Hierarchical Counterexamples for DTMCs

Nils Jansen, Erika Ábrahám, Jens Katelaan, Ralf Wimmer, Joost-Pieter Katoen, and Bernd Becker

Software Modeling and Verification, Theory of Hybrid Systems
RWTH Aachen University, Germany

Computer Architecture
University of Freiburg, Germany

March 27, 2011

ROCKS Workshop

Contents

1 Motivation

2 SCC-based Model Checking

3 Counterexample Generation

4 Implementation and Case Studies

5 Conclusion & Future Work

Counterexamples

Model Checking

- Shows correctness of a system
- Reveals defectiveness of a system

Counterexamples for LTL properties

- Are delivered by Model Checking for defective systems
- Consist of single traces through a system

Counterexamples in the probabilistic setting

- Are not computed during Model Checking
- Consist of (large or infinite) sets of paths

Probabilistic Counterexamples

Some state-of-the-art methods

- Search for paths in order of their probability *(Damman, Han, and Katoen 2008)*
- Find minimal counterexamples
- Use the abstraction of SCCs *(Andrés, D'Argenio and van Rossum, 2008)*

Counterexamples are represented

- By enumeration of the paths
- By regular expressions

Hierarchical Counterexample Generation

Method

- SCC-based Model Checking [QEST'10]
- If property was falsified:
  - Search on abstract system
  - Hierarchical concretization
  - Two different search approaches

Representation

- Critical subsystem

Advantages

- Compact representation $\rightsquigarrow$ Usability
- Abstract counterexamples $\rightsquigarrow$ Treatment of large systems
- Hierarchical approach $\rightsquigarrow$ Omission of system parts

Contents

1 Motivation

2 SCC-based Model Checking

3 Counterexample Generation

4 Implementation and Case Studies

5 Conclusion & Future Work

SCC-based Model Checking - Summary

Problem

- Model Checking DTMCs against unbounded reachability properties
- Target nodes are absorbing.

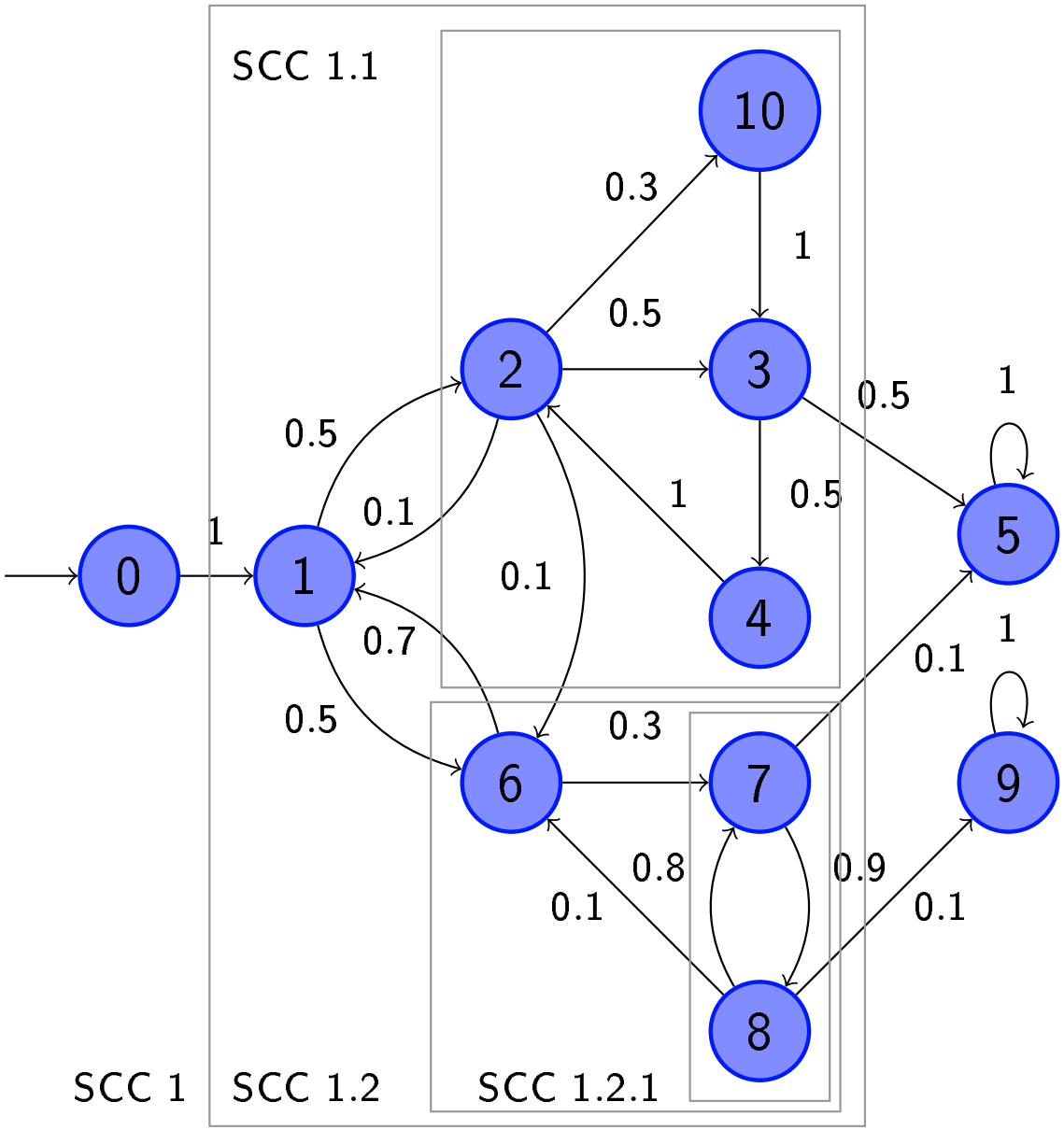
Idea

- Reduce each (nested) SCC to an abstract node whose outgoing edges carry the whole probability mass.

Recursive algorithm

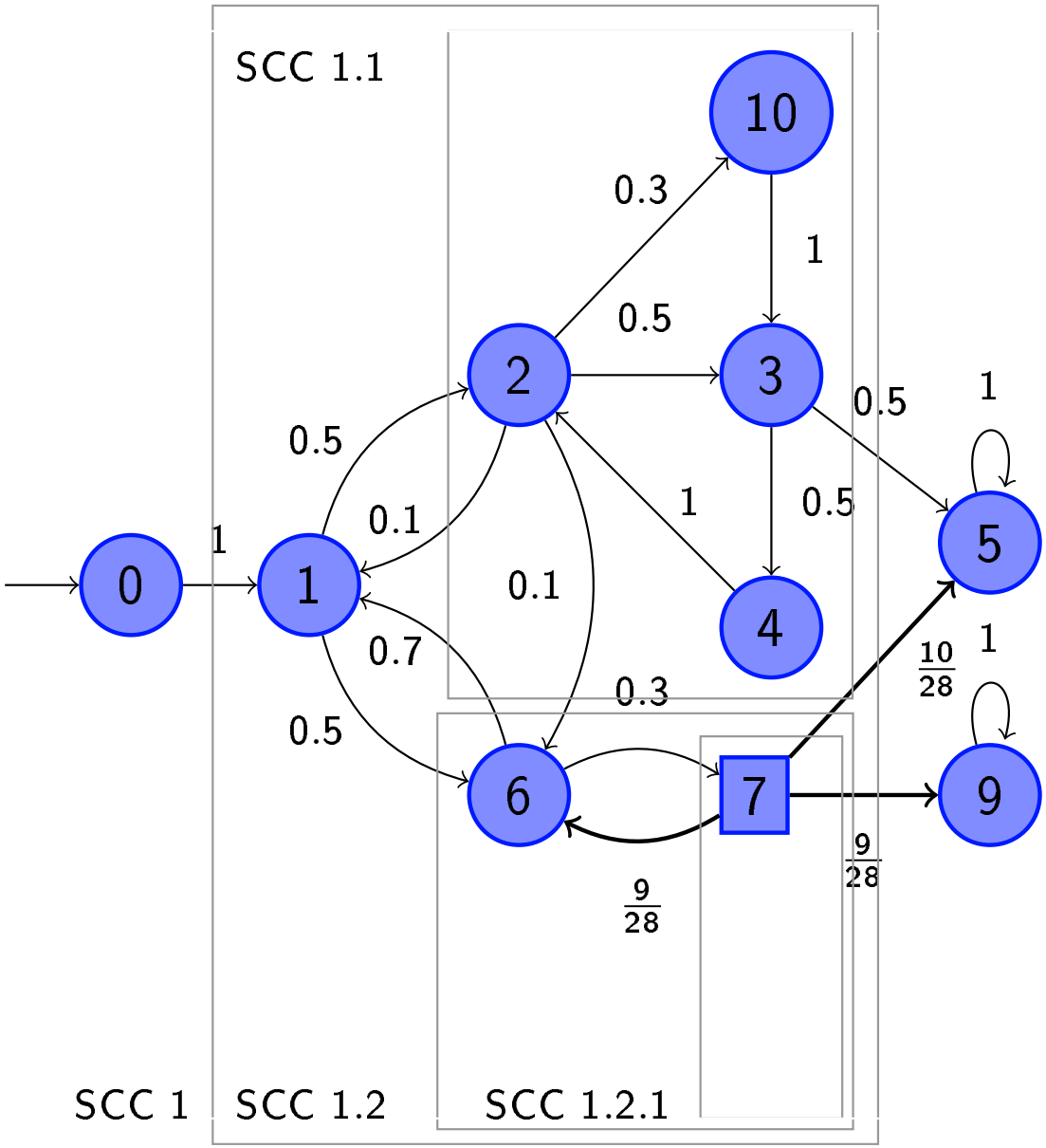
- Bottom-Up computing starting with "minimal SCCs"
- Exploiting specific properties of Markov Chains

SCC-based Model Checking - Example



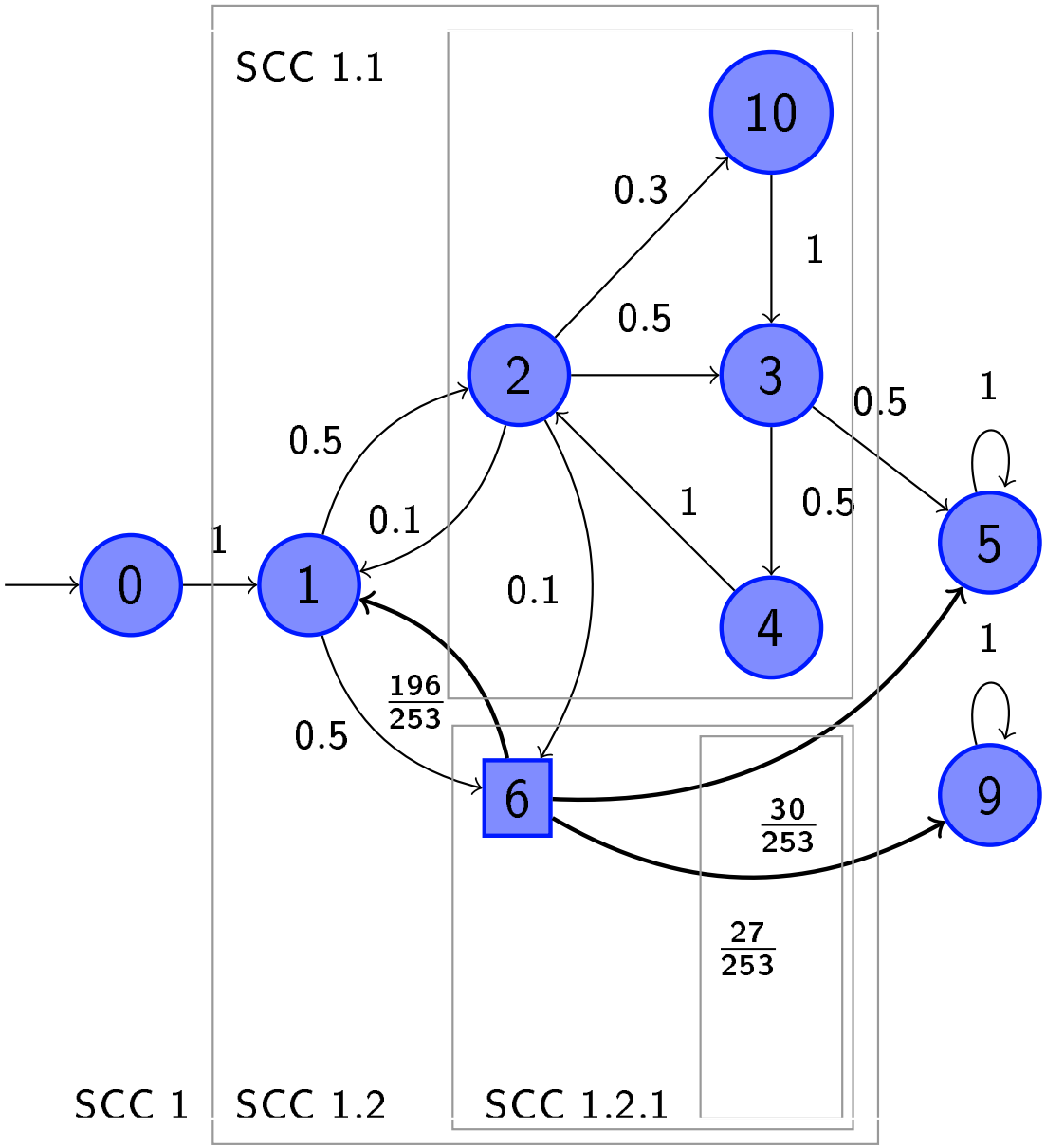
(Nested) SCCs of M

SCC-based Model Checking - Example



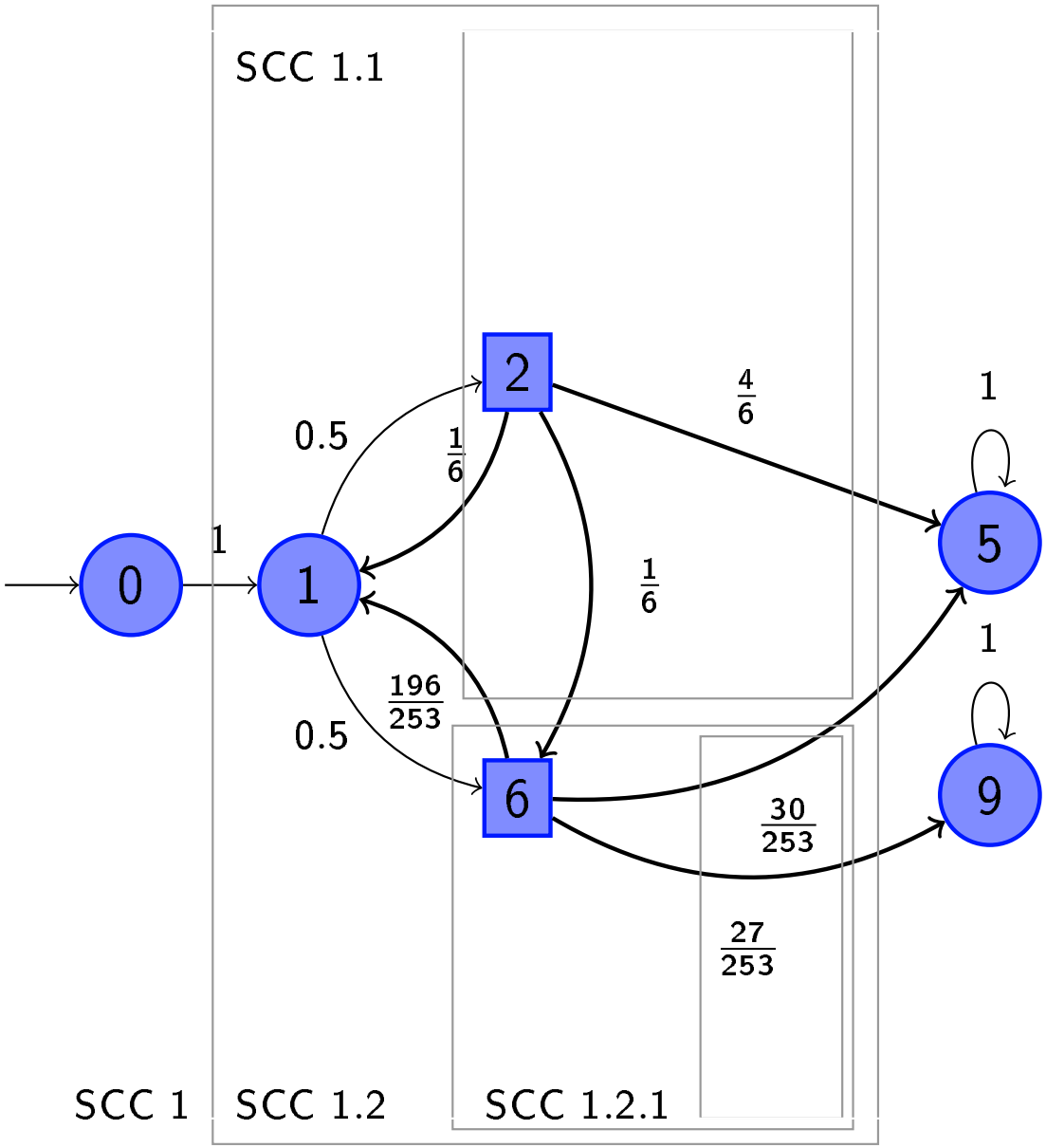
Abstraction of *SCC* 1.2.1

SCC-based Model Checking - Example



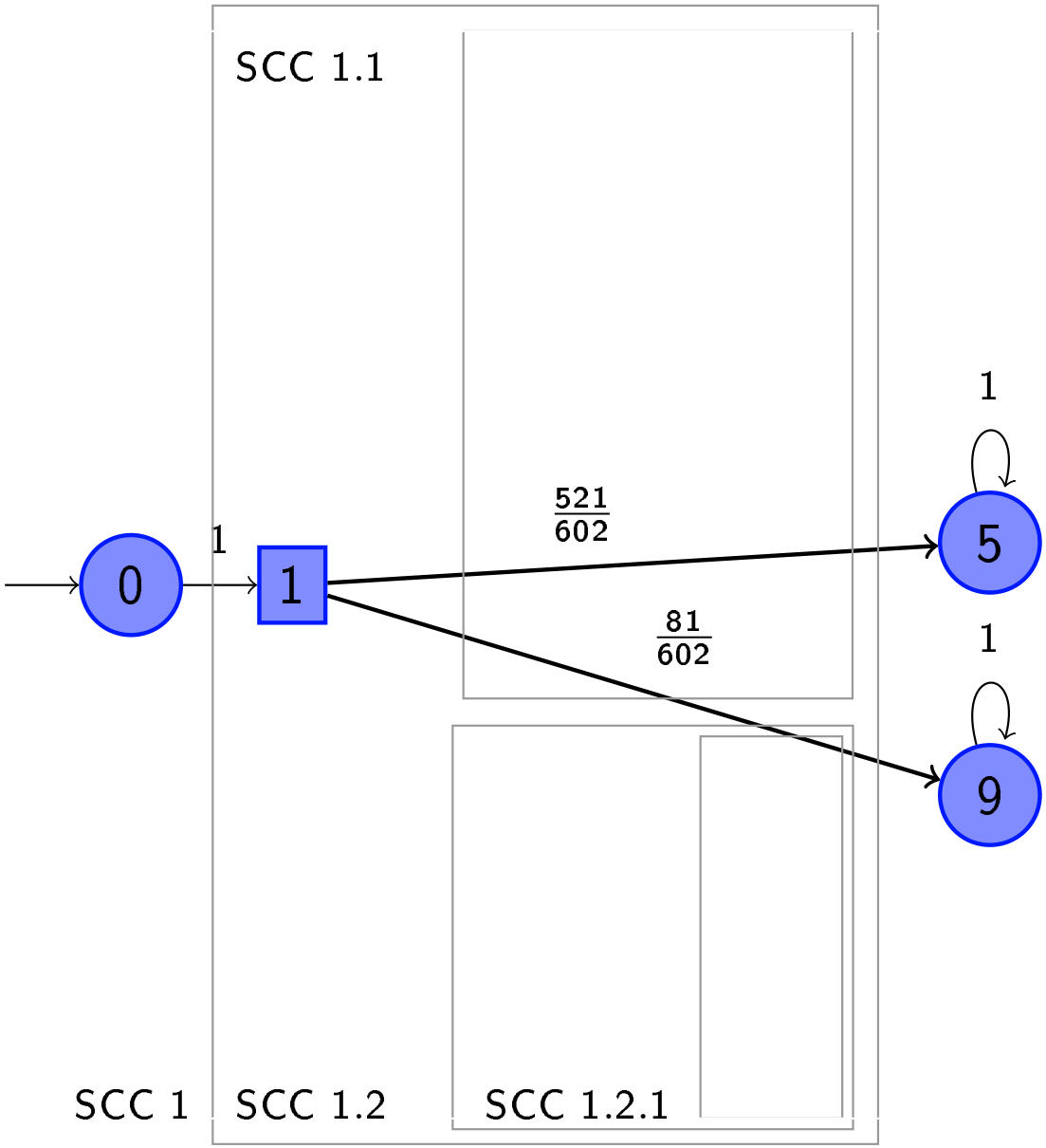
Abstraction of *SCC* 1.2

SCC-based Model Checking - Example



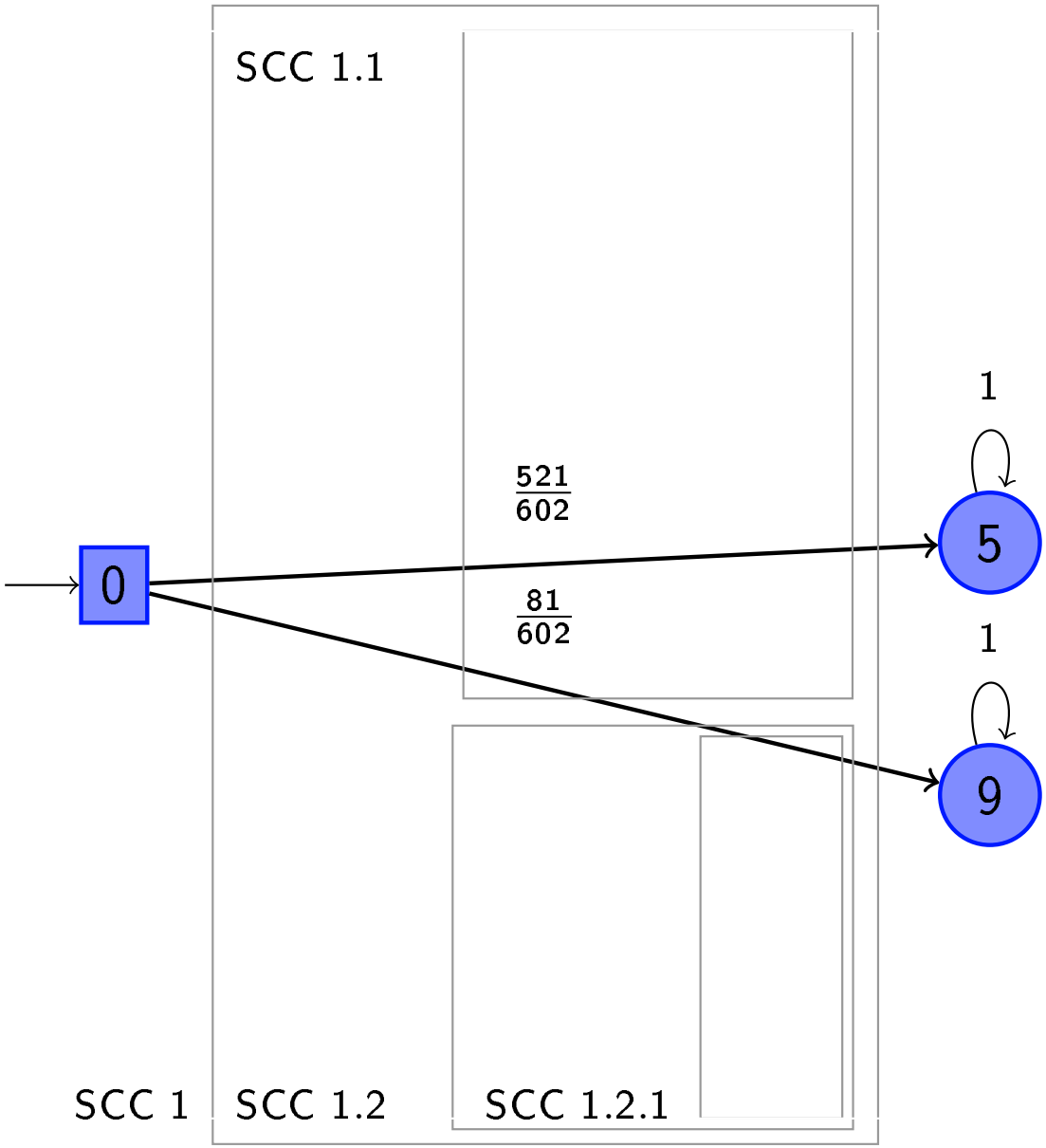
Abstraction of *SCC* 1.1

SCC-based Model Checking - Example



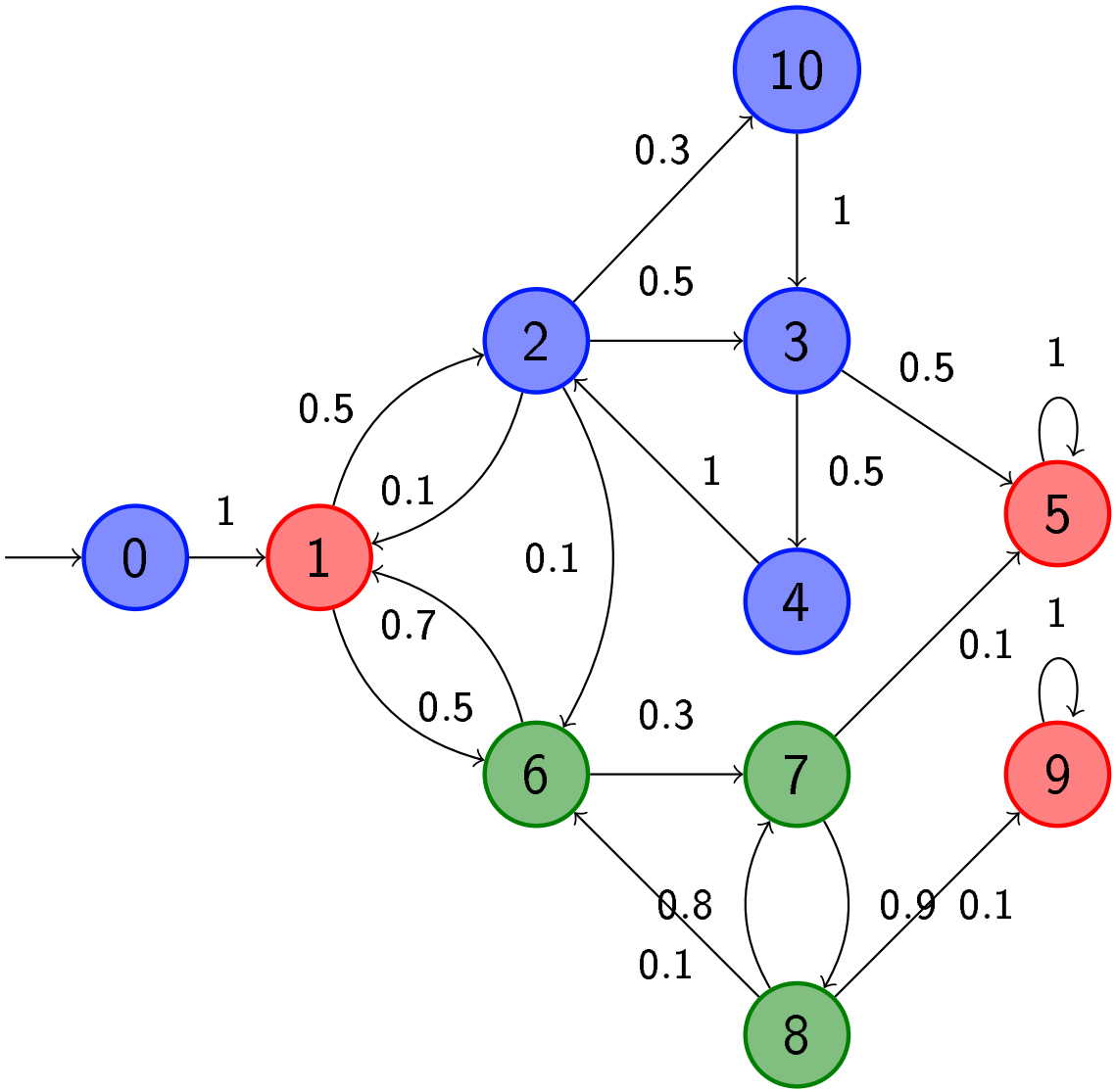
Abstraction of *SCC* 1

SCC-based Model Checking - Example



Path reduction

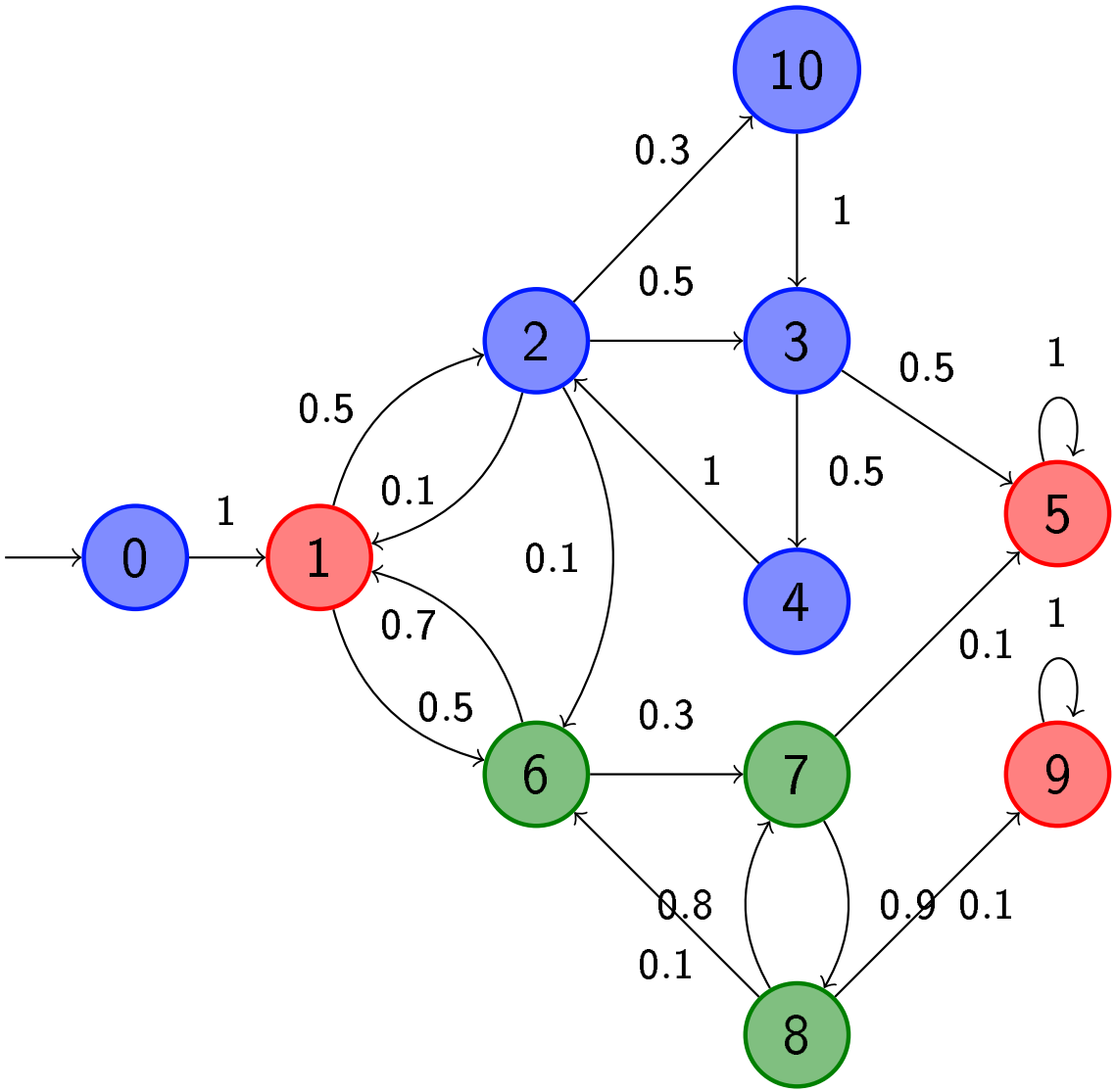
Formalisms



DTMC $M = (S, I, P, L)$

$S' \subseteq S$, $Out^M(S')$

Formalisms



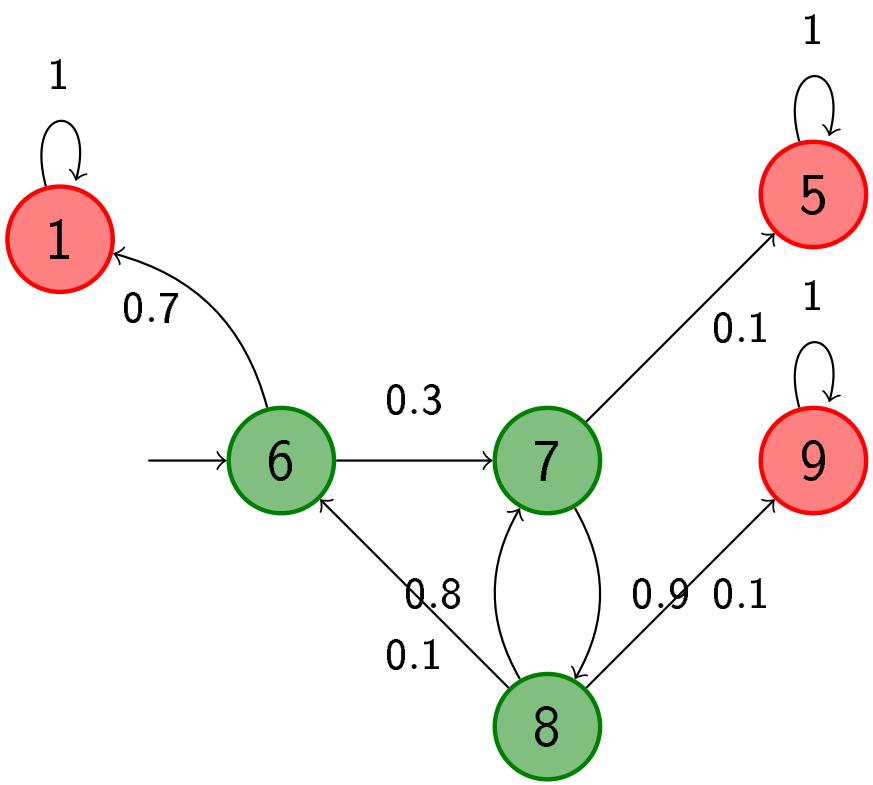
DTMC $M = (S, I, P, L)$

$S' \subseteq S$, $Out^M(S')$

Induced DTMC:

$M' = DTMC(S', M)$

Formalisms



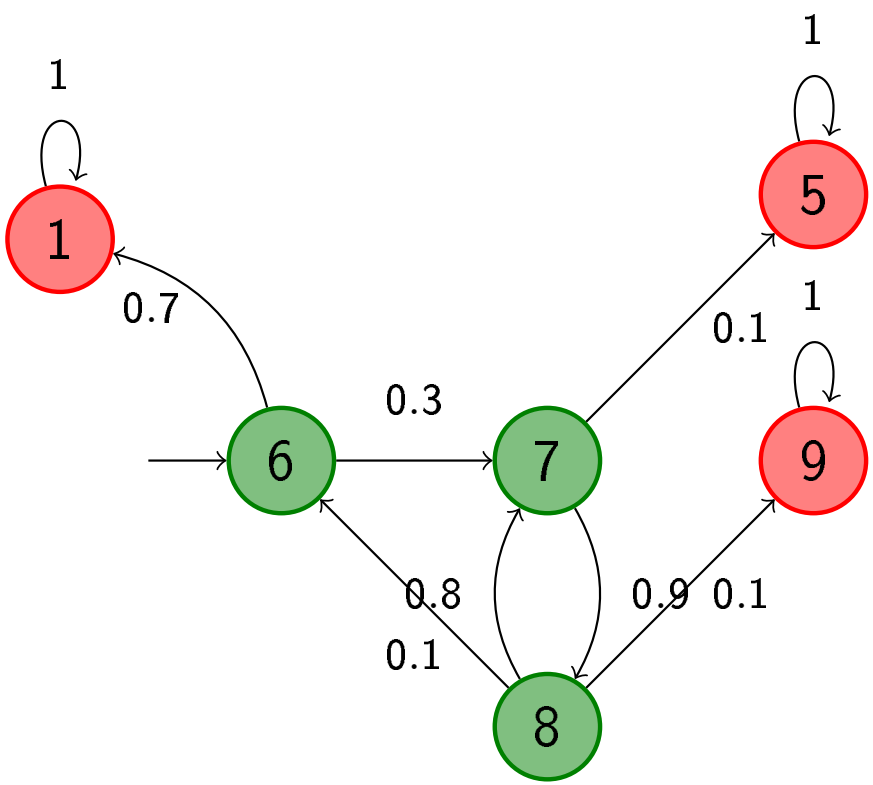
DTMC $M = (S, I, P, L)$

$S' \subseteq S$, $Out^M(S')$

Induced DTMC:

$M' = DTMC(S', M)$

Formalisms



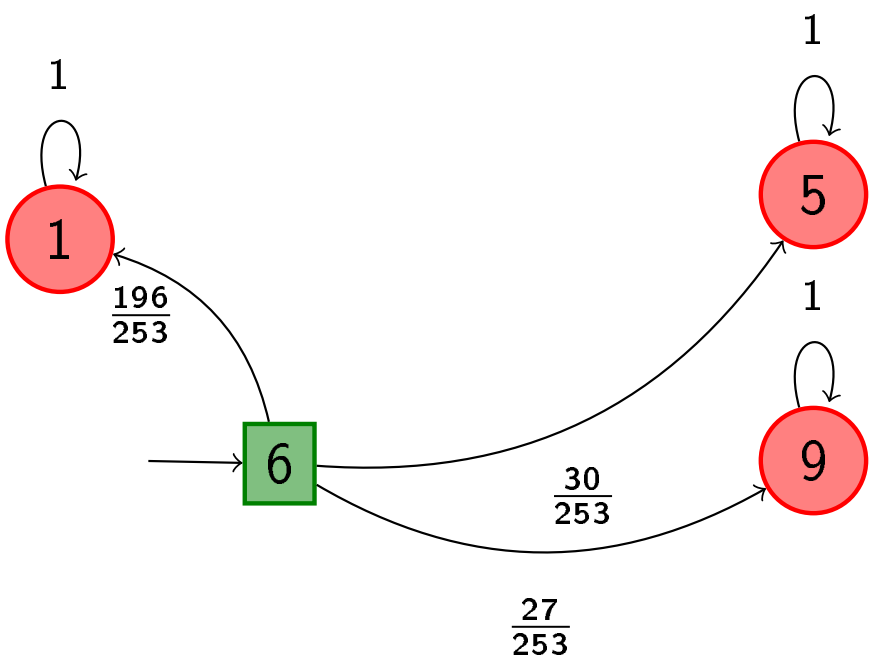
DTMC $M = (S, I, P, L)$

$S' \subseteq S$, $Out^M(S')$

Induced DTMC:
$M' = DTMC(S', M)$

Abstraction of M':
$Abs(M') = M_{abs}$

Formalisms



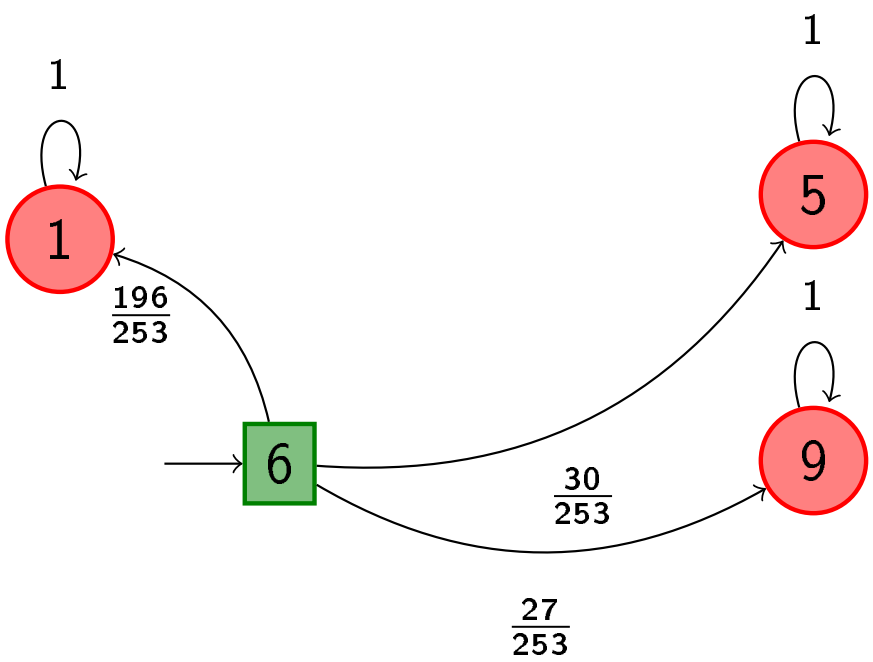
DTMC $M = (S, I, P, L)$

$S' \subseteq S$, $Out^M(S')$

Induced DTMC:
$M' = DTMC(S', M)$

Abstraction of M':
$Abs(M') = M_{abs}$

Formalisms



DTMC $M = (S, I, P, L)$

$S' \subseteq S$, $Out^M(S')$

Induced DTMC:

$M' = DTMC(S', M)$

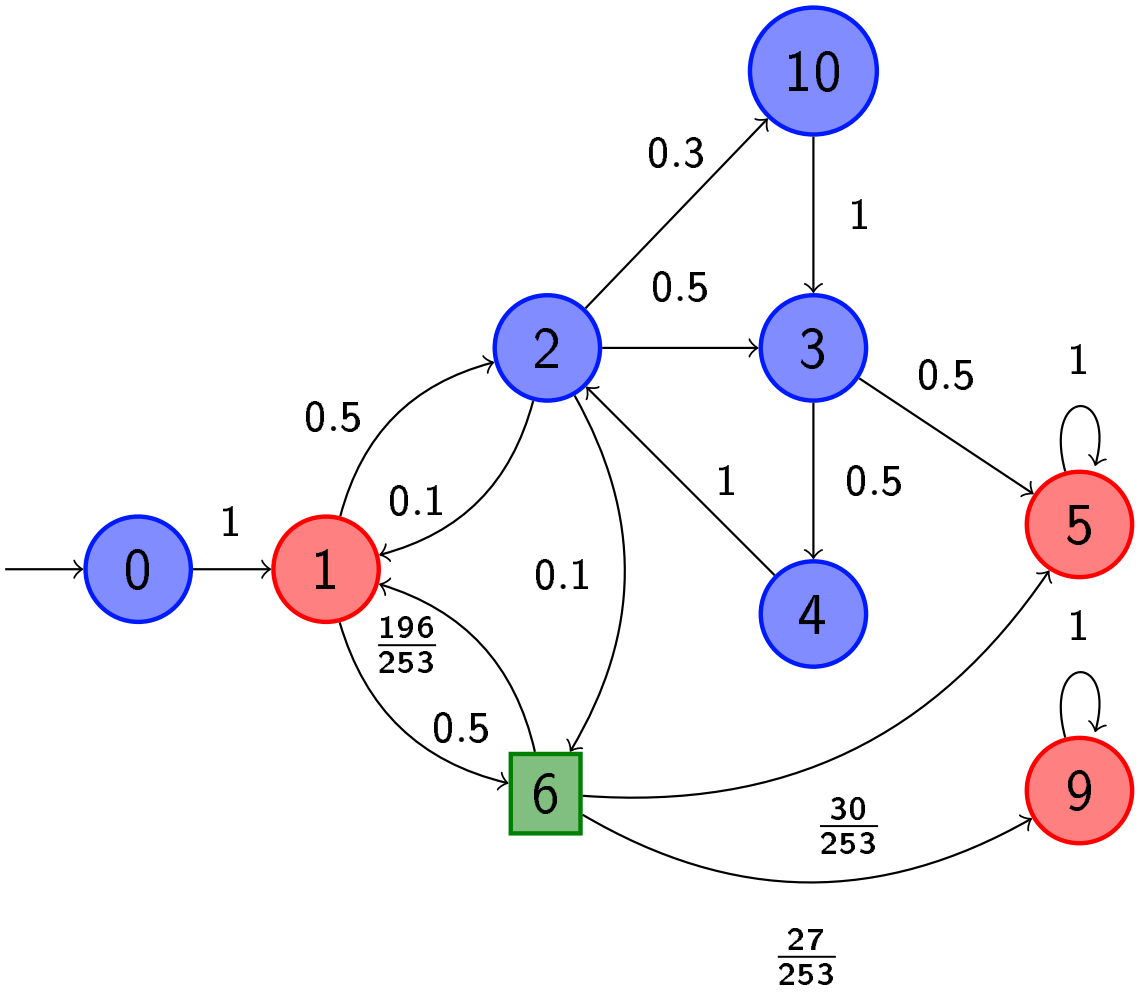
Abstraction of M':

$Abs(M') = M_{abs}$

Substitution:

$M[M_{abs}/M']$

Formalisms



DTMC $M = (S, I, P, L)$

$S' \subseteq S$, $Out^M(S')$

Induced DTMC:

$M' = DTMC(S', M)$

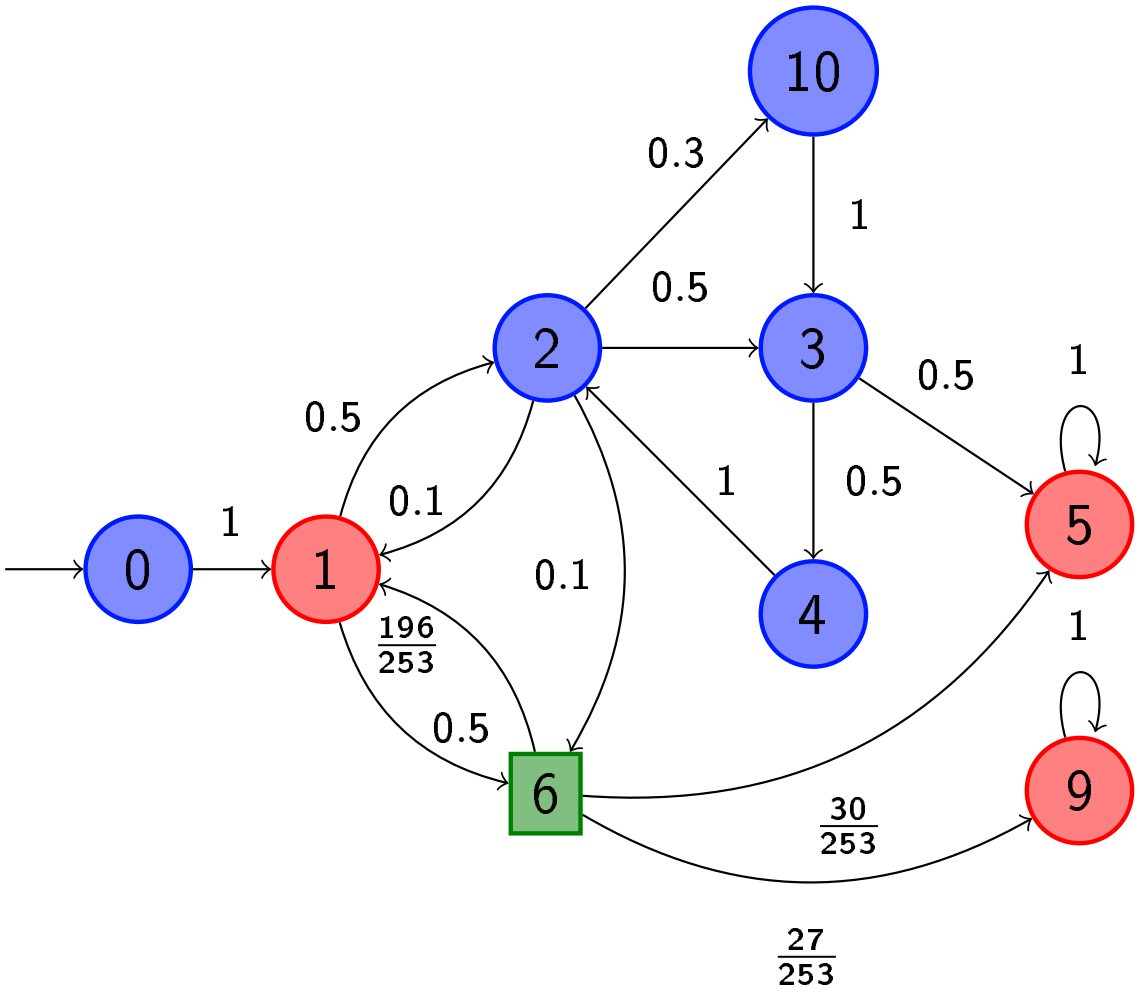
Abstraction of M':

$Abs(M') = M_{abs}$

Substitution:

$M[M_{abs}/M']$

Formalisms



DTMC $M = (S, I, P, L)$

$S' \subseteq S$, $Out^M(S')$

Induced DTMC:

$M' = DTMC(S', M)$

Abstraction of M':

$Abs(M') = M_{abs}$

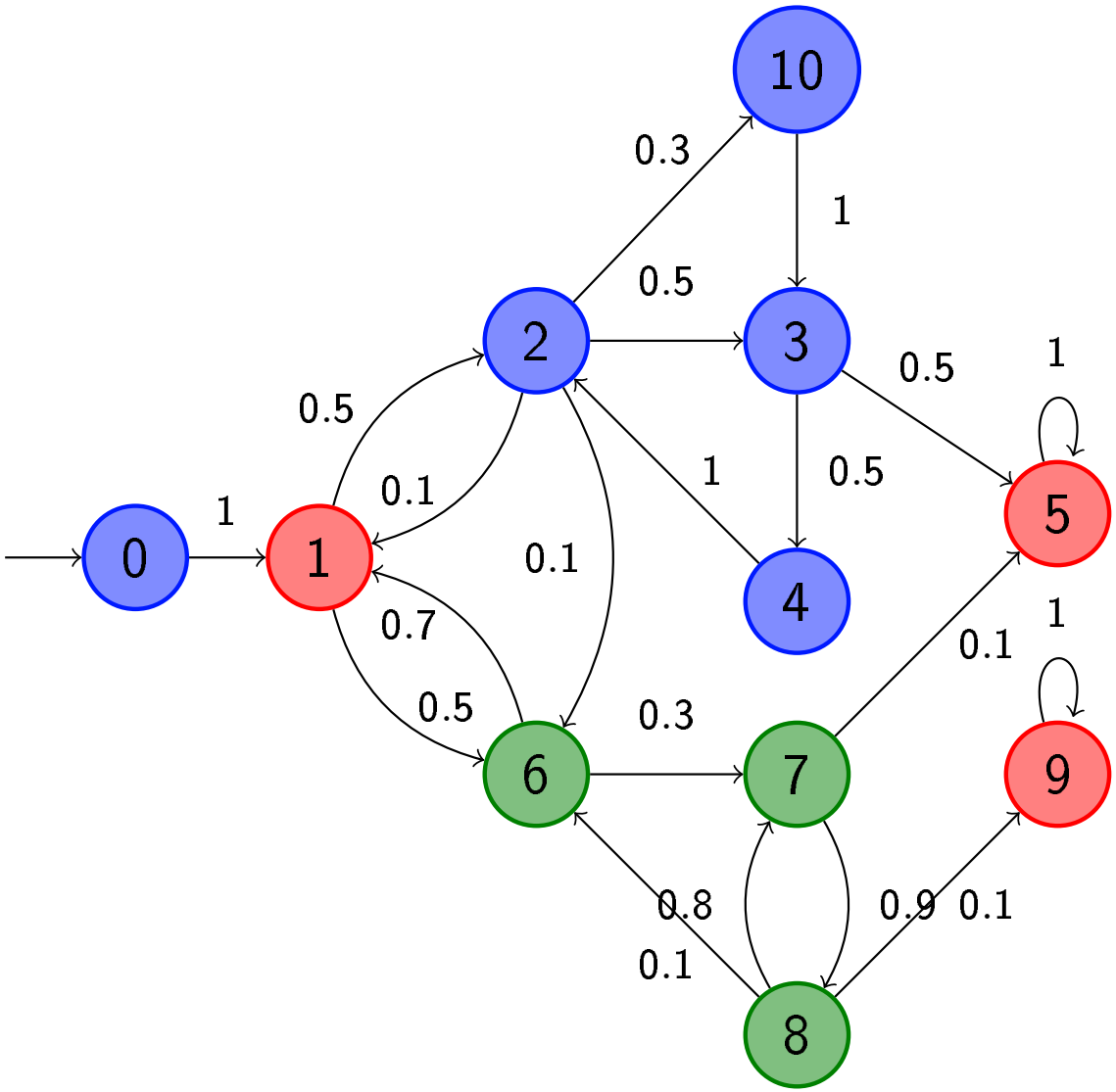
Substitution:

$M[M_{abs}/M']$

Concretization:

$M[M'/M_{abs}]$

Formalisms



DTMC $M = (S, I, P, L)$

$S' \subseteq S$, $Out^M(S')$

Induced DTMC:

$M' = DTMC(S', M)$

Abstraction of M':

$Abs(M') = M_{abs}$

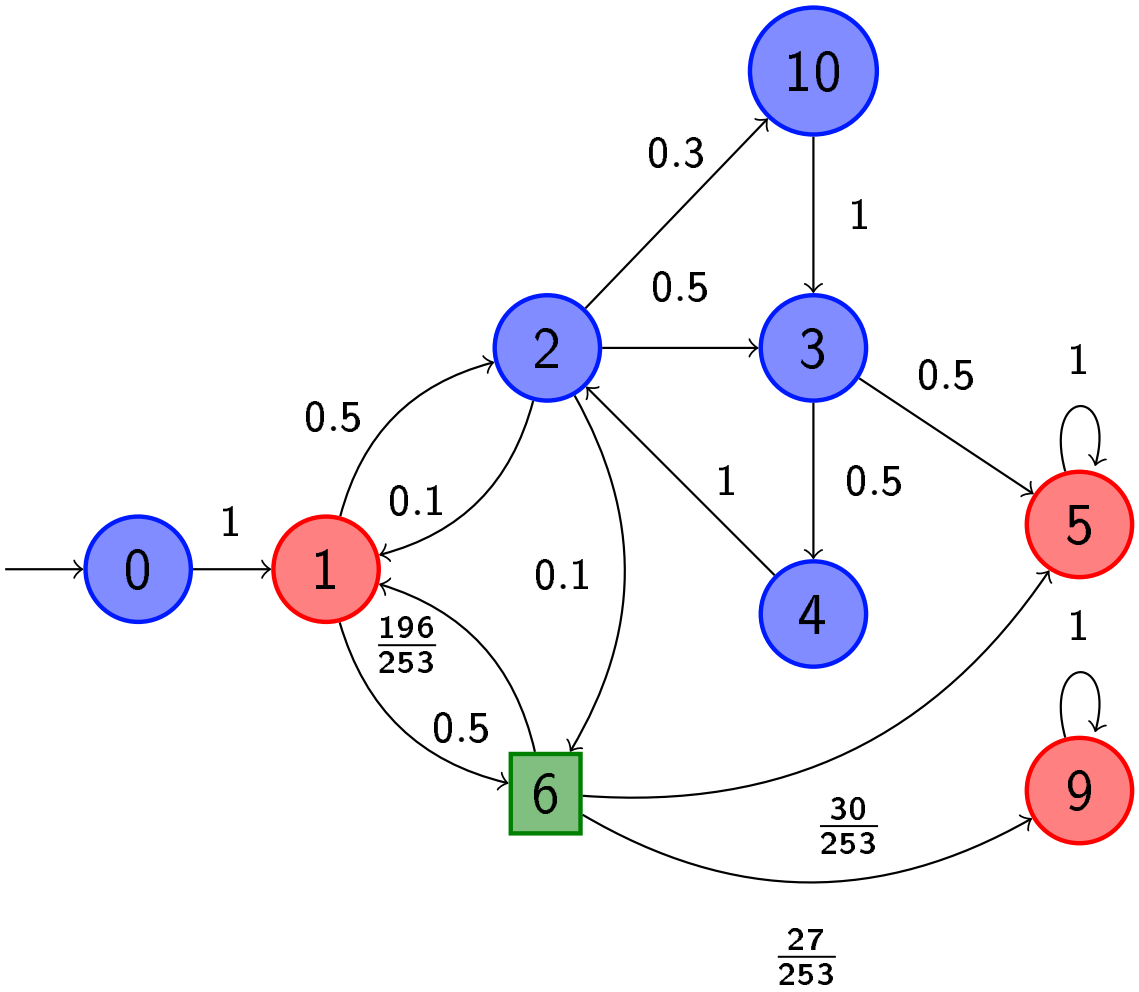
Substitution:

$M[M_{abs}/M']$

Concretization:

$M[M'/M_{abs}]$

Formalisms



DTMC $M = (S, I, P, L)$

$S' \subseteq S$, $Out^M(S')$

Induced DTMC:
$M' = DTMC(S', M)$

Abstraction of M':
$Abs(M') = M_{abs}$

Substitution:
$M[M_{abs}/M']$

Concretization:
$M[M'/M_{abs}]$

Pairs (M', M_{abs}) are saved during procedure.

Contents

1 Motivation

2 SCC-based Model Checking

3 Counterexample Generation

4 Implementation and Case Studies

5 Conclusion & Future Work

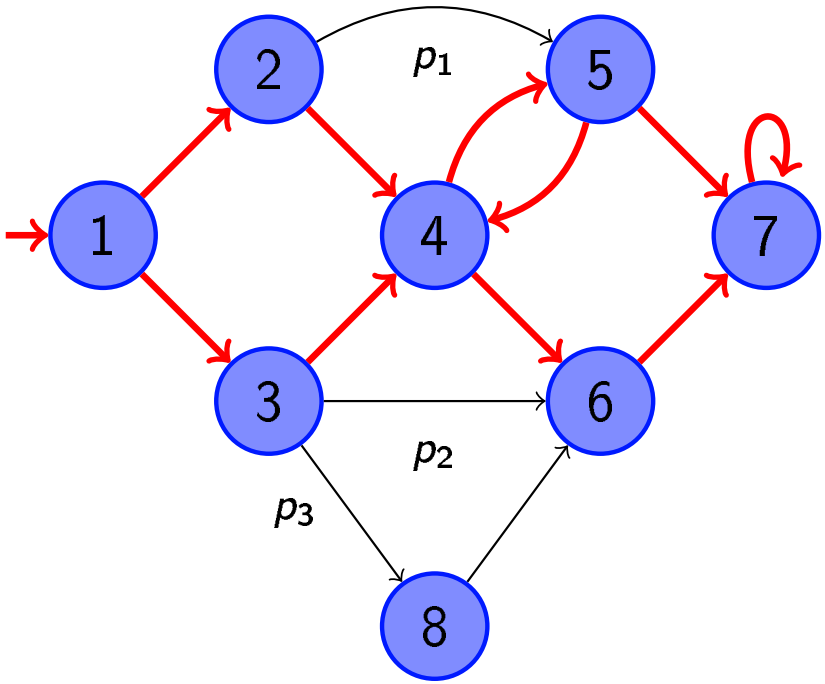
Counterexample Generation - Overview

- Hierarchical concretization of counterexamples
- Search for paths $\rightsquigarrow$ selection of edges
  - Global Search
  - Local Search
- Representation by a critical subsystem

Critical Subsystem

- Subsystem of a DTMC induced by a set of selected edges.
- Set of all paths shall violate a certain PCTL property.

DTMC $M = (S, I, P, L)$,
selection of edges $m \subseteq S \times S$

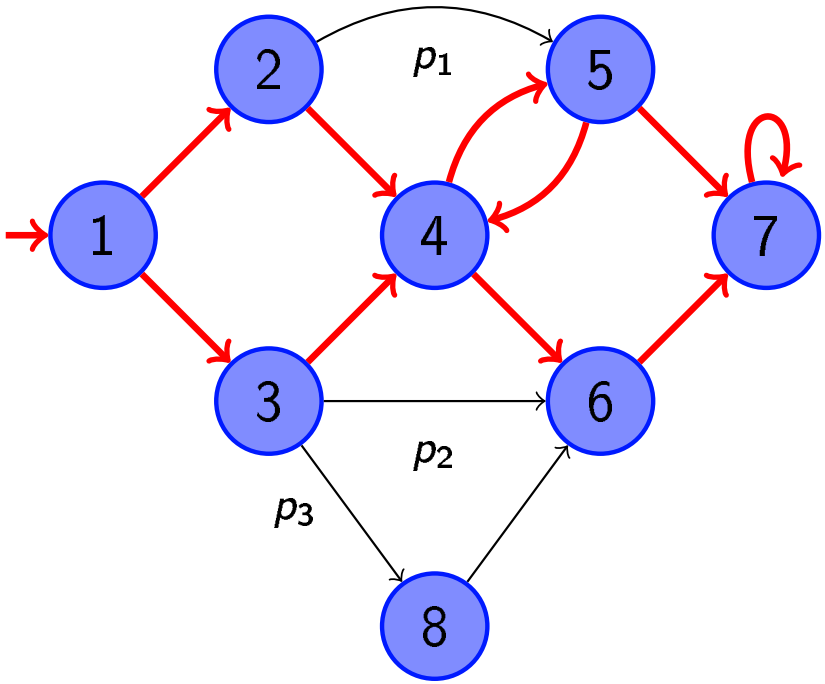


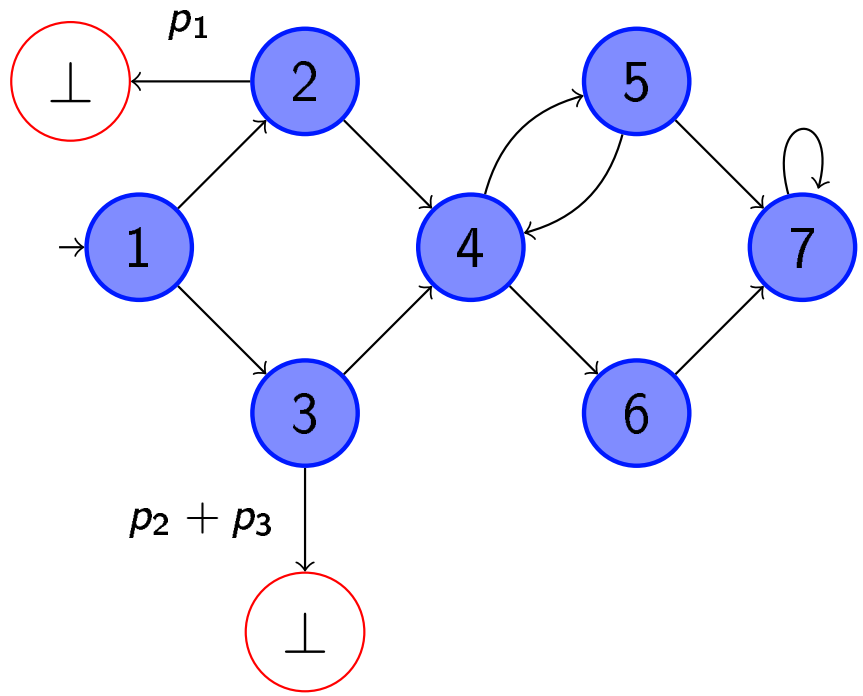
Critical Subsystem

- Subsystem of a DTMC induced by a set of selected edges.
- Set of all paths shall violate a certain PCTL property.

DTMC $M = (S, I, P, L)$,
selection of edges $m \subseteq S \times S$

$closure^M(m)$





Hierarchical Algorithm - Overview

- Input: Abstract DTMC M
- Concretize one or more states of M (using heuristics)
- Find a critical subsystem M_{ce}

- Global Search

- Local Search

Hierarchical Algorithm - Overview

- Input: Abstract DTMC M
- Concretize one or more states of M (using heuristics)
- Find a critical subsystem M_{ce}
  - Find a certain path π
  - Edges along π are selected $\rightsquigarrow$ m
  - Compute $closure_M(m)$
  - Search for more paths until $closure_M(m)$ has enough probability mass
- Global Search
- Local Search

Hierarchical Algorithm - Overview

- Input: Abstract DTMC M
- Concretize one or more states of M (using heuristics)
- Find a critical subsystem M_{ce}
  - Find a certain path π
  - Edges along π are selected $\rightsquigarrow$ m
  - Compute $closure_M(m)$
  - Search for more paths until $closure_M(m)$ has enough probability mass
- Global Search
  - Searches for the most probable paths through the current critical subsystem
- Local Search

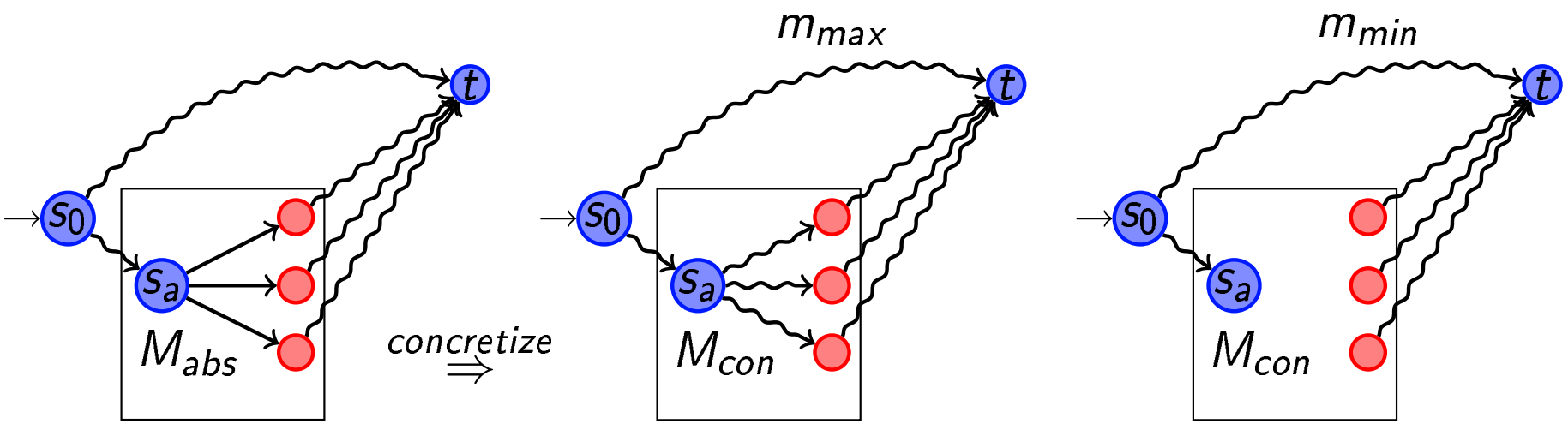
Hierarchical Algorithm - Overview

- Input: Abstract DTMC M
- Concretize one or more states of M (using heuristics)
- Find a critical subsystem M_{ce}
  - Find a certain path π
  - Edges along π are selected $\rightsquigarrow$ m
  - Compute $closure_M(m)$
  - Search for more paths until $closure_M(m)$ has enough probability mass
- Global Search
  - Searches for the most probable paths through the current critical subsystem
- Local Search
  - Searches for most probable path fragments that connect already selected paths
  - Search is restricted to the recently concretized parts

Edge Selection

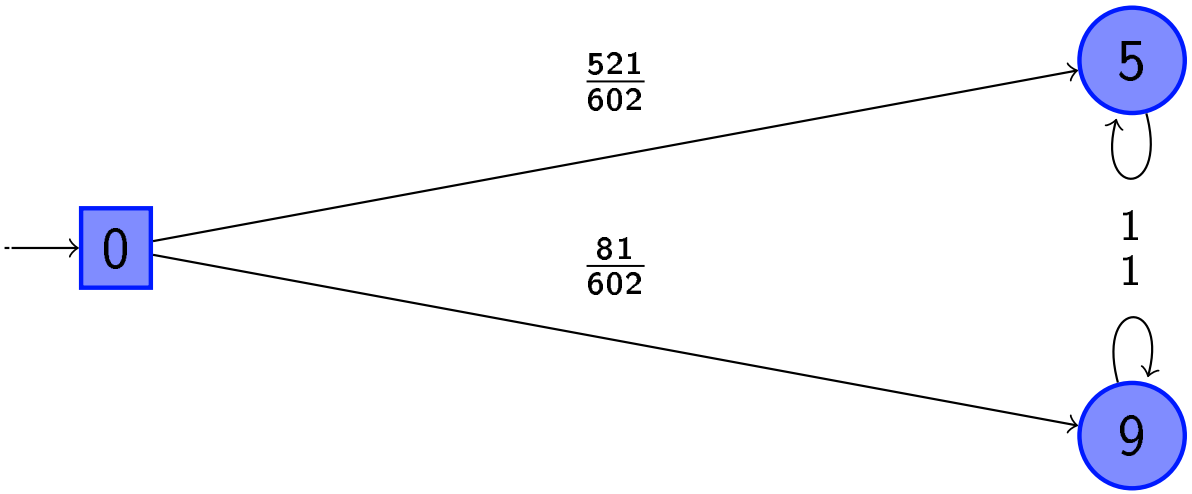
During one search iteration

- Selection m_{max} induces the maximum probability mass
- Selection m_{min} induces the minimum probability mass



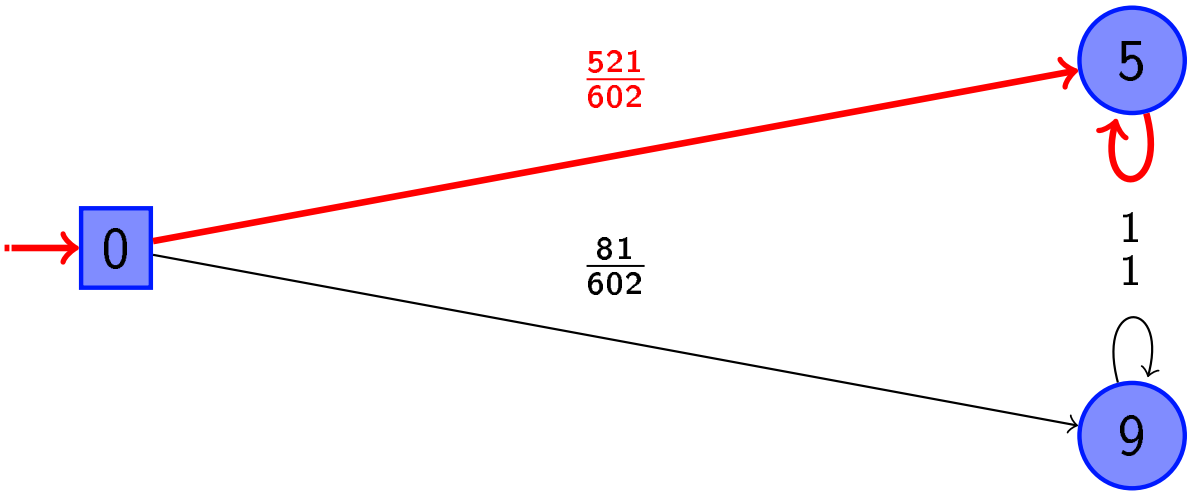
- m_{min} is extended by edges of m_{max}

Global Search - Example



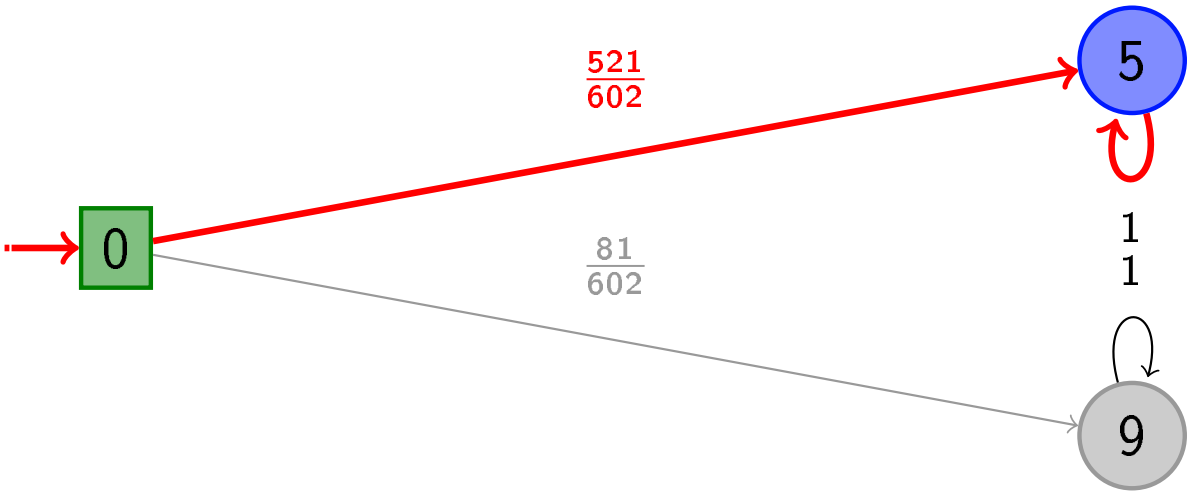
Input:
Abstract DTMC,
Target state 5,
Bound $\mathbb{P}_{<0.3}(\Diamond 5)$

Global Search - Example



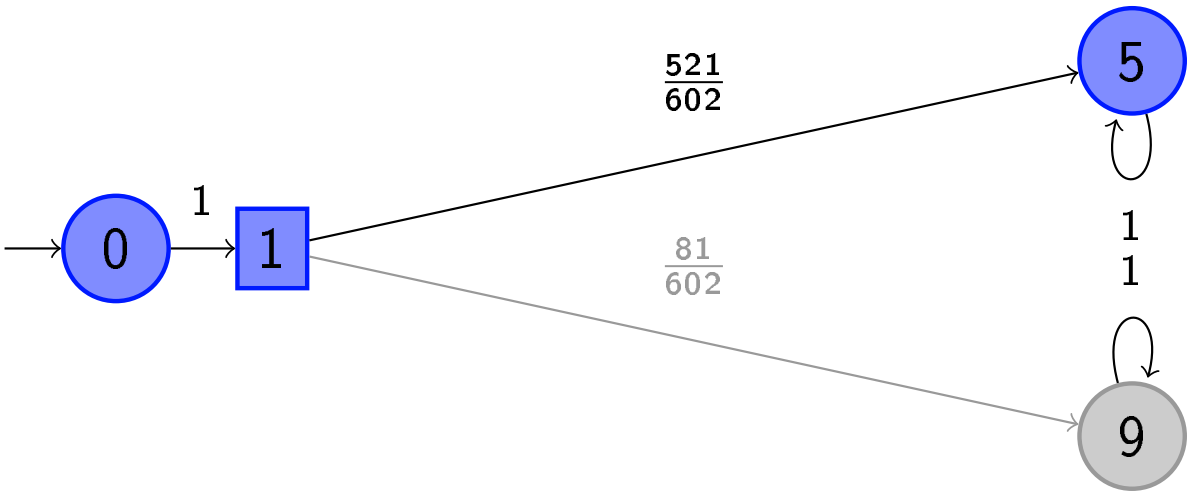
Initial most probable path

Global Search - Example



Concretize 0

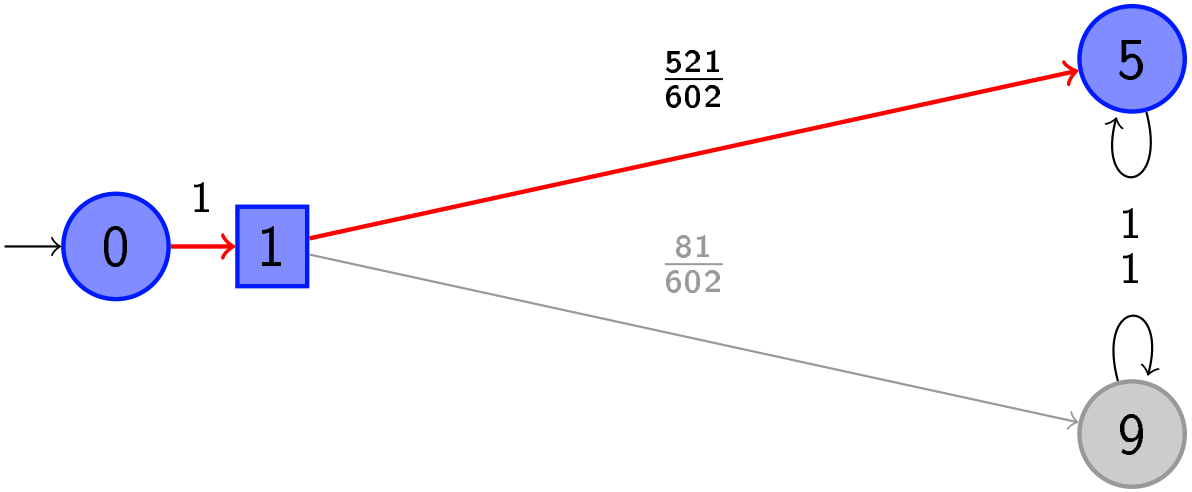
Global Search - Example



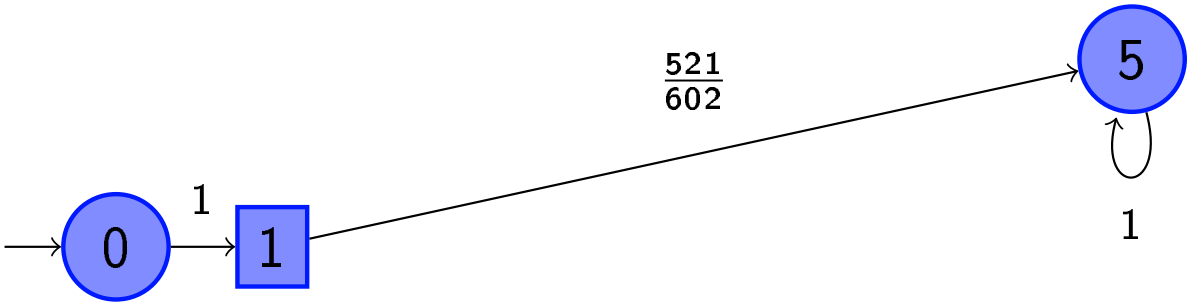
Selection m_{max}

Selection m_{min}

Global Search - Example

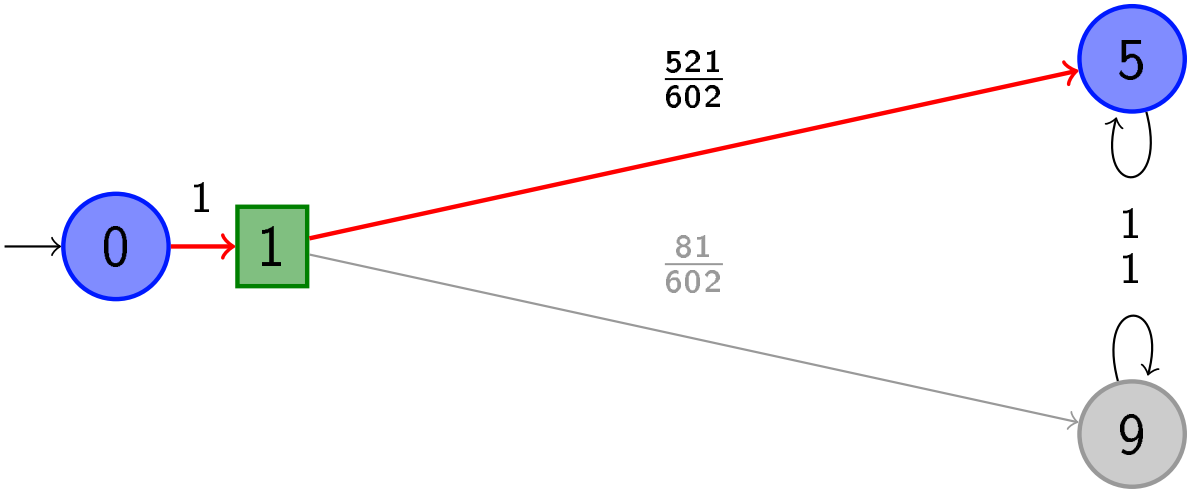


Search for most probable paths

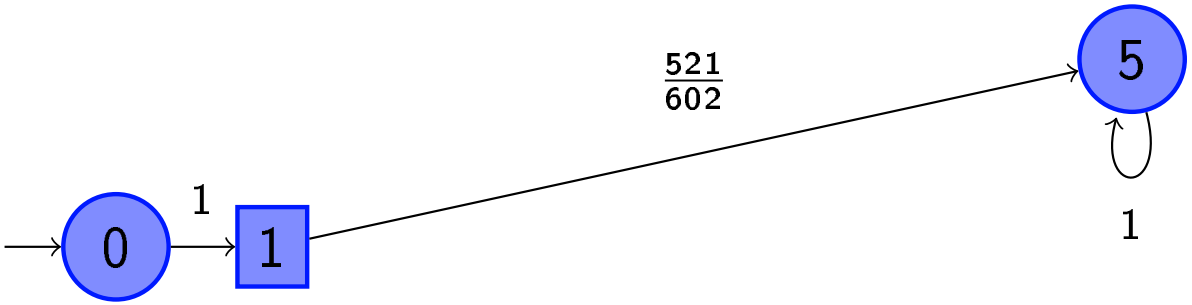


Current critical subsystem

Global Search - Example

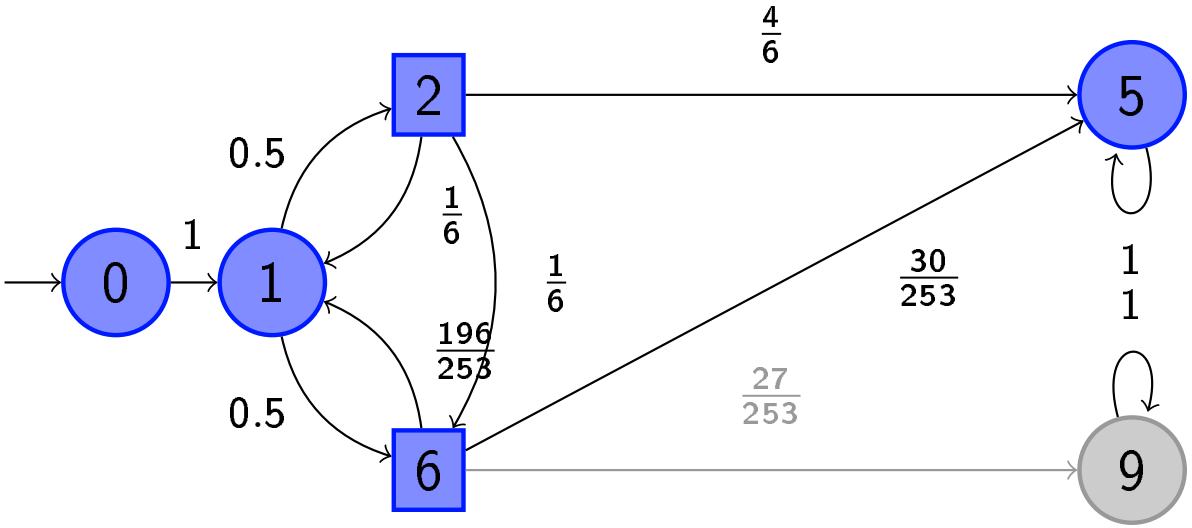


Concretize 1



Current critical subsystem

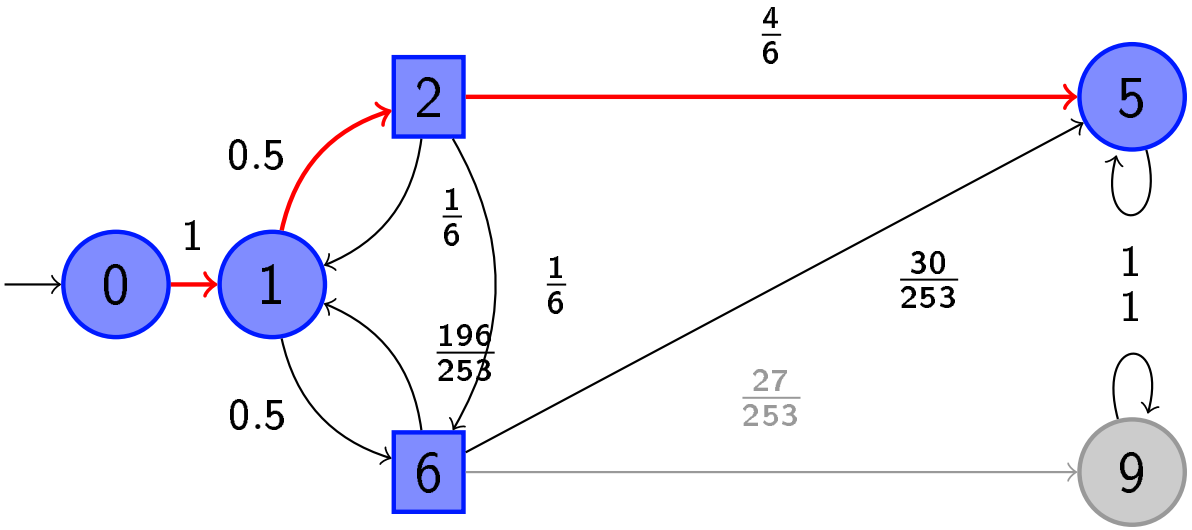
Global Search - Example



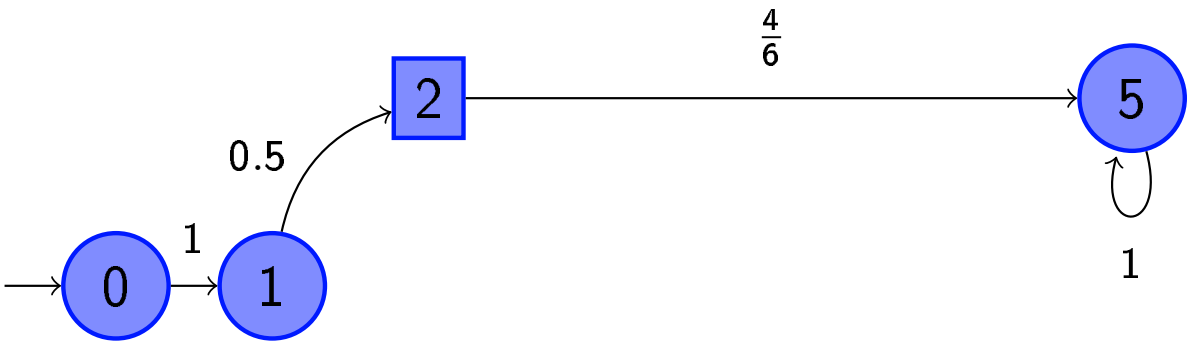
Selection m_{max}

Selection m_{min}

Global Search - Example

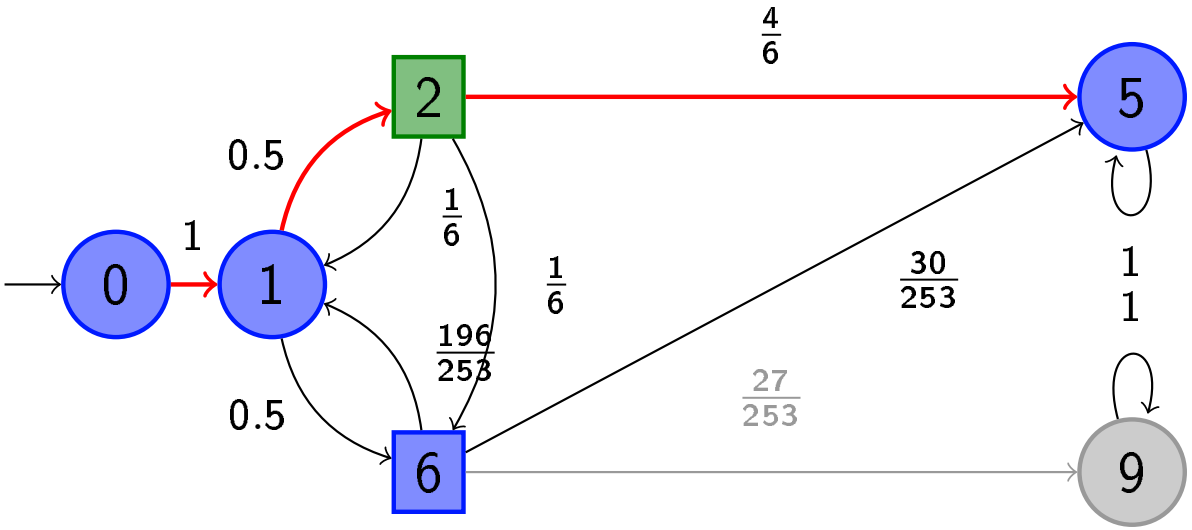


Search for most probable paths

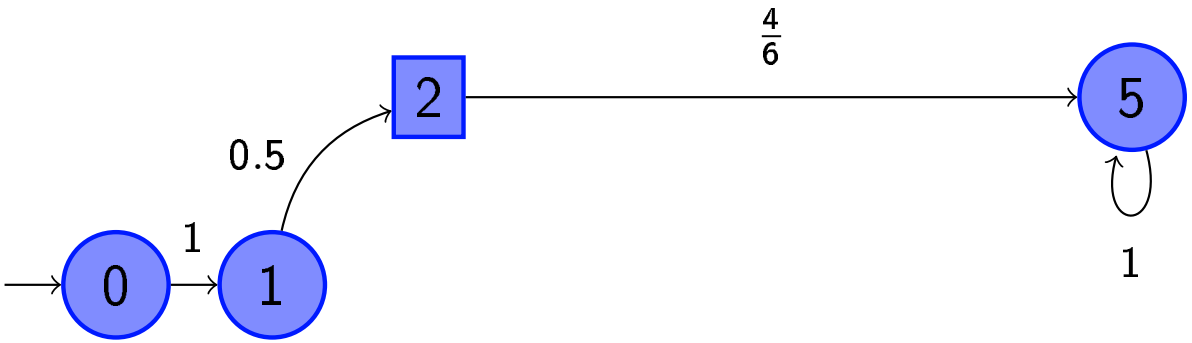


Current critical subsystem

Global Search - Example

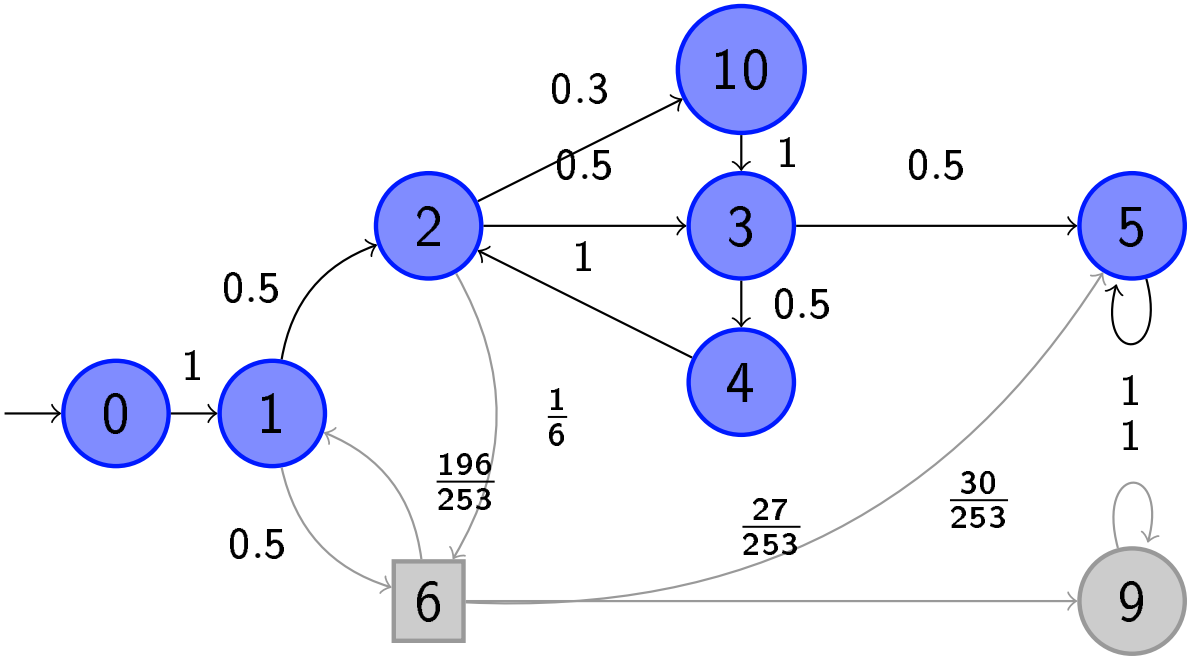


Concretize 2



Current critical subsystem

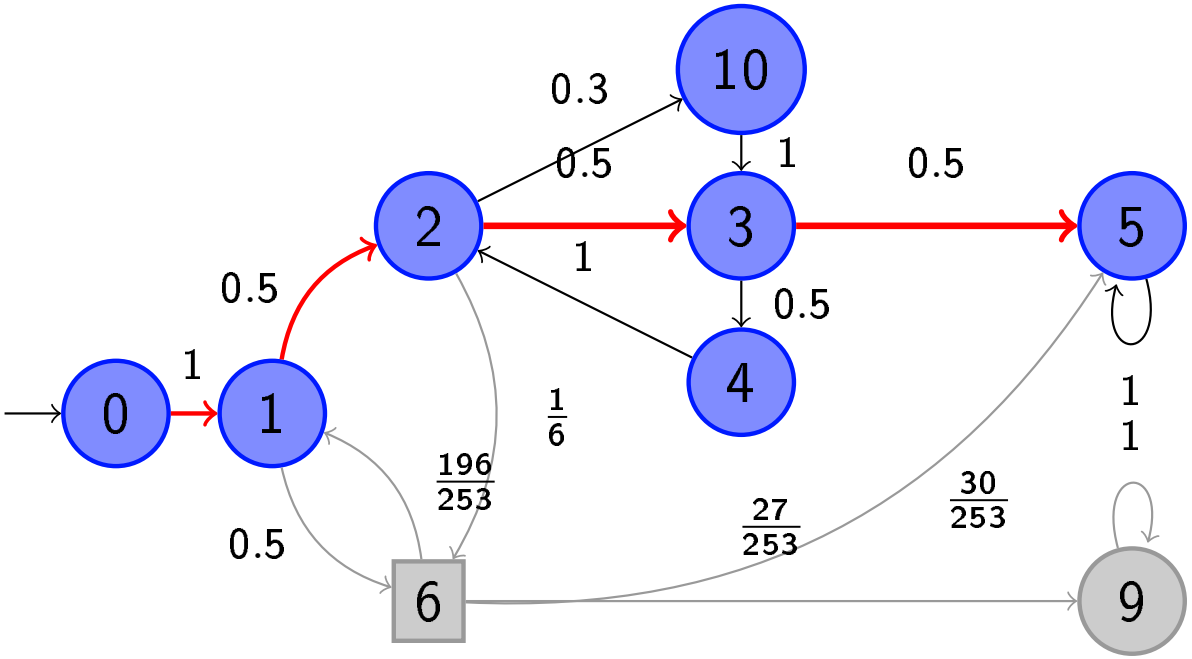
Global Search - Example



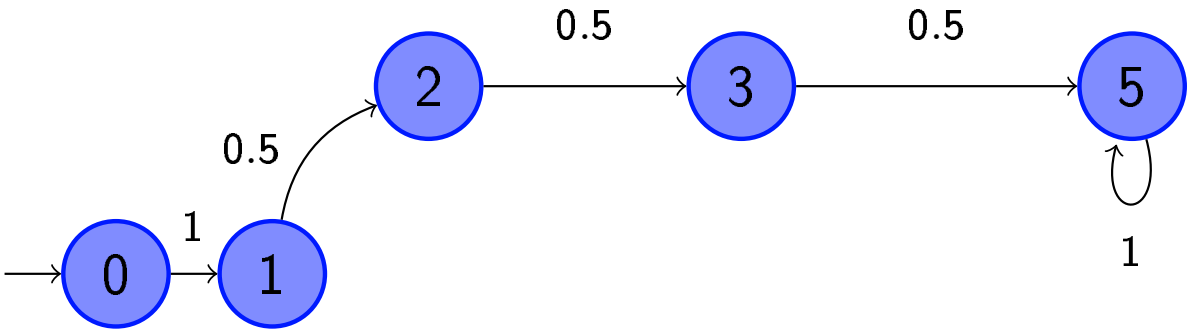
Selection m_{max}

Selection m_{min}

Global Search - Example

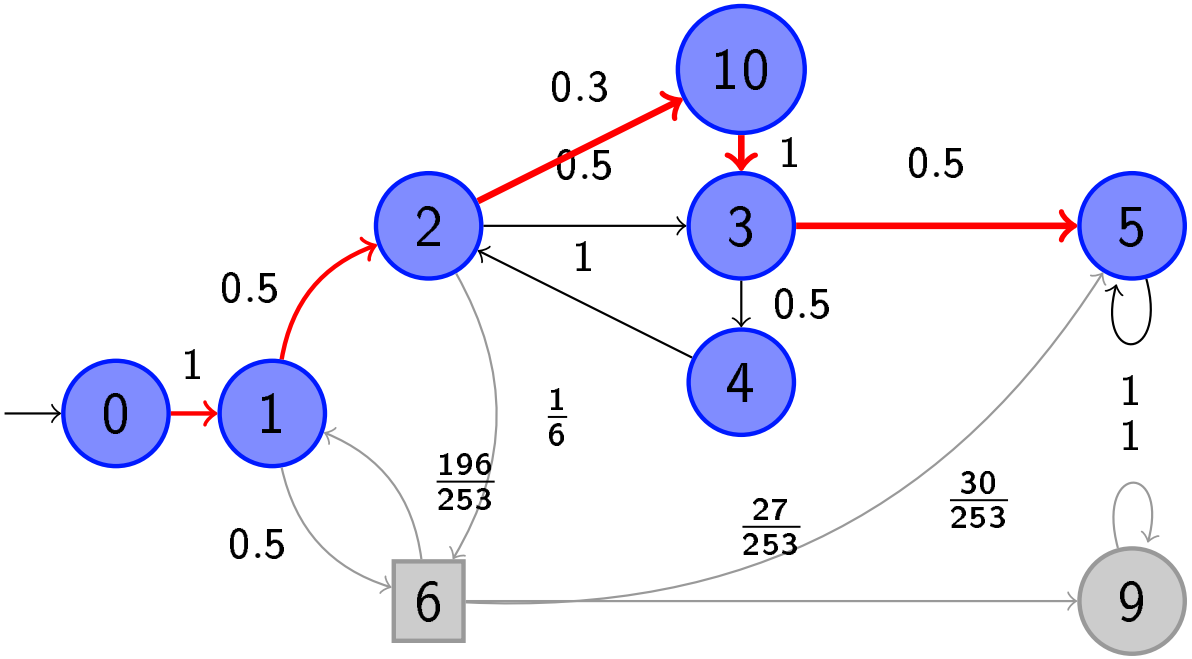


Search for most probable paths

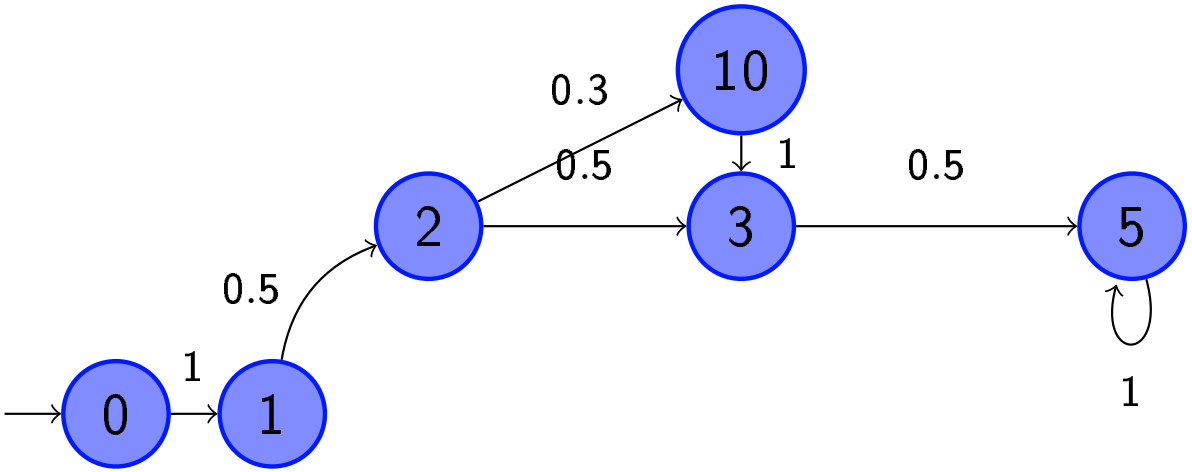


Probability mass: 0.125

Global Search - Example

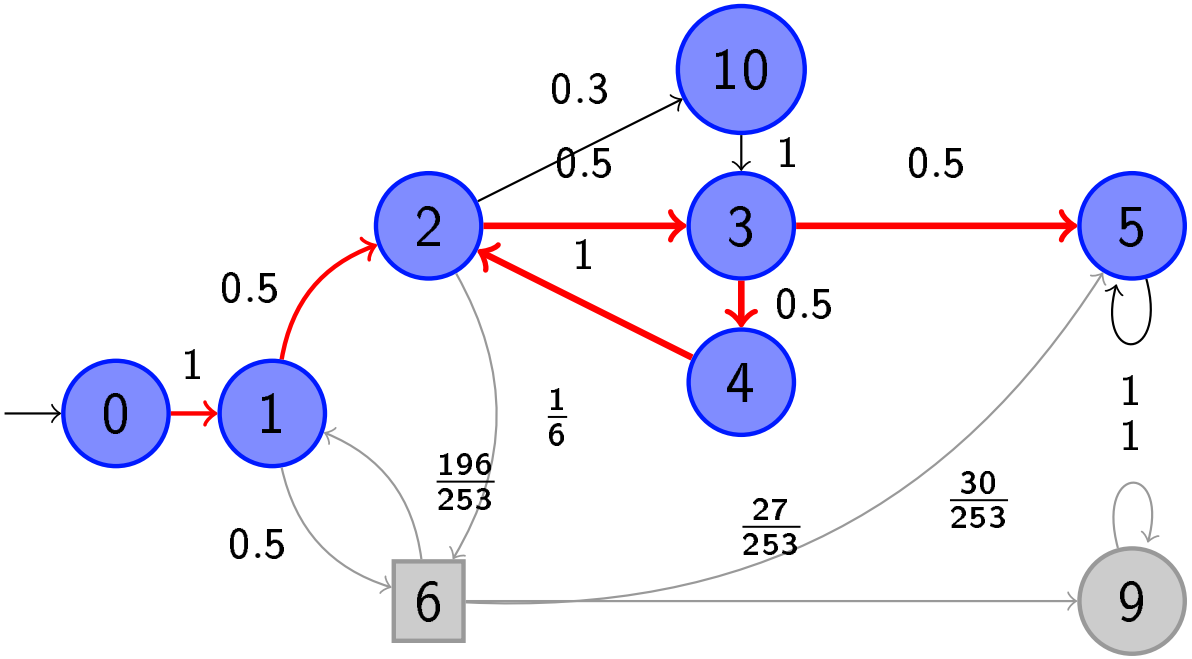


Search for most probable paths

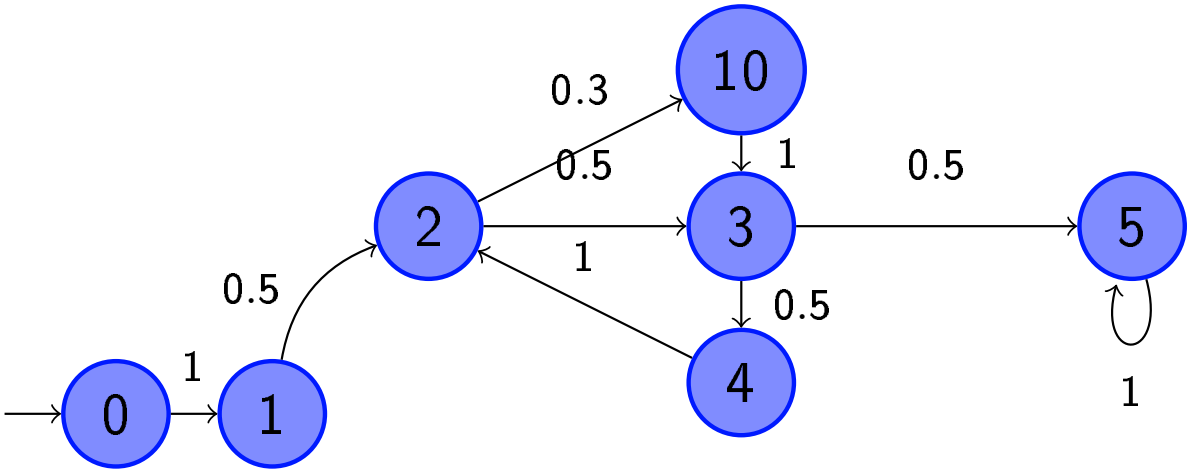


Probability mass: 0.2

Global Search - Example



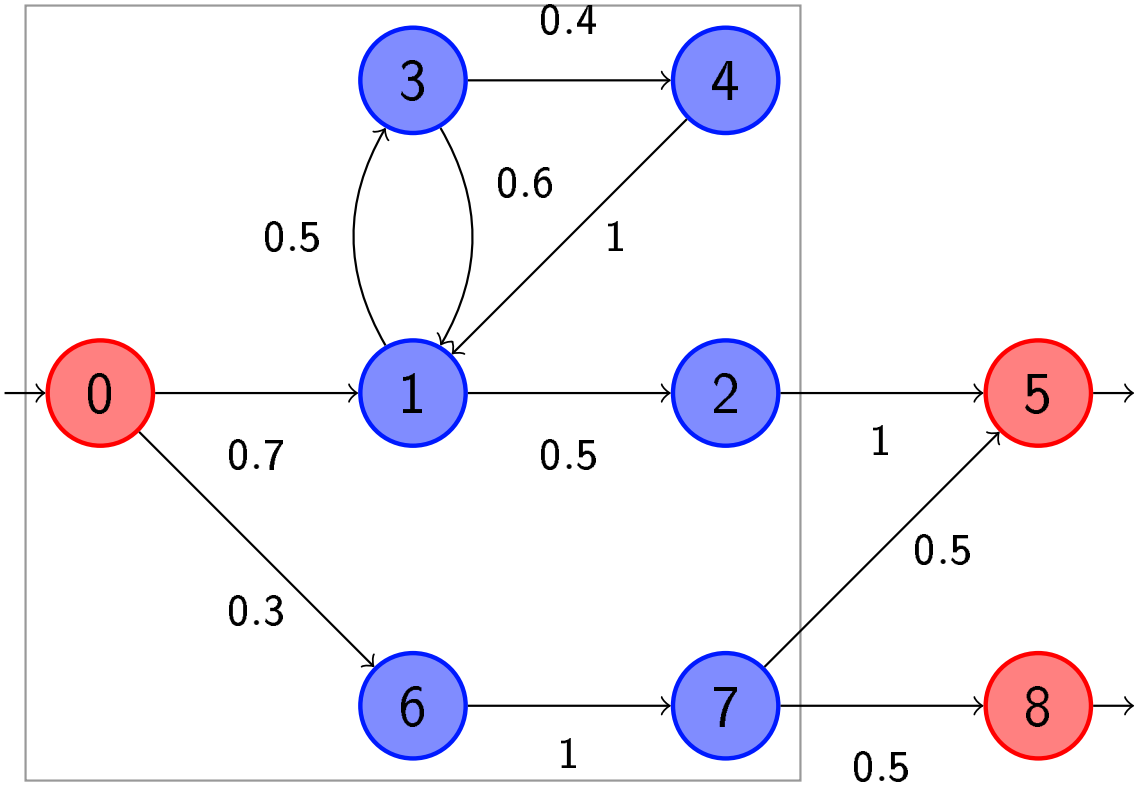
Search for most probable paths



Probability mass: $\frac{1}{3}$

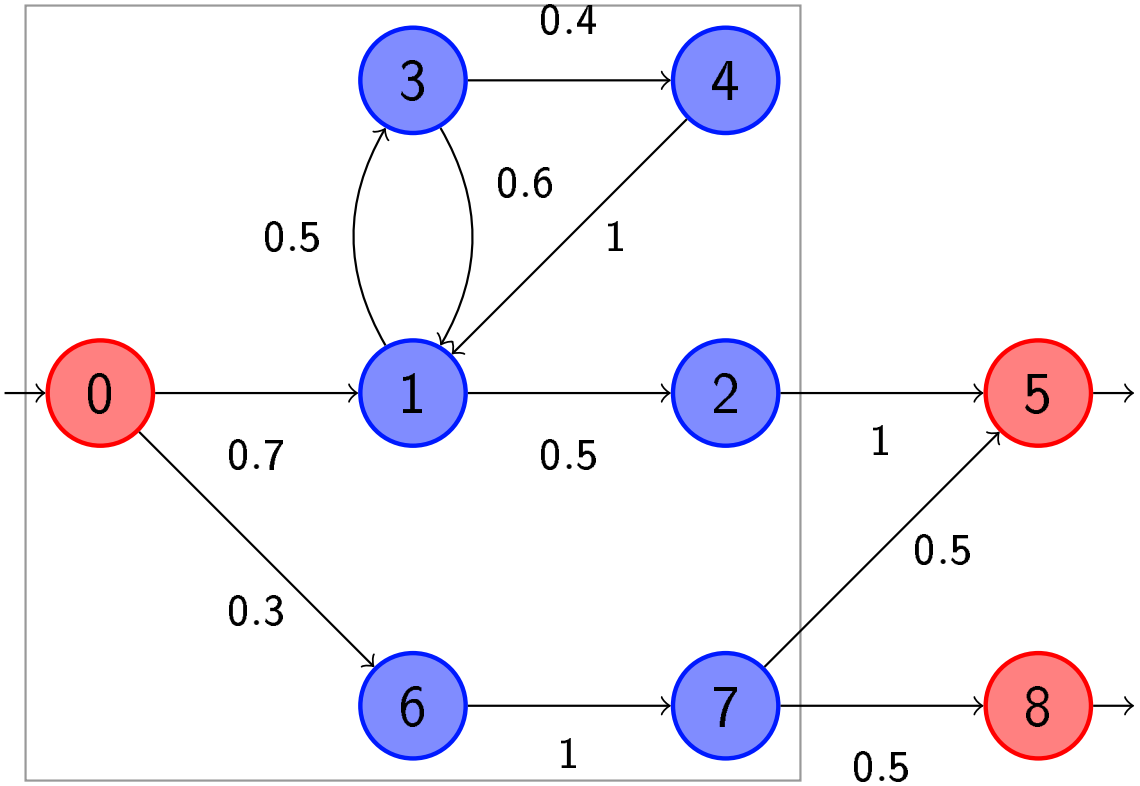
Local Search - Intuition

- Search for most probable path fragments
- "Blow up" paths



Local Search - Intuition

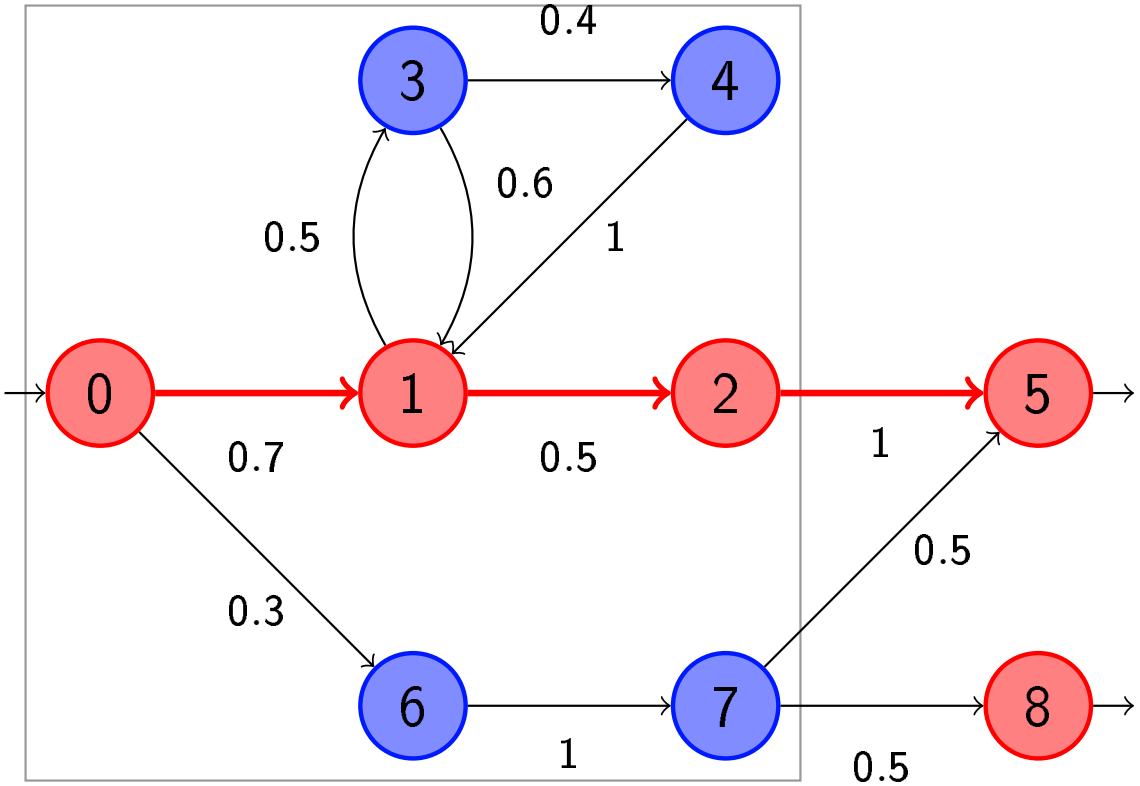
- Search for most probable path fragments
- "Blow up" paths



Connect red states along most probable paths

Local Search - Intuition

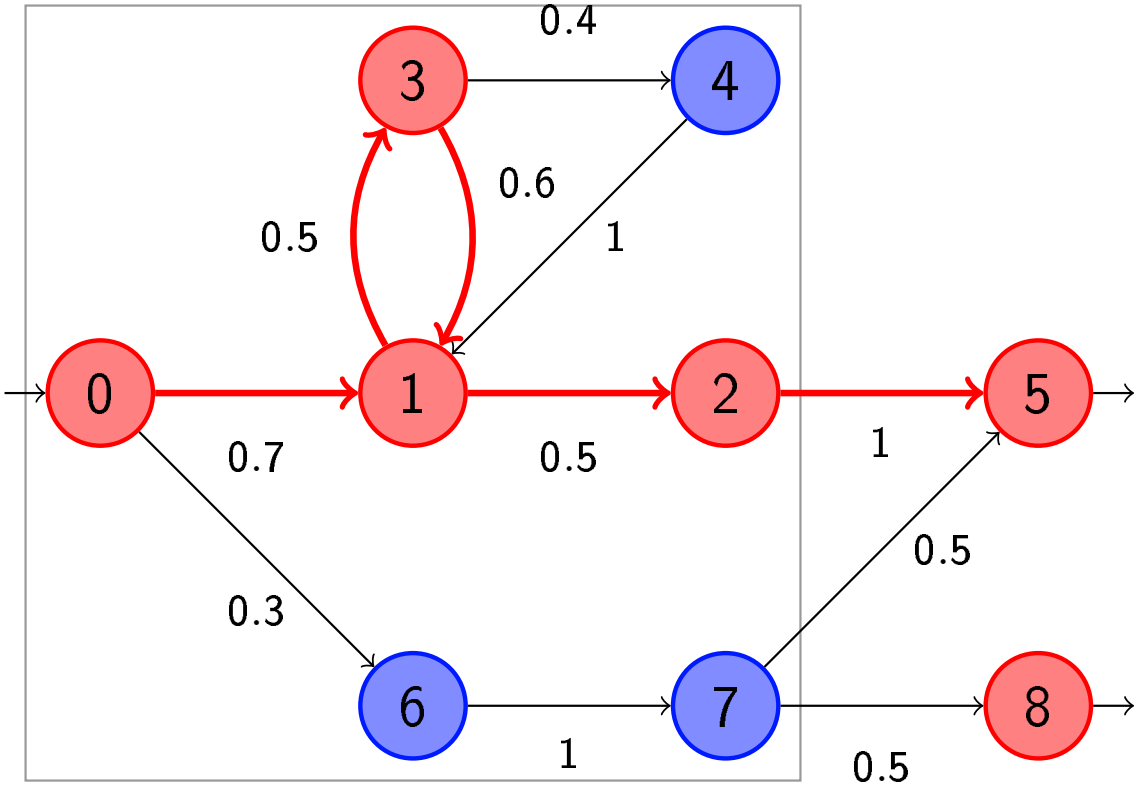
- Search for most probable path fragments
- "Blow up" paths



Connect red states along most probable paths

Local Search - Intuition

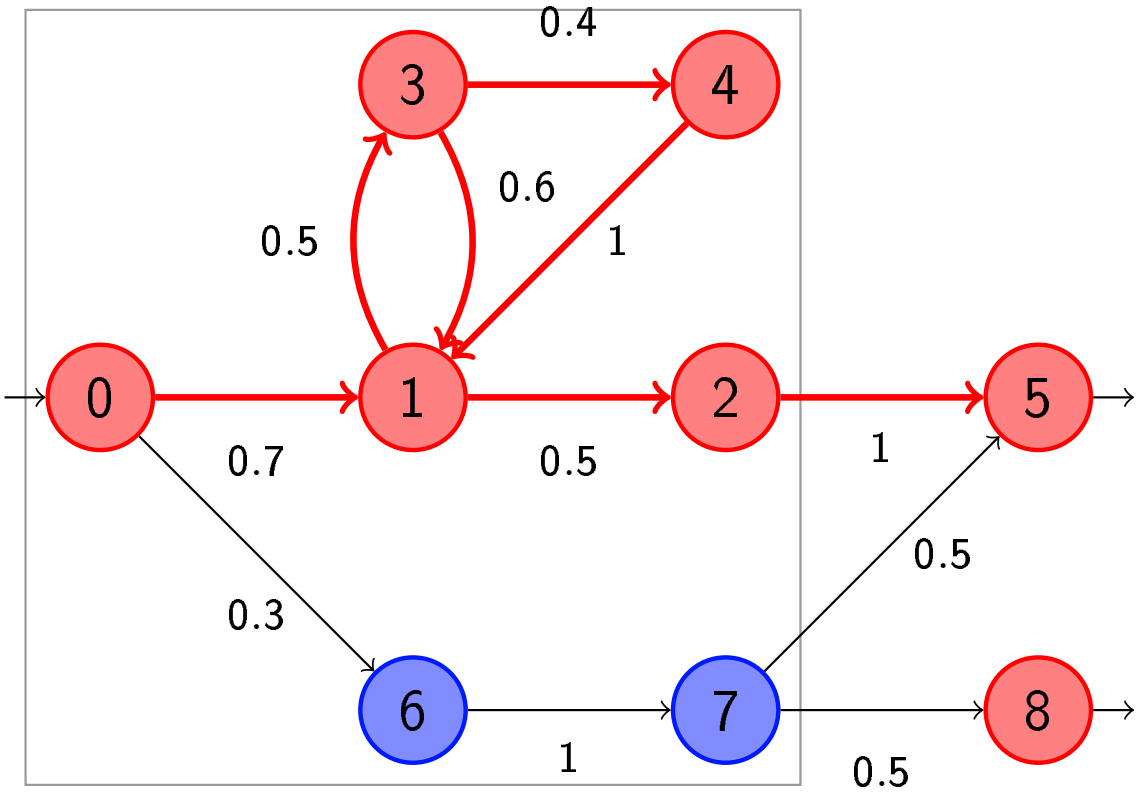
- Search for most probable path fragments
- "Blow up" paths



Connect red states along most probable paths

Local Search - Intuition

- Search for most probable path fragments
- "Blow up" paths



Connect red states along most probable paths

Critical subsystem induces probability mass of 0.7

Contents

1 Motivation

2 SCC-based Model Checking

3 Counterexample Generation

4 Implementation and Case Studies

5 Conclusion & Future Work

Implementation - Overview

- Prototype implementation in C++
- Two general approaches:
  - Hierarchical concretization of counterexamples
  - Counterexample search on the concrete DTMC
- Two different search algorithms:
  - Global Search
  - Local Search
- Closure computations on top of a search for the most probable paths
- Result is a (possibly abstract) subsystem that induces violation of the given property

Implementation Details

- ...are boring

Case Studies

Counterexample generation for case studies

- Synchronous leader election protocol *(Itai, Rodeh 1990)*
- Crowds protocol *(Reiter, Rubin 1998)*

The Crowds Protocol

- Protocol for anonymous communication in networks
- n users divided into good and bad members
- Random forwarding of messages to destination or other member
- A run of r message deliveries is modelled
- Our models are parameterized by n and r
- Fixed good-to-bad ratio, fixed forwarding probabilities
- Model Checking: Probabilitiy that a member is identified (i.e. not anonymous)?

Case Studies - Overview

Show applicability and advantages of

- Representation as critical subsystem
- Both search approaches
- The hierarchical approach
- Heuristics

Critical subsystems

States: 3515, Transitions: 6035 (Crowds 5 / 4)

Model checking result $\mathbb{P}(M) : 0.2346$

Probability bound p	0.15	0.15	0.15
counterexample as	set of paths	subsystem	subsystem
algorithm	path search	global	local
# computed paths	488644	958	98
# computed closures	-	182	98
# states M_{ce}	1071	632	171

- For $p \to \mathbb{P}(M)$ the size of counterexample path sets increases rapidly
- The critical subsystem automatically includes all loop iterations etc
  $\Longrightarrow$ many improbable paths are omitted

Critical subsystems

States: 3515, Transitions: 6035 (Crowds 5 / 4)

Model checking result $\mathbb{P}(M) : 0.2346$

Probability bound p	0.15	0.15	0.15
counterexample as	set of paths	subsystem	subsystem
algorithm	path search	global	local
# computed paths	**488644**	**958**	**98**
# computed closures	-	182	98
# states M_{ce}	1071	632	171

- For $p \to \mathbb{P}(M)$ the size of counterexample path sets increases rapidly
- The critical subsystem automatically includes all loop iterations etc
  $\Longrightarrow$ many improbable paths are omitted

Critical subsystems

States: 3515, Transitions: 6035 (Crowds 5 / 4)

Model checking result $\mathbb{P}(M) : 0.2346$

Probability bound p	0.15	0.15	0.15
counterexample as	set of paths	subsystem	subsystem
algorithm	path search	global	local
# computed paths	488644	958	98
# computed closures	-	182	98
# states M_{ce}	**1071**	**632**	**171**

- For $p \to \mathbb{P}(M)$ the size of counterexample path sets increases rapidly
- The critical subsystem automatically includes all loop iterations etc
  $\Longrightarrow$ many improbable paths are omitted

Critical subsystems

States: 3515, Transitions: 6035 (Crowds 5 / 4)

Model checking result $\mathbb{P}(M)$: **0.2346**

Probability bound p	0.15	0.15	0.15	**0.23**
counterexample as	set of paths	subsystem	subsystem	subsystem
algorithm	path search	global	local	global
# computed paths	488644	958	98	151639
# computed closures	-	182	98	623
# states M_{ce}	1071	632	171	1071

- For $p \to \mathbb{P}(M)$ the size of counterexample path sets increases rapidly
- The critical subsystem automatically includes all loop iterations etc
  $\Longrightarrow$ many improbable paths are omitted

Critical subsystems

States: 3515, Transitions: 6035 (Crowds 5 / 4)

Model checking result $\mathbb{P}(M) : 0.2346$

Probability bound p	0.15	0.15	0.15	0.23
counterexample as	set of paths	subsystem	subsystem	subsystem
algorithm	path search	global	local	global
# computed paths	488644	958	98	151639
# computed closures	-	182	98	623
# states M_{ce}	**1071**	632	171	**1071**

- For $p \to \mathbb{P}(M)$ the size of counterexample path sets increases rapidly
- The critical subsystem automatically includes all loop iterations etc
  $\Longrightarrow$ many improbable paths are omitted

(Closure-based) Global and local search

States: 18817, Transitions: 32677 (Crowds 5 / 6)

Model checking result $\mathbb{P}(M) : 0.4270$

Probability bound p	0.2	0.25
# paths global	3007	56657
# closures global	302	767
# states M_{ce}	663	2047
$\mathbb{P}(M_{ce})$	0.2002	0.2500
# paths/clos. local	202	798
# states M_{ce}	326	1439
$\mathbb{P}(M_{ce})$	0.2001	0.2508

- Global search finds many unnecessary paths (loops etc.)
- Local search finds only significant path fragments, but is expensive

(Closure-based) Global and local search

States: 18817, Transitions: 32677 (Crowds 5 / 6)

Model checking result $\mathbb{P}(M) : 0.4270$

Probability bound p	0.2	0.25
# paths global	3007	56657
# closures global	302	767
# states M_{ce}	663	2047
$\mathbb{P}(M_{ce})$	**0.2002**	**0.2500**
# paths/clos. local	202	798
# states M_{ce}	326	1439
$\mathbb{P}(M_{ce})$	**0.2001**	**0.2508**

- Global search finds many unnecessary paths (loops etc.)
- Local search finds only significant path fragments, but is expensive

(Closure-based) Global and local search

States: 18817, Transitions: 32677 (Crowds 5 / 6)

Model checking result $\mathbb{P}(M) : 0.4270$

Probability bound p	0.2	0.25
# paths global	**3007**	**56657**
# closures global	302	767
# states M_{ce}	663	2047
$\mathbb{P}(M_{ce})$	0.2002	0.2500
# paths/clos. local	**202**	**798**
# states M_{ce}	326	1439
$\mathbb{P}(M_{ce})$	0.2001	0.2508

- Global search finds many unnecessary paths (loops etc.)
- Local search finds only significant path fragments, but is expensive

The hierarchical approach

States: 18817, Transitions: 32677 (Crowds 5 / 6)
Model checking result $\mathbb{P}(M) : 0.4270$
(Hierarchical) SCCs: 756

Probabilitiy bound p	0.2	0.25	0.3	0.35
# refinement steps	13	21	23	35
# paths global	13525	55770	205362	3641675
# closures global	728	1729	2197	4944
# concretized SCCs	37	88	105	224
# states M_{ce}	457	1109	1363	3036

- Idea: Identify whole components of the system that form a counterexample
- The hierarchical approach does this by selecting / discarding SCCs

The hierarchical approach

States: 18817, Transitions: 32677 (Crowds 5 / 6)
Model checking result $\mathbb{P}(M) : 0.4270$
(Hierarchical) SCCs: 756

Probabilitiy bound p	**0.2**	**0.25**	**0.3**	**0.35**
# refinement steps	13	21	23	35
# paths global	13525	55770	205362	3641675
# closures global	728	1729	2197	4944
# concretized SCCs	37	88	105	224
# states M_{ce}	457	1109	1363	3036

- Idea: Identify whole components of the system that form a counterexample
- The hierarchical approach does this by selecting / discarding SCCs

The hierarchical approach

States: 18817, Transitions: 32677 (Crowds 5 / 6)
Model checking result $\mathbb{P}(M) : 0.4270$
(Hierarchical) SCCs: 756

Probabilitiy bound p	0.2	0.25	0.3	0.35
# refinement steps	13	21	23	35
# paths global	13525	55770	205362	3641675
# closures global	728	1729	2197	4944
# concretized SCCs	**37**	**88**	**105**	**224**
# states M_{ce}	457	1109	1363	3036

- Idea: Identify whole components of the system that form a counterexample
- The hierarchical approach does this by selecting / discarding SCCs

The hierarchical approach

States: 18817, Transitions: 32677 (Crowds 5 / 6)
Model checking result $\mathbb{P}(M) : 0.4270$
(Hierarchical) SCCs: 756

Probabilitiy bound p	0.2	0.25	0.3	0.35
# refinement steps	**13**	**21**	**23**	**35**
# paths global	**13525**	**55770**	**205362**	**3641675**
# closures global	**728**	**1729**	**2197**	**4944**
# concretized SCCs	37	88	105	224
# states M_{ce}	457	1109	1363	3036

- Idea: Identify whole components of the system that form a counterexample
- The hierarchical approach does this by selecting / discarding SCCs

Heuristics

- How many abstract states should be concretized per step?
- In which order should abstract states be concretized?

$\rightsquigarrow$ Heuristics:

- Select abstract states with most probable edges
- Number of concretized states parametrized by the number of available states

Heuristics

States: 18817, Transitions: 32677, Bound p: 0.2 (Crowds 5 / 6)

Search Type	global	global
conc-Heur	single	single
choose-Heur	prob	None
#Nodes	458	457
#Refinement	37	37
#Paths	38379	594881
#Closures	728	729

Heuristics:

- $\sqrt{}$: $\sqrt{k}$ out of k possible SCCs are concretized in 1 step
- single: Only 1 SCC is concretized per refinement step.
- prob: SCCs are concretized in order w.r.t. their average output probabilities

Heuristics

States: 18817, Transitions: 32677, Bound p: 0.2 (Crowds 5 / 6)

Search Type	global	global	global	global
conc-Heur	single	single	$\surd$	$\surd$
choose-Heur	prob	None	prob	None
#Nodes	458	457	457	457
#Refinement	37	37	13	10
#Paths	38379	594881	13525	912455
#Closures	728	729	728	730

Heuristics:

- $\surd$: $\sqrt{k}$ out of k possible SCCs are concretized in 1 step
- single: Only 1 SCC is concretized per refinement step.
- prob: SCCs are concretized in order w.r.t. their average output probabilities

Heuristics

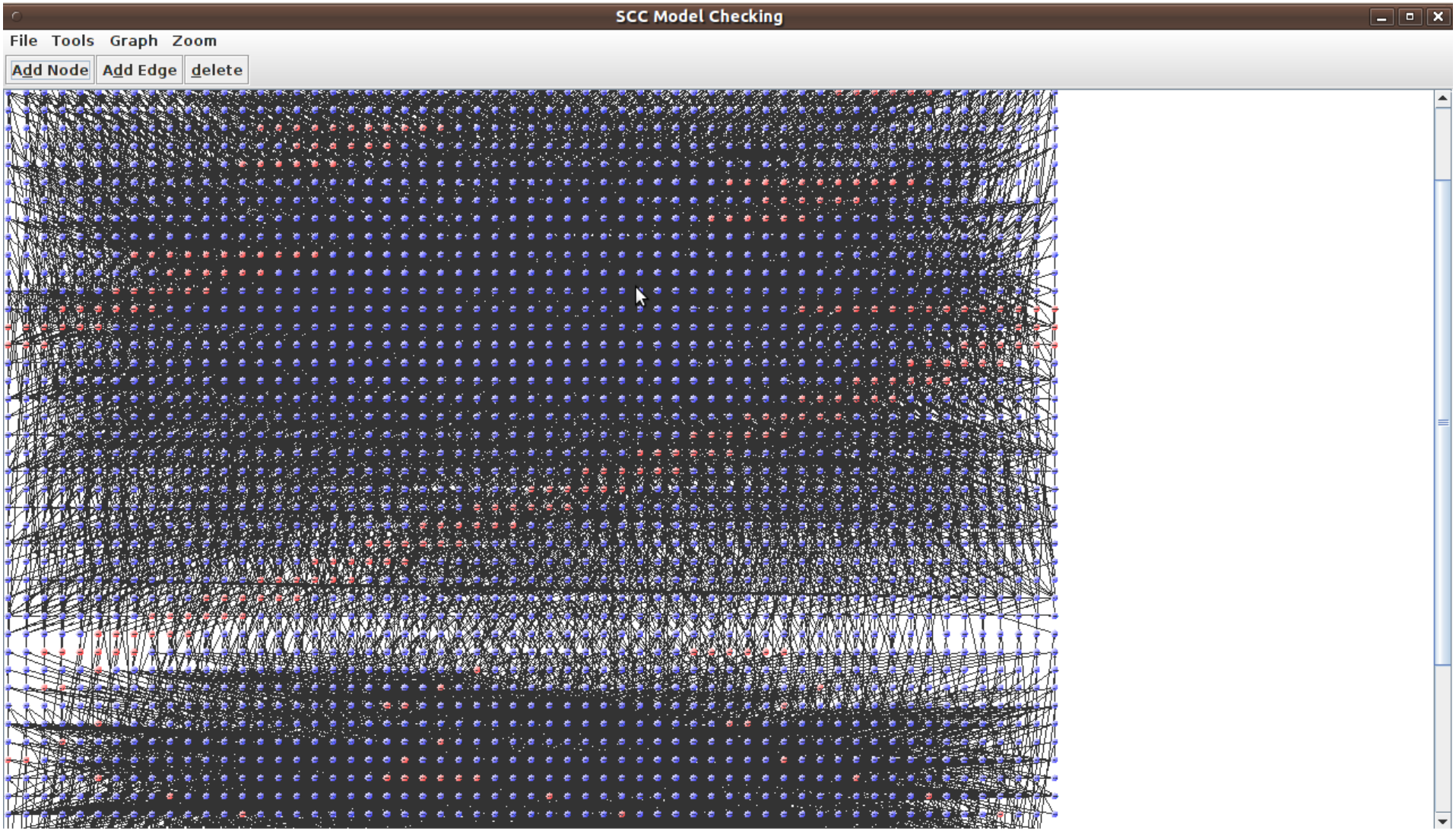
States: 18817, Transitions: 32677, Bound p: 0.2 (Crowds 5 / 6)

Search Type	global	global	global	global	local	local
conc-Heur	single	single	$\surd$	$\surd$	single	$\surd$
choose-Heur	prob	None	prob	None	None	None
#Nodes	458	457	457	457	347	319
#Refinement	37	37	13	10	28	9
#Paths	38379	594881	13525	912455	545	496
#Closures	728	729	728	730	545	496

Heuristics:

- $\surd$: $\sqrt{k}$ out of k possible SCCs are concretized in 1 step
- single: Only 1 SCC is concretized per refinement step.
- prob: SCCs are concretized in order w.r.t. their average output probabilities

GUI - In Progress



Contents

1 Motivation

2 SCC-based Model Checking

3 Counterexample Generation

4 Implementation and Case Studies

5 Conclusion & Future Work

Conclusion & Future Work

We did

We are working on

We will work on

Conclusion & Future Work

We did

- Model Checking for DTMCs with a resulting abstract system
- Counterexample Generation
  - both on abstract and concrete systems
  - with a very compact representation
  - with promising test results

We are working on

We will work on

Conclusion & Future Work

We did

- Model Checking for DTMCs with a resulting abstract system
- Counterexample Generation
  - both on abstract and concrete systems
  - with a very compact representation
  - with promising test results

We are working on

- a user interface
- an optimized implementation
- further heuristics

We will work on

Conclusion & Future Work

We did

- Model Checking for DTMCs with a resulting abstract system
- Counterexample Generation
  - both on abstract and concrete systems
  - with a very compact representation
  - with promising test results

We are working on

- a user interface
- an optimized implementation
- further heuristics

We will work on

- Case studies:
  - Retain information
  - Reduce size
- Complexity analysis

Thank you!