UNIVERSITY OF TWENTE.

Formal Methods & Tools.

Confluence versus Partial Order Reduction in Statistical Model Checking and Simulation

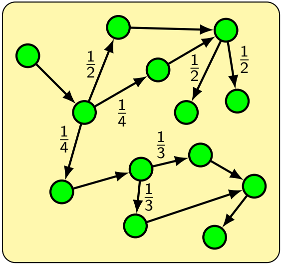
Mark Timmer

September 13, 2012

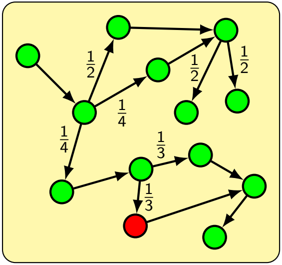
Aachen 2012

Joint work with Arnd Hartmanns

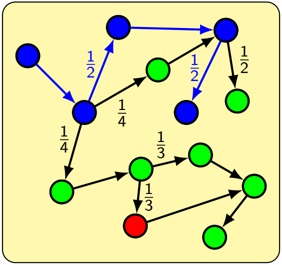
(Statistical) Model Checking



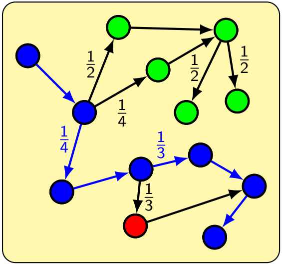
(Statistical) Model Checking



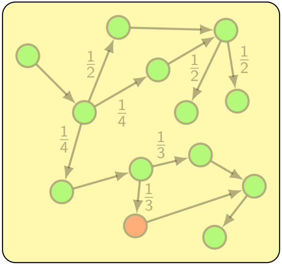
(Statistical) Model Checking

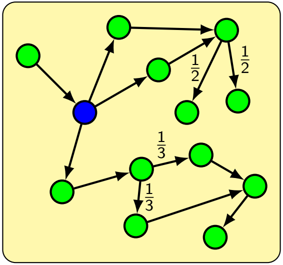


(Statistical) Model Checking

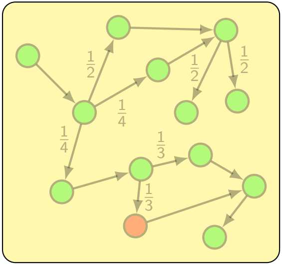


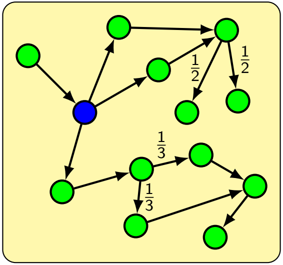
(Statistical) Model Checking





(Statistical) Model Checking

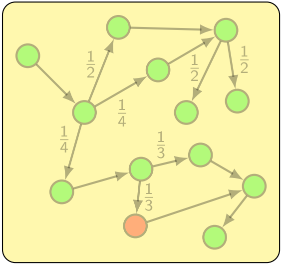


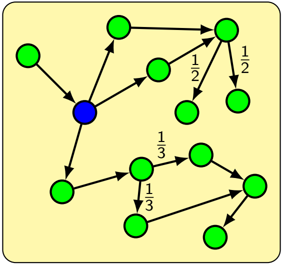


Problem with the presence of nondeterminism:

- No single probability: minimum and maximum

(Statistical) Model Checking





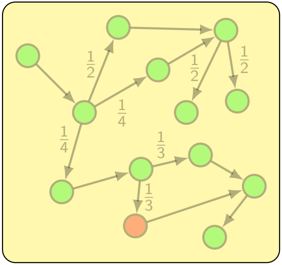
Problem with the presence of nondeterminism:

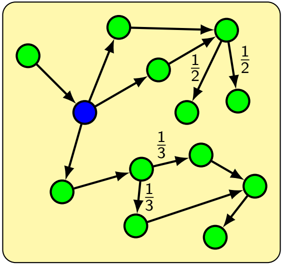
- No single probability: minimum and maximum

Possible solutions:

- Assume a (uniform) distribution
- Show the nondeterminism to be spurious

(Statistical) Model Checking





Problem with the presence of nondeterminism:

- No single probability: minimum and maximum

Possible solutions:

- Assume a (uniform) distribution
- Show the nondeterminism to be spurious $\Longleftarrow$

Reduction techniques for nondeterminism

Reduction techniques:

- Partial order reduction (ample sets)
- Confluence reduction

Reduction techniques for nondeterminism

Reduction techniques:

- Partial order reduction (ample sets)
- Confluence reduction

Reduction function F:

$F(s) \subseteq \text{enabled}(s)$

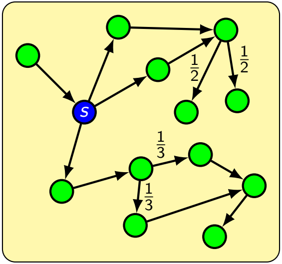
Reduction techniques for nondeterminism

Reduction techniques:

- Partial order reduction (ample sets)
- Confluence reduction

Reduction function F:

$F(s) \subseteq \text{enabled}(s)$



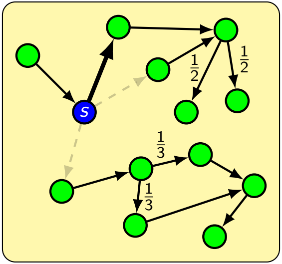
Reduction techniques for nondeterminism

Reduction techniques:

- Partial order reduction (ample sets)
- Confluence reduction

Reduction function F:

$F(s) \subseteq \mathrm{enabled}(s)$



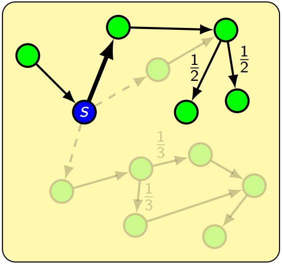
Reduction techniques for nondeterminism

Reduction techniques:

- Partial order reduction (ample sets)
- Confluence reduction

Reduction function F:

$F(s) \subseteq \text{enabled}(s)$



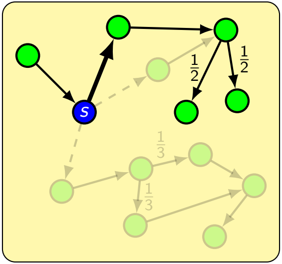
Reduction techniques for nondeterminism

Reduction techniques:

- Partial order reduction (ample sets)
- Confluence reduction

Reduction function F:

$F(s) \subseteq \text{enabled}(s)$

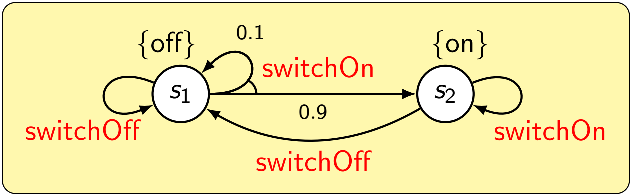


Important:

reduced system should be equivalent

The model: Probabilistic Automata

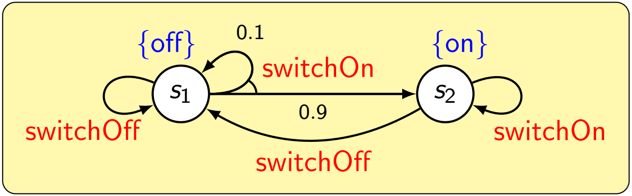
Probabilistic Automaton:



- Non-deterministically choose a transition
- Probabilistically choose the next state

The model: Probabilistic Automata

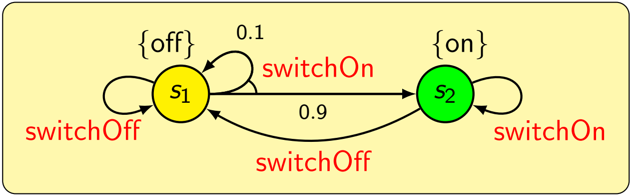
Probabilistic Automaton:



- Non-deterministically choose a transition
- Probabilistically choose the next state

The model: Probabilistic Automata

Probabilistic Automaton:



- Non-deterministically choose a transition
- Probabilistically choose the next state

Contents

Behaviour preservation by invisible steps

Invisible transitions in confluence reduction:

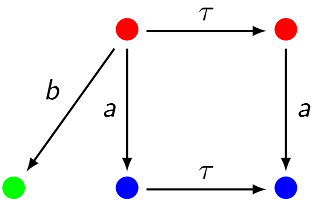
- Deterministic
- Stuttering

Behaviour preservation by invisible steps

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible τ-steps might disable behaviour...

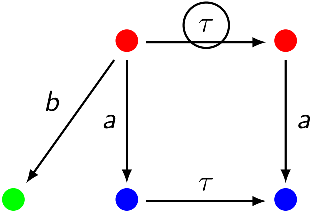


Behaviour preservation by invisible steps

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible τ-steps might disable behaviour. . .



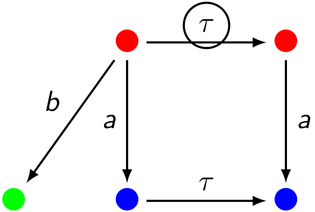
Behaviour preservation by invisible steps

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible τ-steps might disable behaviour...

...though often, they connect equivalent states



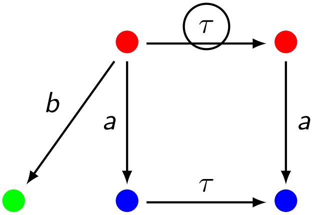
Behaviour preservation by invisible steps

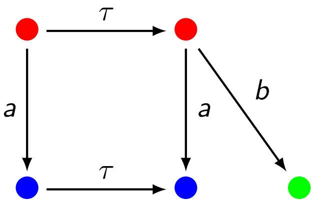
Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible τ-steps might disable behaviour...

...though often, they connect equivalent states





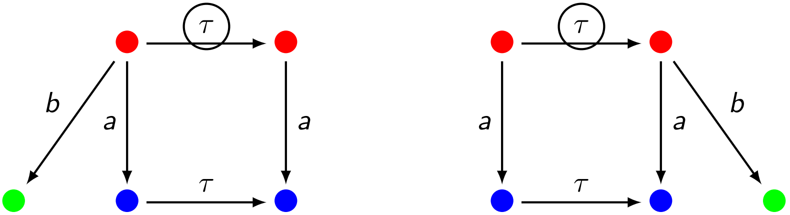
Behaviour preservation by invisible steps

Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible τ-steps might disable behaviour...

...though often, they connect equivalent states



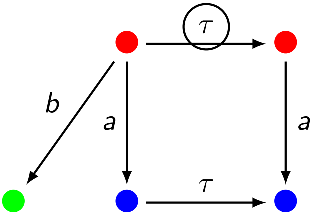
Behaviour preservation by invisible steps

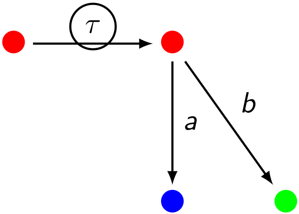
Invisible transitions in confluence reduction:

- Deterministic
- Stuttering

Invisible τ-steps might disable behaviour...

...though often, they connect equivalent states





Non-probabilistic and probabilistic confluence reduction

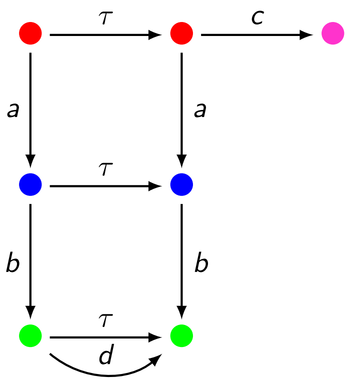
Confluence reduction:

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.

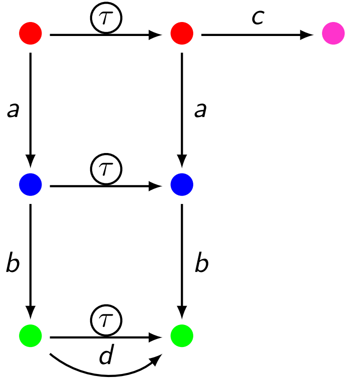
Non-probabilistically:



Non-probabilistic and probabilistic confluence reduction

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

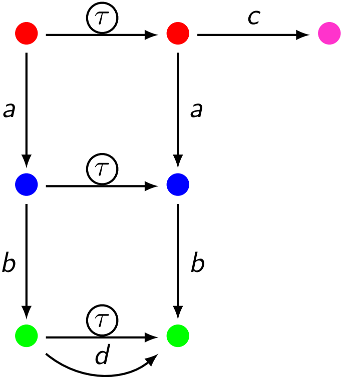
Non-probabilistically:



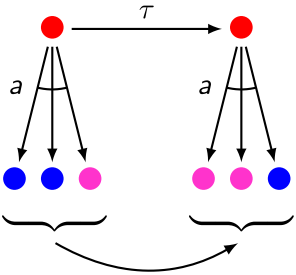
Non-probabilistic and probabilistic confluence reduction

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

Non-probabilistically:



Probabilistically:

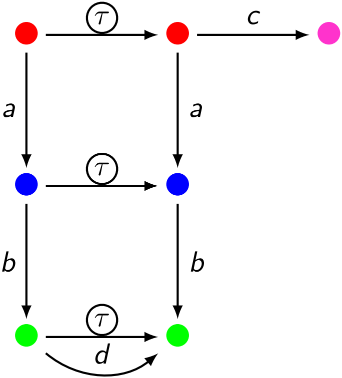


Non-probabilistic and probabilistic confluence reduction

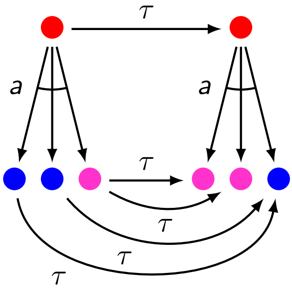
Confluence reduction:

denoting a subset of the invisible transitions as confluent.

Non-probabilistically:



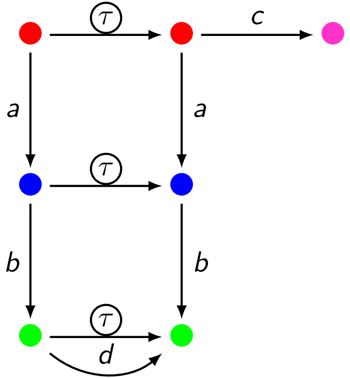
Probabilistically:



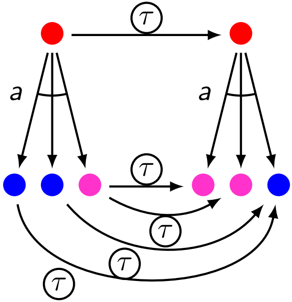
Non-probabilistic and probabilistic confluence reduction

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

Non-probabilistically:



Probabilistically:

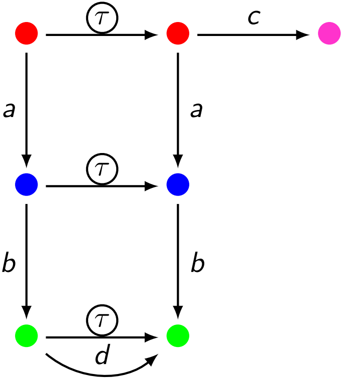


Non-probabilistic and probabilistic confluence reduction

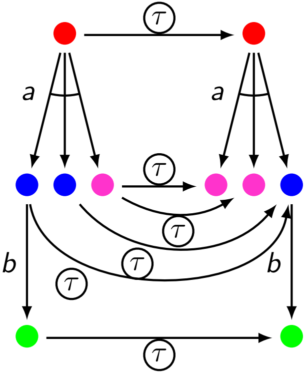
Confluence reduction:

denoting a subset of the invisible transitions as confluent.

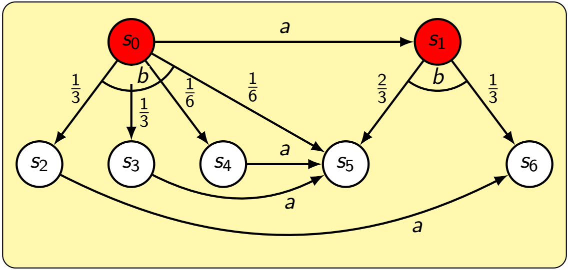
Non-probabilistically:



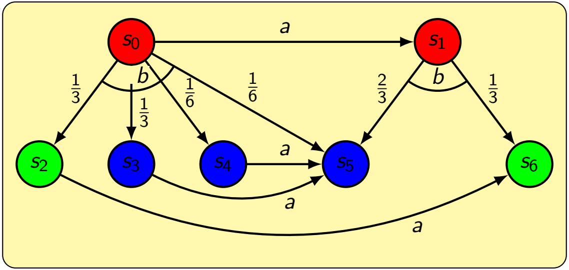
Probabilistically:



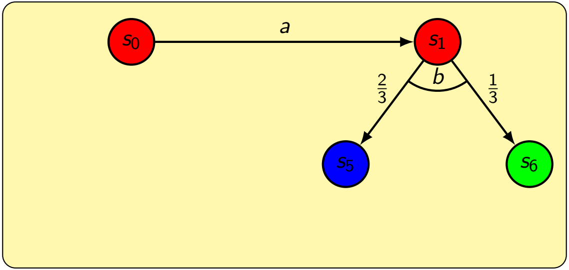
Probabilistic example



Probabilistic example



Probabilistic example

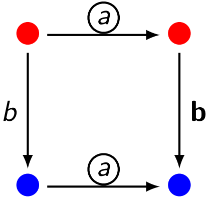


Alterations to the concept of confluence

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

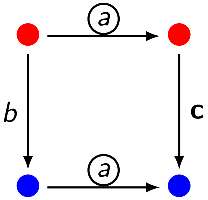
Alterations to the concept of confluence

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



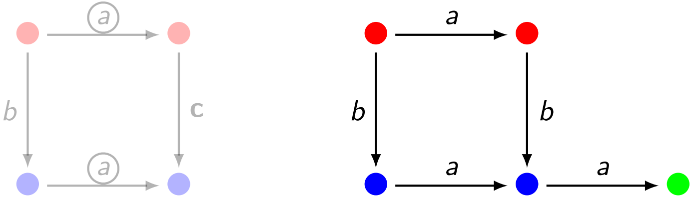
Alterations to the concept of confluence

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



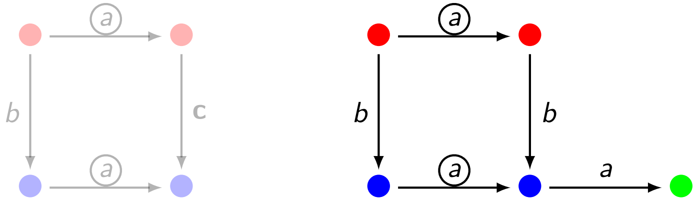
Alterations to the concept of confluence

- 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

Alterations to the concept of confluence

- More liberal notion of equivalence of distributions

Definition (Old)

Distributions μ and ν are $\mathcal{T}$-equivalent, if there exists a partitioning $\mathrm{spt}(\mu) = \biguplus_{i=1}^{n} S_i$ of the support of μ and an ordering $\mathrm{spt}(\nu) = \{s_1, \dots, s_n\}$ of the support of ν, such that $\forall 1 \leq i \leq n$

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

Alterations to the concept of confluence

- More liberal notion of equivalence of distributions

Definition (Old)

Distributions μ and ν are $\mathcal{T}$-equivalent, if there exists a partitioning $\text{spt}(\mu) = \biguplus_{i=1}^{n} S_i$ of the support of μ and an ordering $\text{spt}(\nu) = \{s_1, \dots, s_n\}$ of the support of ν, such that $\forall 1 \leq i \leq n$

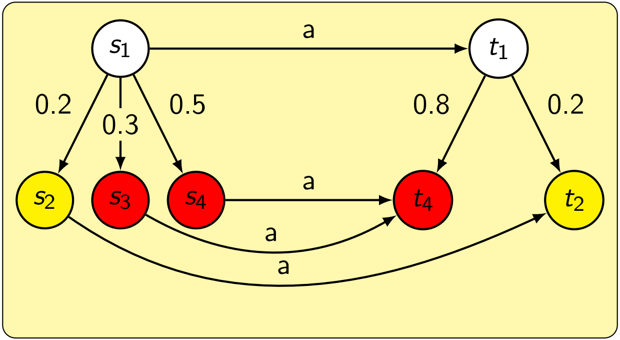
$$\mu(S_i) = \nu(s_i) \wedge (S_i = \{s_i\} \vee \forall s \in S_i \, . \, \exists a \in \Sigma \, . \, s \xrightarrow{a} s_i \in \mathcal{T}).$$

Definition (New)

Distributions μ and ν are $\mathcal{T}$-equivalent, if $\mu \equiv_R \nu$ for the smallest equivalence relation R containing the set

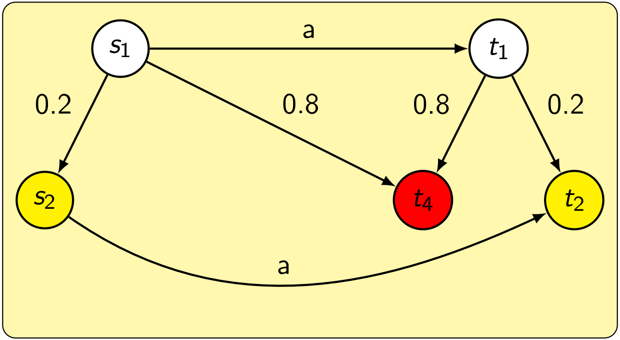
$$\{(s,t) \mid s \in \text{spt}(\mu), t \in \text{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}$$

Alterations to the concept of confluence



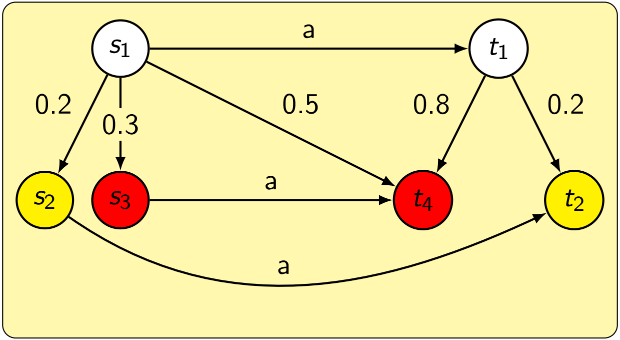
$$\mu(S_i) = \nu(s_i) \land (S_i = \{s_i\} \lor \forall s \in S_i \,.\, \exists a \in \Sigma \,.\, s \xrightarrow{a} s_i \in \mathcal{T})$$

Alterations to the concept of confluence



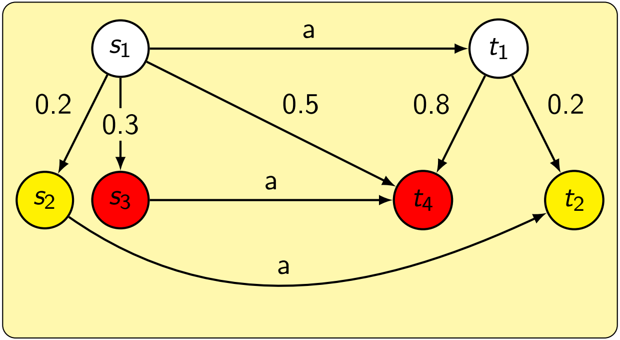
$$\mu(S_i) = \nu(s_i) \land (S_i = \{s_i\} \lor \forall s \in S_i \,.\, \exists a \in \Sigma \,.\, s \xrightarrow{a} s_i \in \mathcal{T})$$

Alterations to the concept of confluence



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

Alterations to the concept of confluence



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

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

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:

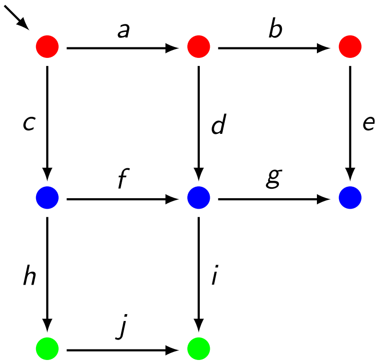
1. Simulate until reaching a nondeterministic choice
2. Check for each outgoing transition if it is confluent
   - If one choice is confluent, take it and continue
   - If no choice is confluent, abort

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

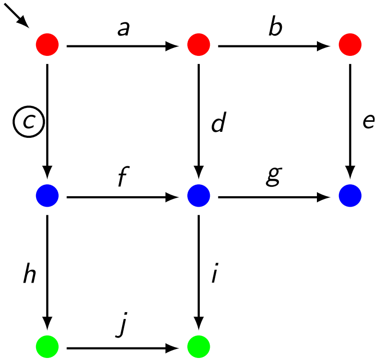
To check if a transition is confluent:

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

Checking a transition for confluence

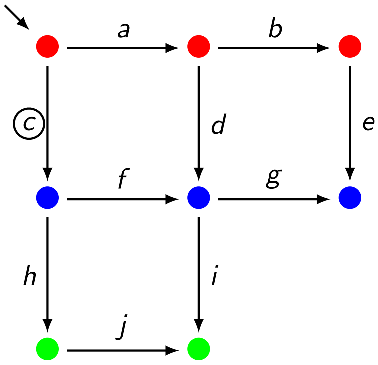


Checking a transition for confluence



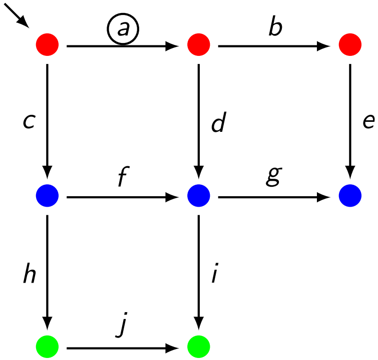
- Check if c is confluent

Checking a transition for confluence



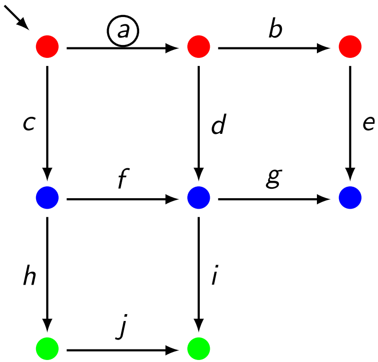
- Check if c is confluent
  - No; it is not invisible

Checking a transition for confluence



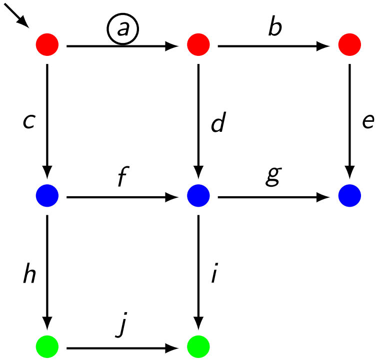
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent

Checking a transition for confluence



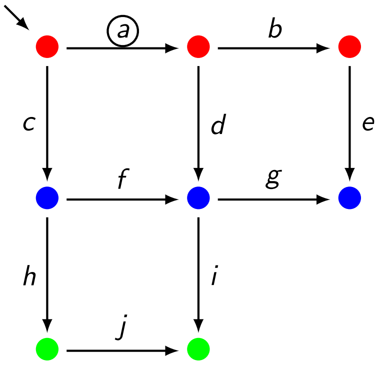
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible

Checking a transition for confluence



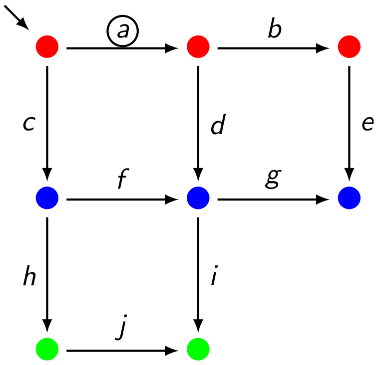
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?

Checking a transition for confluence



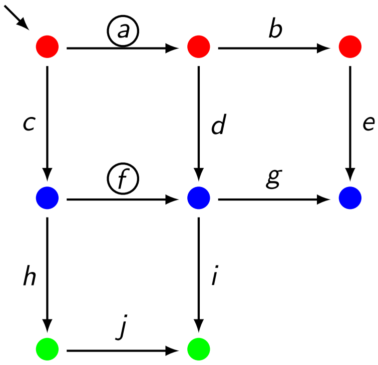
- Check if *c* is confluent
  - No; it is not invisible
- Check if *a* is confluent
  - It is invisible
  - Is the *c*-transition mimicked?
    - Possibly by the *d*-transition

Checking a transition for confluence



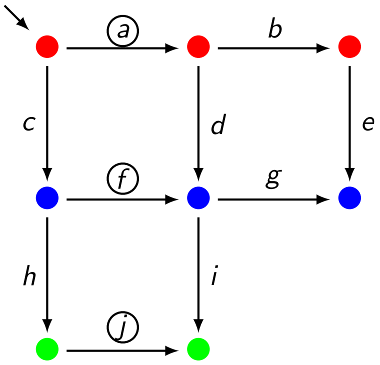
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?
    - Possibly by the d-transition
    - But then f has to be confluent: check this
    - ...

Checking a transition for confluence



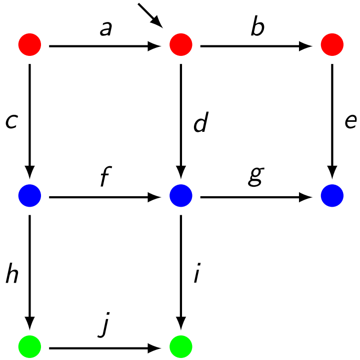
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?
    - Possibly by the d-transition
    - But then f has to be confluent: check this
    - ...

Checking a transition for confluence



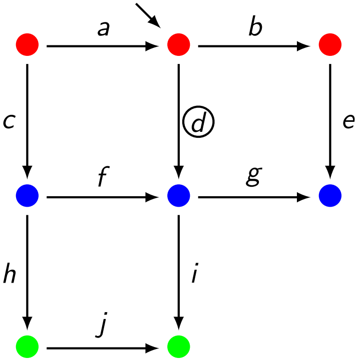
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?
    - Possibly by the d-transition
    - But then f has to be confluent: check this
    - ...

Checking a transition for confluence



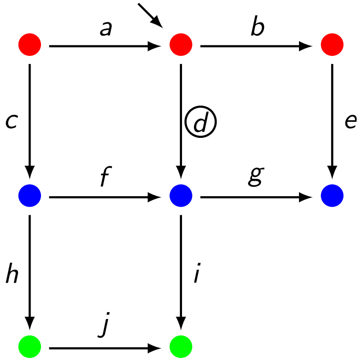
- Check if c is confluent
  - No; it is not invisible
- Check if a is confluent
  - It is invisible
  - Is the c-transition mimicked?
    - Possibly by the d-transition
    - But then f has to be confluent: check this
    - ...

Checking a transition for confluence



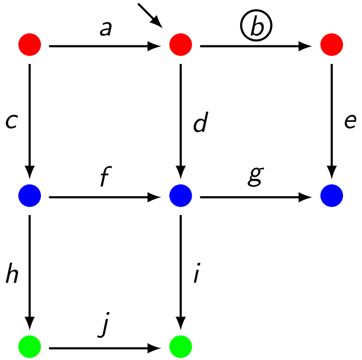
- Check if d is confluent

Checking a transition for confluence



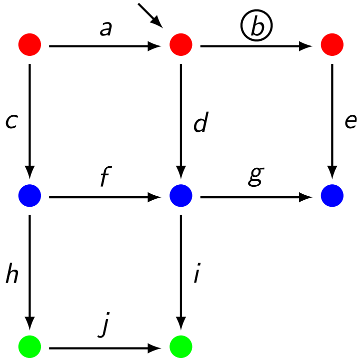
- Check if *d* is confluent
  - No; it is not invisible

Checking a transition for confluence



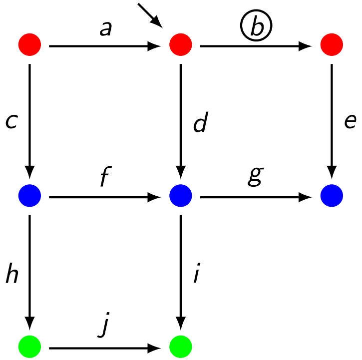
- Check if d is confluent
  - No; it is not invisible
- Check if b is confluent

Checking a transition for confluence



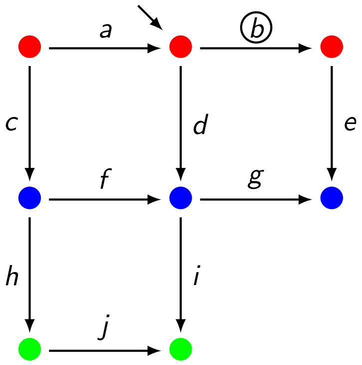
- Check if d is confluent
  - No; it is not invisible
- Check if b is confluent
  - It is invisible

Checking a transition for confluence



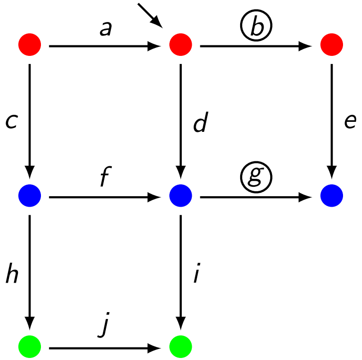
- Check if *d* is confluent
  - No; it is not invisible
- Check if *b* is confluent
  - It is invisible
  - Is the *d*-transition mimicked?

Checking a transition for confluence



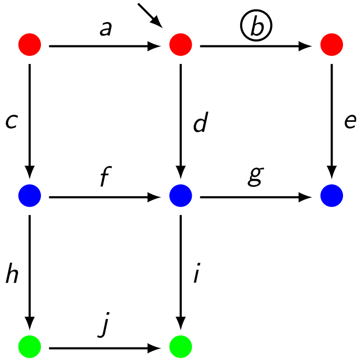
- Check if *d* is confluent
  - No; it is not invisible
- Check if *b* is confluent
  - It is invisible
  - Is the *d*-transition mimicked?
    - Possibly by the *e*-transition

Checking a transition for confluence



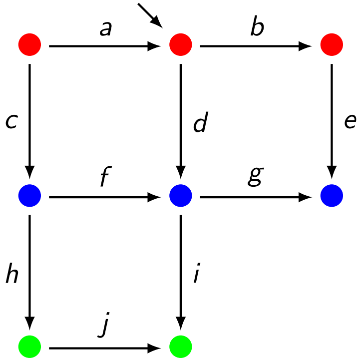
- Check if d is confluent
  - No; it is not invisible
- Check if b is confluent
  - It is invisible
  - Is the d-transition mimicked?
    - Possibly by the e-transition
    - But then g has to be confluent: check this
    - ...

Checking a transition for confluence



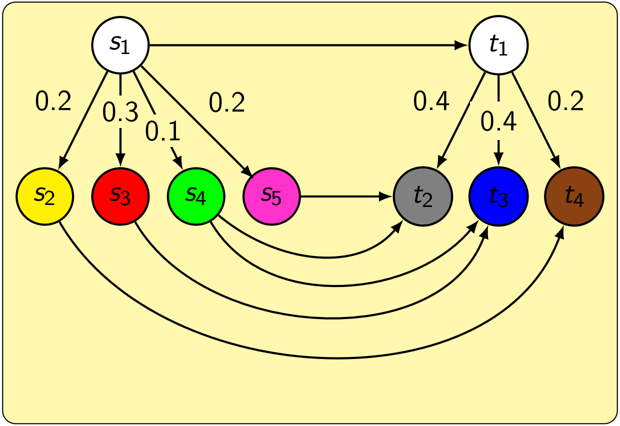
- Check if d is confluent
  - No; it is not invisible
- Check if b is confluent
  - It is invisible
  - Is the d-transition mimicked?
    - Possibly by the e-transition
    - But then g has to be confluent: check this
    - ...

Checking a transition for confluence



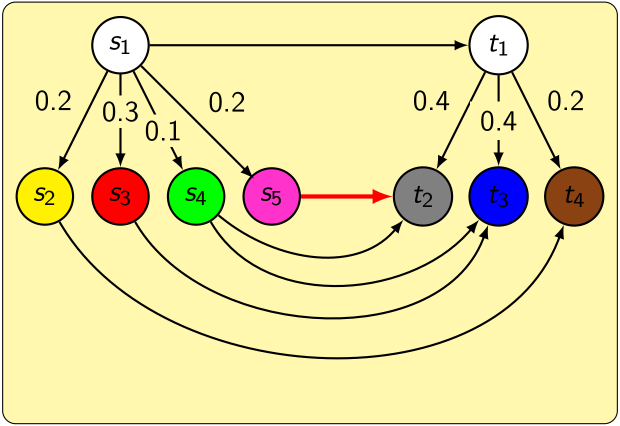
- Check if *d* is confluent
  - No; it is not invisible
- Check if *b* is confluent
  - It is invisible
  - Is the *d*-transition mimicked?
    - Possibly by the *e*-transition
    - But then *g* has to be confluent: check this
    - ...

Detecting equivalence of transitions



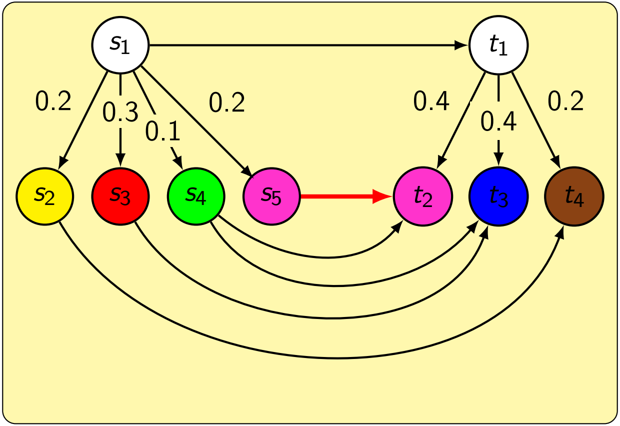
$$\boxed{\{(s,t) \mid s \in \mathrm{spt}(\mu), t \in \mathrm{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}}$$

Detecting equivalence of transitions



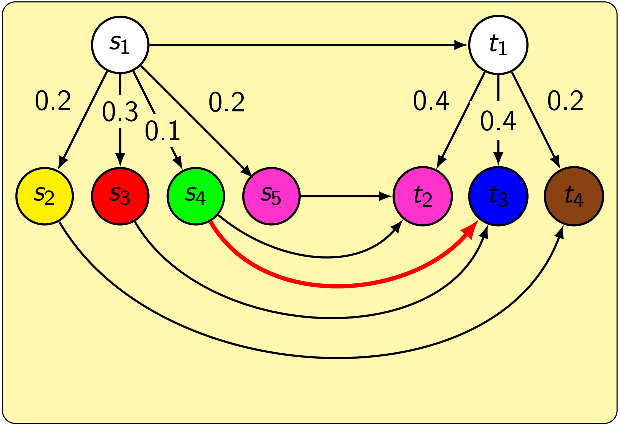
$$\boxed{\{(s, t) \mid s \in \mathrm{spt}(\mu), t \in \mathrm{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}}$$

Detecting equivalence of transitions



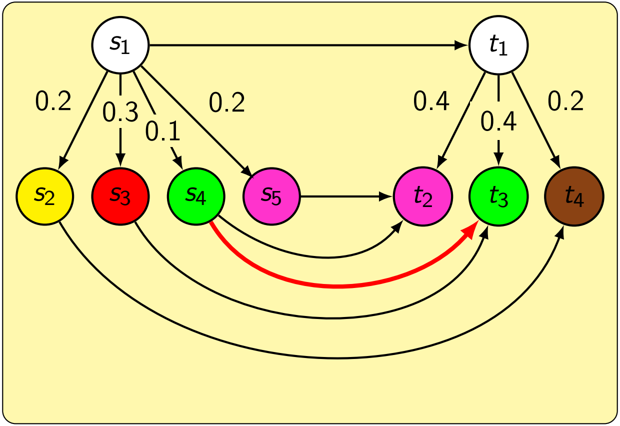
$$\boxed{\{(s, t) \mid s \in \mathrm{spt}(\mu), t \in \mathrm{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}}$$

Detecting equivalence of transitions



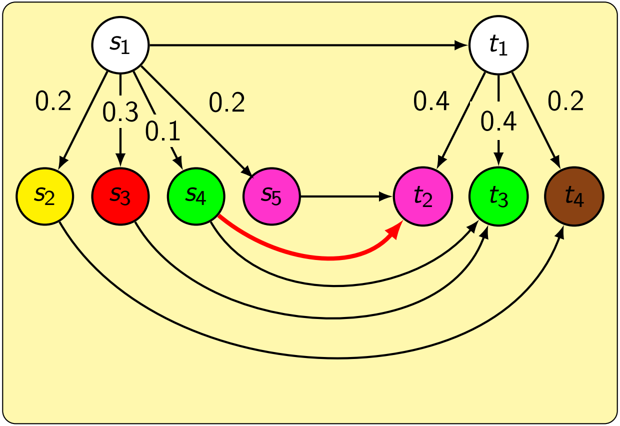
$$\boxed{\{(s,t) \mid s \in \mathrm{spt}(\mu), t \in \mathrm{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}}$$

Detecting equivalence of transitions



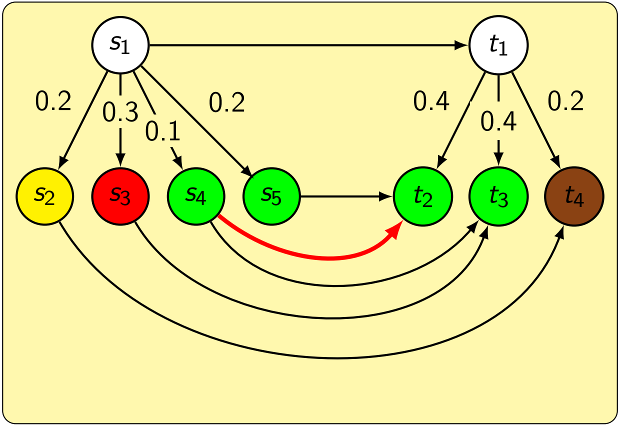
$$\boxed{\{(s,t) \mid s \in \mathrm{spt}(\mu), t \in \mathrm{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}}$$

Detecting equivalence of transitions



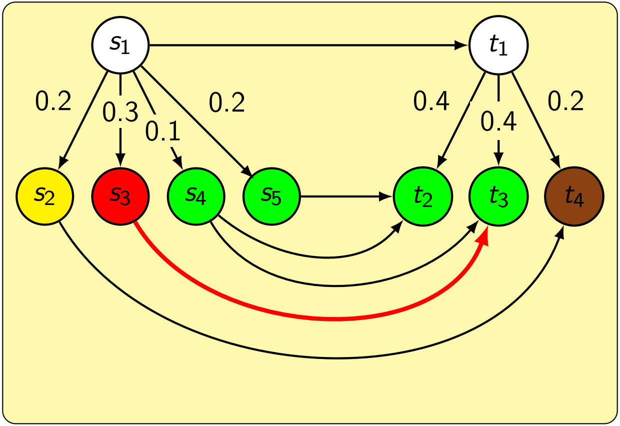
$$\boxed{\{(s,t) \mid s \in \mathrm{spt}(\mu), t \in \mathrm{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}}$$

Detecting equivalence of transitions



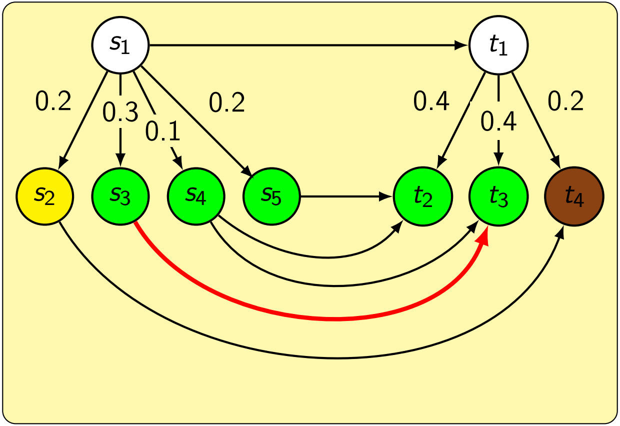
$$\boxed{\{(s,t) \mid s \in \mathrm{spt}(\mu), t \in \mathrm{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}}$$

Detecting equivalence of transitions



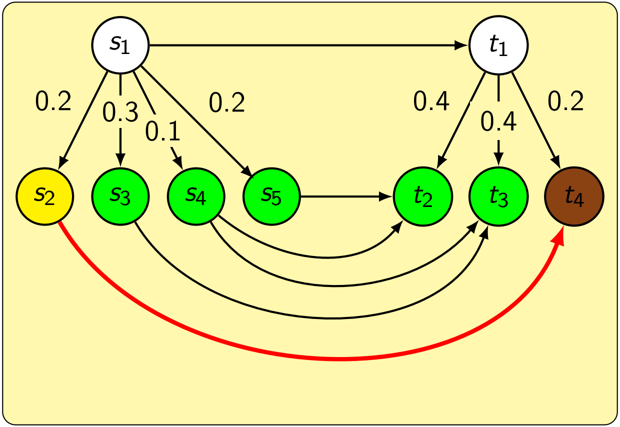
$$\boxed{\{(s, t) \mid s \in \mathrm{spt}(\mu), t \in \mathrm{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}}$$

Detecting equivalence of transitions



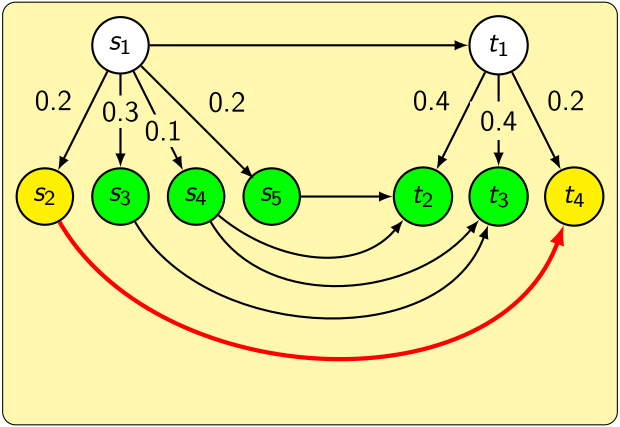
$$\boxed{\{(s,t) \mid s \in \mathrm{spt}(\mu), t \in \mathrm{spt}(\nu), s \xrightarrow{a} t \in \mathcal{T}\}}$$

Detecting equivalence of transitions



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

Detecting equivalence of transitions



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

Implementation

modes: a discrete-event simulator for the MODEST language

- Statistical model checking of deterministic systems
  - Partial order reduction
  - Confluence reduction

Implementation

modes: a discrete-event simulator for the MODEST language

- Statistical model checking of deterministic systems
  - Partial order reduction
  - Confluence reduction

Three case studies:

- Dining Cryptographers
- IEEE 802.3 CSMA/CD
- Binary Exponential Backoff

Case study: Dining Cryptographers

Table : Confluence simulation runtime compared

model	simulation	
(N)	uniform	confluence
3	3 s	13 s
4	4 s	66 s
5	5 s	338 s

Partial order reduction was not able to resolve the nondetermism.

Case study: CSMA/CD

Table : Confluence simulation runtime compared

model	simulation		model checking	
(RED, BC_{MAX})	uniform	confluence	states	time
$(2, 1)$	6 s	18 s	15283	11 s
$(1, 1)$	6 s	18 s	30256	51 s
$(1, 2)$	11 s	48 s	194818	214 s

Partial order reduction was not able to resolve the nondetermism.
(for confluence, probabilistic transitions needed to be synchronised)

Case study: Binary Exponential Backoff

Table : Confluence simulation runtime compared

model	simulation		
(K, N, H)	uniform	partial order	confluence
$(4, 3, 3)$	1 s	2 s	2 s
$(8, 7, 4)$	14 s	18 s	16 s

Conclusions

- We improved on the notion of confluence reduction:
  - Independent of action labels
  - Independent of global behaviour
  - More liberal equivalence of distributions

Conclusions

- We improved on the notion of confluence reduction:
  - Independent of action labels
  - Independent of global behaviour
  - More liberal equivalence of distributions
- We provided an on-the-fly detection algorithm for SMC
- We implemented the new technique in MODEST

Conclusions

- We improved on the notion of confluence reduction:
  - Independent of action labels
  - Independent of global behaviour
  - More liberal equivalence of distributions
- We provided an on-the-fly detection algorithm for SMC
- We implemented the new technique in MODEST
- Case studies show that confluence reduction reduces more and slightly faster than partial order reduction
- More models can now statistically be checked

Questions

Questions?