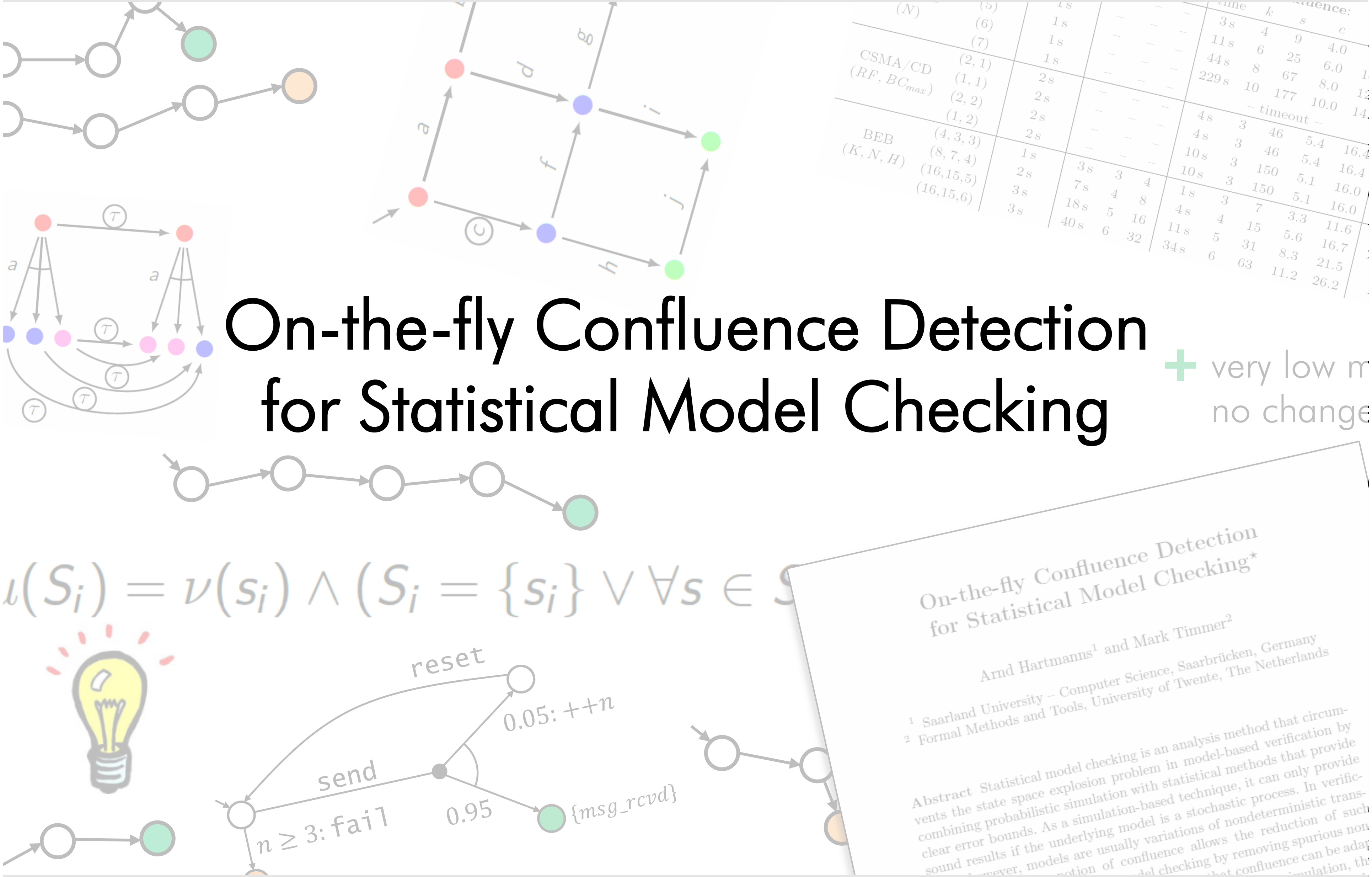
On-the-fly Confluence Detection for Statistical Model Checking



Arnd Hartmanns and Mark Timmer
Saarland University, Germany   University of Twente, The Netherlands

Model Checking

$\diamond msg\_rcvd$?

LTS Model ⟷ model checking ⟷ Requirements

model-based testing

WANTED

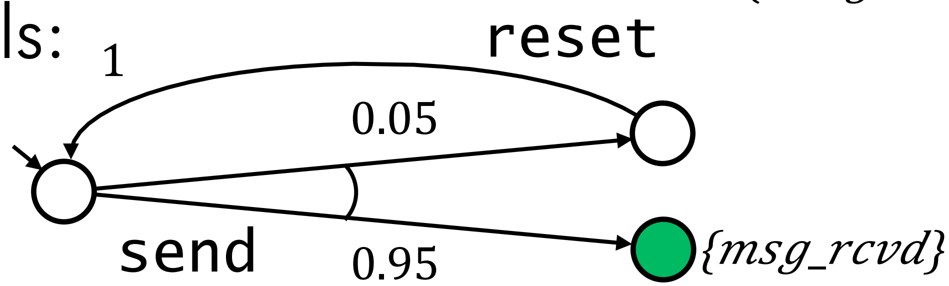
System/implementation

Problem: State-space explosion **memory consumption**

Probabilistic Model Checking

$P(\diamond msg\_rcvd) \geq 0.99$?

DTMC models:

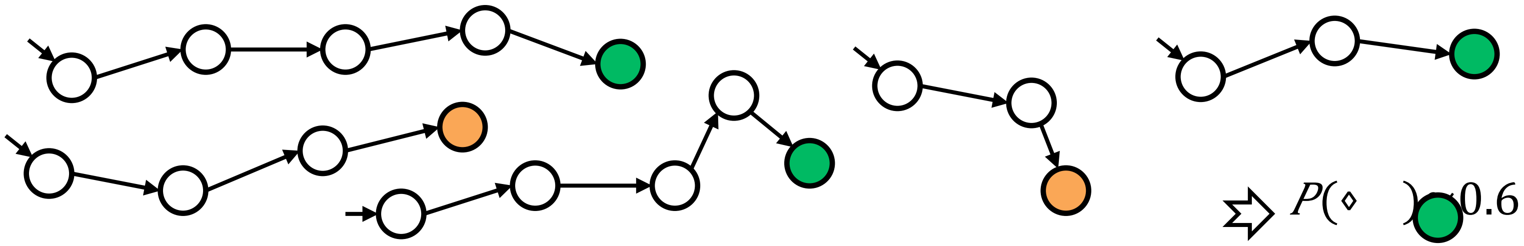


runtime stability

Problem: State-space explosion plus numerical complexity

Statistical Model Checking SMC

SMC = Simulation + Statistics



…confidence intervals, Chernoff-Hoeffding bound, SPRT…

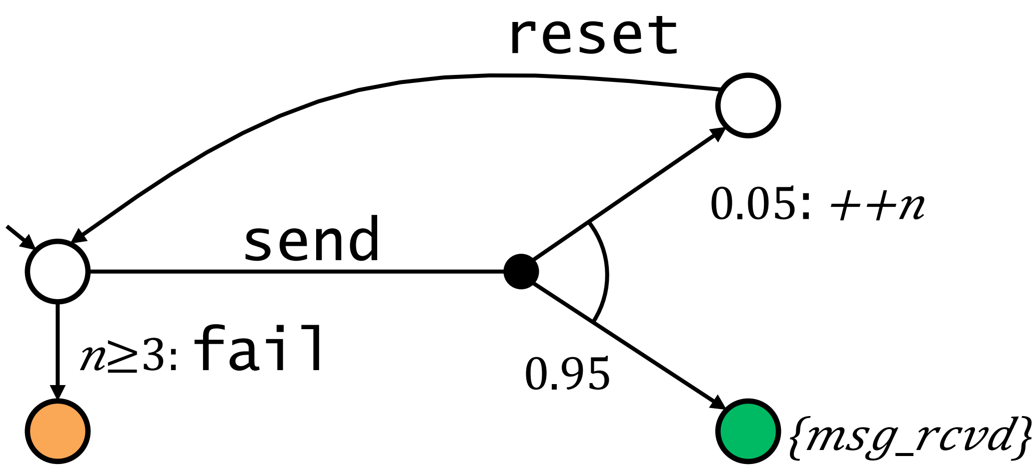
error bounds: e.g. result is ϵ-correct with probability δ

\+ constant memory usage (store only current state)
no numeric surprises (e.g. with imprecise arithmetics)

− runtime strongly dependent on desired accuracy

Statistical Model Checking **versus Nondeterminism**

MDP**/PA** models:



➪ need to resolve nondeterministic choices

$P_{min}(\Diamond msg\_rcvd) \geq 0.99$?

$P_{max}(\Diamond msg\_rcvd) \geq 1$?

Standard technique:

☹ implicit uniformly distributed resolution

➪ some value $\in [P_{min}, P_{max}]$

widely implemented: PRISM, UPPAAL, ...

Previous approaches to SMC for MDPs

Partial order reduction-based method: "POR"

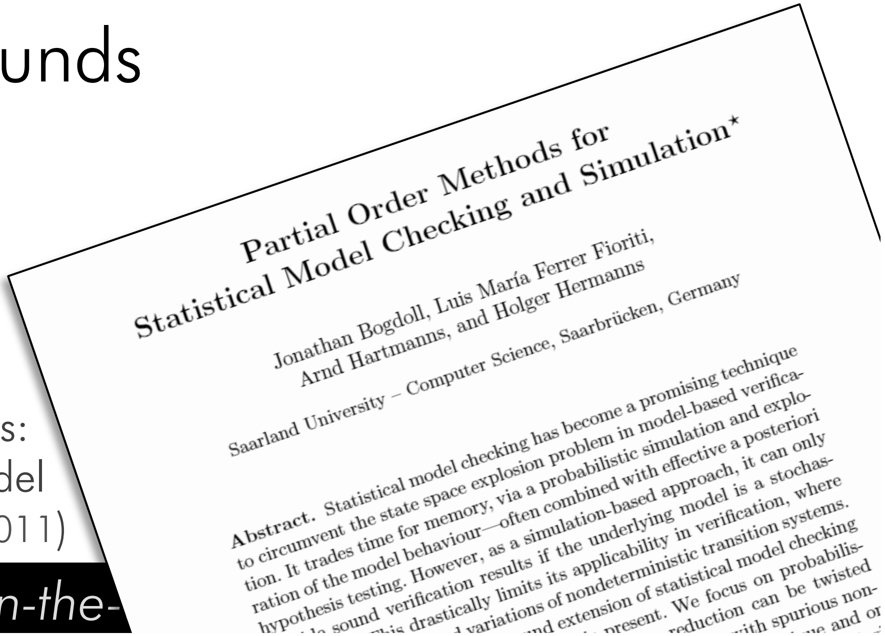
Nondeterminism may be **spurious**
= irrelevant for the results, i.e. $P_{max} = P_{min}$

⇨ check for spuriousness on-the-fly and ignore

\+ very low memory overhead
no change to SMC error bounds

– spurious interleavings only

Bogdoll, Ferrer Fioriti, H., Hermanns:
Partial Order Methods for Statistical Model
Checking and Simulation (FMOODS/FORTE 2011)



Partial Order Methods for
Statistical Model Checking and Simulation*

Jonathan Bogdoll, Luis María Ferrer Fioriti,
Arnd Hartmanns, and Holger Hermanns

Saarland University – Computer Science, Saarbrücken, Germany

Abstract. Statistical model checking has become a promising technique to circumvent the state space explosion problem in model-based verification. It trades time for memory, via a probabilistic simulation and exploration of the model behaviour—often combined with effective a posteriori hypothesis testing. However, as a simulation-based approach, it can only ... sound verification results if the underlying model is a stochas... This drastically limits its applicability in verification, where ... variations of nondeterministic transition systems. ... and extension of statistical model checking ... present. We focus on probabilis... reduction can be twisted ... with spurious non-...

Previous approaches to SMC for MDPs

Learning-based method:

technique from AI

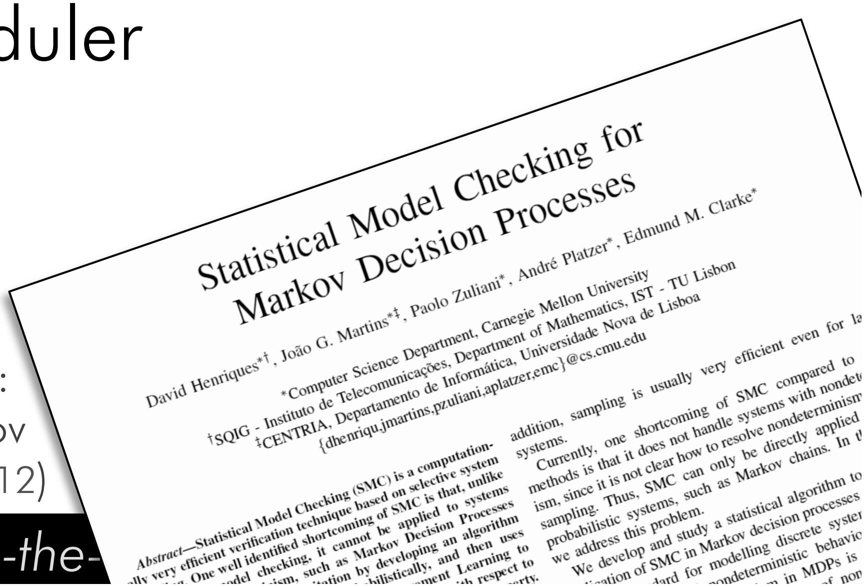
Use reinforcement learning to obtain memoryless scheduler using simulation

⇨ use that scheduler for SMC for $P{\downarrow}max\,(bounded\ LTL)$

+ works for every MDP

– memory usage to store scheduler
  no error bounds, converges
  to actual result only

Henriques, Martins, Zuliani, Platzer, Clarke:
Statistical Model Checking for Markov
Decision Processes (QEST 2012)


Statistical Model Checking for Markov Decision Processes

David Henriques*†, João G. Martins*‡, Paolo Zuliani*, André Platzer*, Edmund M. Clarke*

*Computer Science Department, Carnegie Mellon University
†SQIG - Instituto de Telecomunicações, Department of Mathematics, IST - TU Lisbon
‡CENTRIA, Departamento de Informática, Universidade Nova de Lisboa
{dhenriqu,jmartins,pzuliani,aplatzer,emc}@cs.cmu.edu

Outline

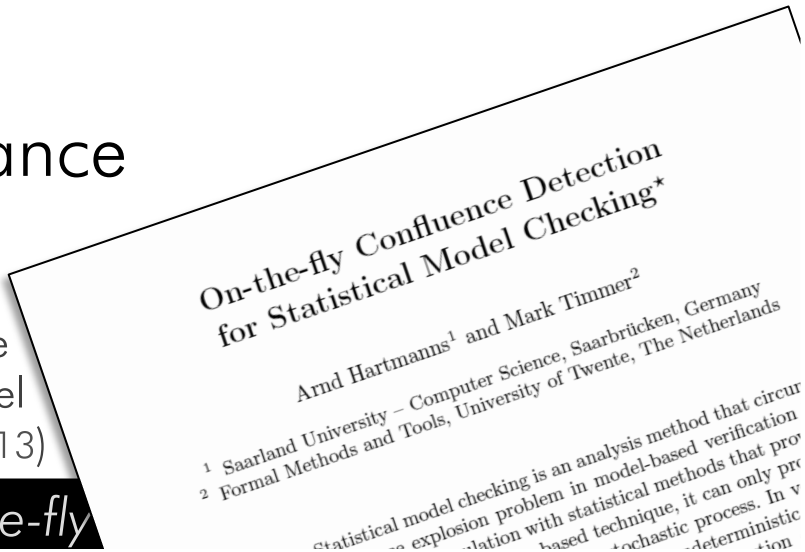
In this talk: a new method
based on **on-the-fly confluence detection** 

1 **Probabilistic Confluence** (MT)
Adaption to SMC & advantages over POR

2 **On-the-fly Detection** (MT)
A correct algorithm for use during simulation

3 **Evaluation**
Tools, applicability, performance

H., T.: On-the-fly Confluence Detection for Statistical Model Checking (to appear at NFM 2013)


On-the-fly Confluence Detection for Statistical Model Checking*

Arnd Hartmanns[1] and Mark Timmer[2]

[1] Saarland University – Computer Science, Saarbrücken, Germany
[2] Formal Methods and Tools, University of Twente, The Netherlands

Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:

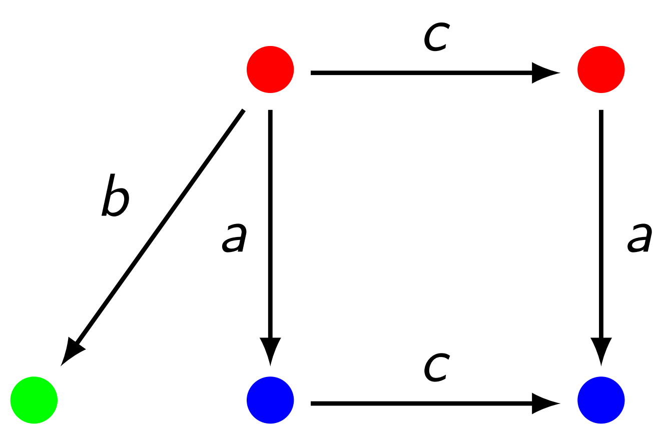
- Deterministic
- Stuttering

Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible transitions might disable behaviour...

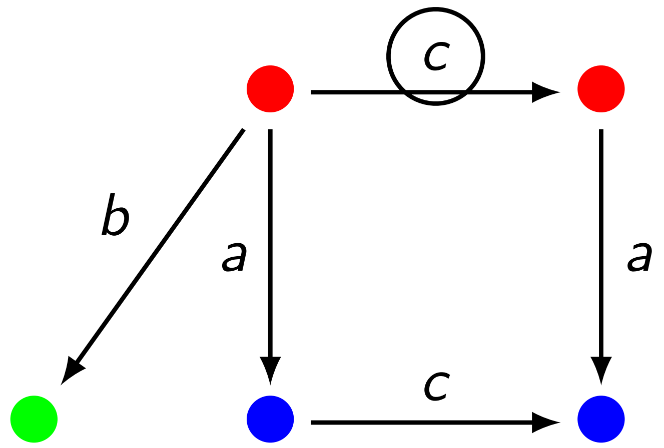


Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible transitions might disable behaviour. . .



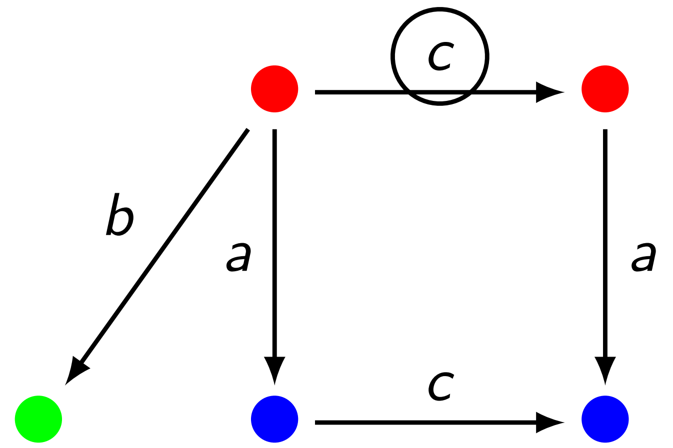
Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible transitions might disable behaviour...

...though often, they connect equivalent states



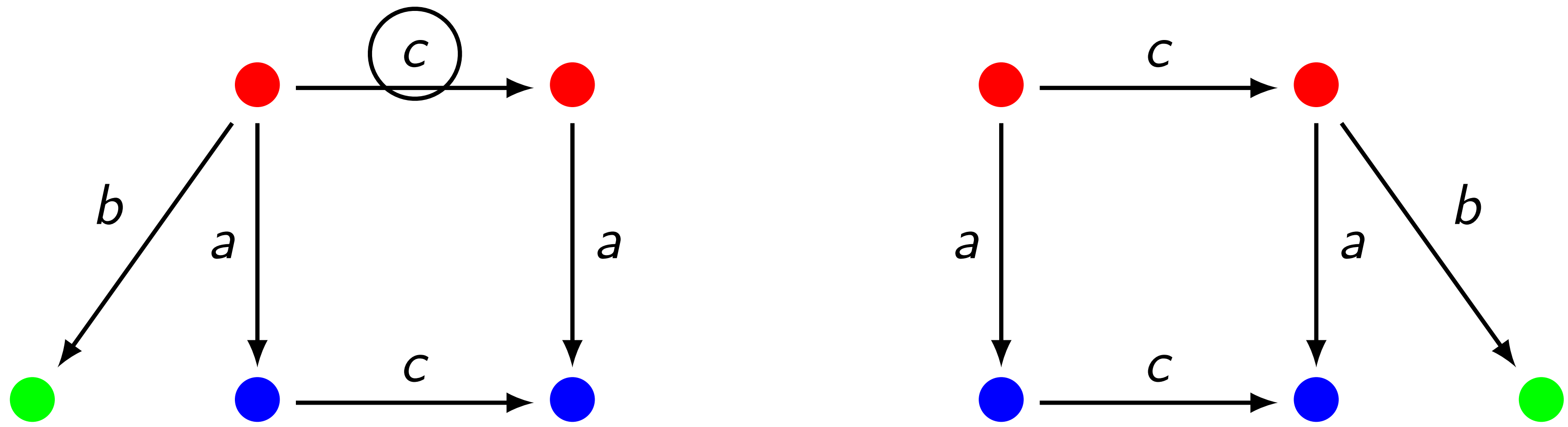
Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible transitions might disable behaviour...

...though often, they connect equivalent states



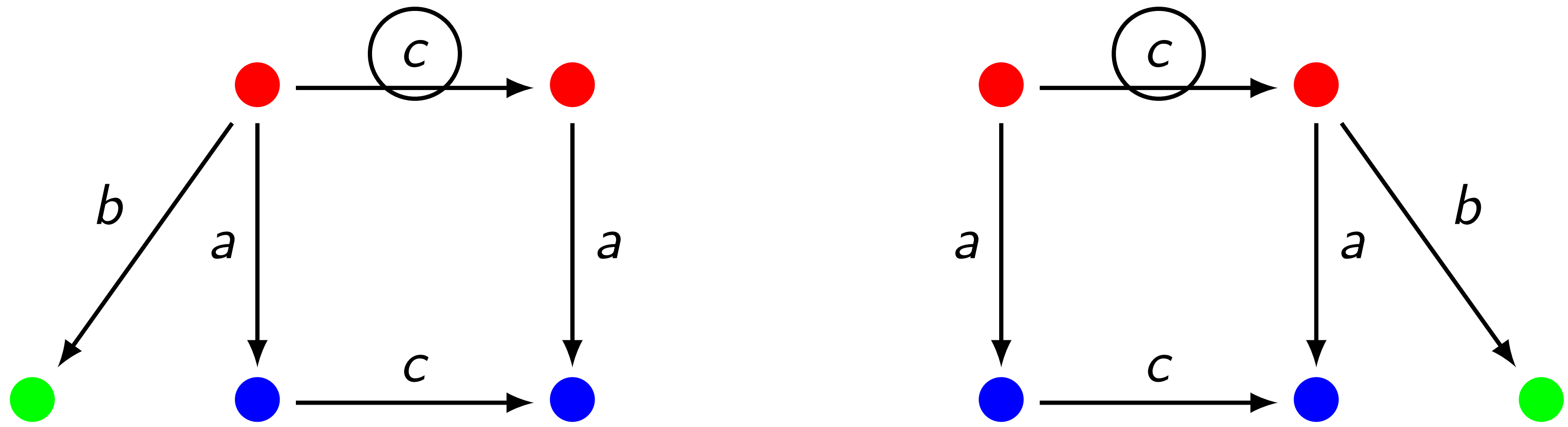
Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible transitions might disable behaviour...

...though often, they connect equivalent states



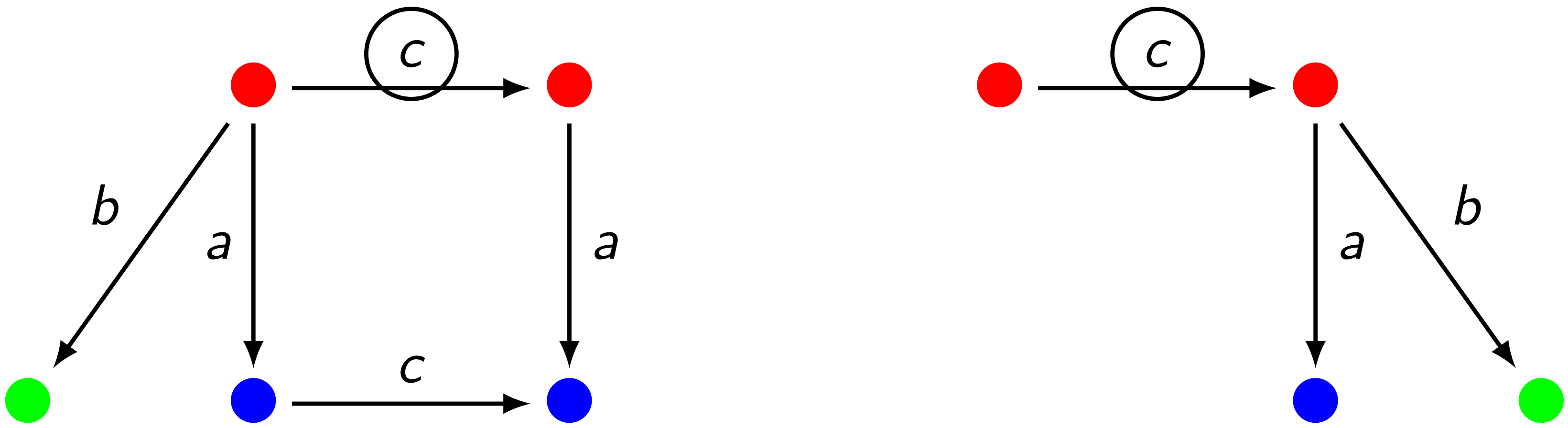
Invisible transitions connecting equivalent states

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible transitions might disable behaviour...

...though often, they connect equivalent states



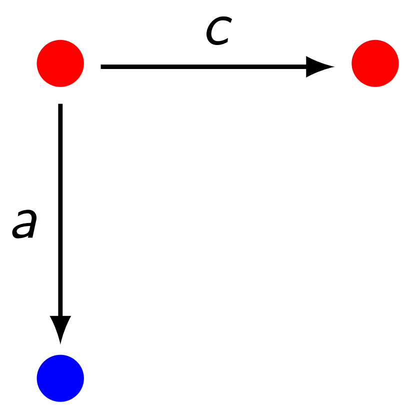
Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
denoting a subset of the invisible transitions as confluent.

Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
denoting a subset of the invisible transitions as confluent.

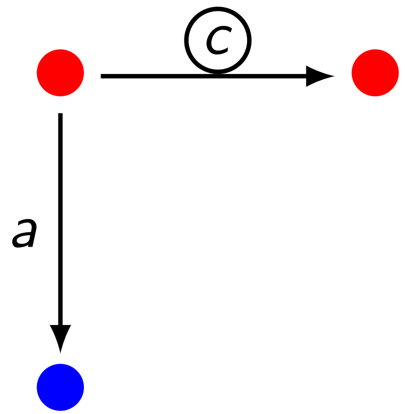
Non-probabilistically:



Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
denoting a subset of the invisible transitions as confluent.

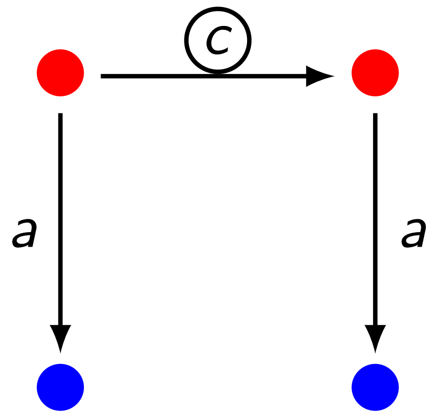
Non-probabilistically:



Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
denoting a subset of the invisible transitions as confluent.

Non-probabilistically:

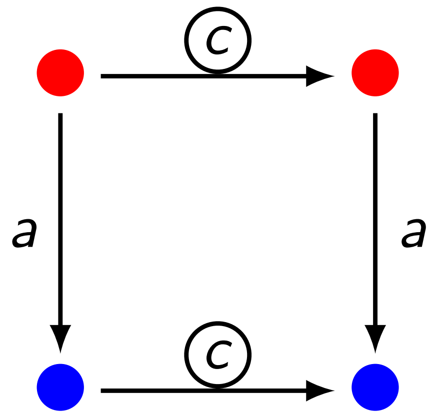


Non-probabilistic and probabilistic confluence reduction

Confluence reduction:

denoting a subset of the invisible transitions as confluent.

Non-probabilistically:

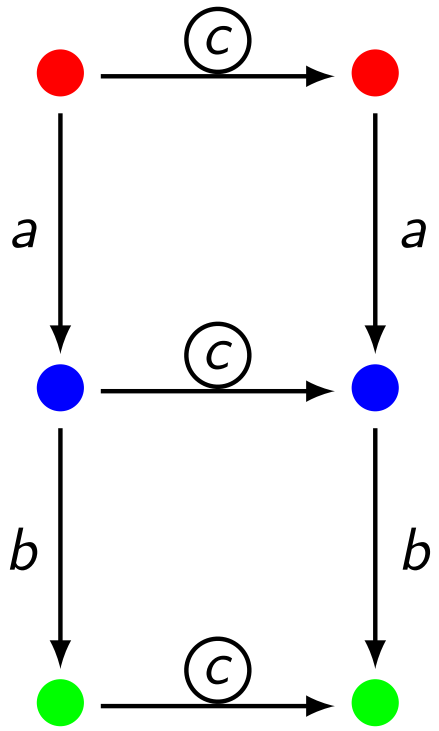


Non-probabilistic and probabilistic confluence reduction

Confluence reduction:

denoting a subset of the invisible transitions as confluent.

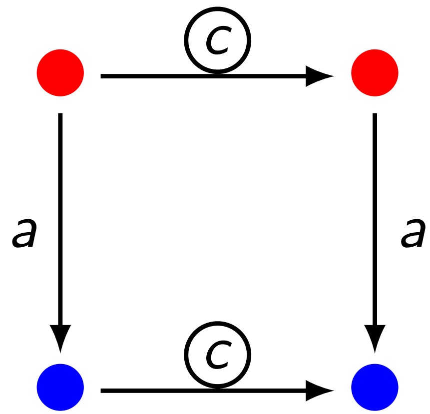
Non-probabilistically:



Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
denoting a subset of the invisible transitions as confluent.

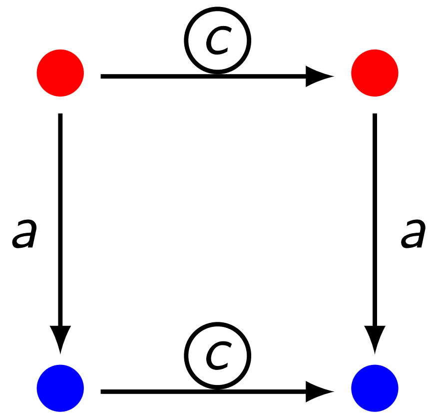
Non-probabilistically:



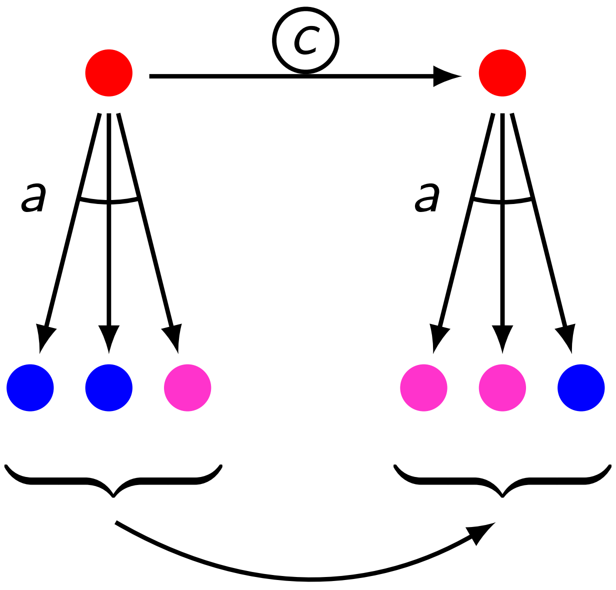
Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
denoting a subset of the invisible transitions as confluent.

Non-probabilistically:



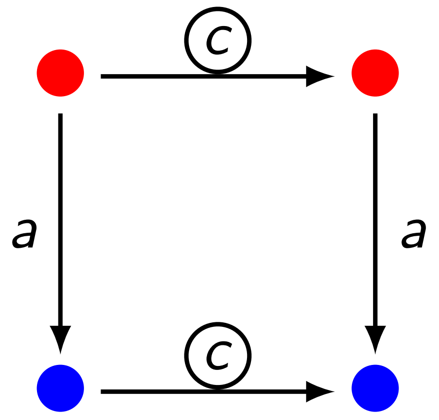
Probabilistically:



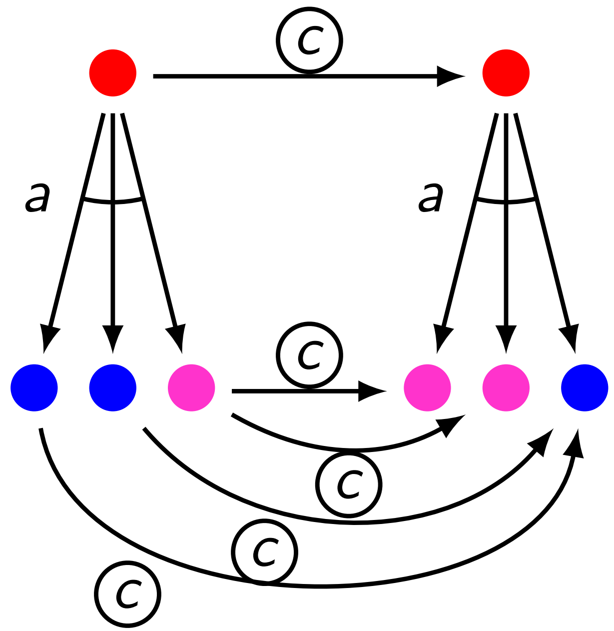
Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
denoting a subset of the invisible transitions as confluent.

Non-probabilistically:



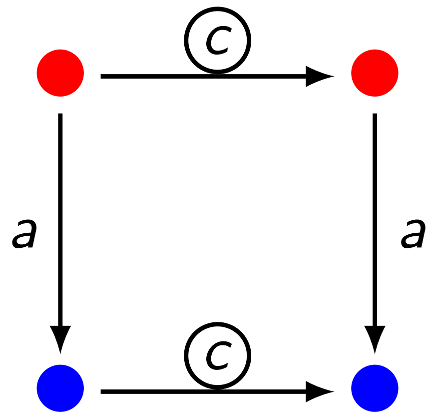
Probabilistically:



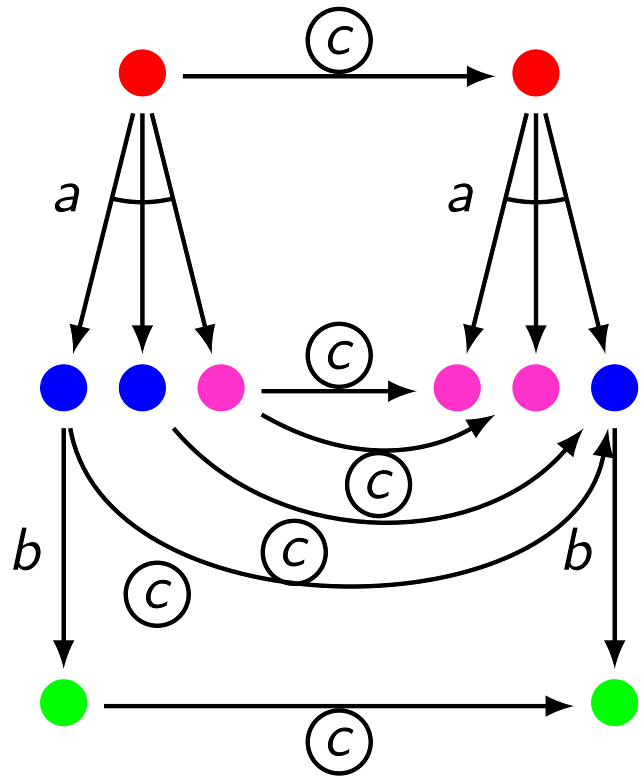
Non-probabilistic and probabilistic confluence reduction

Confluence reduction:
denoting a subset of the invisible transitions as confluent.

Non-probabilistically:



Probabilistically:



Confluence versus Partial Order Reduction

Partial Order Reduction:

- Preservation of probabilistic LTL$\backslash_X$
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic

Confluence versus Partial Order Reduction

Partial Order Reduction:

- Preservation of probabilistic LTL$\backslash_X$ (or PCTL$^*\backslash_X$)
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic (or not)

Confluence versus Partial Order Reduction

Partial Order Reduction:

- Preservation of probabilistic LTL$\backslash_X$ (or PCTL$^*\backslash_X$)
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic (or not)

Disadvantage: not defined for concrete state spaces

Confluence versus Partial Order Reduction

Partial Order Reduction:

- Preservation of probabilistic $\mathrm{LTL}\backslash_X$ (or $\mathrm{PCTL}^*\backslash_X$)
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic (or not)

Disadvantage: not defined for concrete state spaces

Confluence Reduction:

- Preservation of $\mathrm{PCTL}^*\backslash_X$
- Based on confluent transitions (commuting diagrams)

Confluence versus Partial Order Reduction

Partial Order Reduction:

- Preservation of probabilistic LTL$\backslash x$ (or PCTL$^*\backslash x$)
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic (or not)

Disadvantage: not defined for concrete state spaces

Confluence Reduction:

- Preservation of PCTL$^*\backslash x$
- Based on confluent transitions (commuting diagrams)

Disadvantage: confluent transitions not allowed to be probabilistic

Confluence versus Partial Order Reduction

Partial Order Reduction:

- Preservation of probabilistic LTL$\backslash X$ (or PCTL$^*\backslash X$)
- Based on independent actions and ample sets
- Ample actions are allowed to be probabilistic (or not)

Disadvantage: not defined for concrete state spaces

Confluence Reduction:

- Preservation of PCTL$^*\backslash X$
- Based on confluent transitions (commuting diagrams)

Disadvantage: confluent transitions not allowed to be probabilistic

Relating the notions:

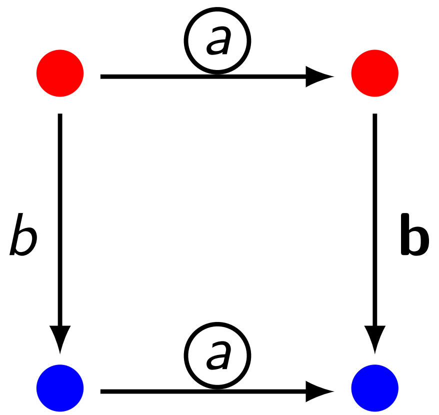
- Confluence reduction subsumes branching-time POR
- Confluence reduction and linear-time POR are incomparable

Alterations to the concept of confluence

- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- More liberal notion of equivalence of distributions

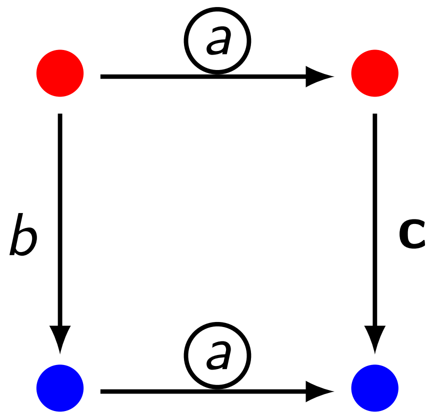
Alterations to the concept of confluence

- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- More liberal notion of equivalence of distributions



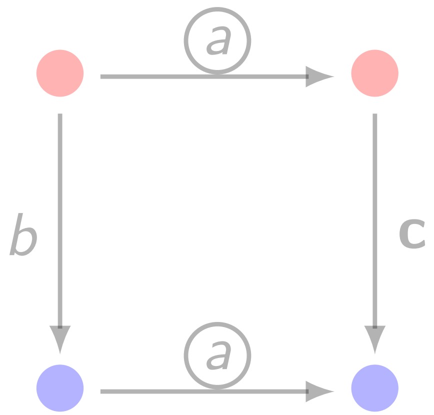
Alterations to the concept of confluence

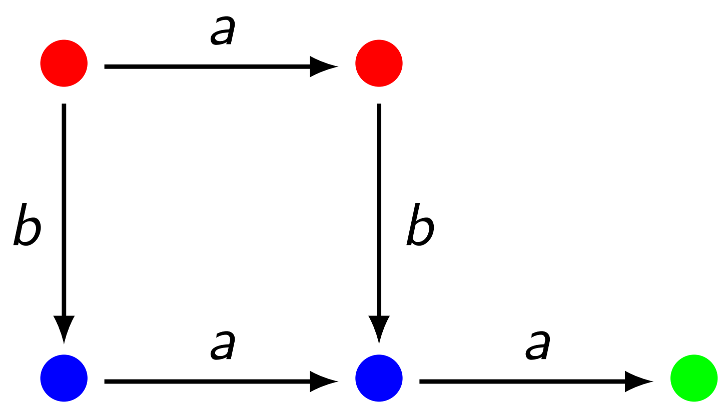
- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- More liberal notion of equivalence of distributions



Alterations to the concept of confluence

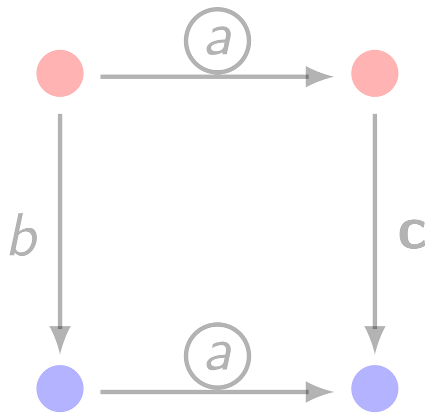
- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- More liberal notion of equivalence of distributions

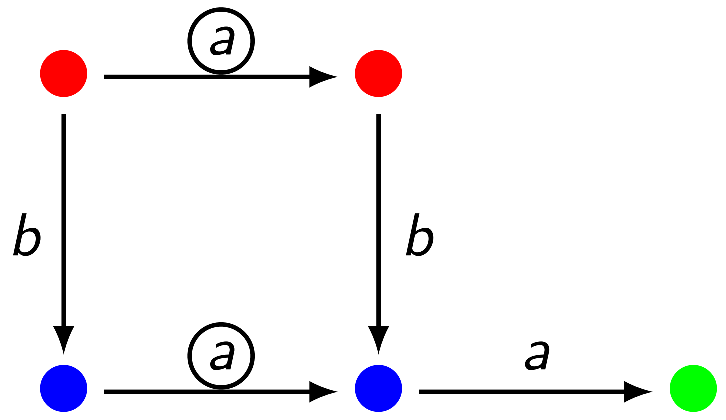




Alterations to the concept of confluence

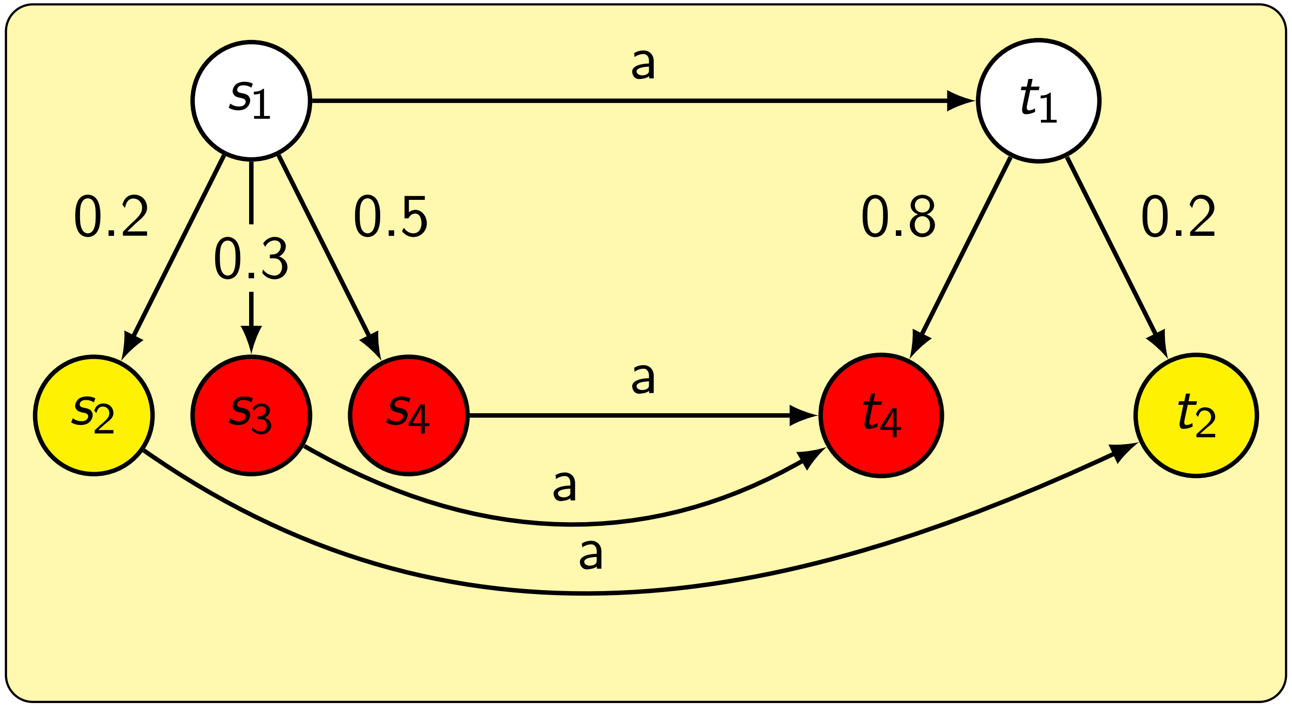
- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- More liberal notion of equivalence of distributions





Alterations to the concept of confluence

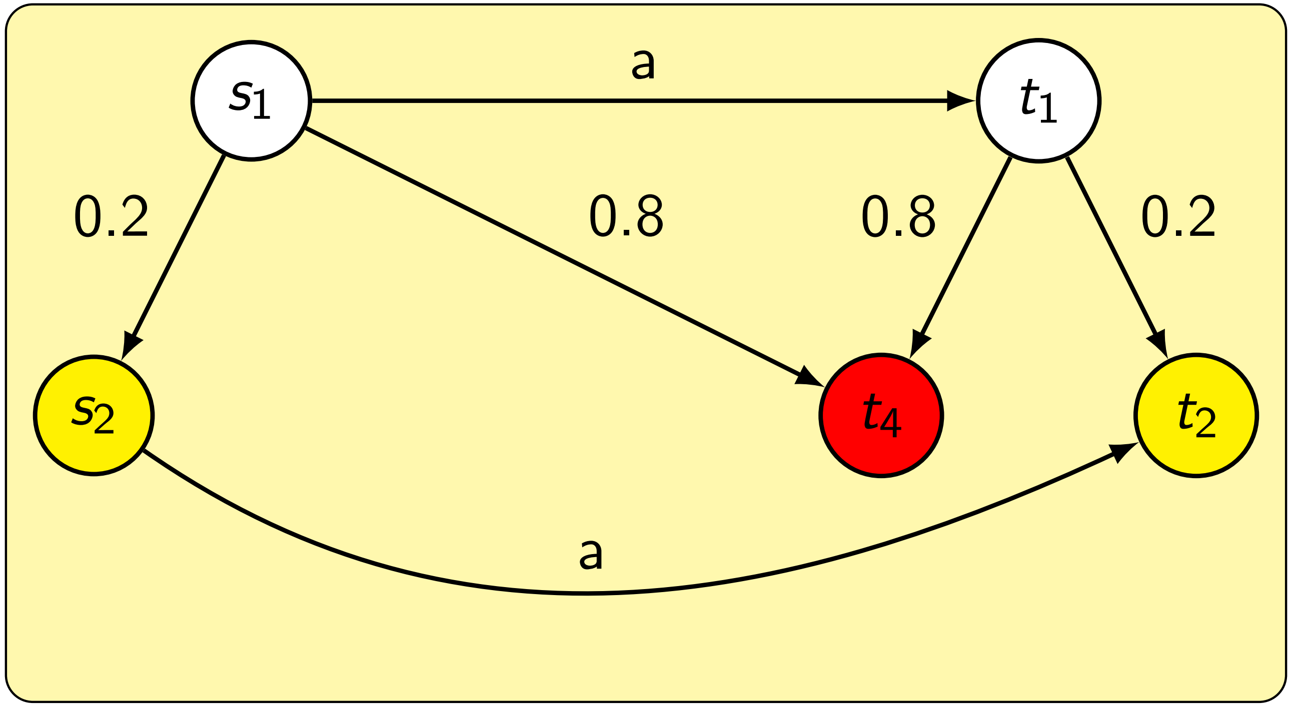
- More liberal notion of equivalence of distributions



$$\mu(S_i) = \nu(s_i) \wedge (S_i = \{s_i\} \vee \forall s \in S_i \,.\, \exists a \in \Sigma \,.\, s \xrightarrow{a} s_i \in \mathcal{T})$$

Alterations to the concept of confluence

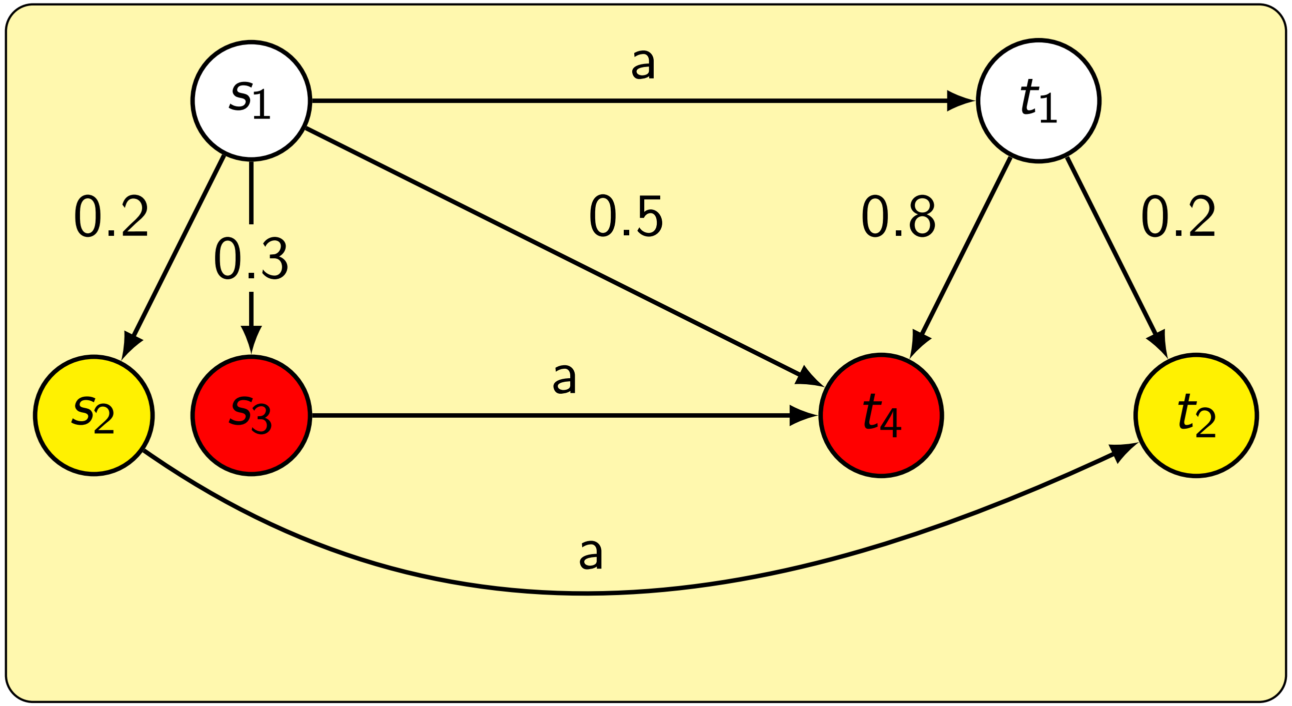
- More liberal notion of equivalence of distributions



$$\mu(S_i) = \nu(s_i) \wedge (S_i = \{s_i\} \vee \forall s \in S_i \,.\, \exists a \in \Sigma \,.\, s \xrightarrow{a} s_i \in \mathcal{T})$$

Alterations to the concept of confluence

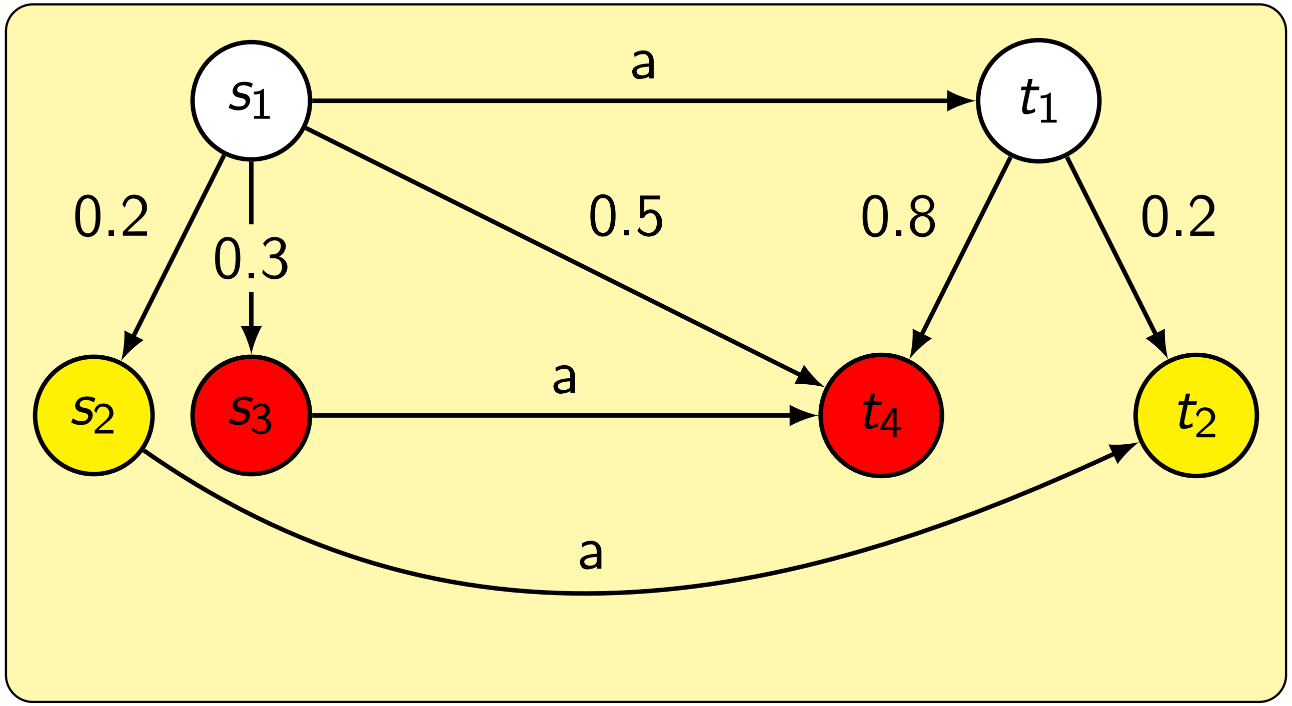
- More liberal notion of equivalence of distributions



$$\mu(S_i) = \nu(s_i) \wedge (S_i = \{s_i\} \vee \forall s \in S_i \,.\, \exists a \in \Sigma \,.\, s \xrightarrow{a} s_i \in \mathcal{T})$$

Alterations to the concept of confluence

- More liberal notion of equivalence of distributions



$$\cancel{\mu(S_i) = \nu(s_i) \wedge (S_i = \{s_i\} \vee \forall s \in S_i \,.\, \exists a \in \Sigma \,.\, s \xrightarrow{a} s_i \in \mathcal{T})}$$

$$\boxed{\{(s,t) \mid s \in \mathrm{spt}(\mu), t \in \mathrm{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}}$$

Correctness of confluence reduction

Even though

- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- We have a more liberal notion of equivalence of distributions

Correctness of confluence reduction

Even though

- Transitions may be mimicked by differently-labelled transitions
- Transitions only have to be invisible locally
- We have a more liberal notion of equivalence of distributions

Still we find:

Theorem

Confluent transitions can be given priority, preserving $PCTL^*_X$.

On-the-fly detection of confluence

Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice

On-the-fly detection of confluence

Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice
2. Check for each outgoing transition if it is confluent
   - If one choice is confluent, take it and continue
   - If no choice is confluent, abort

On-the-fly detection of confluence

Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice
2. Check for each outgoing transition if it is confluent
    - If one choice is confluent, take it and continue
    - If no choice is confluent, abort

(Possibly, if confluence fails, attempt the same using POR)

On-the-fly detection of confluence

Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice
2. Check for each outgoing transition if it is confluent
   - If one choice is confluent, take it and continue
   - If no choice is confluent, abort

(Possibly, if confluence fails, attempt the same using POR)

To check if a transition is confluent:

- Check if it is invisible

On-the-fly detection of confluence

Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice
2. Check for each outgoing transition if it is confluent
    - If one choice is confluent, take it and continue
    - If no choice is confluent, abort

(Possibly, if confluence fails, attempt the same using POR)

To check if a transition is confluent:

- Check if it is invisible
- Check if all its neighbouring transitions are mimicked
    - For this, additional transitions might need to be confluent

On-the-fly detection of confluence

Simulation using on-the-fly confluence detection:

1. Simulate until reaching a nondeterministic choice
2. Check for each outgoing transition if it is confluent
   - If one choice is confluent, take it and continue
   - If no choice is confluent, abort

(Possibly, if confluence fails, attempt the same using POR)

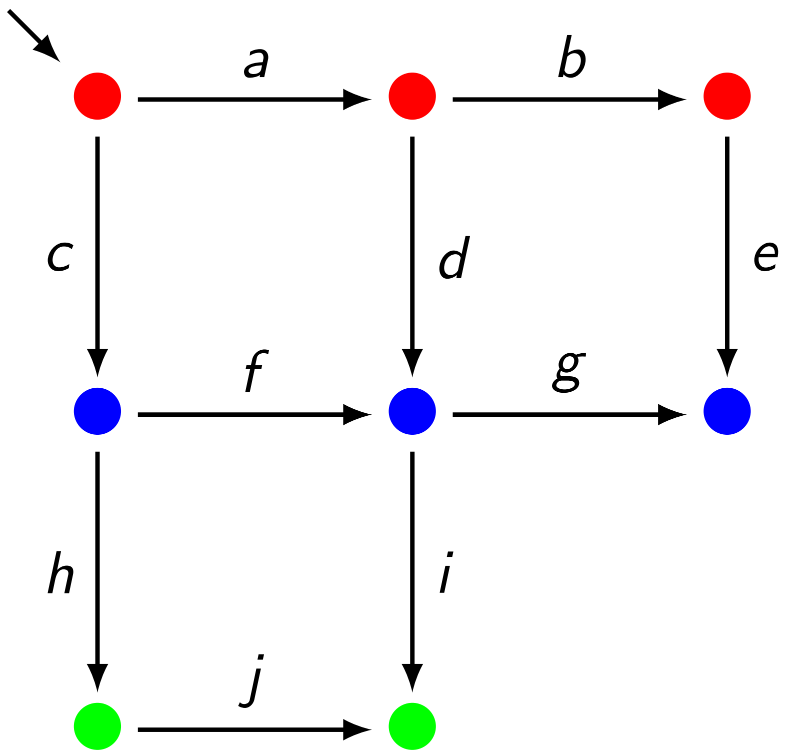
To check if a transition is confluent:

- Check if it is invisible
- Check if all its neighbouring transitions are mimicked
  - For this, additional transitions might need to be confluent

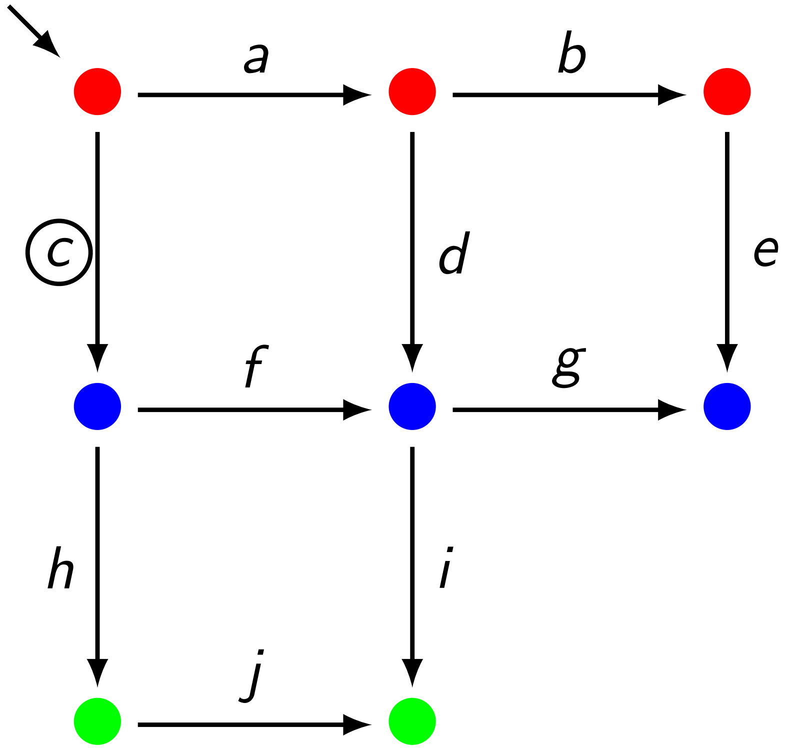
Careful: do not ignore visible behaviour forever (ignoring problem)

- Check if at least once every l steps a state is fully expanded

Checking a transition for confluence

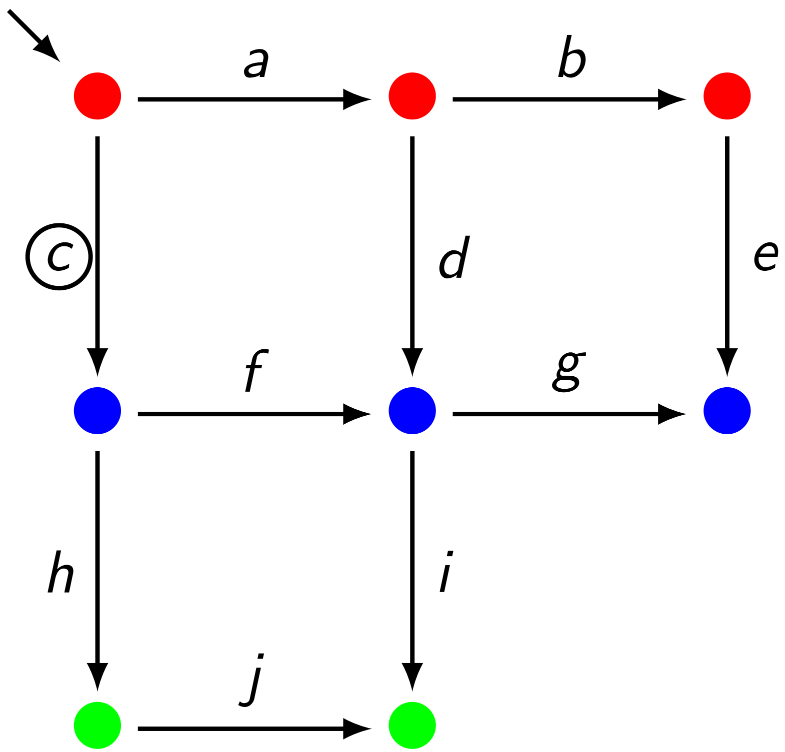


Checking a transition for confluence



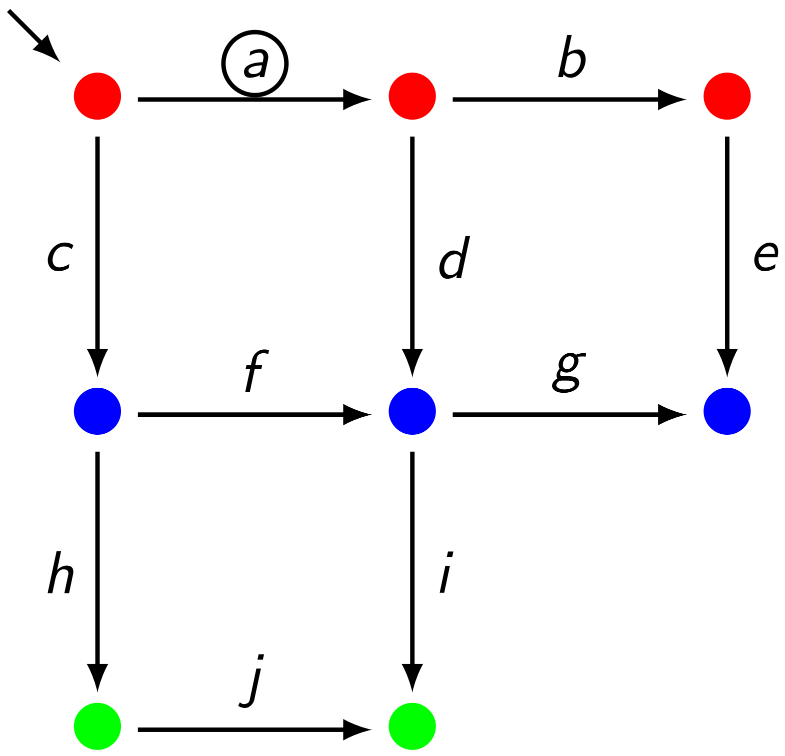
- Check if c is confluent

Checking a transition for confluence



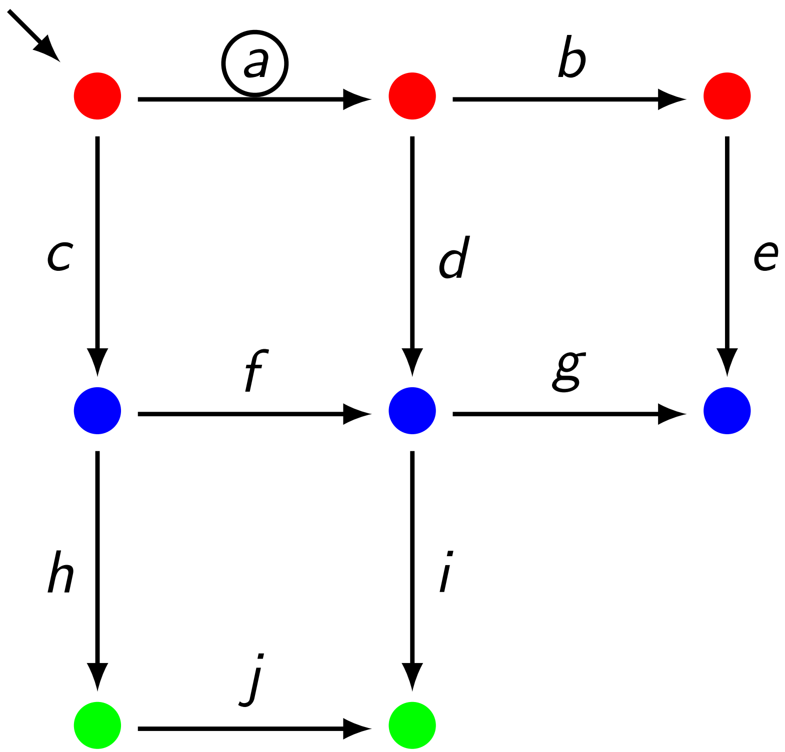
- Check if c is confluent
  - No; it is not invisible

Checking a transition for confluence



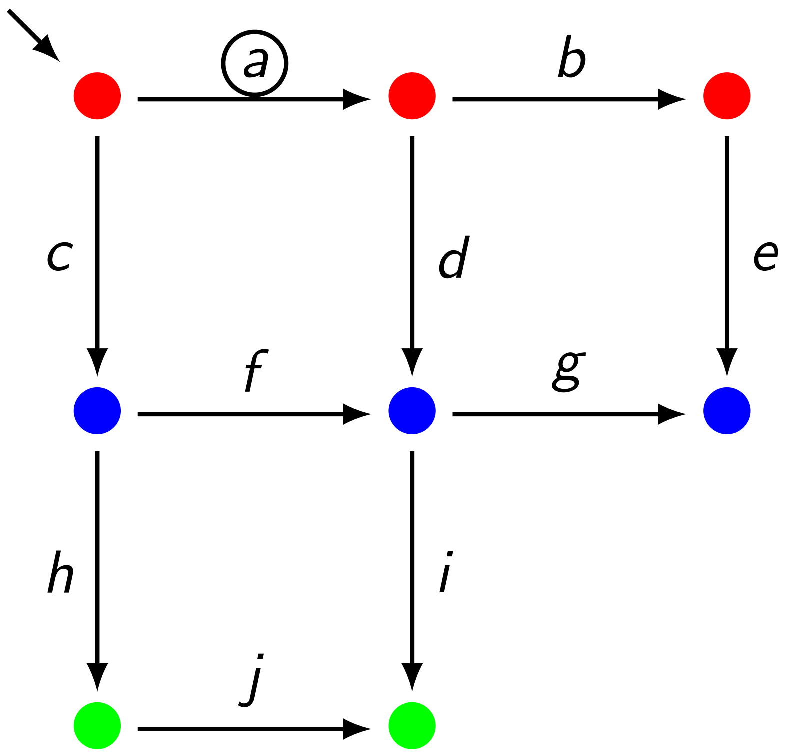
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent

Checking a transition for confluence



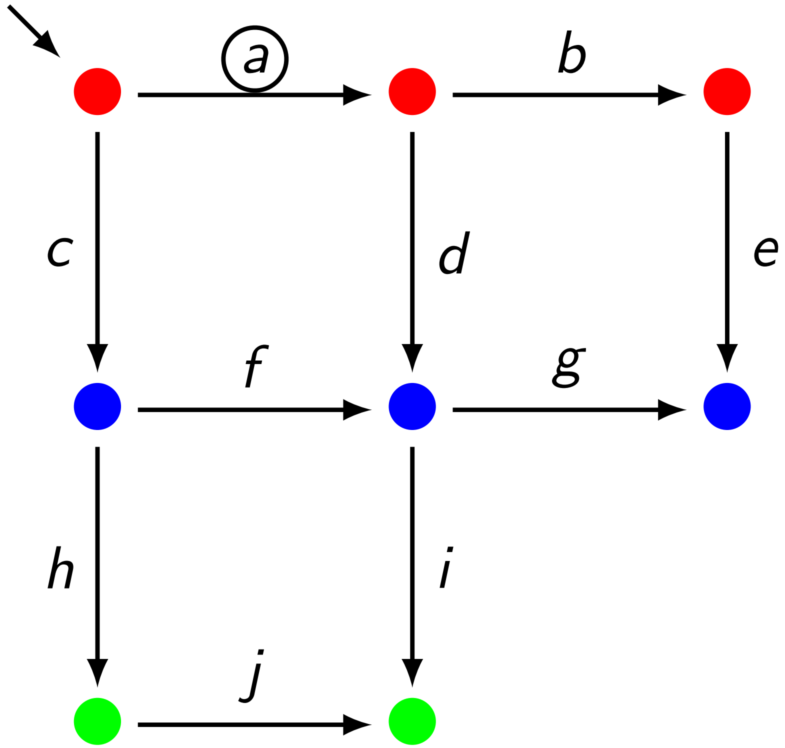
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible

Checking a transition for confluence



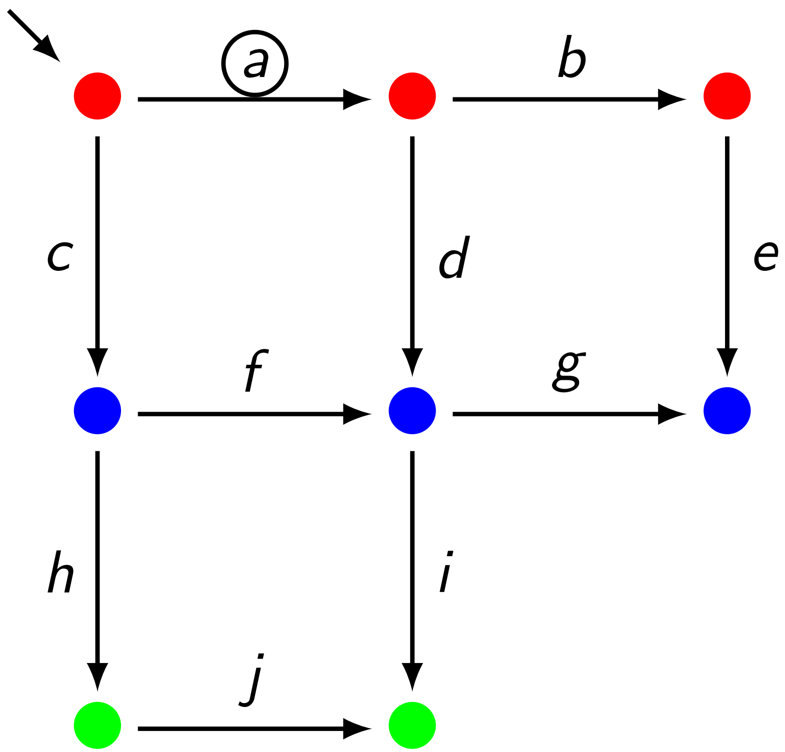
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?

Checking a transition for confluence



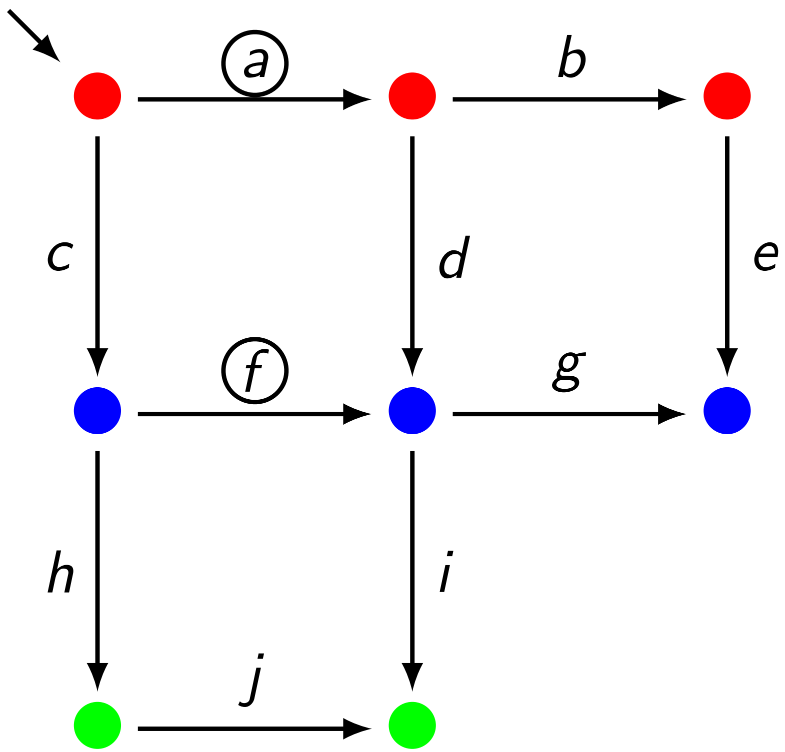
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?
    - Possibly by the d-transition

Checking a transition for confluence



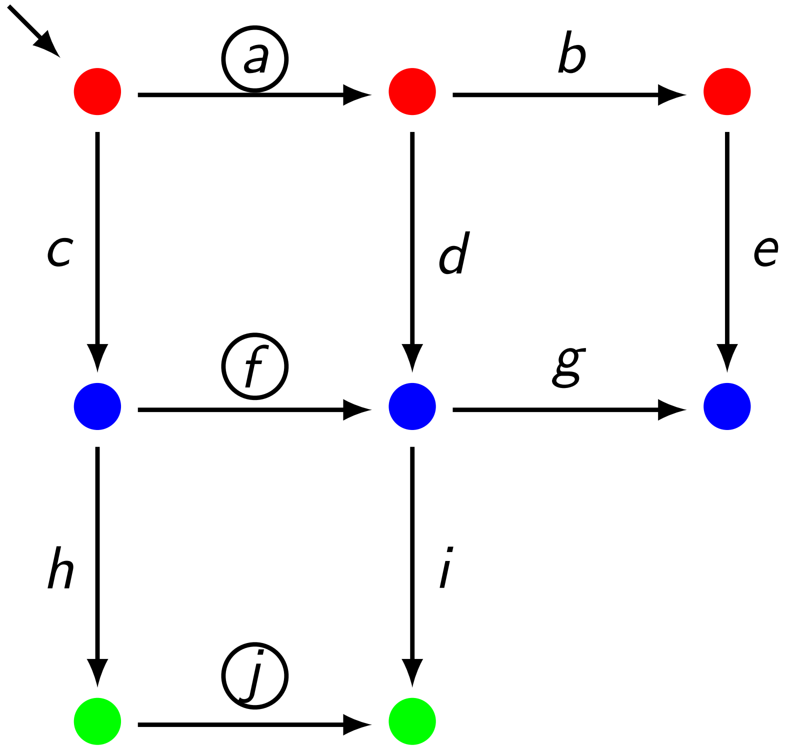
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?
    - Possibly by the d-transition
    - But then f has to be confluent: check this
    - . . .

Checking a transition for confluence



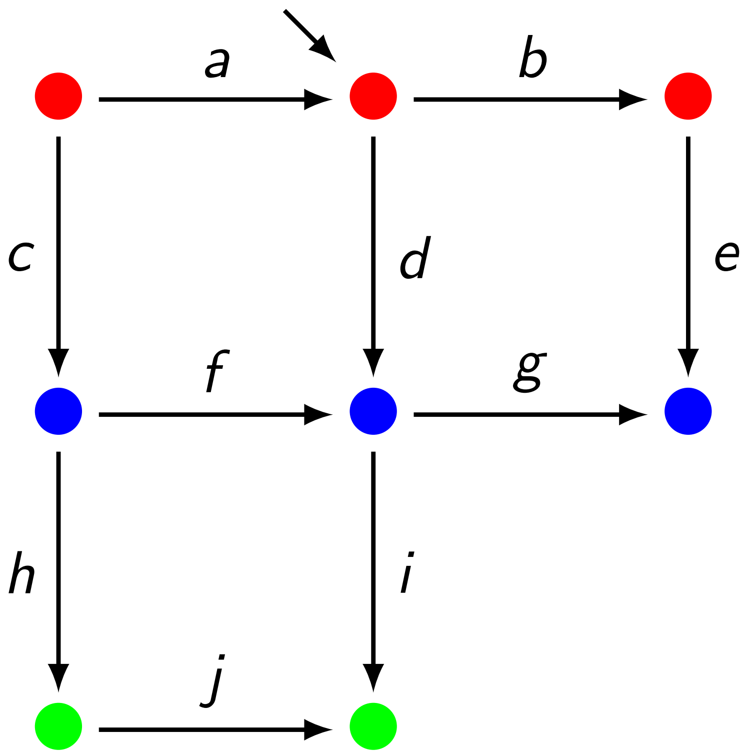
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?
    - Possibly by the d-transition
    - But then f has to be confluent: check this
    - ...

Checking a transition for confluence



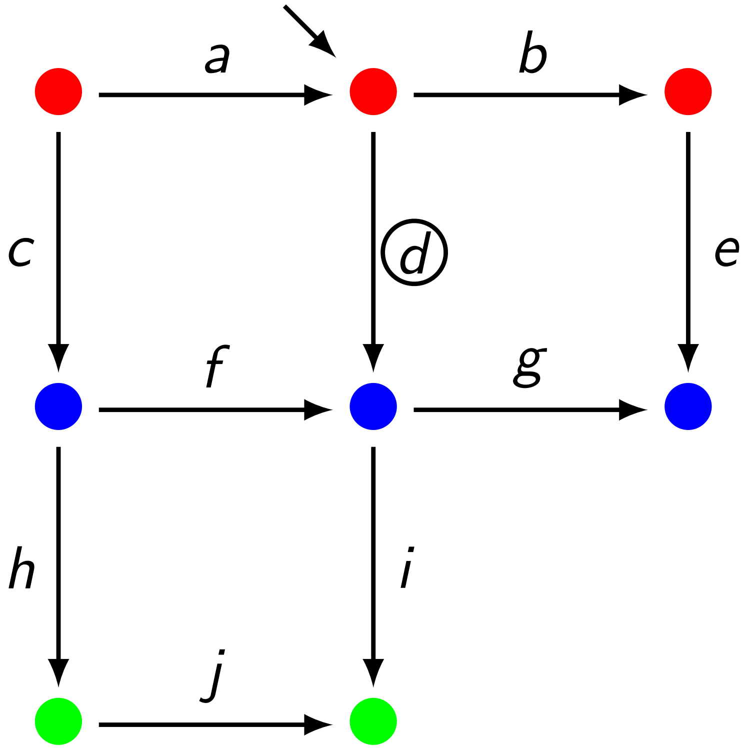
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?
    - Possibly by the d-transition
    - But then f has to be confluent: check this
    - ...

Checking a transition for confluence



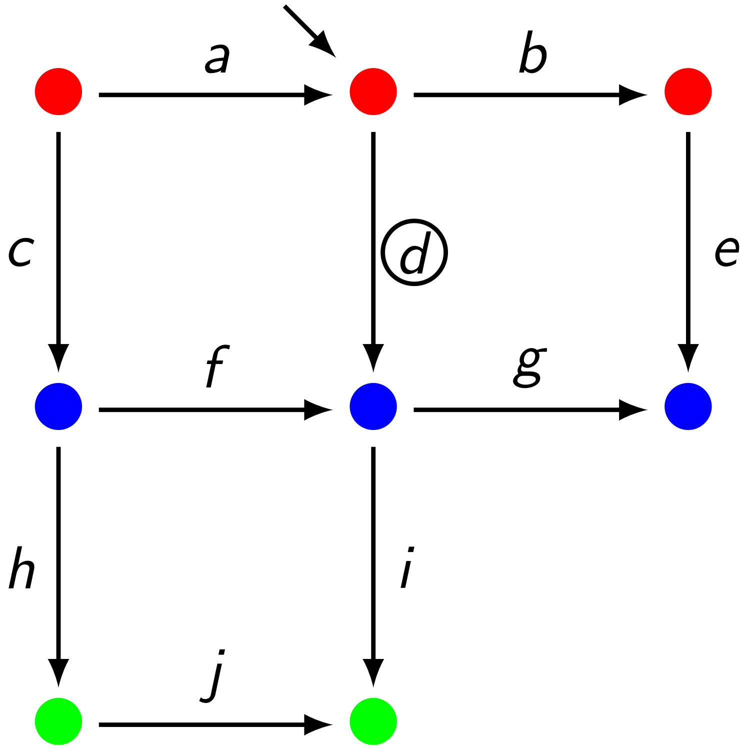
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?
    - Possibly by the d-transition
    - But then f has to be confluent: check this
    - ...

Checking a transition for confluence



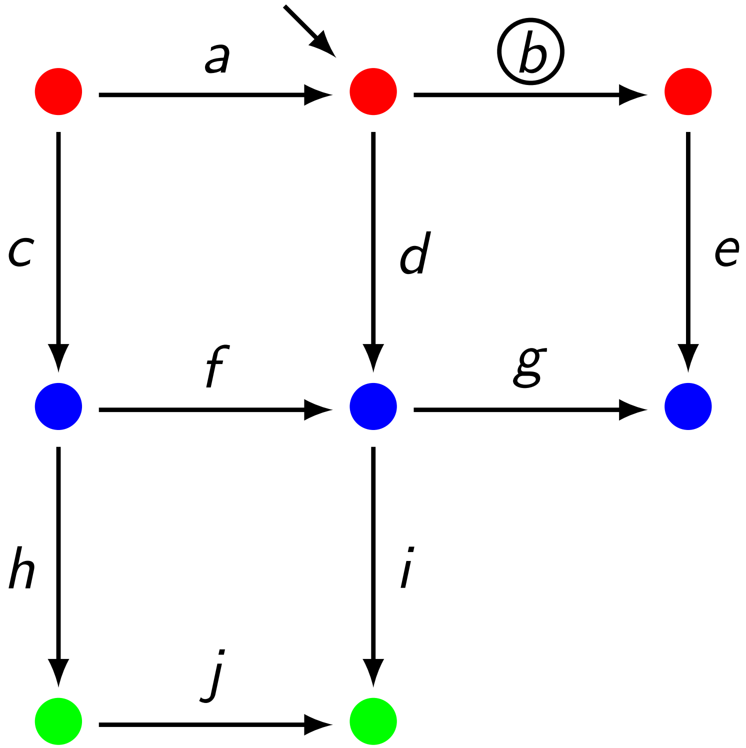
- Check if d is confluent

Checking a transition for confluence



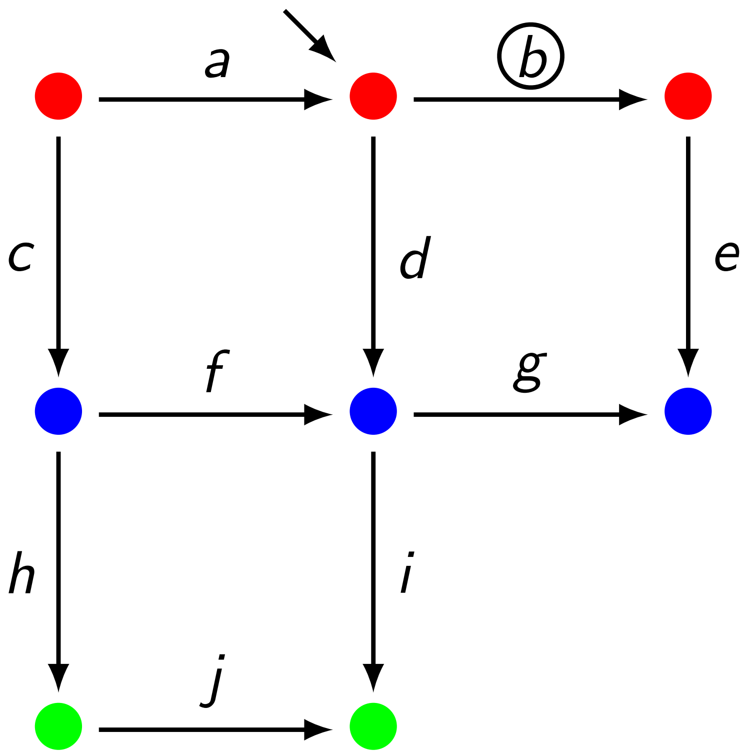
- Check if d is confluent
  - No; it is not invisible

Checking a transition for confluence



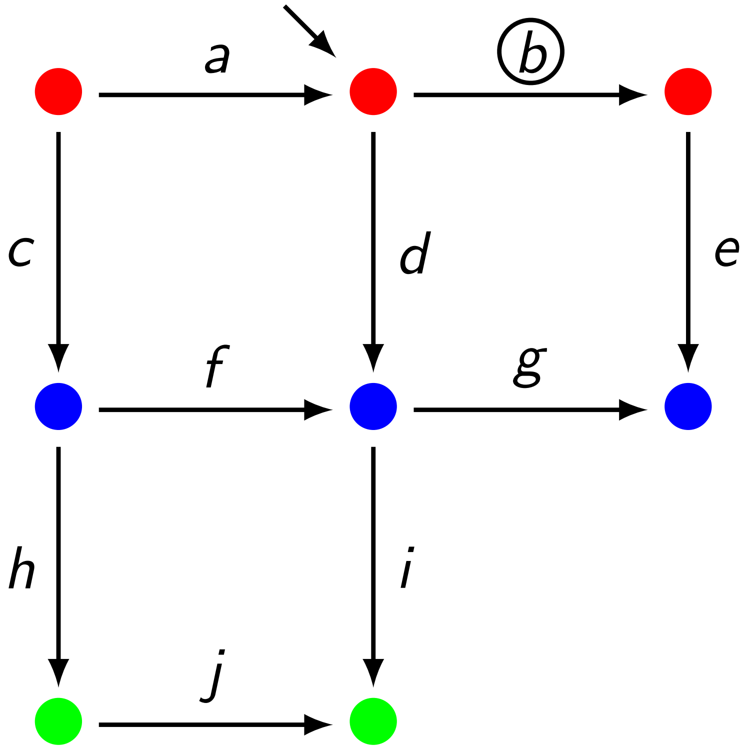
- Check if d is confluent
  - No; it is not invisible
- Check if b is confluent

Checking a transition for confluence



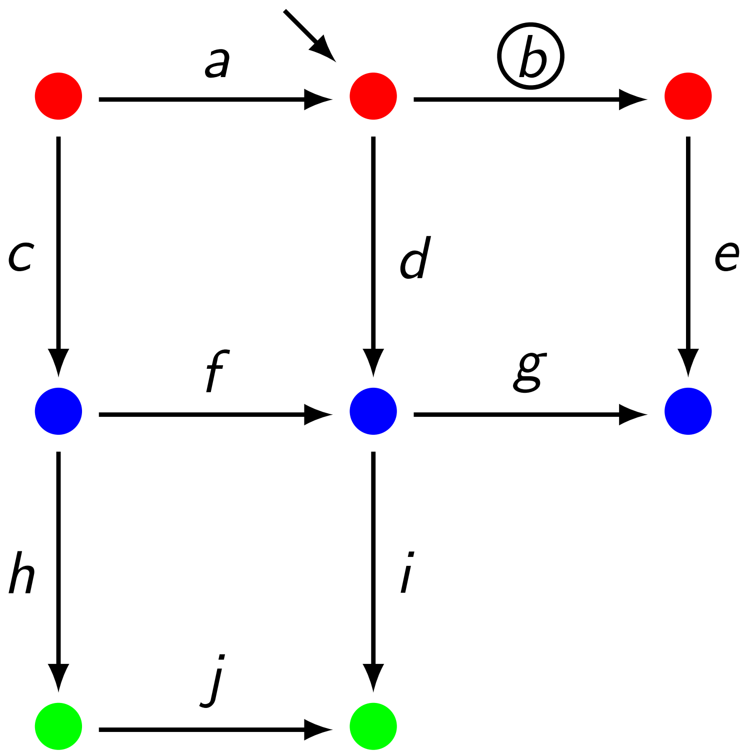
- Check if d is confluent
  - No; it is not invisible
- Check if b is confluent
  - It is invisible

Checking a transition for confluence



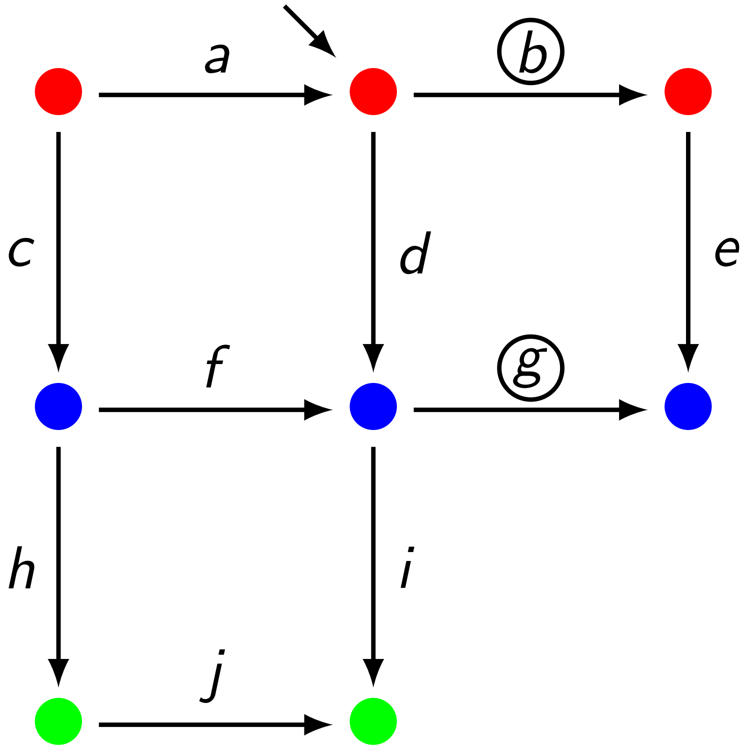
- Check if d is confluent
  - No; it is not invisible
- Check if b is confluent
  - It is invisible
  - Is the d-transition mimicked?

Checking a transition for confluence



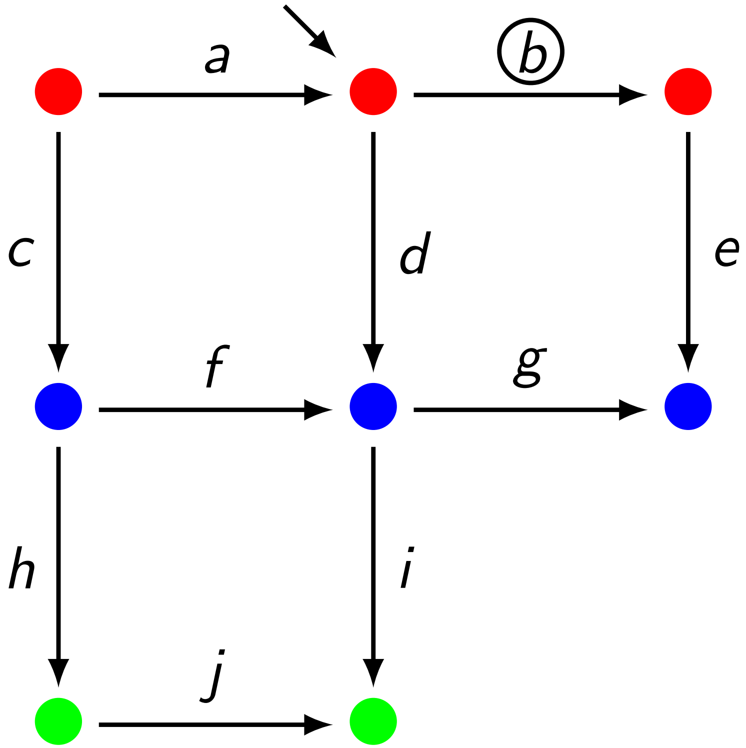
- Check if *d* is confluent
  - No; it is not invisible
- Check if *b* is confluent
  - It is invisible
  - Is the *d*-transition mimicked?
    - Possibly by the *e*-transition

Checking a transition for confluence



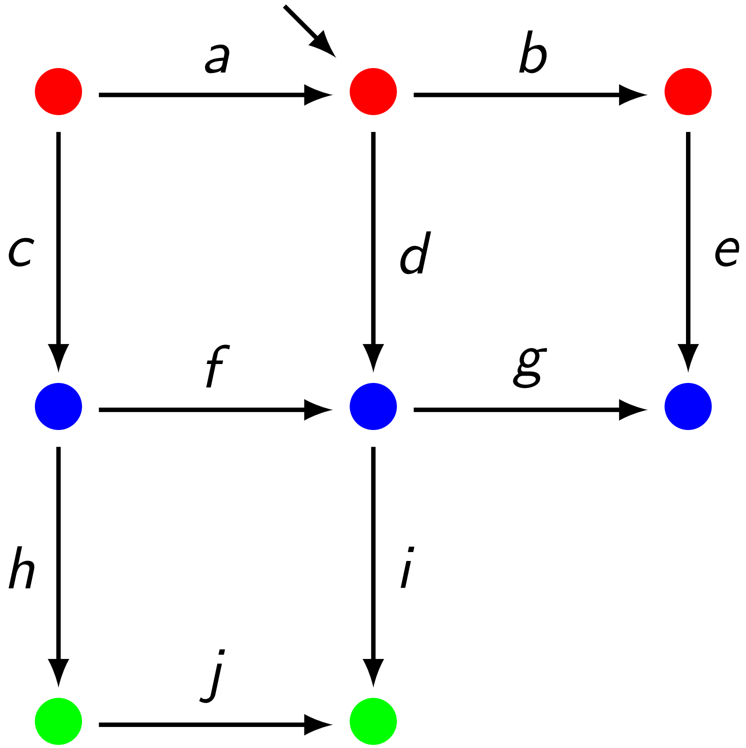
- Check if d is confluent
  - No; it is not invisible
- Check if b is confluent
  - It is invisible
  - Is the d-transition mimicked?
    - Possibly by the e-transition
    - But then g has to be confluent: check this

Checking a transition for confluence



- Check if d is confluent
  - No; it is not invisible
- Check if b is confluent
  - It is invisible
  - Is the d-transition mimicked?
    - Possibly by the e-transition
    - But then g has to be confluent: check this
    - It is not; abort

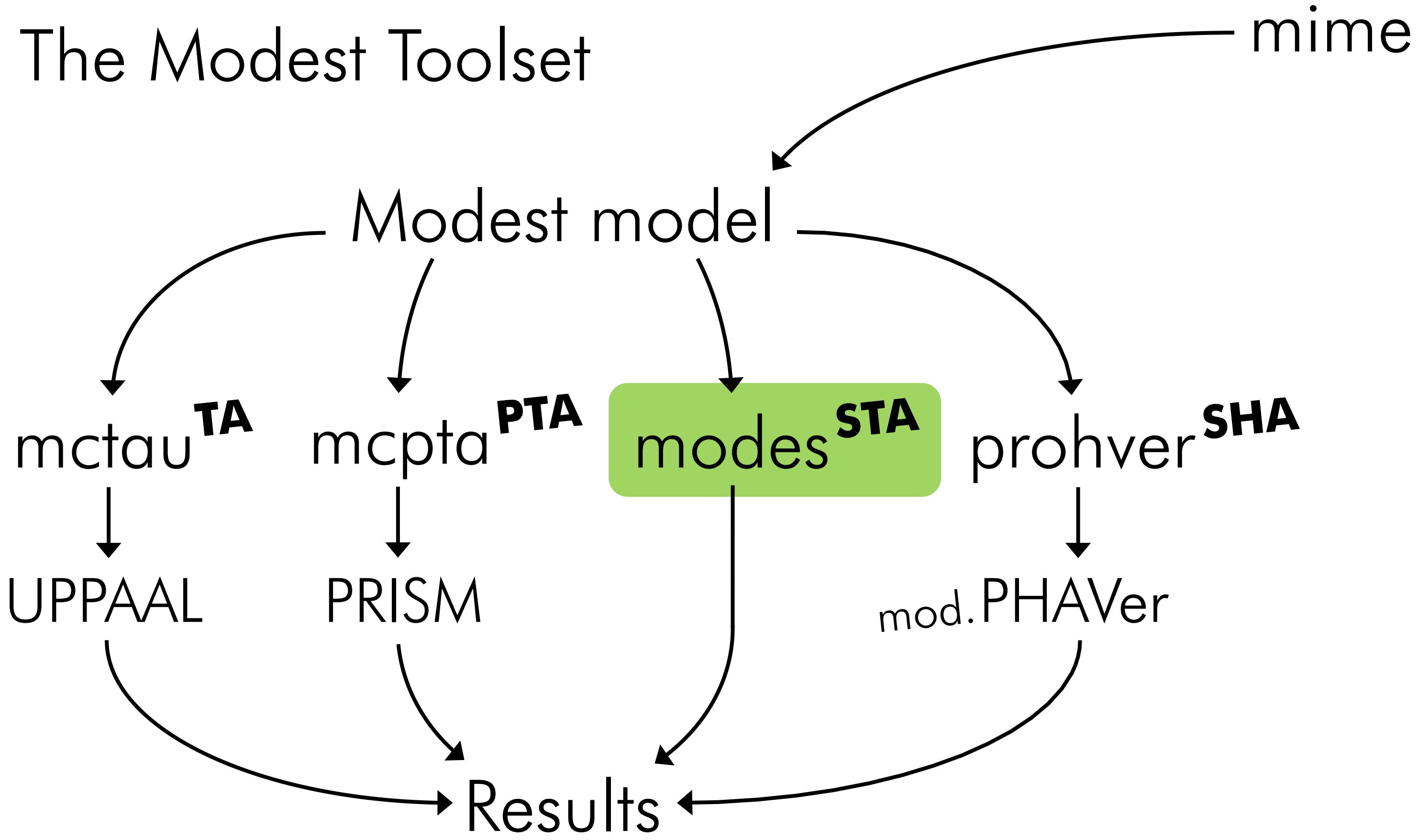
Checking a transition for confluence

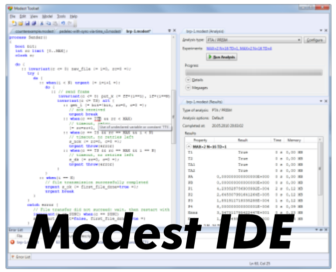


- Check if *d* is confluent
  - No; it is not invisible
- Check if *b* is confluent
  - It is invisible
  - Is the *d*-transition mimicked?
    - Possibly by the *e*-transition
    - But then *g* has to be confluent: check this
    - It is not; abort

Tool Support

The Modest Toolset





modes: SMC for Modest/STA

⇨ MDP as special case

POR + **Confluence** NEW

www.modestchecker.net

Evaluation

Examples

	POR	confluence
Dining Cryptographers **PRISM model** N cryptographers, two neighbours each Nondeterminism: communication order	✗	✓
CSMA/CD **"DPTA"** Two senders, one shared channel, collisions Nondeterministic choice of station inside channel	✗	✓
BEB (Bounded Exponential Backoff) Detailed MDP model of exponential backoff K: max. backoff, N: n° of retries, H: n° of hosts **huge state space**	✓	✓

Evaluation

Results

reachability properties

(10000 runs ⇒$\epsilon<0.01$, $\delta>0.98$)

model	params	uniform: time	partial order: time	partial order: k	partial order: s	**confluence:** time	confluence: k	confluence: s	confluence: c	confluence: t	model checking: states	model checking: time
dining crypto-graphers (N)	(3)	1 s	–	–	–	3 s	4	9	4.0	8.0	609	1 s
	(4)	1 s	–	–	–	11 s	6	25	6.0	10.0	3 841	2 s
	(5)	1 s	–	–	–	44 s	8	67	8.0	12.0	23 809	7 s
	(6)	1 s	–	–	–	229 s	10	177	10.0	14.0	144 705	26 s
	(7)	1 s	–	–	–	– timeout –					864 257	80 s
CSMA/CD (RF, BC_{max})	(2, 1)	2 s	–	–	–	4 s	3	46	5.4	16.4	15 283	11 s
	(1, 1)	2 s	–	–	–	4 s	3	46	5.4	16.4	30 256	49 s
	(2, 2)	2 s	–	–	–	10 s	3	150	5.1	16.0	98 533	52 s
	(1, 2)	2 s	–	–	–	10 s	3	150	5.1	16.0	194 818	208 s
BEB (K, N, H)	(4, 3, 3)	1 s	3 s	3	4	1 s	3	7	3.3	11.6	$> 10^3$	> 0 s
	(8, 7, 4)	2 s	7 s	4	8	4 s	4	15	5.6	16.7	$> 10^7$	> 7 s
	(16,15,5)	3 s	18 s	5	16	11 s	5	31	8.3	21.5	– memout –	
	(16,15,6)	3 s	40 s	6	32	34 s	6	63	11.2	26.2	– memout –	

+ performance on BEB & CSMA/CD models
  + vs. model-checking

+ a bit faster than POR

− does not work well for dining cryptographers

Conclusion

A new approach to SMC for MDPs
based on **on-the-fly confluence detection**

– detect confluence on-the-fly on the concrete state space
– handle more kinds of nondeterminism than POR method

approach	nondeterminism	probabilities	memory	error bounds
POR	spurious interleavings	$P{\downarrow}max = P{\downarrow}min$	$s \ll n$	unchanged
⇨ confluence	confluent spurious	$P{\downarrow}max = P{\downarrow}min$	$s \ll n$	unchanged
learning	any	$P{\downarrow}max$ only	$s \to n$	convergence

See also

www.modestchecker.net

& H., T.: On-the-fly Confluence Detection for Statistical Model Checking (to appear at NFM 2013)