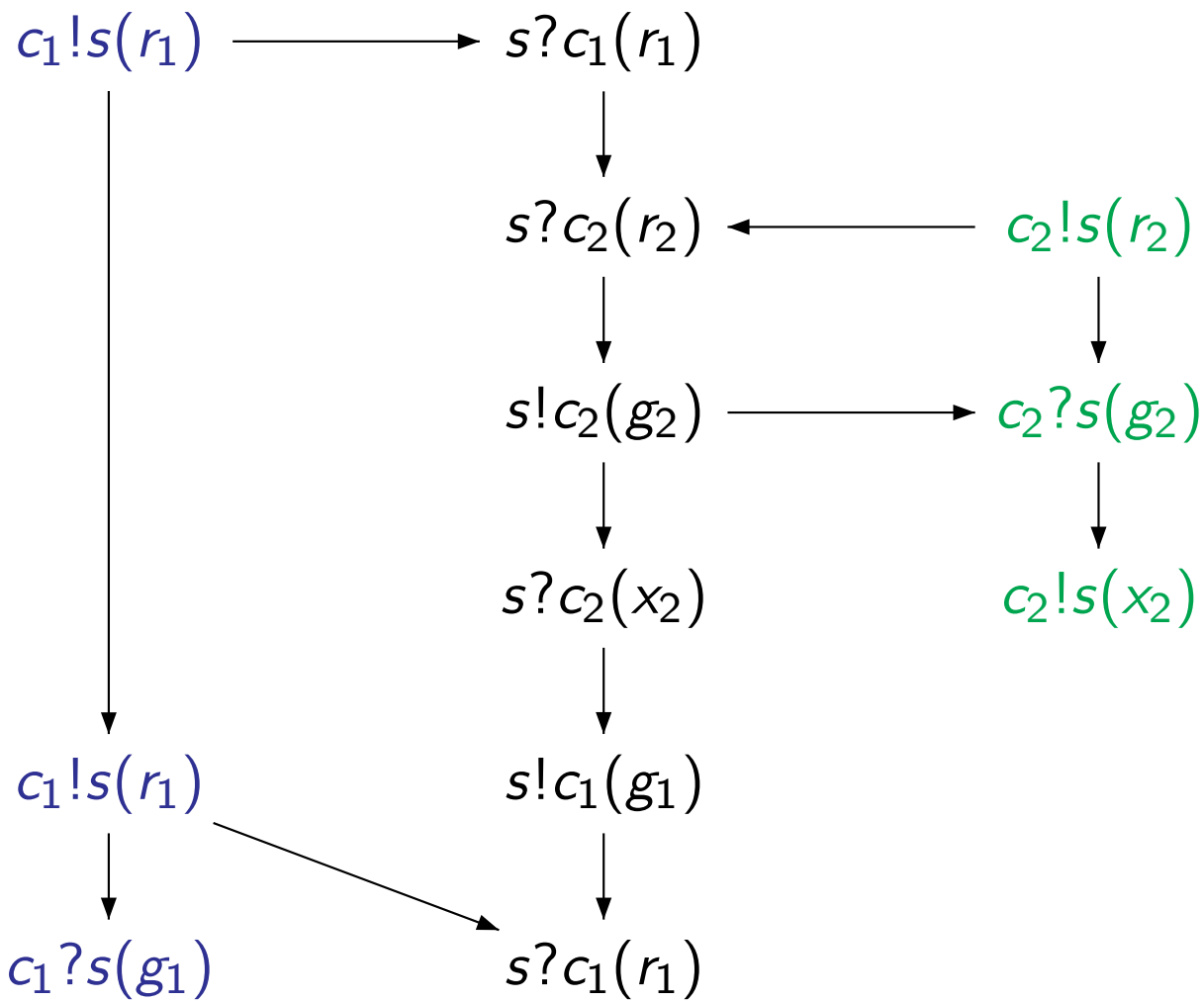
Linearizations to MSCs

$c_2!s(r_2)$ $c_1!s(r_1)$ $c_1!s(r_1)$ $s?c_1(r_1)$ $s?c_2(r_2)$ $s!c_2(g_2)$ $c_2?s(g_2)$ $c_2!s(x_2)$
$s?c_2(x_2)$ $s!c_1(g_1)$ $c_1?s(g_1)$ $s?c_1(r_1)$



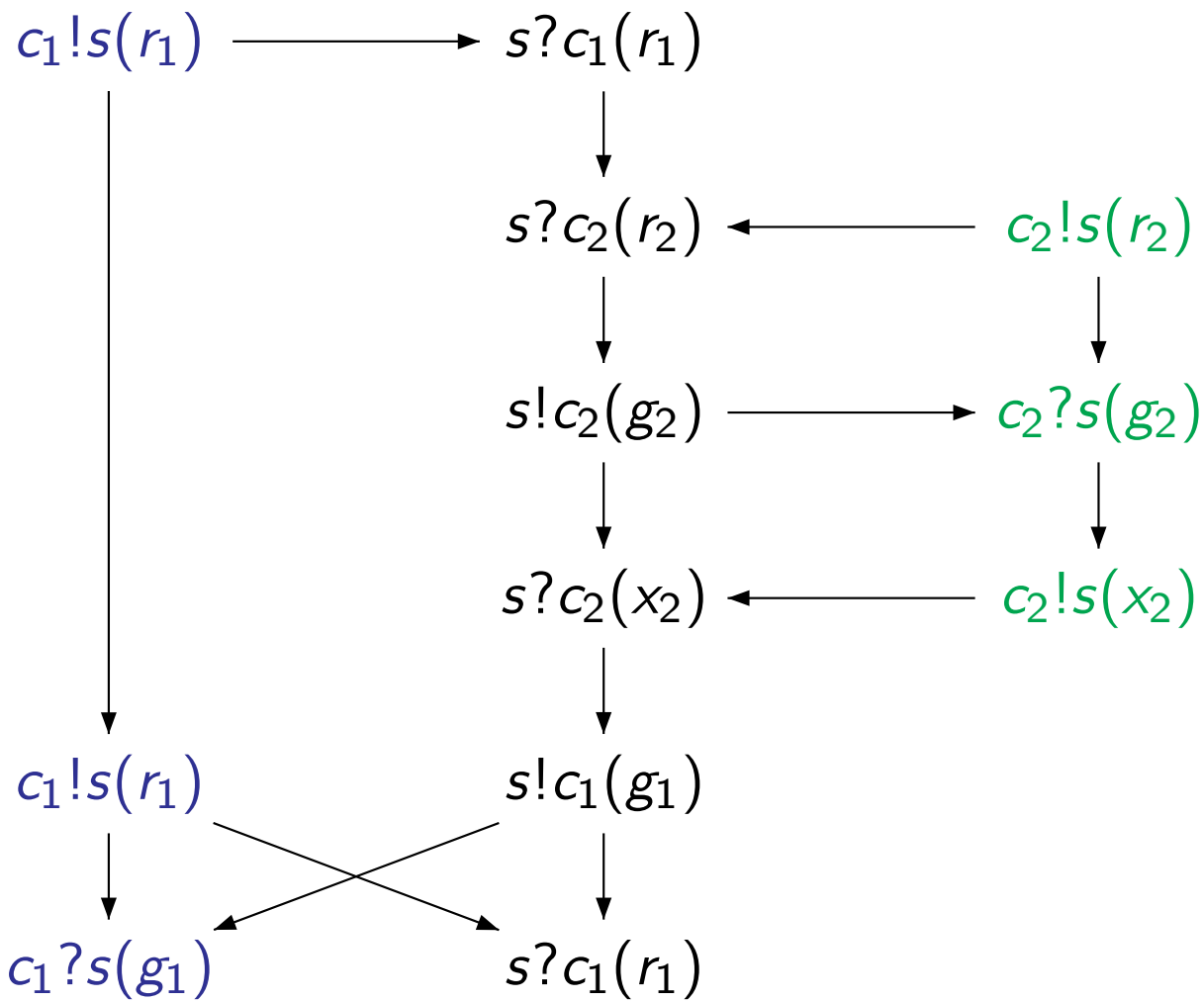
Linearizations to MSCs

$c_2!s(r_2)$ $c_1!s(r_1)$ $c_1!s(r_1)$ $s?c_1(r_1)$ $s?c_2(r_2)$ $s!c_2(g_2)$ $c_2?s(g_2)$ $c_2!s(x_2)$
$s?c_2(x_2)$ $s!c_1(g_1)$ $c_1?s(g_1)$ $s?c_1(r_1)$



Linearizations to MSCs

$c_2!s(r_2)$ $c_1!s(r_1)$ $c_1!s(r_1)$ $s?c_1(r_1)$ $s?c_2(r_2)$ $s!c_2(g_2)$ $c_2?s(g_2)$ $c_2!s(x_2)$
$s?c_2(x_2)$ $s!c_1(g_1)$ $c_1?s(g_1)$ $s?c_1(r_1)$



Languages of MSCs

- A language of MSCs is a set of MSCs (over a fixed set of processes and message alphabet).

Languages of MSCs

- A language of MSCs is a set of MSCs (over a fixed set of processes and message alphabet).
- MSCs are labelled partial orders, and an MSC can be reconstructed from any of its linearizations.

Languages of MSCs

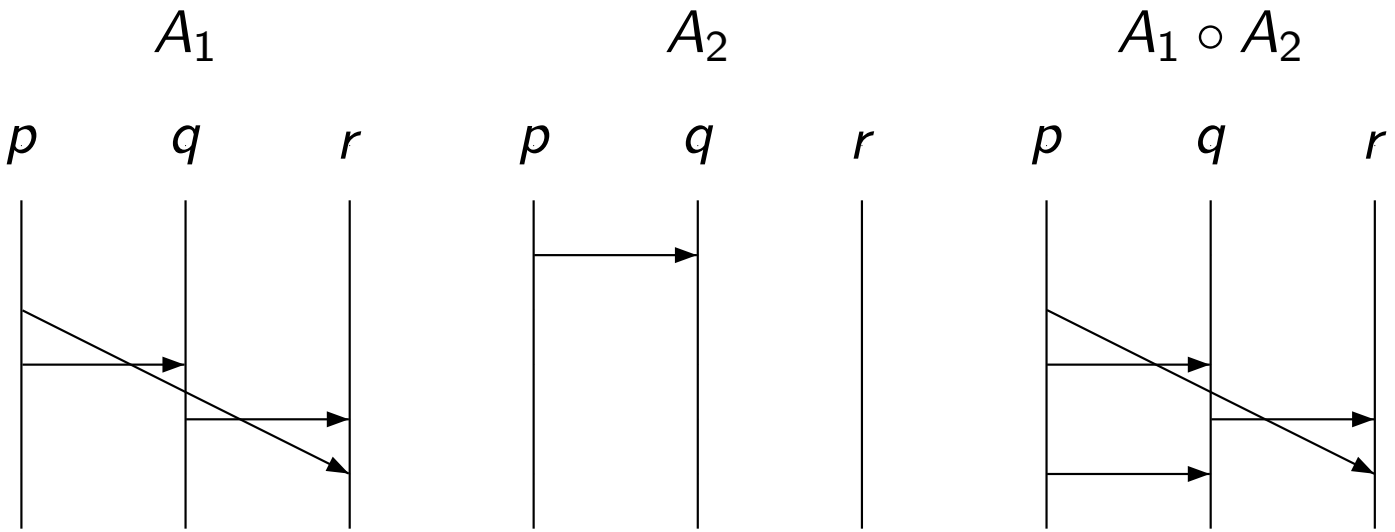
- A language of MSCs is a set of MSCs (over a fixed set of processes and message alphabet).
- MSCs are labelled partial orders, and an MSC can be reconstructed from any of its linearizations.
- An MSC language L can also be thought of as the word language of the linearizations of the MSCs in L.

Languages of MSCs

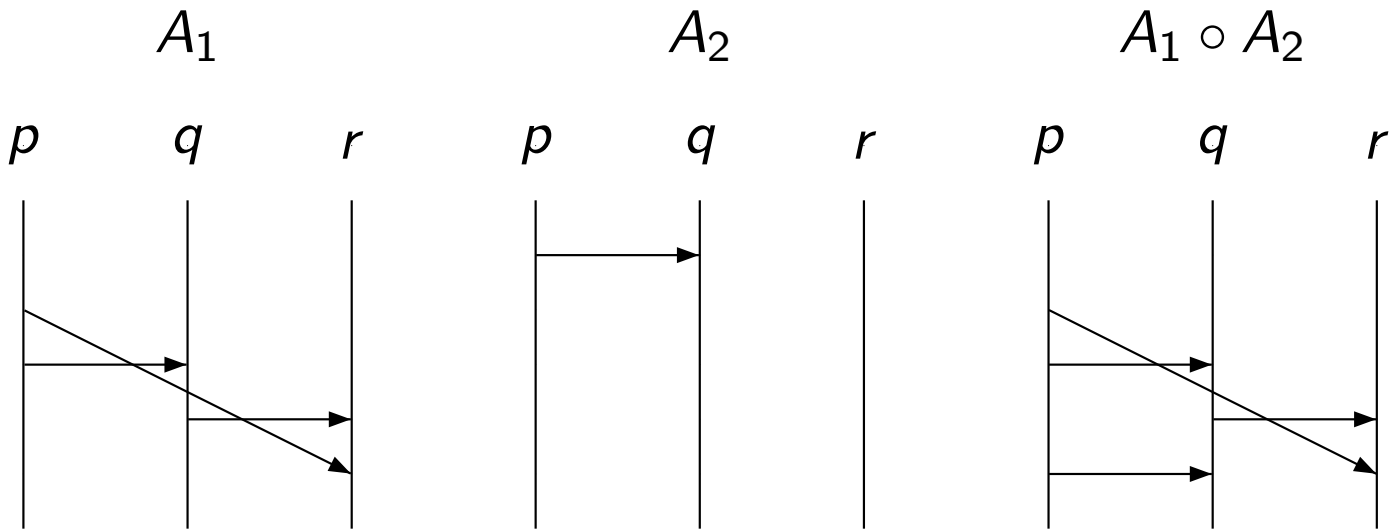
- A language of MSCs is a set of MSCs (over a fixed set of processes and message alphabet).
- MSCs are labelled partial orders, and an MSC can be reconstructed from any of its linearizations.
- An MSC language L can also be thought of as the word language of the linearizations of the MSCs in L.

How do we describe MSC languages?

Concatenation of MSCs

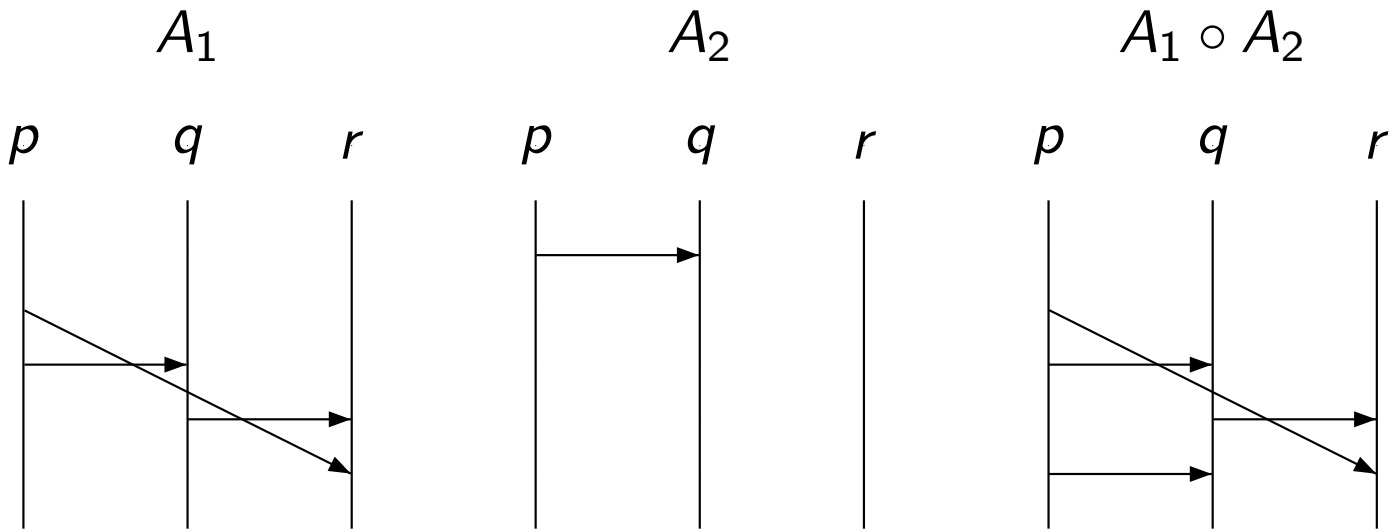


Concatenation of MSCs



Every event from A_2 in any process p occurs after all the events from A_1 in the process p have already occurred.

Concatenation of MSCs



Every event from A_2 in any process p occurs after all the events from A_1 in the process p have already occurred.

p!r p!q q?p q!r r?q p!q q?p r?p

is a linearization of $A_1 \circ A_2$.

Message Sequence Graphs

- A finite state automaton

Message Sequence Graphs

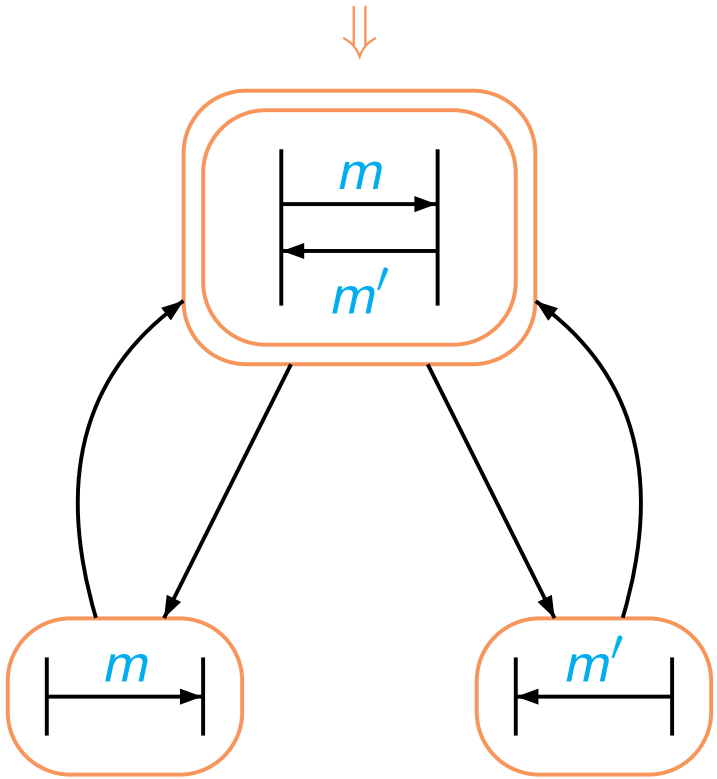
- A finite state automaton
- Each state is labelled by an MSC

Message Sequence Graphs

- A finite state automaton
- Each state is labelled by an MSC
- Each (legal) path in the automaton generates a MSC

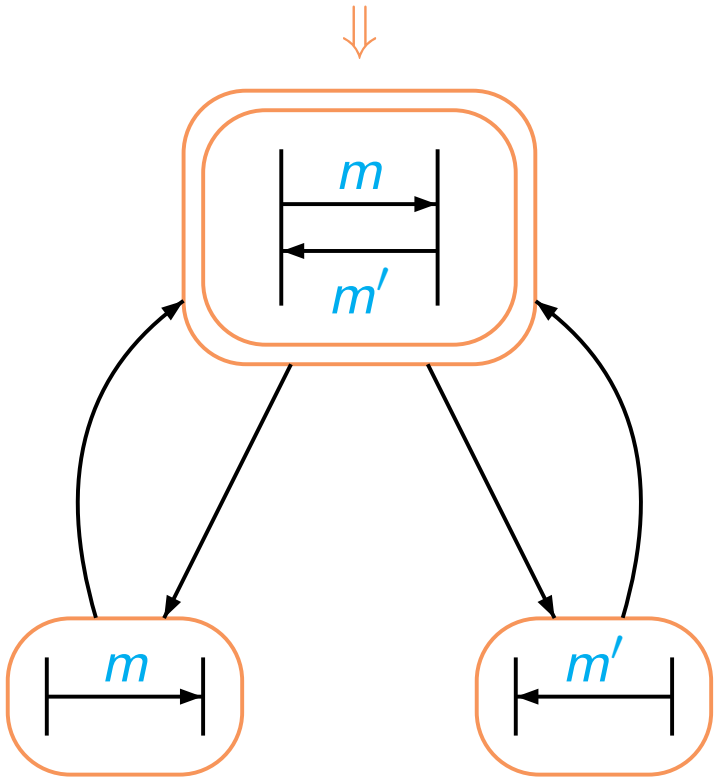
Message Sequence Graphs

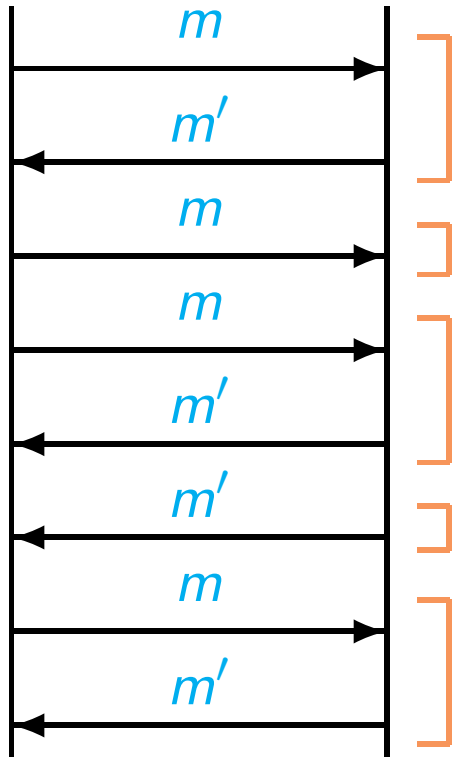
- A finite state automaton
- Each state is labelled by an MSC
- Each (legal) path in the automaton generates a MSC



Message Sequence Graphs

- A finite state automaton
- Each state is labelled by an MSC
- Each (legal) path in the automaton generates a MSC





Message Sequence Graphs

- MSGs are part of the UML definition and are used in practice

Message Sequence Graphs

- MSGs are part of the UML definition and are used in practice
- They are used to *specify* the desired (or undesired behaviours) of a system.

Message Sequence Graphs

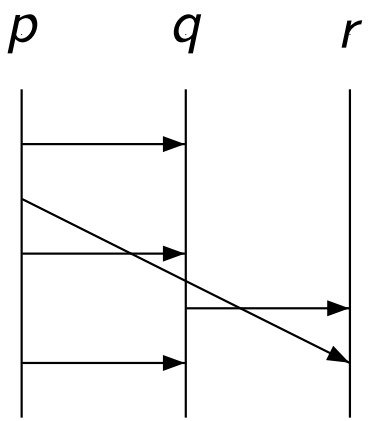
- MSGs are part of the UML definition and are used in practice
- They are used to *specify* the desired (or undesired behaviours) of a system.
- They are global descriptions and do not give any guarantee about implementability.

Boundedness

A linearization of an MSC is B-bounded if no channel has more than B messages at any point.

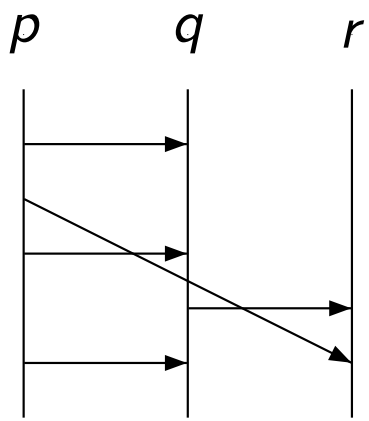
Boundedness

A linearization of an MSC is B-bounded if no channel has more than B messages at any point.



Boundedness

A linearization of an MSC is B-bounded if no channel has more than B messages at any point.



The linearization

p!q q?p p!r p!q q?p q!r r?q p!q q?p r?p

is 1-bounded while the linearization

p!q p!r p!q p!q q?p q?p p!r q?p r?q r?p

is 3-bounded.

Boundedness ...

An MSC is existentially B-bounded if one of its linearizations is B-bounded.

Boundedness ...

An MSC is existentially B-bounded if one of its linearizations is B-bounded.

> *The execution of the MSC can be scheduled in such a way that buffers are all B-bounded.*

Boundedness ...

An MSC is existentially B-bounded if one of its linearizations is B-bounded.

> *The execution of the MSC can be scheduled in such a way that buffers are all B-bounded.*

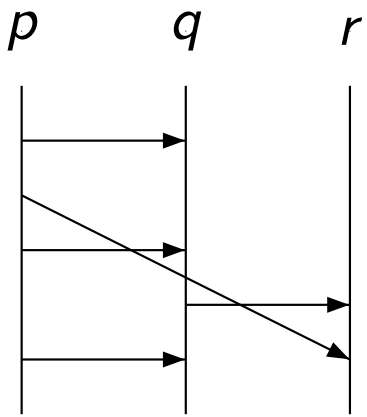
An MSC is universally B-bounded if all of its linearizations are B-bounded.

Boundedness ...

An MSC is existentially B-bounded if one of its linearizations is B-bounded.

> *The execution of the MSC can be scheduled in such a way that buffers are all B-bounded.*

An MSC is universally B-bounded if all of its linearizations are B-bounded.



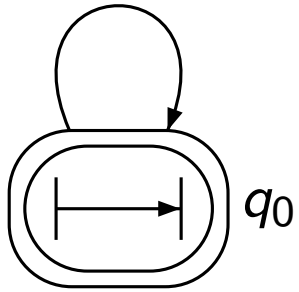
An existentially 1-bounded and universally 3-bounded MSC.

Boundedness ...

An MSG is existentially B-bounded if every MSC it generates is existentially B-bounded.

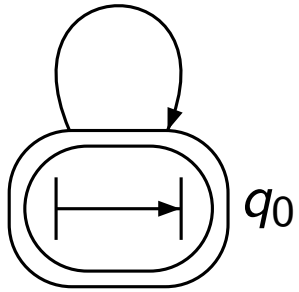
Boundedness ...

An MSG is existentially B-bounded if every MSC it generates is existentially B-bounded.



Boundedness ...

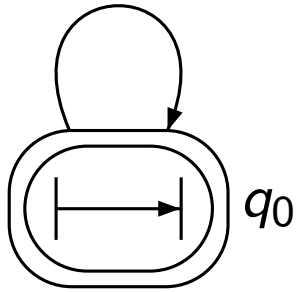
An MSG is existentially B-bounded if every MSC it generates is existentially B-bounded.



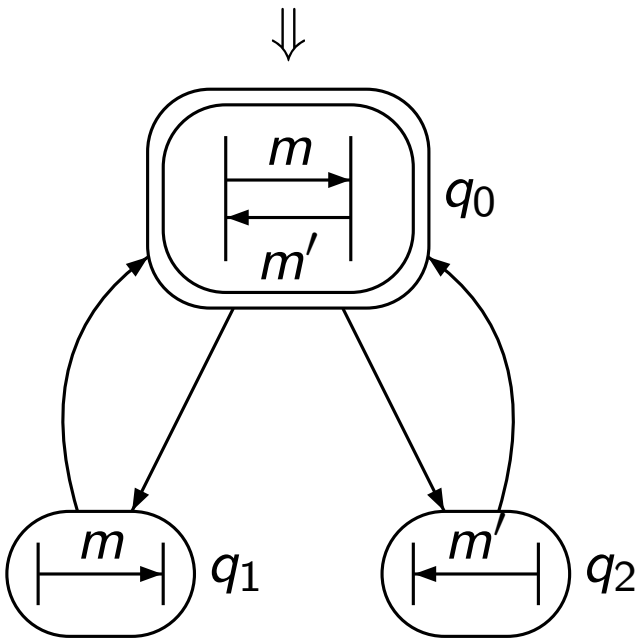
An MSG is universally B-bounded if every MSC it generates is universally B-bounded.

Boundedness ...

An MSG is existentially B-bounded if every MSC it generates is existentially B-bounded.

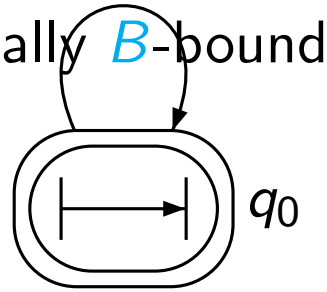


An MSG is universally B-bounded if every MSC it generates is universally B-bounded.

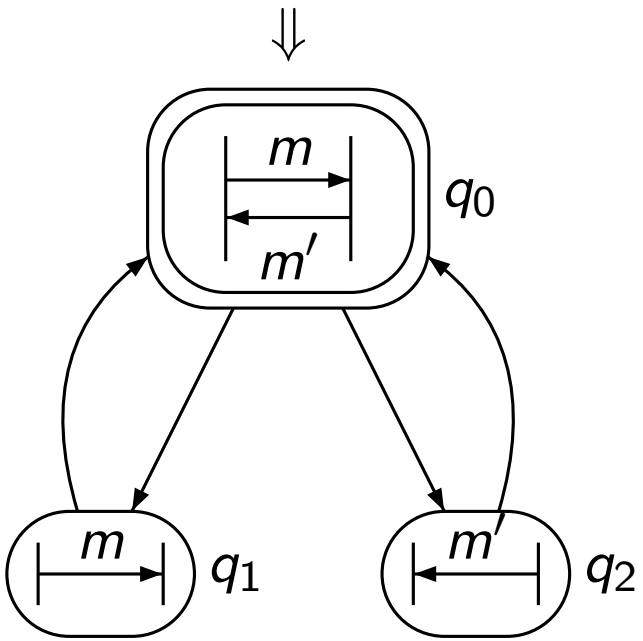


Boundedness ...

An MSG is existentially bounded if there exists a B such that every MSC it generates is existentially B-bounded.



An MSG is universally bounded if there exists a B such that every MSC it generates is B-bounded.



Deciding Boundedness

Can we check whether the language of an MSG is implementable with bounded buffers?

Deciding Boundedness

Can we check whether the language of an MSG is implementable with bounded buffers?

- Every MSG is existentially B-bounded for some B.

Deciding Boundedness

Can we check whether the language of an MSG is implementable with bounded buffers?

- Every MSG is existentially B-bounded for some B.
- Checking whether an MSG is existentially B-bounded for a given B is decidable.

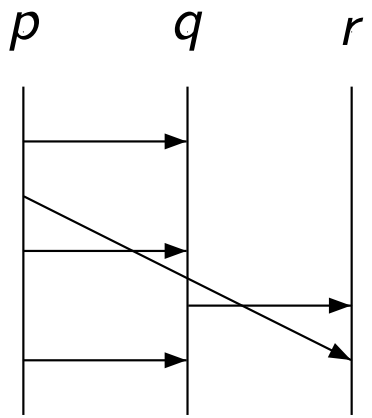
Deciding Boundedness

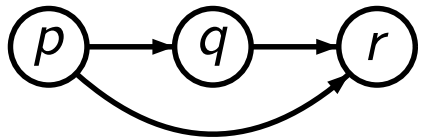
Can we check whether the language of an MSG is implementable with bounded buffers?

- Every MSG is existentially B-bounded for some B.
- Checking whether an MSG is existentially B-bounded for a given B is decidable.
- Checking whether an MSG is universally bounded (B-bounded for a given B) is decidable.

Communication graph of an MSC

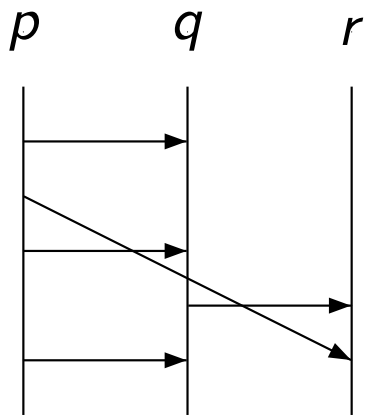
Nodes are the processes. An edge from p to q if there is a message from p to q.

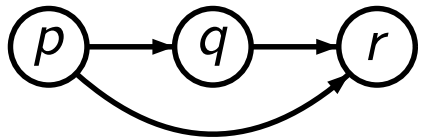




Communication graph of an MSC

Nodes are the processes. An edge from p to q if there is a message from p to q.





An MSG is locally strongly connected if and only if the MSC generated by any loop has a communication graph that is the disjoint union of SCCs.

SCCs and boundedness

- Suppose p and q lie in the same SCC in the communication graph of M and further suppose that this SCC has k processes

SCCs and boundedness

- Suppose p and q lie in the same SCC in the communication graph of M and further suppose that this SCC has k processes
- In M^k, the first q event guaranteed to be below the maximum event of p.

SCCs and boundedness

- Suppose p and q lie in the same SCC in the communication graph of M and further suppose that this SCC has k processes

- In M^k, the first q event guaranteed to be below the maximum event of p.
- In M^{k+1}, the first $q?p$ event corresponding to the first $p!q$ event (if any) is guaranteed to be below the maximum event of p.

  The first message from p to q is acknowledged within M^{k+1}.

SCCs and boundedness

- Suppose p and q lie in the same SCC in the communication graph of M and further suppose that this SCC has k processes
- In M^k, the first q event guaranteed to be below the maximum event of p.
- In M^{k+1}, the first $q?p$ event corresponding to the first $p!q$ event (if any) is guaranteed to be below the maximum event of p.

  The first message from p to q is acknowledged within M^{k+1}.

- The channel from p to q is bounded by the size of M^{k+1}.

Regularity of linearizations

When do we say that a language of MSCs is regular?

Regularity of linearizations

When do we say that a language of MSCs is regular?

1. Finite state implementations.

Regularity of linearizations

When do we say that a language of MSCs is regular?

1. Finite state implementations.
2. Can be analysed. Reachability, model-checking, ...

Regularity of linearizations

When do we say that a language of MSCs is regular?

1. Finite state implementations.
2. Can be analysed. Reachability, model-checking, ...
3. Robust characterizations via logics, ...

Regularity of linearizations

When do we say that a language of MSCs is regular?

1. Finite state implementations.
2. Can be analysed. Reachability, model-checking, ...
3. Robust characterizations via logics, ...

A language of MSCs is said to be regular if its set of linearizations forms a regular (word) language.

Regularity of linearizations

When do we say that a language of MSCs is regular?

1. Finite state implementations.
2. Can be analysed. Reachability, model-checking, ...
3. Robust characterizations via logics, ...

A language of MSCs is said to be regular if its set of linearizations forms a regular (word) language.

Other definitions are possible ...

Boundedness and Regularity

Every regular language of MSCs is universally bounded.

Boundedness and Regularity

Every regular language of MSCs is universally bounded.

The number of different buffer configurations is bounded by the number of states of any FA accepting the linearizations.

Boundedness and Regularity

Every regular language of MSCs is universally bounded.

The number of different buffer configurations is bounded by the number of states of any FA accepting the linearizations.

> *Let A be a finite automaton for the linearizations with s as the initial state. If w and v lead to the same state q from s and u is word that runs from q to some final state then ...*

Boundedness and Regularity

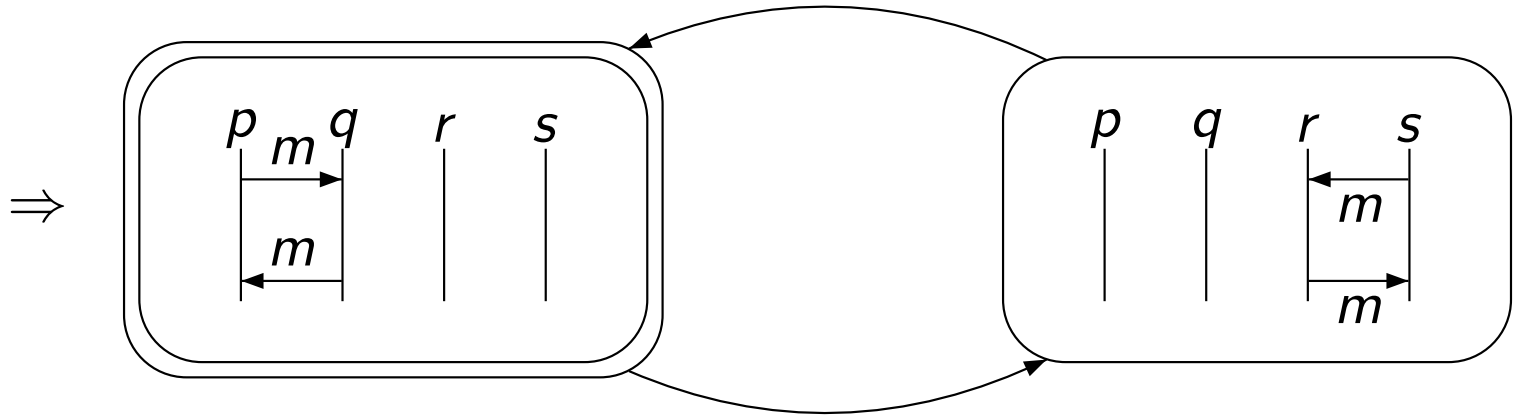
Every regular language of MSCs is universally bounded.

The number of different buffer configurations is bounded by the number of states of any FA accepting the linearizations.

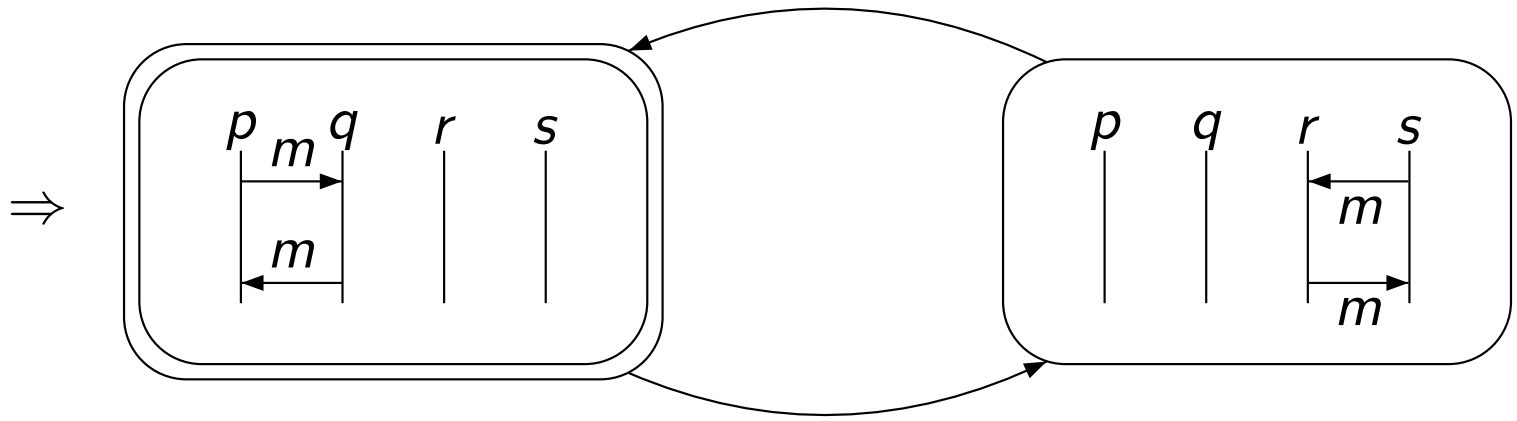
> *Let A be a finite automaton for the linearizations with s as the initial state. If w and v lead to the same state q from s and u is word that runs from q to some final state then ...*

What about the converse?

Iteration of concurrent events

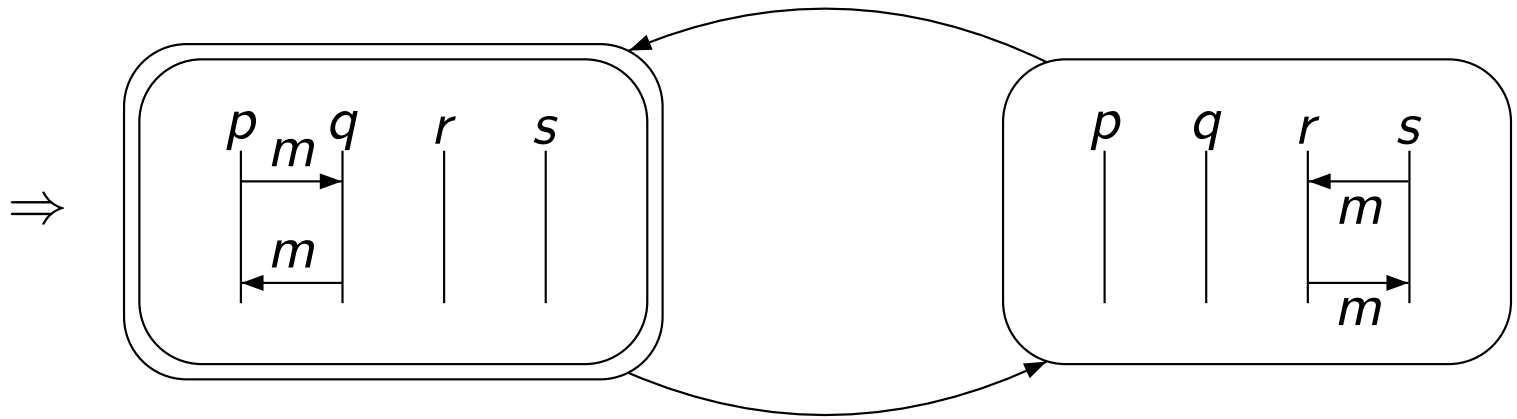


Iteration of concurrent events



The only constraint on relating the $p!q(m)$ events and the $r!s(m)$ is that they must be equal in number.

Iteration of concurrent events



The only constraint on relating the $p!q(m)$ events and the $r!s(m)$ is that they must be equal in number.

Thus even bounded MSGs can exhibit nonregular behaviour.

Regularity ...

Can we check whether a MSG describes a regular language or not?

Regularity ...

Can we check whether a MSG describes a regular language or not?

Answer: NO.

Regularity ...

Can we check whether a MSG describes a regular language or not?

Answer: NO.

The proof follows from a similar result for *Mazurkiewicz Traces* due to Sakarovich.

Regularity ...

Can we check whether a MSG describes a regular language or not?

Answer: NO.

The proof follows from a similar result for *Mazurkiewicz Traces* due to Sakarovich.

What about sufficient conditions for regularity?

Locally synchronized MSGs

An MSC is said to be locally synchronized if its communication graph has a single nontrivial SCC.

Locally synchronized MSGs

An MSC is said to be locally synchronized if its communication graph has a single nontrivial SCC.

An MSG is said to be locally synchronized if every closed walk in the MSG generates a locally synchronized MSC.

Locally synchronized MSGs

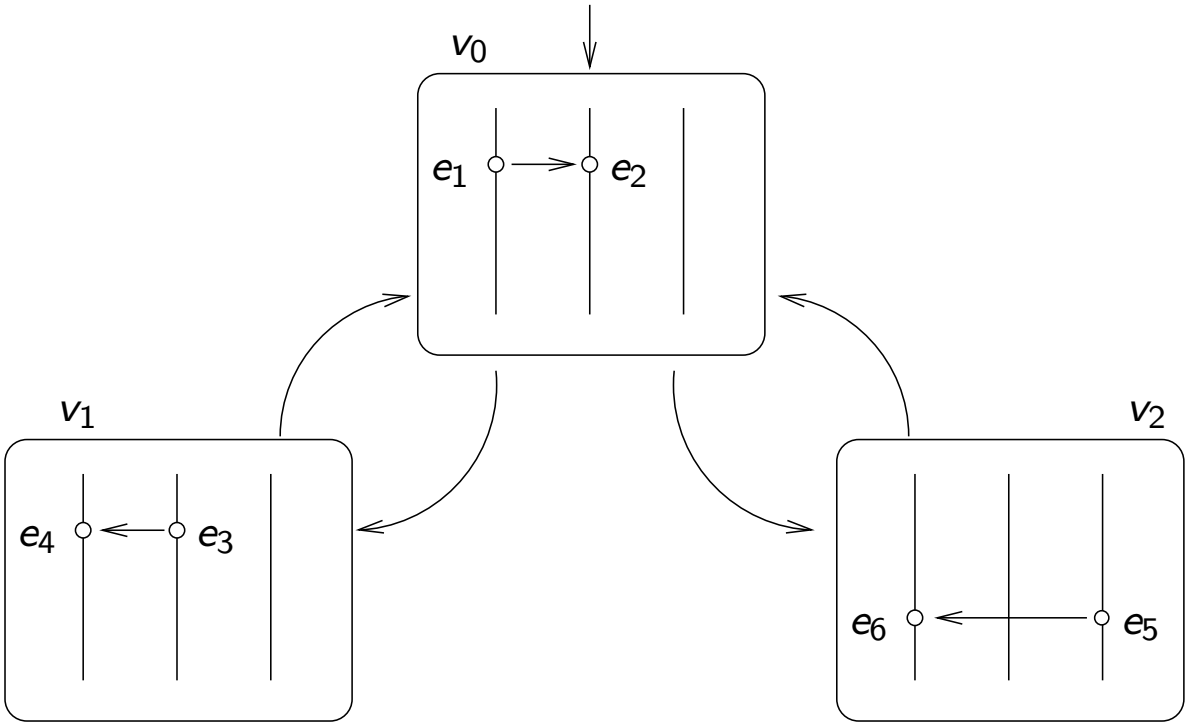
An MSC is said to be locally synchronized if its communication graph has a single nontrivial SCC.

An MSG is said to be locally synchronized if every closed walk in the MSG generates a locally synchronized MSC.

Theorem: Every locally synchronized MSG generates a regular MSC language.

Locally synchronized MSGs ...

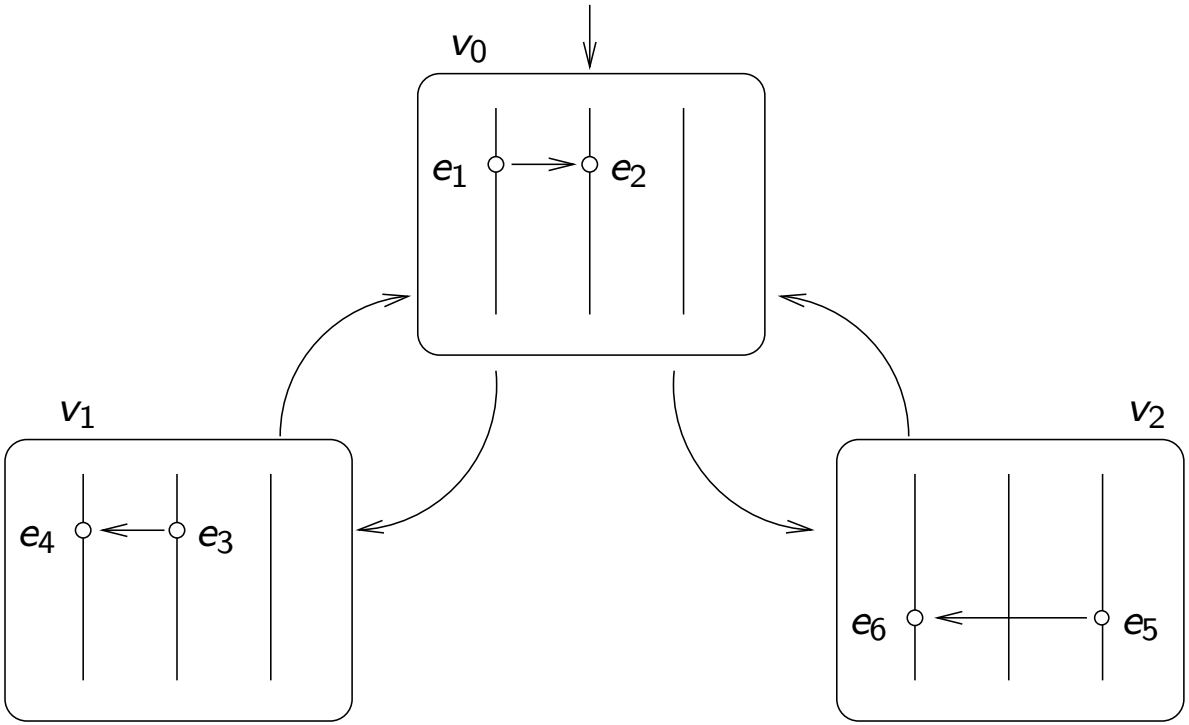
- Every linearization of the MSCs generated must be exhibited.



$$\mathbf{e_5(e_1e_2e_3e_4)^*e_6}$$

Locally synchronized MSGs ...

- Every linearization of the MSCs generated must be exhibited.

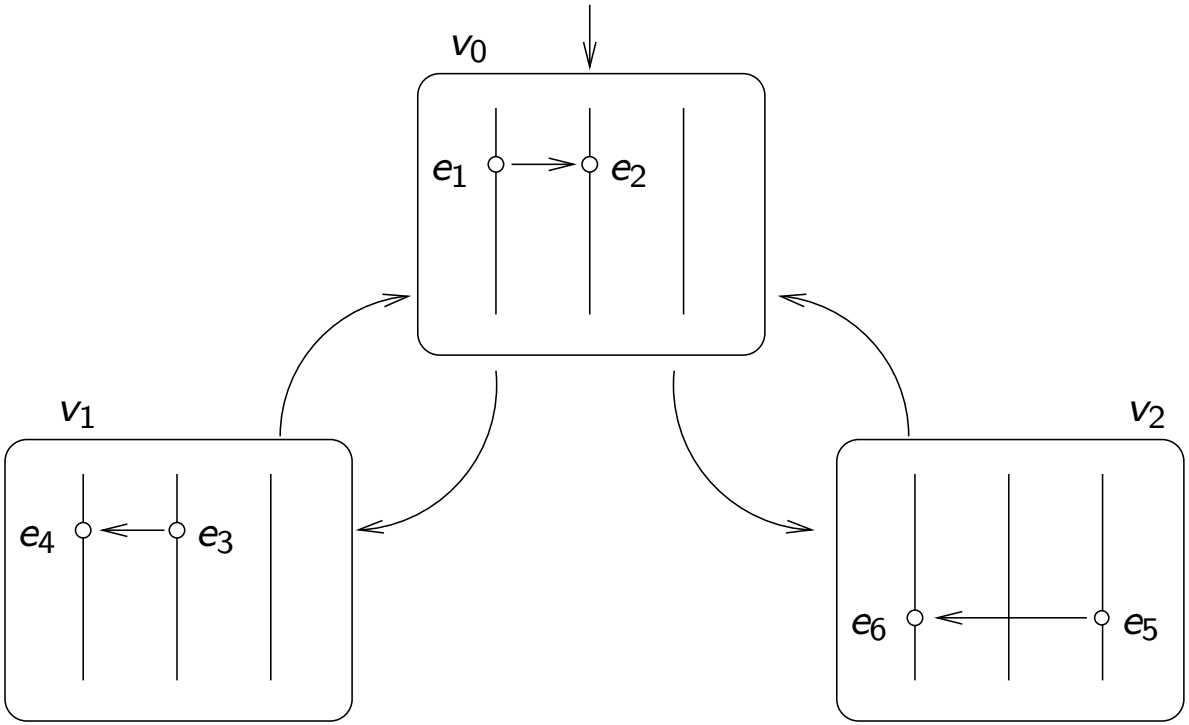


$$\mathbf{e_5(e_1e_2e_3e_4)^*e_6}$$

- How many and which nodes of the path in the MSG do we need to carry with us to generate these sequentializations ?

Locally synchronized MSGs ...

- Every linearization of the MSCs generated must be exhibited.



$$\mathbf{e_5(e_1e_2e_3e_4)^*e_6}$$

- How many and which nodes of the path in the MSG do we need to carry with us to generate these sequentializations ?
- Nodes have to inserted in the middle, deleted from the middle

Locally synchronized MSGs and regularity

There are regular MSC languages that are not generated by MSGs.

Locally synchronized MSGs and regularity

There are regular MSC languages that are not generated by MSGs.

- The language of an MSG is constructed from a finite set of basic building blocks (*atoms*).

Locally synchronized MSGs and regularity

There are regular MSC languages that are not generated by MSGs.

- The language of an MSG is constructed from a finite set of basic building blocks (*atoms*).
- There are regular MSC languages which need infinitely many basic blocks. (eg.)

$$\mathbf{p!r\ p!q\ q?p\ (q!r\ r?q\ r!q\ q?r))^*\ r?p}$$

Locally synchronized MSGs and regularity

There are regular MSC languages that are not generated by MSGs.

- The language of an MSG is constructed from a finite set of basic building blocks (*atoms*).
- There are regular MSC languages which need infinitely many basic blocks. (eg.)

$$\mathbf{p!r\ p!q\ q?p\ (q!r\ r?q\ r!q\ q?r))^*\ r?p}$$

Theorem: Every finitely generated MSC regular language is the language of a locally synchronized MSG.

Message-passing Automata

1. An implementation model for MSCs. A bunch of finite automata communicating via buffered channels.

Message-passing Automata

1. An implementation model for MSCs. A bunch of finite automata communicating via buffered channels.
2. A distributed model of computation.

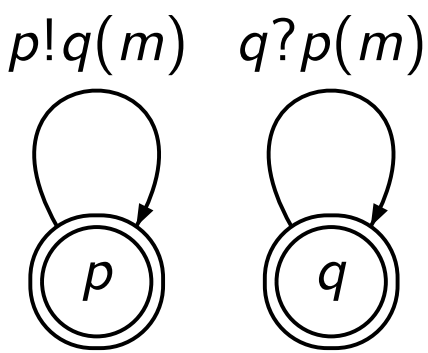
Message-passing Automata

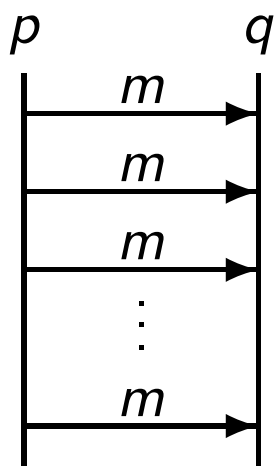
1. An implementation model for MSCs. A bunch of finite automata communicating via buffered channels.
2. A distributed model of computation.

Message-passing Automata

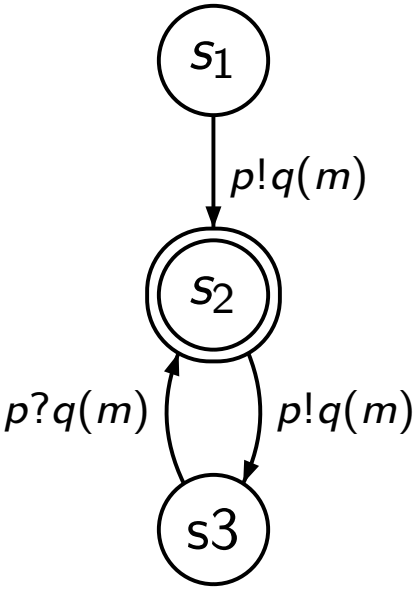
1. An implementation model for MSCs. A bunch of finite automata communicating via buffered channels.
2. A distributed model of computation.

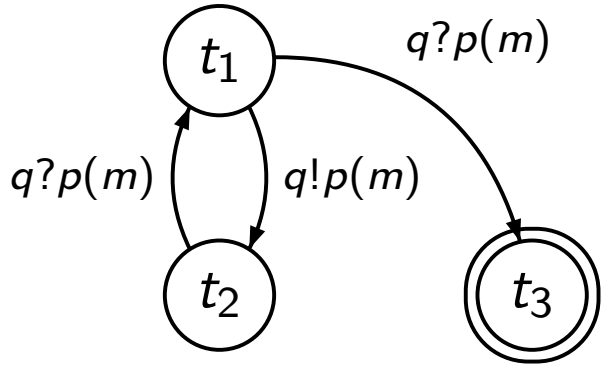
An Example: The producer–consumer system.

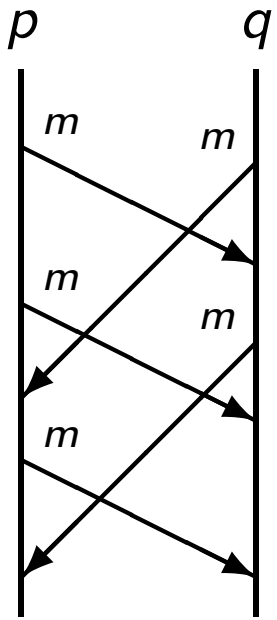




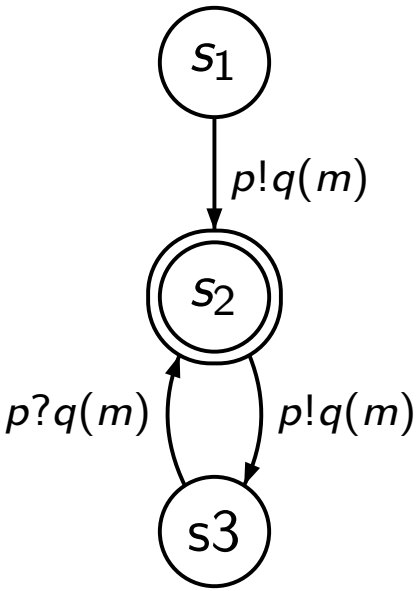
Message-passing Automata ...

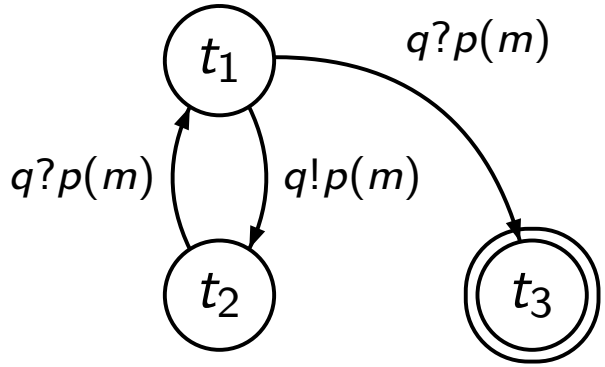


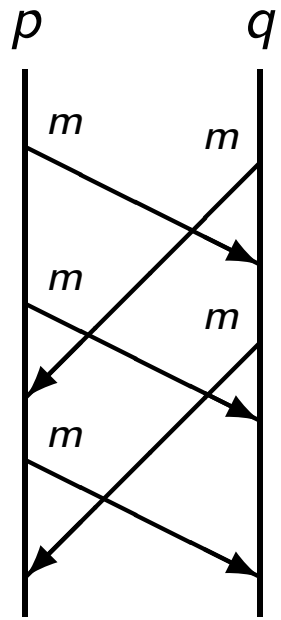




Message-passing Automata ...







MPAs can accept languages that cannot be generated by MSGs.

MSGs to MPAs

When are MSG languages implementable using MPAs?

MSGs to MPAs

When are MSG languages implementable using MPAs?

- MPA languages are product languages. (i.e.) if M is such that for all $p \in P$ there is an $M_p \in L$ such that $M \downarrow P = M_p \downarrow P$ then $M \in L$.

MSGs to MPAs

When are MSG languages implementable using MPAs?

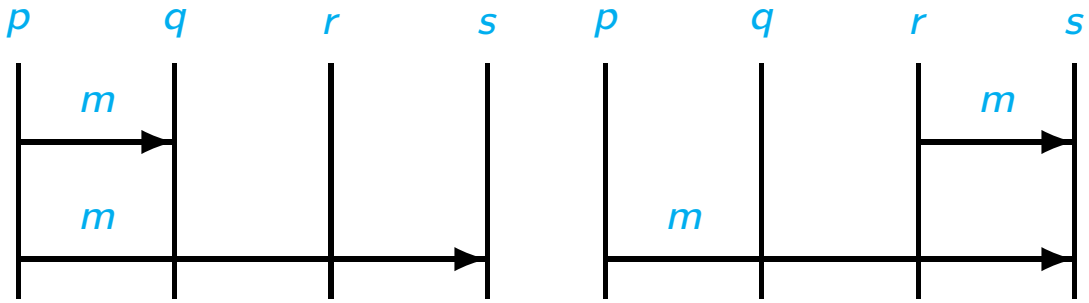
- MPA languages are product languages. (i.e.) if M is such that for all $p \in P$ there is an $M_p \in L$ such that $M \downarrow P = M_p \downarrow P$ then $M \in L$.
- An M of the form exhibited above, is said to be implied by L.

MSGs to MPAs

When are MSG languages implementable using MPAs?

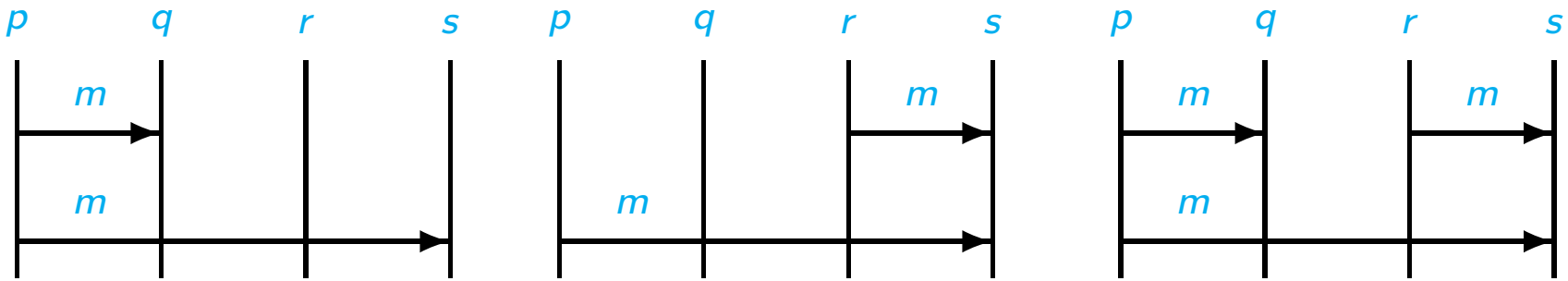
- MPA languages are product languages. (i.e.) if M is such that for all $p \in P$ there is an $M_p \in L$ such that $M \downarrow P = M_p \downarrow P$ then $M \in L$.
- An M of the form exhibited above, is said to be implied by L.
- An MSG language is an MPA language if and only if every MSC implied by its language also belongs to its language (it is implied closed).

Implied closure



M_1 M_2

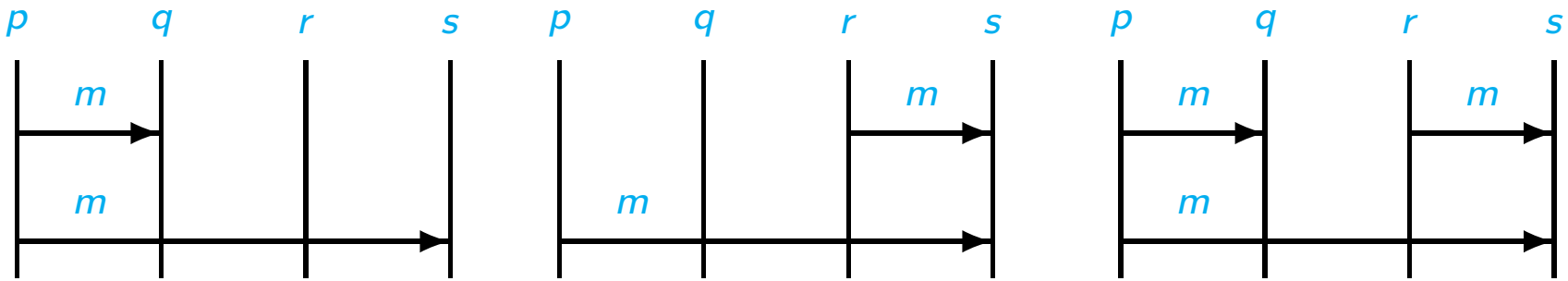
Implied closure



M_1 M_2 M

- p and q believe M is M_1
- r and s believe M is M_2

Implied closure



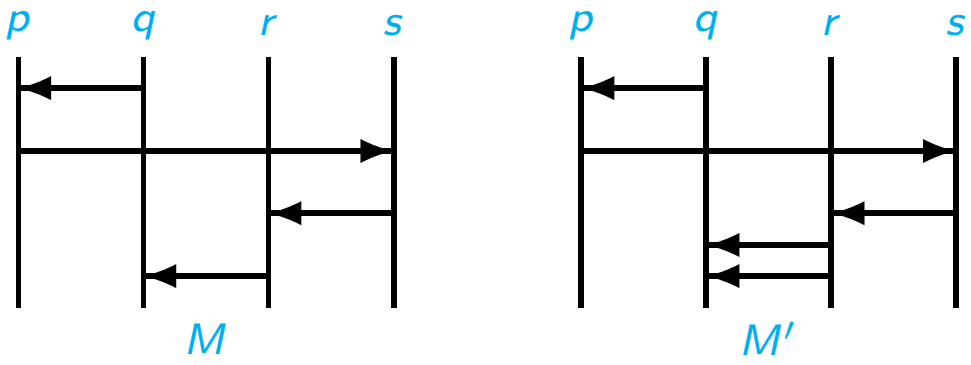
M_1 M_2 M

- p and q believe M is M_1
- r and s believe M is M_2
- M is in the implied closure of $\{M_1, M_2\}$.

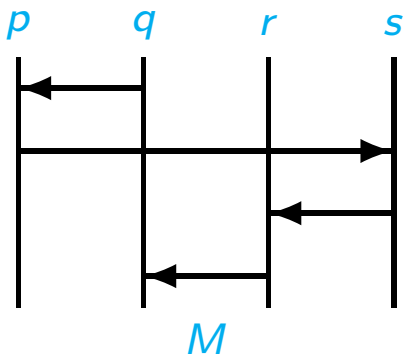
Implied closure

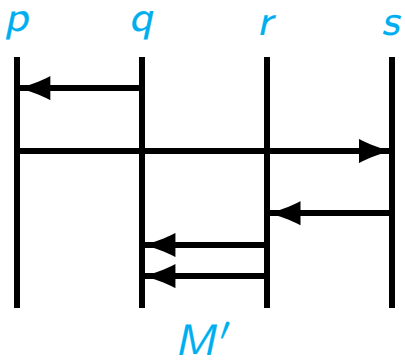
Implied closure

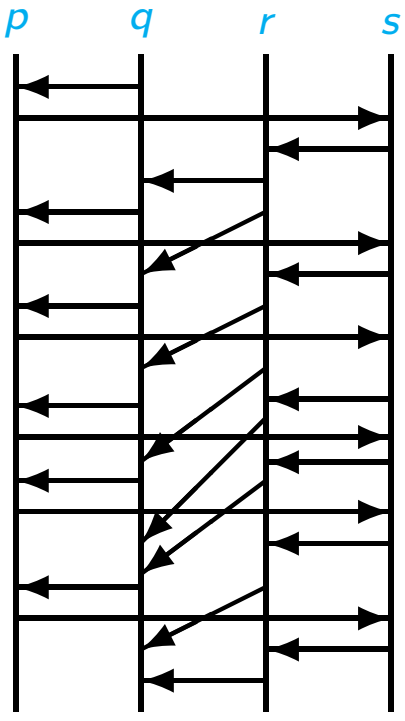
Implied closure



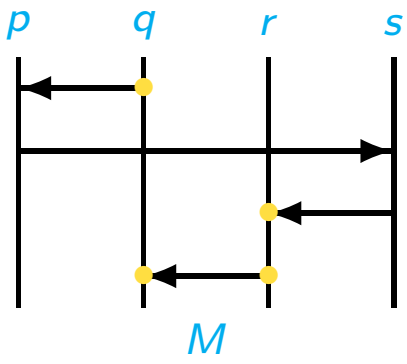
Implied closure

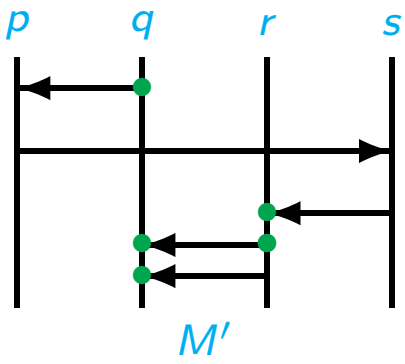


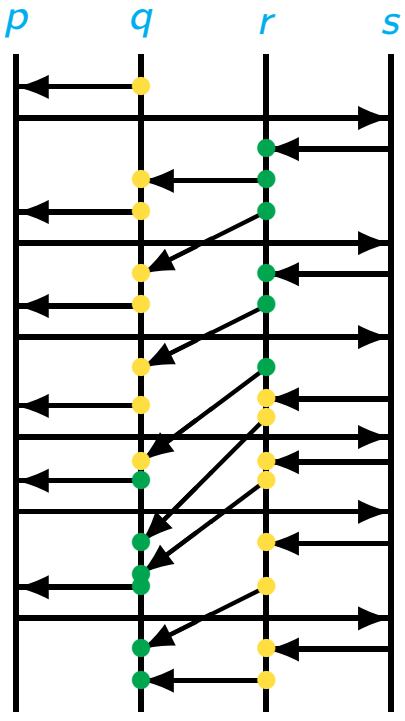




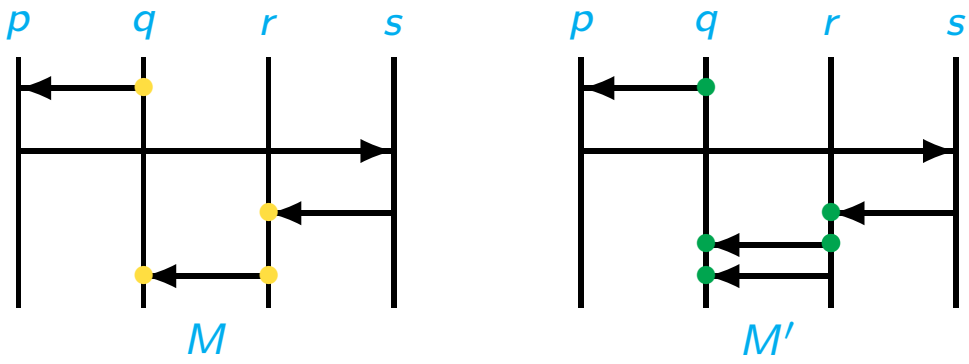
Implied closure



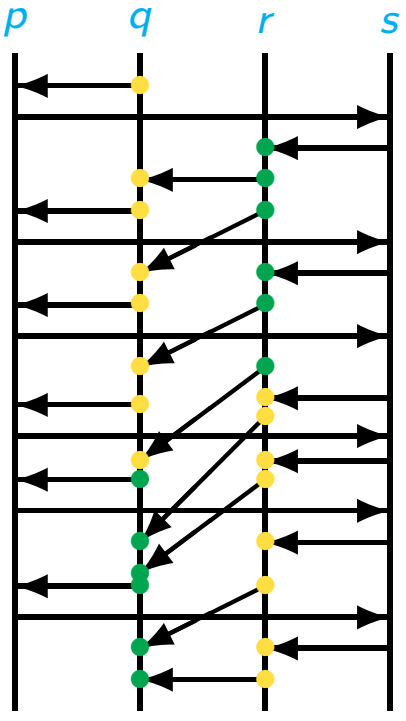




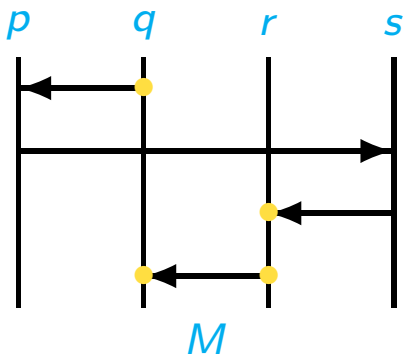
Implied closure

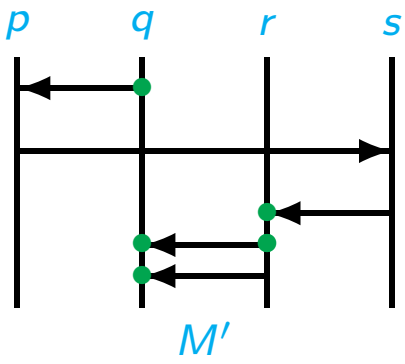


Confusing $M^{2k}M'^{k}$ and $M'^{k}M^{2k}$ generates upto k messages in $p \longrightarrow s$ channel

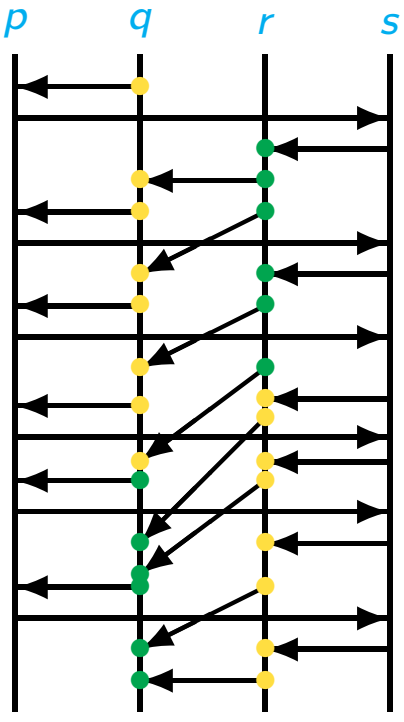


Implied closure



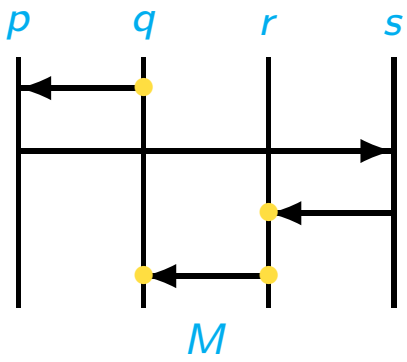


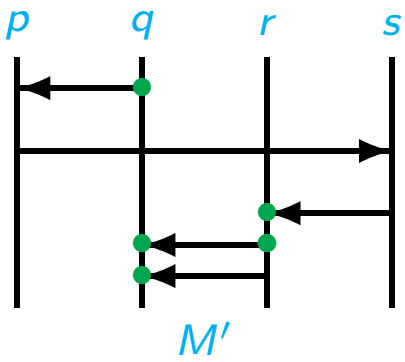
Confusing $M^{2k}M'^{k}$ and $M'^{k}M^{2k}$ generates upto k messages in $p \longrightarrow s$ channel



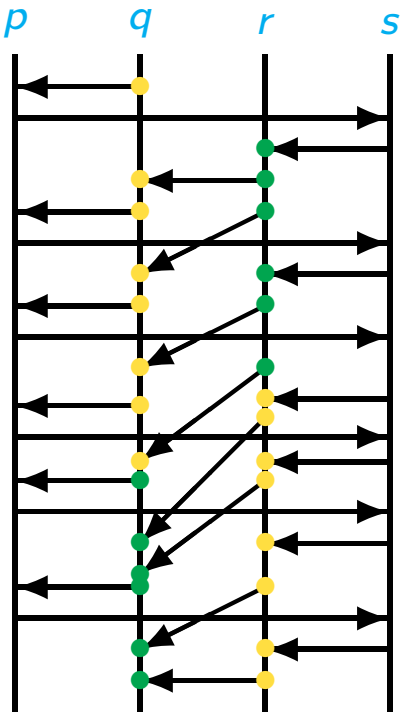
Theorem: Checking whether the language of an MSG is implied closed is undecidable. It is undecidable even for locally synchronized MSGs.

Implied closure





Confusing $M^{2k}M'^{k}$ and $M'^{k}M^{2k}$ generates upto k messages in $p \longrightarrow s$ channel



Theorem: Checking whether the language of an MSG is implied closed is undecidable. It is undecidable even for locally synchronized MSGs.

A weaker notion of implementability yields interesting positive results.

Decision problems for MPAs

- Checking emptiness for MPAs is undecidable.

Decision problems for MPAs

- Checking emptiness for MPAs is undecidable.
- Checking whether the language of an MPA is B-bounded is undecidable.

Decision problems for MPAs

- Checking emptiness for MPAs is undecidable.
- Checking whether the language of an MPA is B-bounded is undecidable.
- Checking whether the language of an MPA is regular is undecidable.

The Model-checking problem

- Positive Model-checking Given a specification language S and an implementation L decide if $S \subseteq L$.

The Model-checking problem

- Positive Model-checking Given a specification language S and an implementation L decide if $S \subseteq L$.

  Are all the positive instances exhibited?

- Negative Model-checking Given a specification language S and an implementation L decide whether $S \cap L = \emptyset$.

The Model-checking problem

- Positive Model-checking Given a specification language S and an implementation L decide if $S \subseteq L$.

  Are all the positive instances exhibited?

- Negative Model-checking Given a specification language S and an implementation L decide whether $S \cap L = \emptyset$.

  Are all the negative instances avoided?

The Model-checking problem ...

The Model-checking problem ...

- If S and L are given as locally synchronized MSGs, both the model checking problems are decidable.

The Model-checking problem ...

- If S and L are given as locally synchronized MSGs, both the model checking problems are decidable.
- If S is given by a locally synchronized MSG and L is given by any MSG, both the model checking problems are decidable.

The Model-checking problem ...

- If S and L are given as locally synchronized MSGs, both the model checking problems are decidable.
- If S is given by a locally synchronized MSG and L is given by any MSG, both the model checking problems are decidable.
  1. Replace each node in the MSG with a linearization.
  2. Let X be the regular language accepted by the resulting finite automaton.
  3. $L \subseteq S$ if and only if $X \subseteq S$ and $L \cap S = \emptyset$ if and only if $X \cap S = \emptyset$.
- These results can be generalized further ...

The Model-checking problem ...

- If S and L are given as locally synchronized MSGs, both the model checking problems are decidable.
- If S is given by a locally synchronized MSG and L is given by any MSG, both the model checking problems are decidable.
  1. Replace each node in the MSG with a linearization.
  2. Let X be the regular language accepted by the resulting finite automaton.
  3. $L \subseteq S$ if and only if $X \subseteq S$ and $L \cap S = \emptyset$ if and only if $X \cap S = \emptyset$.
- These results can be generalized further ...

Model-checking ...

Sufficient conditions for the decidability of model-checking:

Model-checking ...

Sufficient conditions for the decidability of model-checking:

- The system S has a regular set of representatives.

  A regular language R such that the set of MSCs generated by the words in R is L.

Model-checking ...

Sufficient conditions for the decidability of model-checking:

- The system S has a regular set of representatives.

  A regular language R such that the set of MSCs generated by the words in R is L.

- Given B, we can effectively construct $Lin^B(L)$ consisiting of all the B bounded linearizations of MSCs in L.

Globally Cooperative MSGs

Globally Cooperative MSGs

- An MSC is globally cooperative, if the symmetric closure of its communication graph has a single nontrivial SCC.

Globally Cooperative MSGs

- An MSC is globally cooperative, if the symmetric closure of its communication graph has a single nontrivial SCC.
- An MSG is globally cooperative, if every loop in the MSG generates a globally cooperative MSC.

Globally Cooperative MSGs

- An MSC is globally cooperative, if the symmetric closure of its communication graph has a single nontrivial SCC.
- An MSG is globally cooperative, if every loop in the MSG generates a globally cooperative MSC.

Rules out *independent iterations* without insisting on regularity.

Globally Cooperative MSGs

- An MSC is globally cooperative, if the symmetric closure of its communication graph has a single nontrivial SCC.
- An MSG is globally cooperative, if every loop in the MSG generates a globally cooperative MSC.

Rules out *independent iterations* without insisting on regularity.

Theorem: Given B, the set of B bounded linearizations of a GC-MSG is a regular language.

Globally Cooperative MSGs

- An MSC is globally cooperative, if the symmetric closure of its communication graph has a single nontrivial SCC.
- An MSG is globally cooperative, if every loop in the MSG generates a globally cooperative MSC.

Rules out *independent iterations* without insisting on regularity.

Theorem: Given B, the set of B bounded linearizations of a GC-MSG is a regular language.

[One of the many results best proved via a translation to Mazurkiewicz traces.]

Summary

- MSCs describe runs or behaviours of message passing systems.

Summary

- MSCs describe runs or behaviours of message passing systems.
- MSGs, a visual formalism to describe languages of MSCs.

Summary

- MSCs describe runs or behaviours of message passing systems.
- MSGs, a visual formalism to describe languages of MSCs.
- MSGs : regularity is not decidable, but boundedness is.

Summary

- MSCs describe runs or behaviours of message passing systems.
- MSGs, a visual formalism to describe languages of MSCs.
- MSGs : regularity is not decidable, but boundedness is.
- MPAs : An operational model, distributed, ...

Summary

- MSCs describe runs or behaviours of message passing systems.
- MSGs, a visual formalism to describe languages of MSCs.
- MSGs : regularity is not decidable, but boundedness is.
- MPAs : An operational model, distributed, ...
- Verifying implementability for MSGs is undecidable.

Summary

- MSCs describe runs or behaviours of message passing systems.
- MSGs, a visual formalism to describe languages of MSCs.
- MSGs : regularity is not decidable, but boundedness is.
- MPAs : An operational model, distributed, ...
- Verifying implementability for MSGs is undecidable.
- Model checking MSGs w.r.t. regular specifications is decidable.

Summary

- MSCs describe runs or behaviours of message passing systems.
- MSGs, a visual formalism to describe languages of MSCs.
- MSGs : regularity is not decidable, but boundedness is.
- MPAs : An operational model, distributed, ...
- Verifying implementability for MSGs is undecidable.
- Model checking MSGs w.r.t. regular specifications is decidable.
- Model checking MSGs w.r.t. GC-MSG specifications is decidable.

In the next lecture...

- A distributed synthesis theorem w.r.t. a weaker notion of implementability.
- Generalizing the decidability of model-checking beyond GC-MSGs.
- Generalizing the distributed synthesis theorem.
- Monadic Second order Logic (MSO) over MSCs and its relationship to regularity and MPAs.
- Extending MSCs with time.