SCC-based Abstraction in Probabilistic Model Checking and Counterexample Generation

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

Theory of Hybrid Systems
RWTH Aachen

March 9, 2010

Contents

Contents

Model Checking & Counterexamples

- Classical DTMC Model Checking provides no diagnostic information.
- State-of-the-art counterexample generation needs the Model Checking result.
- Goal: Development of a Model Checking Algorithm with simultaneous Counterexample Generation
  - Counterexamples may consist of a big number of paths.
  - $\rightsquigarrow$ Abstract but refineable counterexamples
  - $\rightsquigarrow$ Save information during model checking and abstraction.

Contents

SCC-based Abstraction - Summary

Observation

- Paths through a non-bottom-SCC will leave it with probability 1.
- For unbounded reachability we look at the whole probability mass provided by SCCs.

Idea

- Reduce SCC to abstract states and edges carrying the whole probability mass.

Approaches

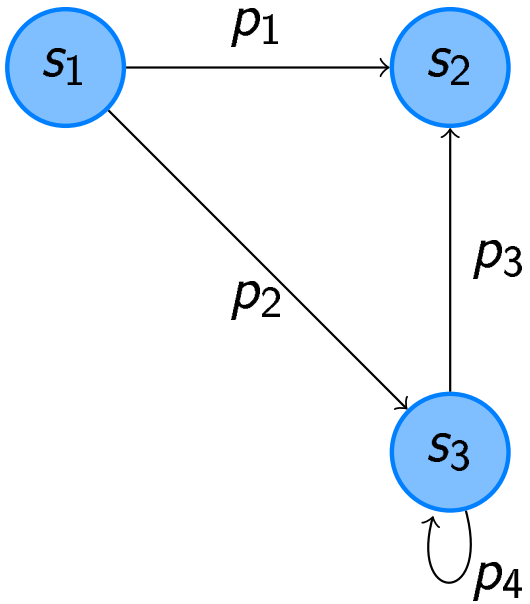
- State elimination
  - Finite Automaton $\longrightarrow$ Regular Expression (*Kleene*)
  - Markov Chain: Compute Probabilities (*Han, Katoen, 2008 and Hahn, Hermanns, Zhang, 2009*)
- Our approach: Recursive Algorithm
  - Bottom-Up computing
  - Use specific properties of Markov Chains

State Elimination

Probability of reaching s_2 (from s_1):

$$p_1 + (p_2 \cdot p_3) + (p_2 \cdot p_4 \cdot p_3) + (p_2 \cdot p_4^2 \cdot p_3) \ldots = p_1 + \sum_{i \in \mathbb{N}} p_2 \cdot p_4^i \cdot p_3$$

$$= p_1 + p_2 \cdot \sum_{i \in \mathbb{N}} p_4 \cdot p_3 = p_1 + p_2 \cdot \frac{1}{1 - p_4} \cdot p_3$$

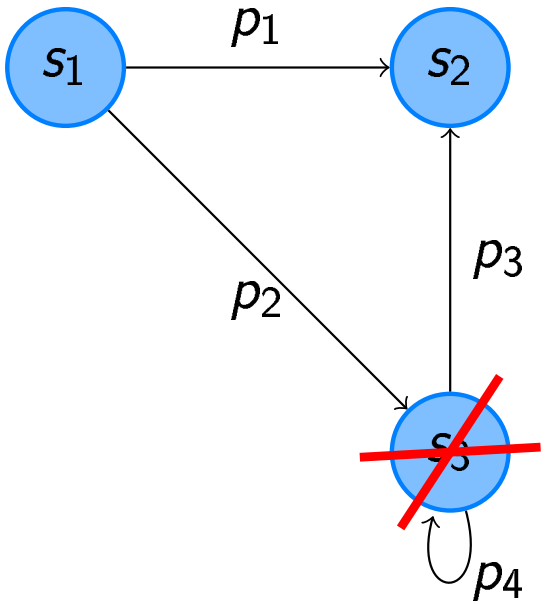


State Elimination

Probability of reaching s_2 (from s_1):

$$p_1 + (p_2 \cdot p_3) + (p_2 \cdot p_4 \cdot p_3) + (p_2 \cdot p_4^2 \cdot p_3) \ldots = p_1 + \sum_{i \in \mathbb{N}} p_2 \cdot p_4^i \cdot p_3$$

$$= p_1 + p_2 \cdot \sum_{i \in \mathbb{N}} p_4 \cdot p_3 = p_1 + p_2 \cdot \frac{1}{1 - p_4} \cdot p_3$$

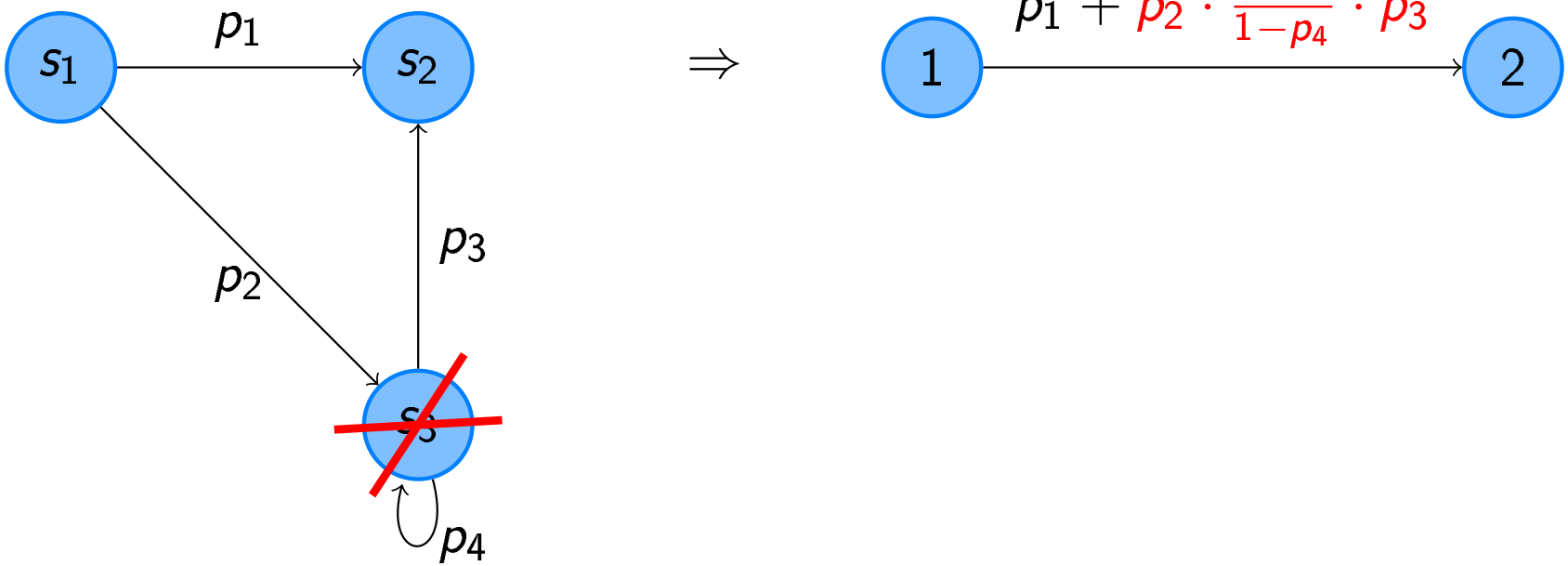


State Elimination

Probability of reaching s_2 (from s_1):

$$p_1 + (p_2 \cdot p_3) + (p_2 \cdot p_4 \cdot p_3) + (p_2 \cdot p_4^2 \cdot p_3) \ldots = p_1 + \sum_{i \in \mathbb{N}} p_2 \cdot p_4^i \cdot p_3$$

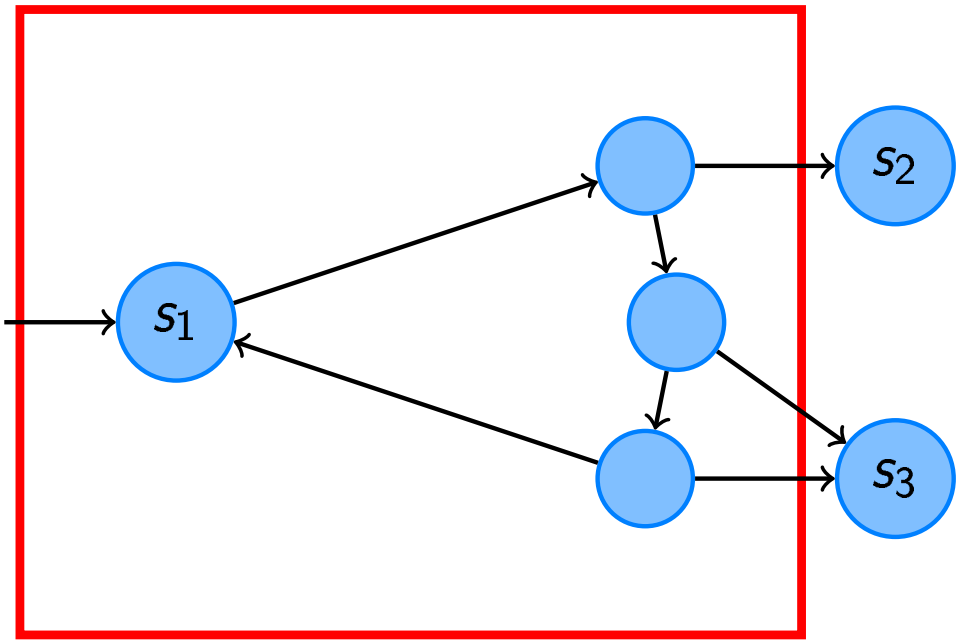
$$= p_1 + p_2 \cdot \sum_{i \in \mathbb{N}} p_4 \cdot p_3 = p_1 + p_2 \cdot \frac{1}{1 - p_4} \cdot p_3$$



Contents

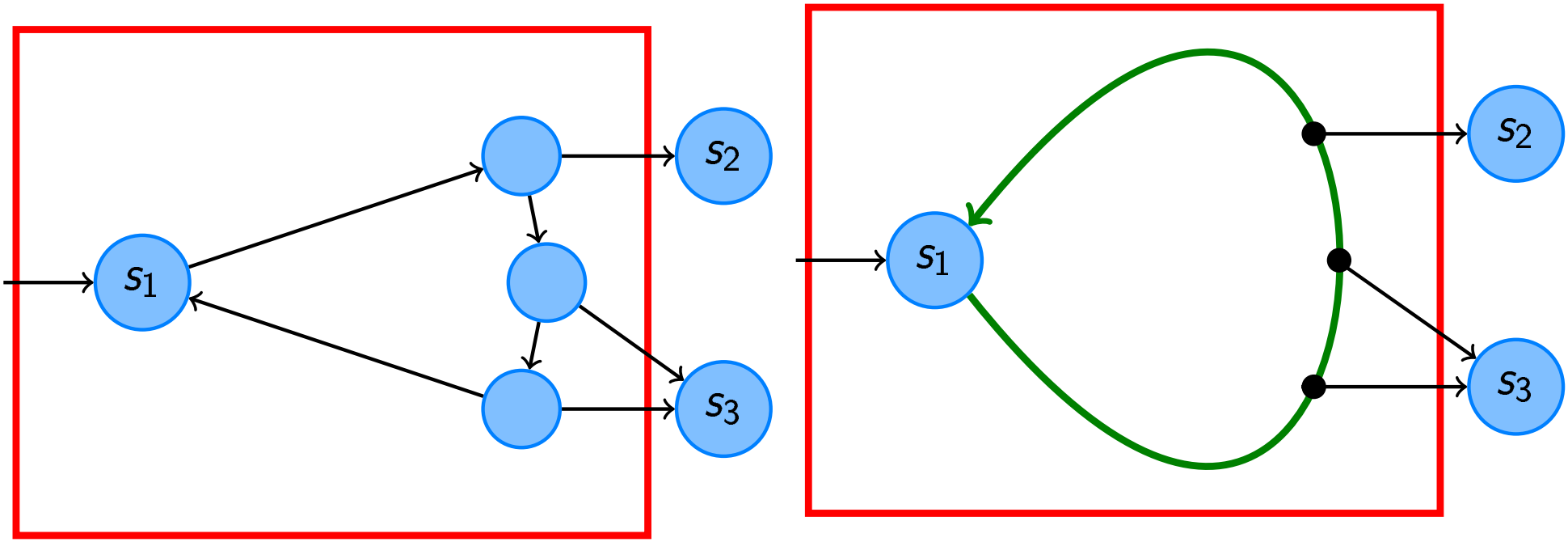
SCC with one ingoing node

Example SCC



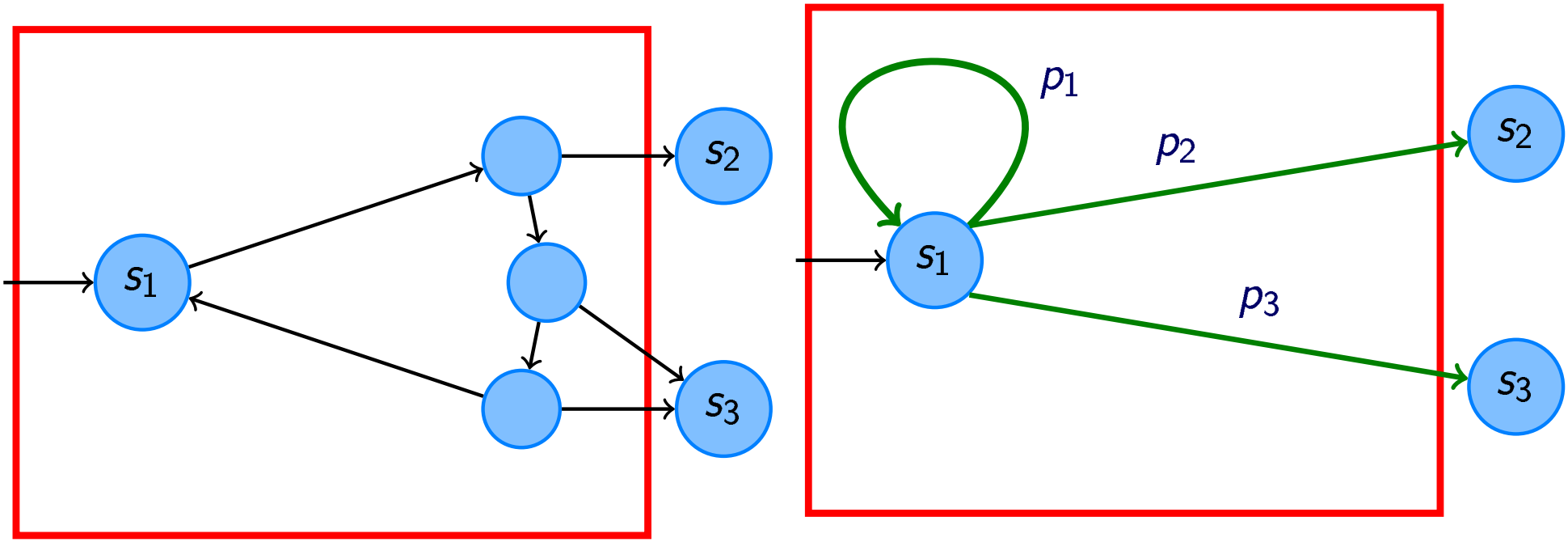
SCC with one ingoing node

Example SCC and abstract view



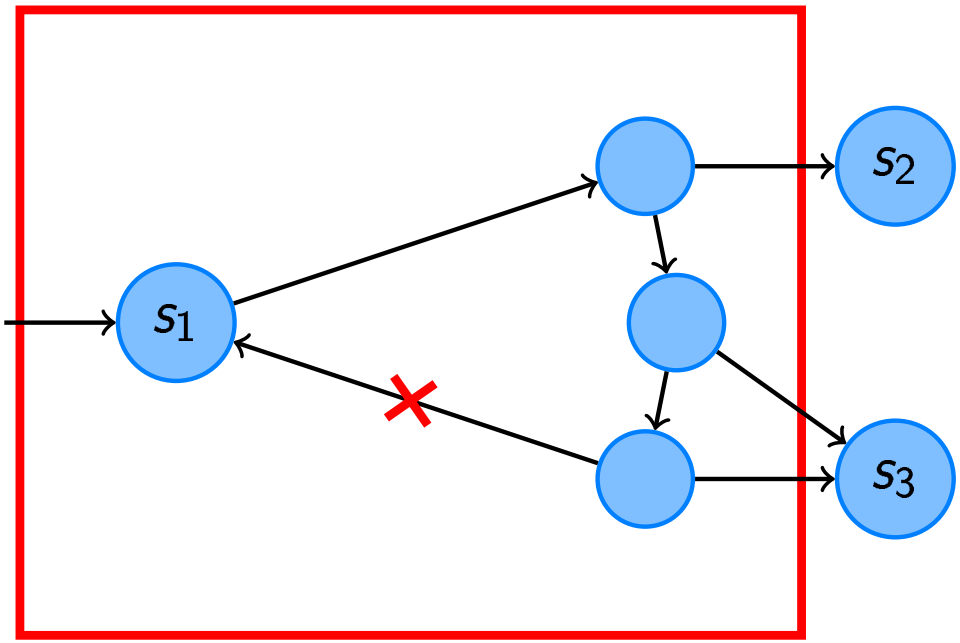
SCC with one ingoing node

Example SCC and more abstract view

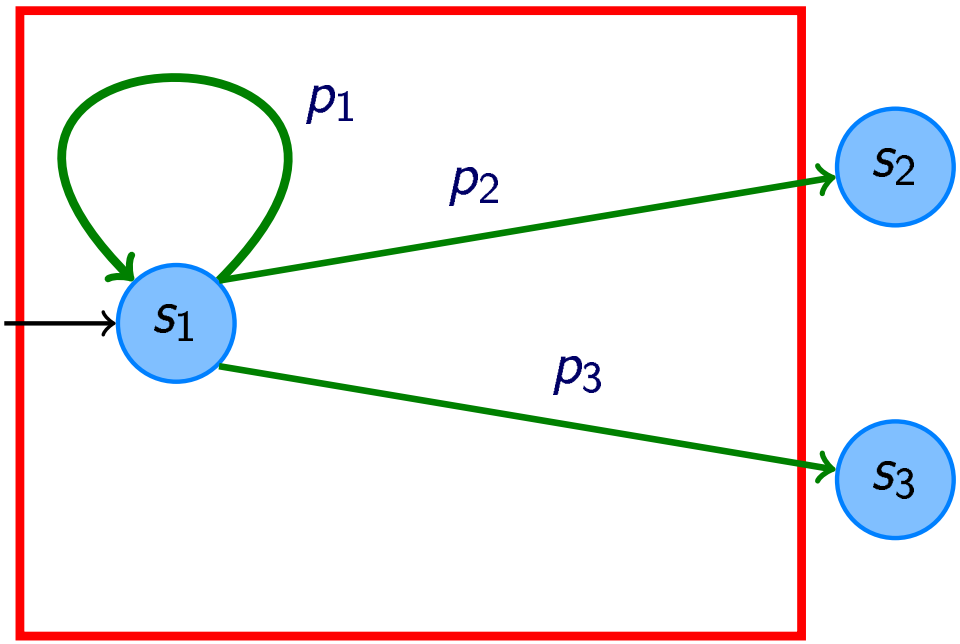


SCC with one ingoing node

Example SCC: Paths leading back to input state are ignored



SCC with one ingoing node



Probability mass of all paths entering and eventually leaving SCC

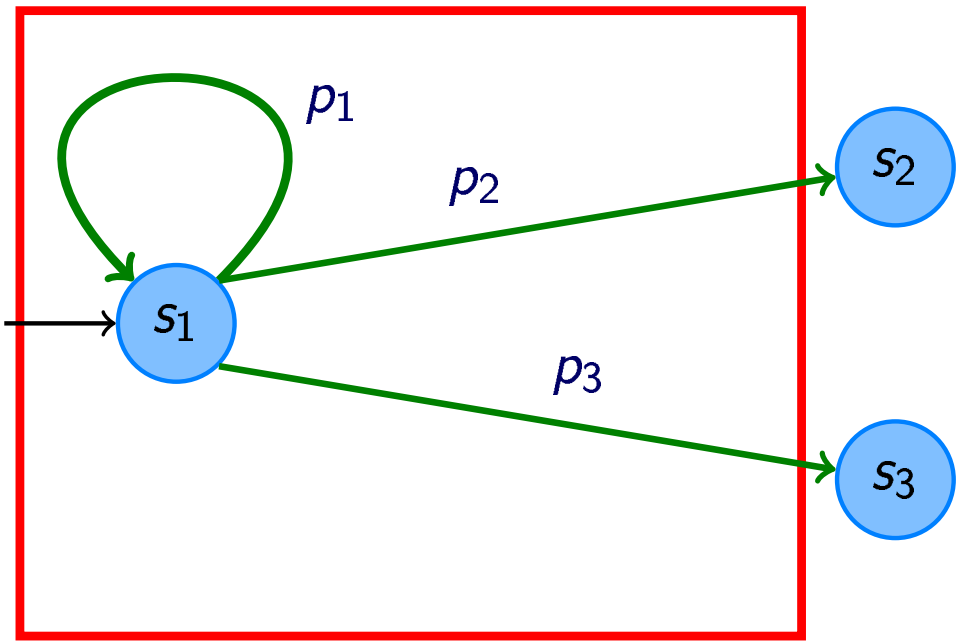
$$\sum_{i \in \mathbb{N}} p_1^i \cdot (p_2 + p_3) = \frac{1}{1 - p_1} \cdot (p_2 + p_3)$$

Prob. for leaving SCC is 1

$$\frac{1}{1 - p_1} \cdot (p_2 + p_3) = 1$$

$$\Leftrightarrow p_1 = 1 - p_2 - p_3$$

SCC with one ingoing node



Probability mass of all paths entering and eventually leaving SCC

$$\sum_{i \in \mathbb{N}} p_1^i \cdot (p_2 + p_3) = \frac{1}{1 - p_1} \cdot (p_2 + p_3)$$

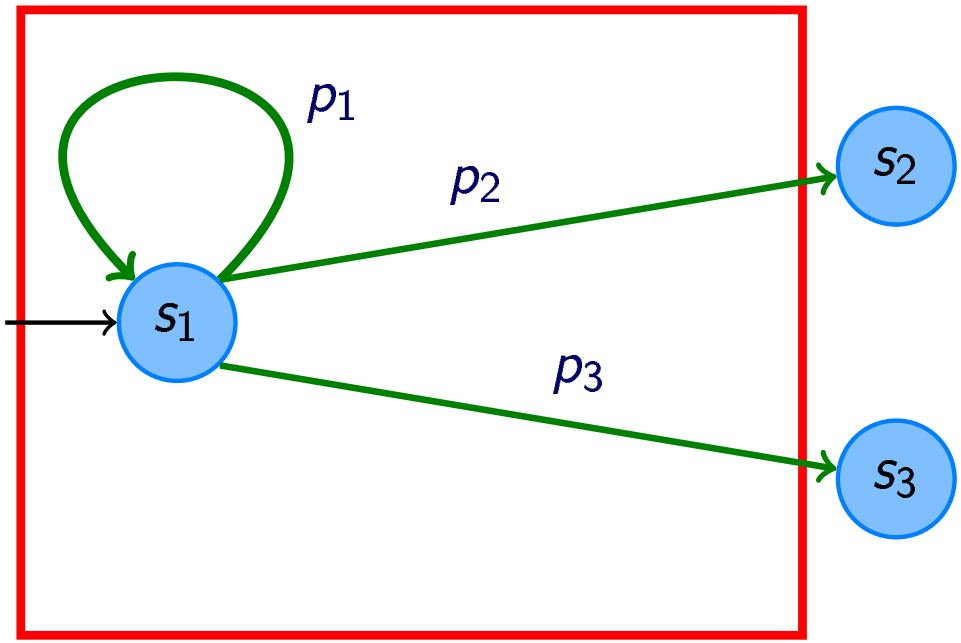
Prob. for leaving SCC is 1

$$\frac{1}{1 - p_1} \cdot (p_2 + p_3) = 1$$

$$\Leftrightarrow p_1 = 1 - p_2 - p_3$$

Probability of coming back to input node can be computed without considering paths that lead back to it!

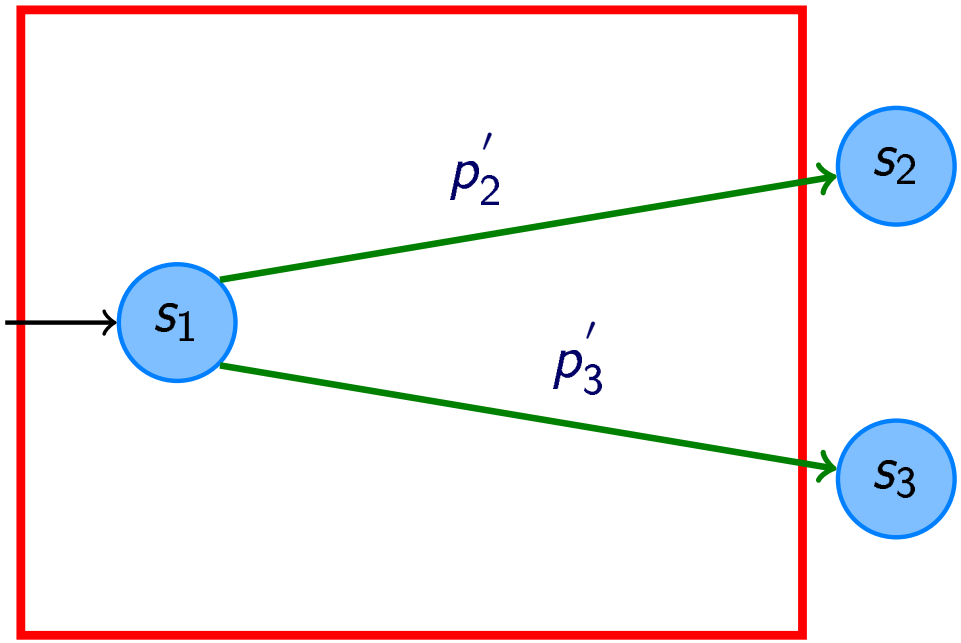
SCC with one ingoing node



Probability of reaching s_2:

$$\frac{1}{1-p_1} \cdot p_2$$

SCC with one ingoing node



Probability of reaching s_2:

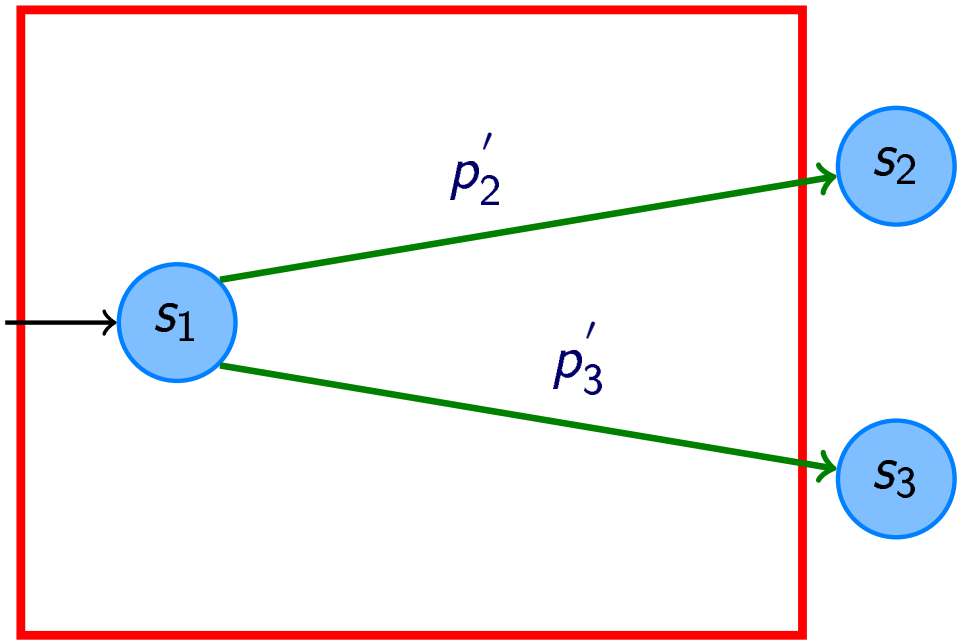
$$\frac{1}{1-p_1} \cdot p_2$$

Discard p_1 and scale p_2 and p_3:

$$p_2' = \frac{p_2}{p_2 + p_3}$$

$$p_3' = \frac{p_3}{p_2 + p_3}$$

SCC with one ingoing node



Probability of reaching s_2:

$$\frac{1}{1-p_1} \cdot p_2$$

Discard p_1 and scale p_2 and p_3:

$$p_2' = \frac{p_2}{p_2 + p_3}$$

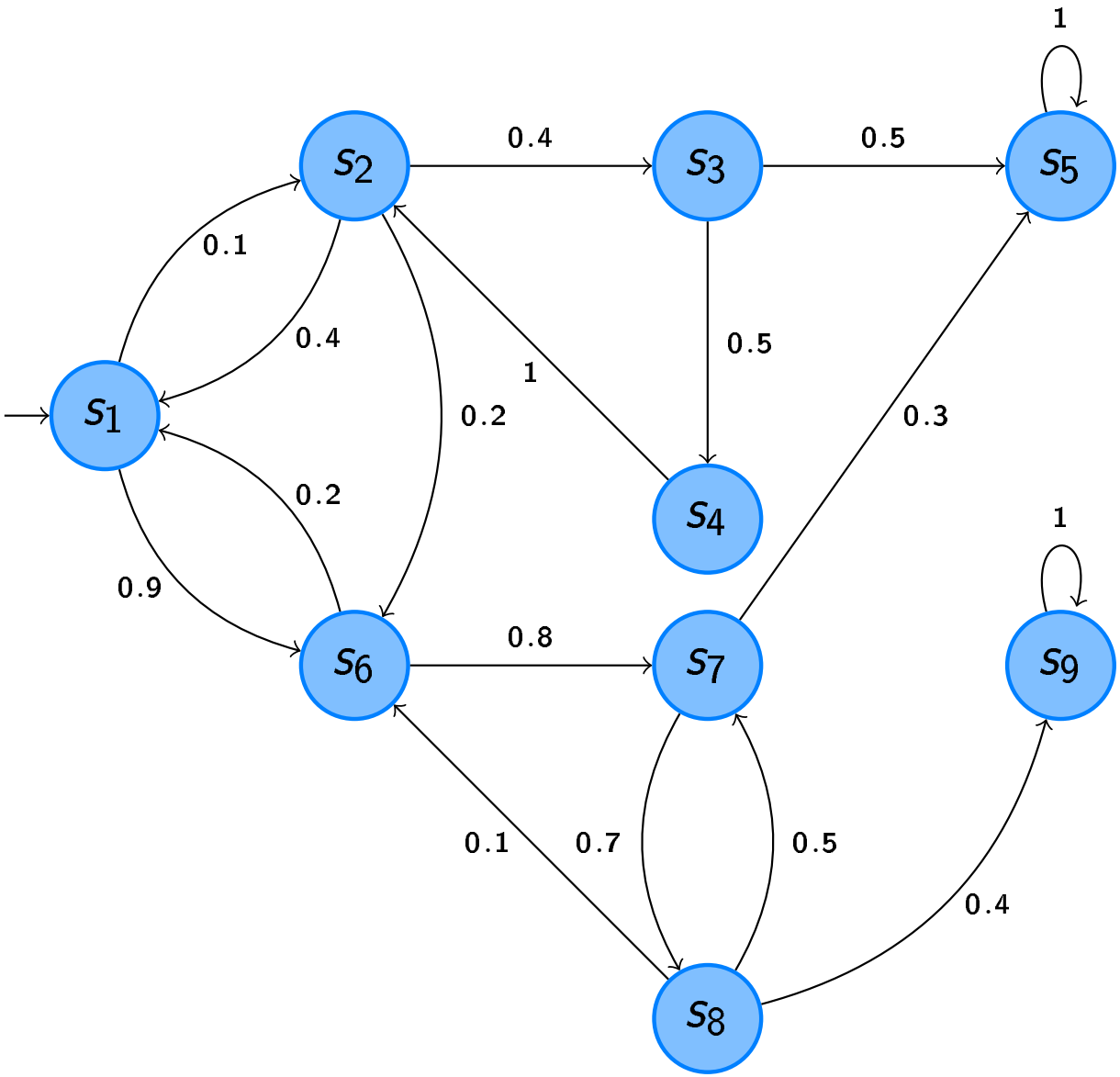
$$p_3' = \frac{p_3}{p_2 + p_3}$$

Scaling preserves reachability probabilities:

(As before: $p_3 = 1 - p_2 - p_1$)

$$p_2' = \frac{p_2}{p_2 + p_3} = \frac{p_2}{p_2 + (1 - p2 - p1)} = \frac{p_2}{1 - p_1}$$

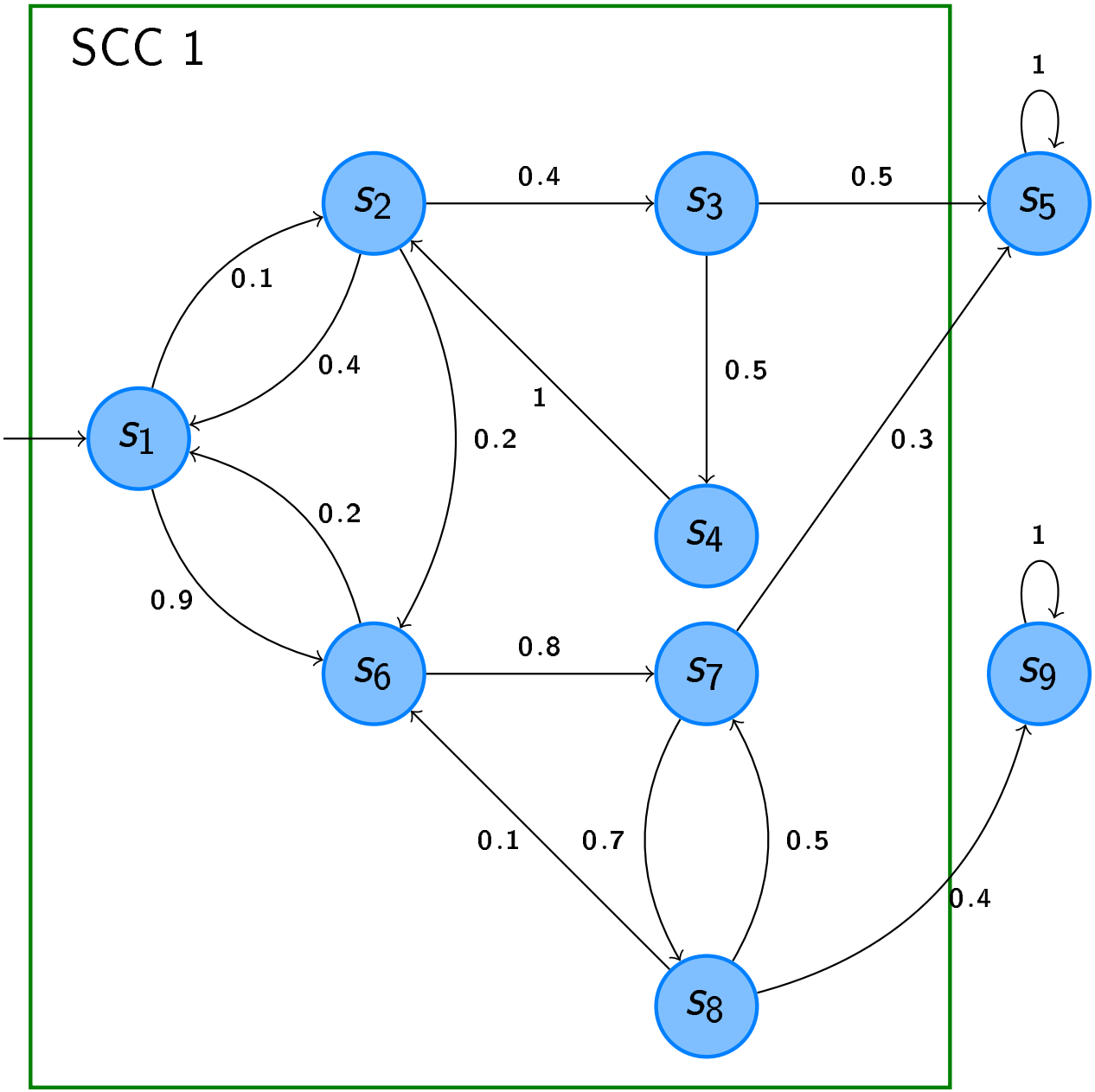
Example



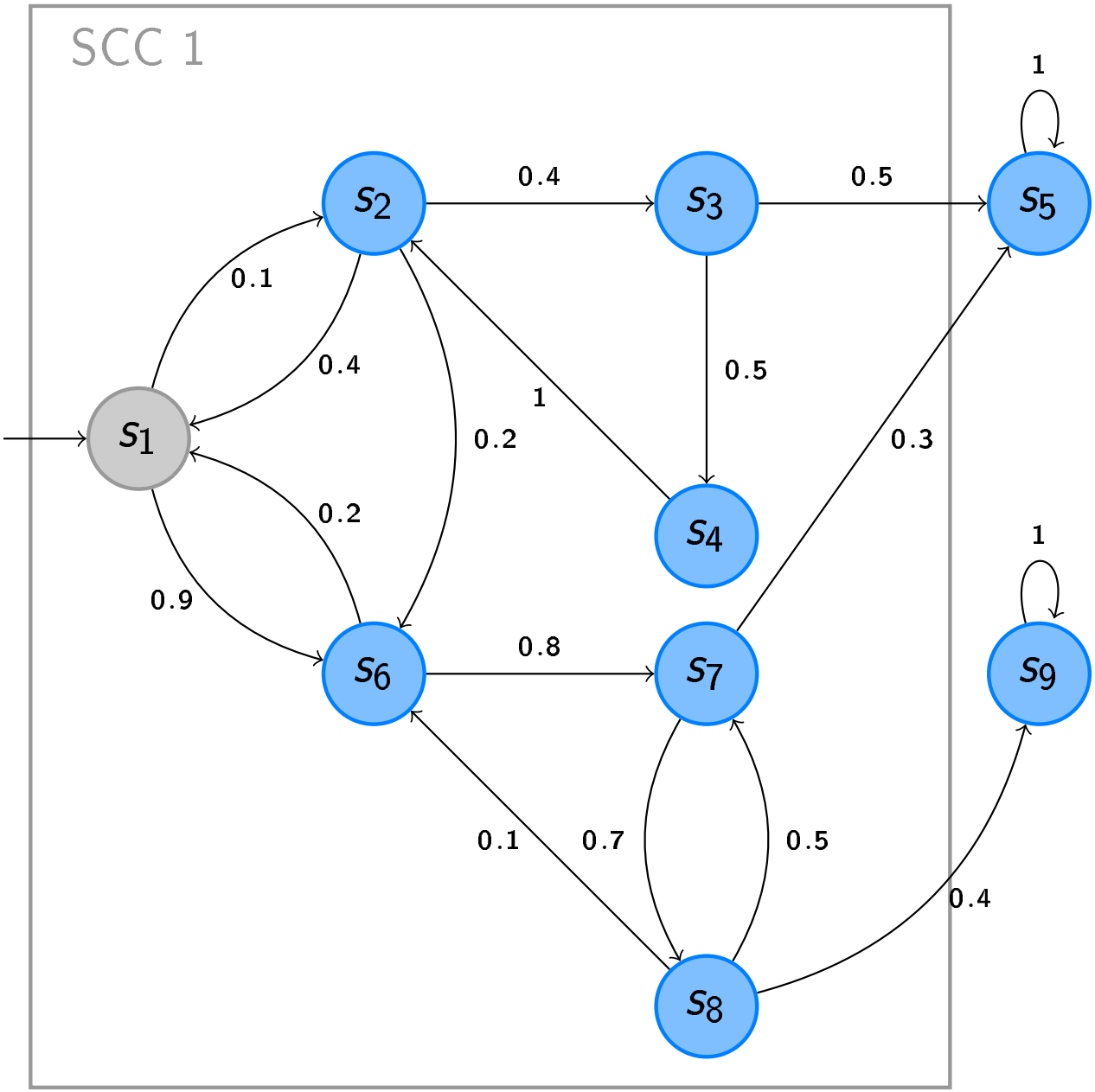
Whole graph:

- Search for SCCs

Example


SCC 1
s_1
s_2
s_3
s_4
s_5
s_6
s_7
s_8
s_9
0.4
0.5
1
0.1
0.4
0.5
1
0.2
0.3
0.2
1
0.9
0.8
0.1
0.7
0.5
0.4

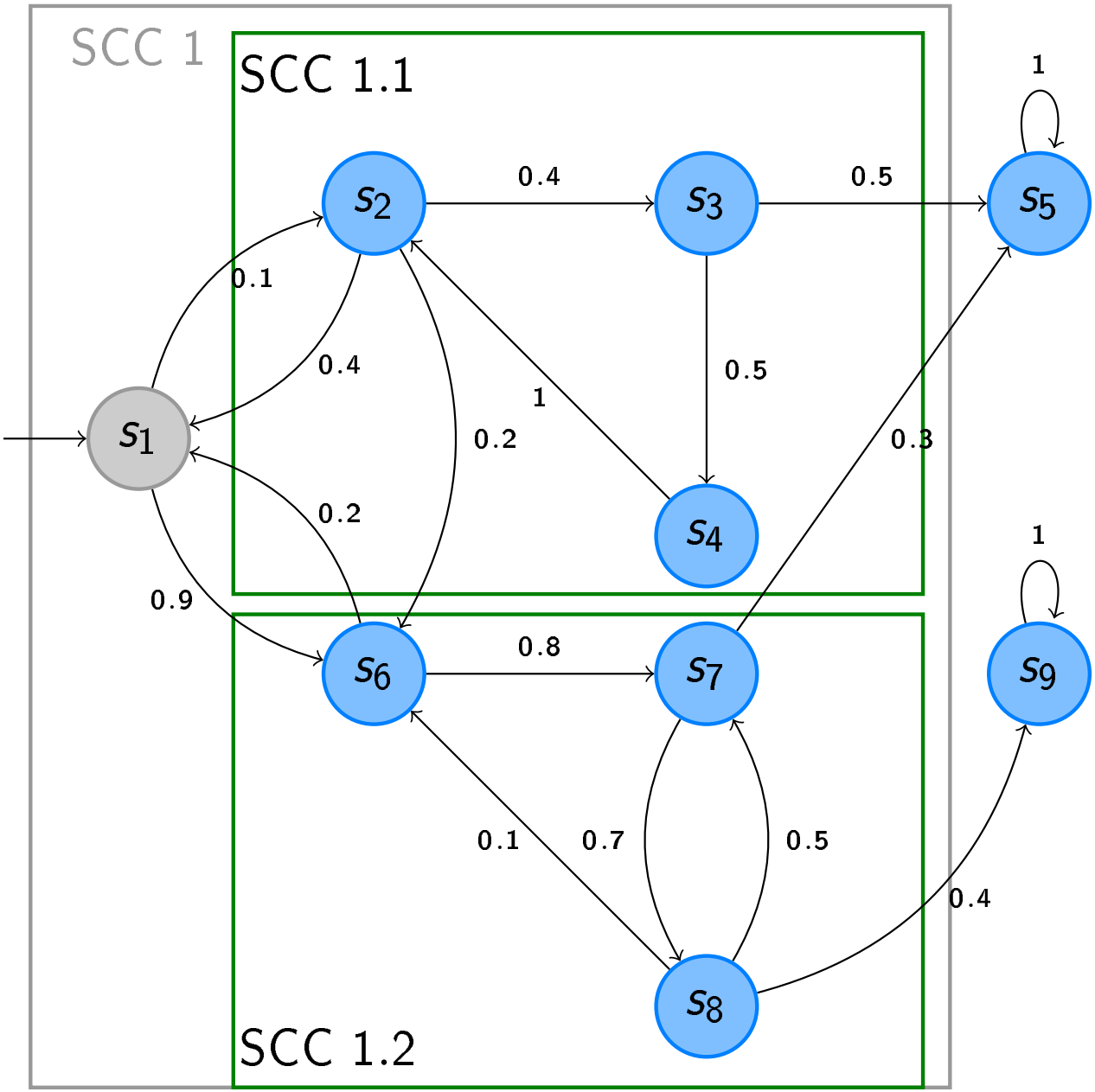
Example



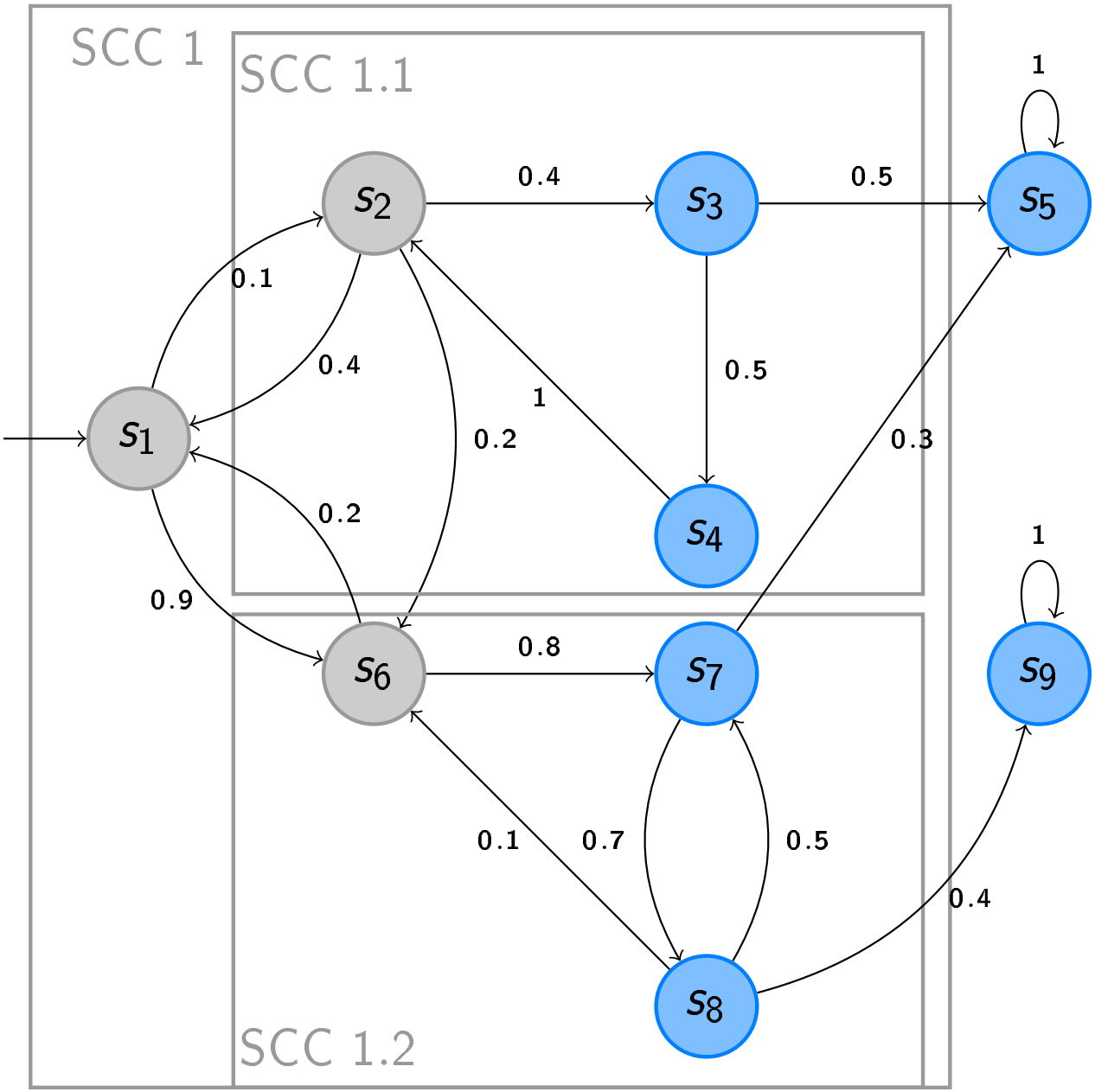
SCC 1:

- Ignore
  Input-Node
- Search for SCCs

Example


SCC 1
SCC 1.1
SCC 1.2
s_1
s_2
s_3
s_4
s_5
s_6
s_7
s_8
s_9
0.4
0.5
1
0.1
0.4
0.5
1
0.2
0.3
0.2
0.9
0.8
1
0.1
0.7
0.5
0.4

Example



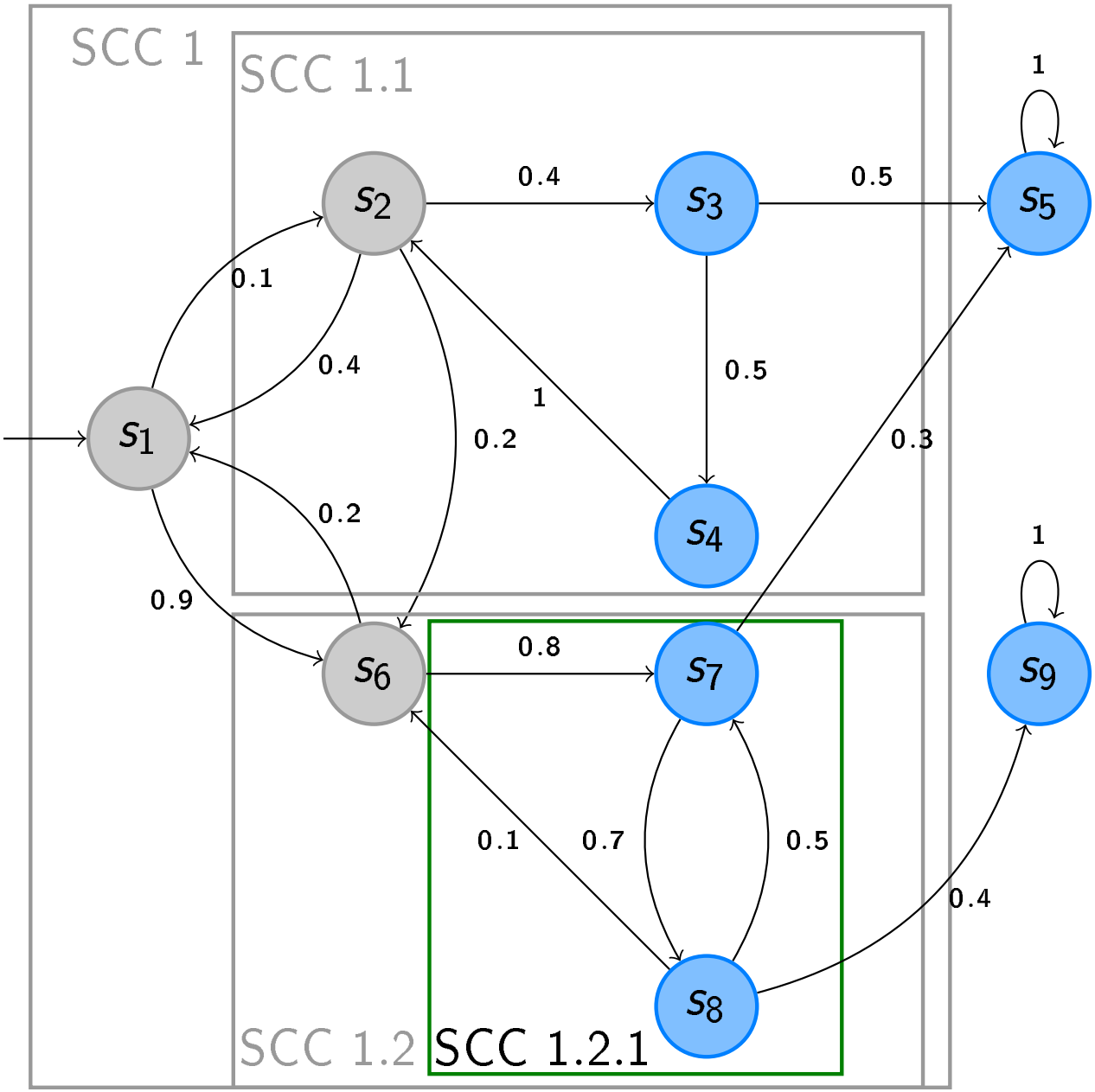
SCC 1.1:

- Ignore Input-Node
- Search for SCCs

SCC 1.2:

- Ignore Input-Node
- Search for SCCs

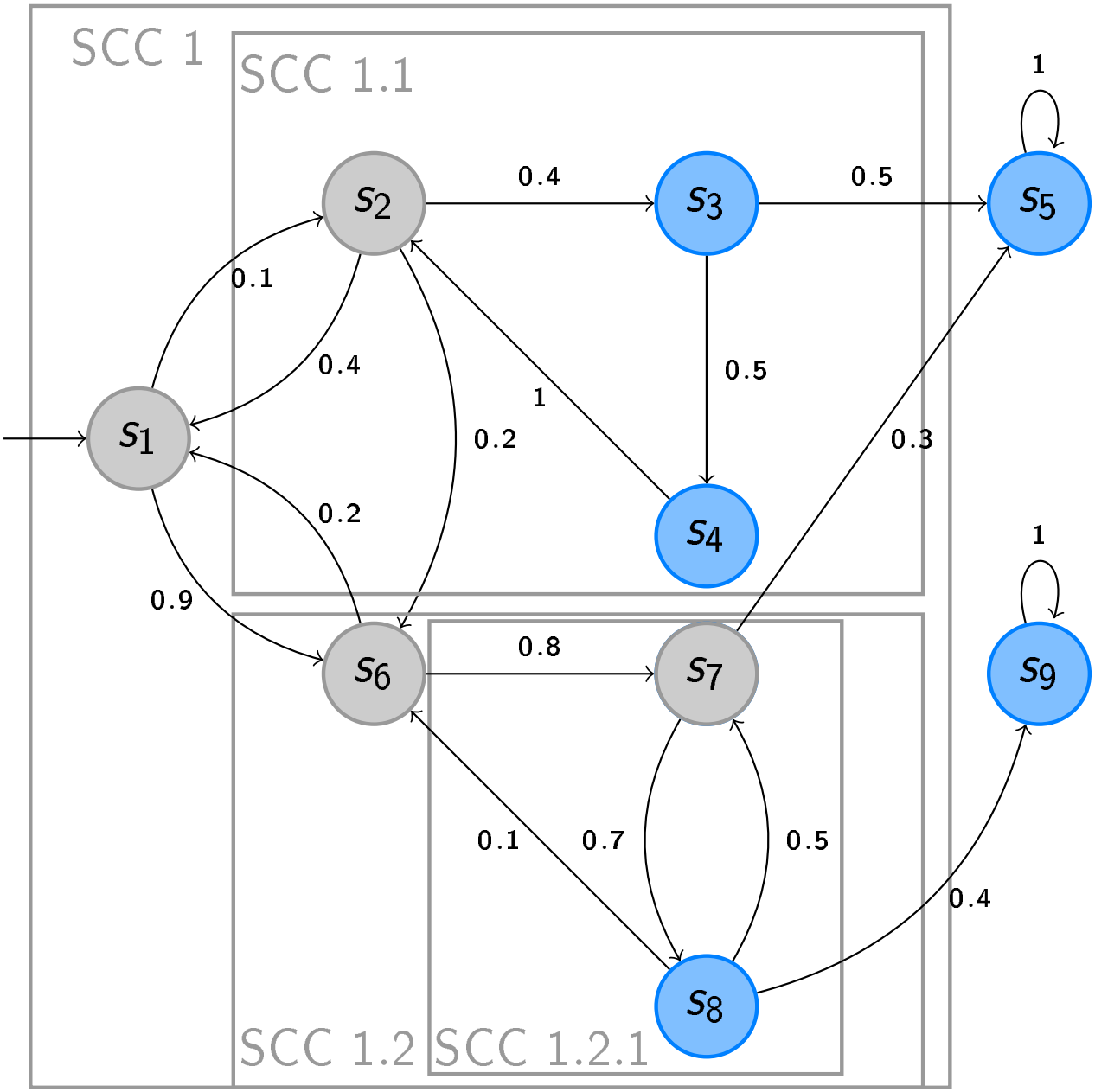
Example



SCC 1.1:

- No SCCs found!

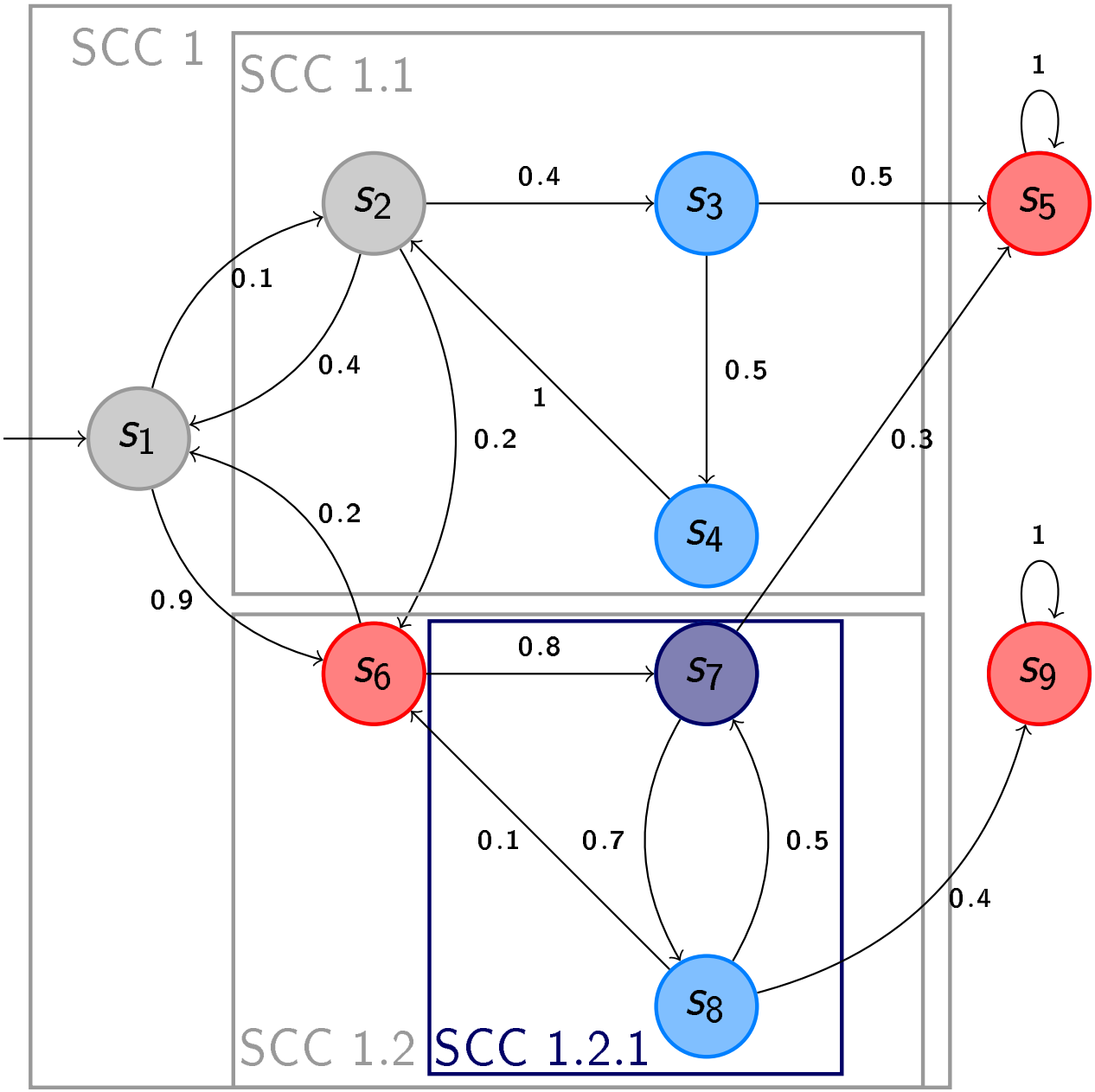
Example



SCC 1.2:

- Ignore Input-Node
- Search for SCCs

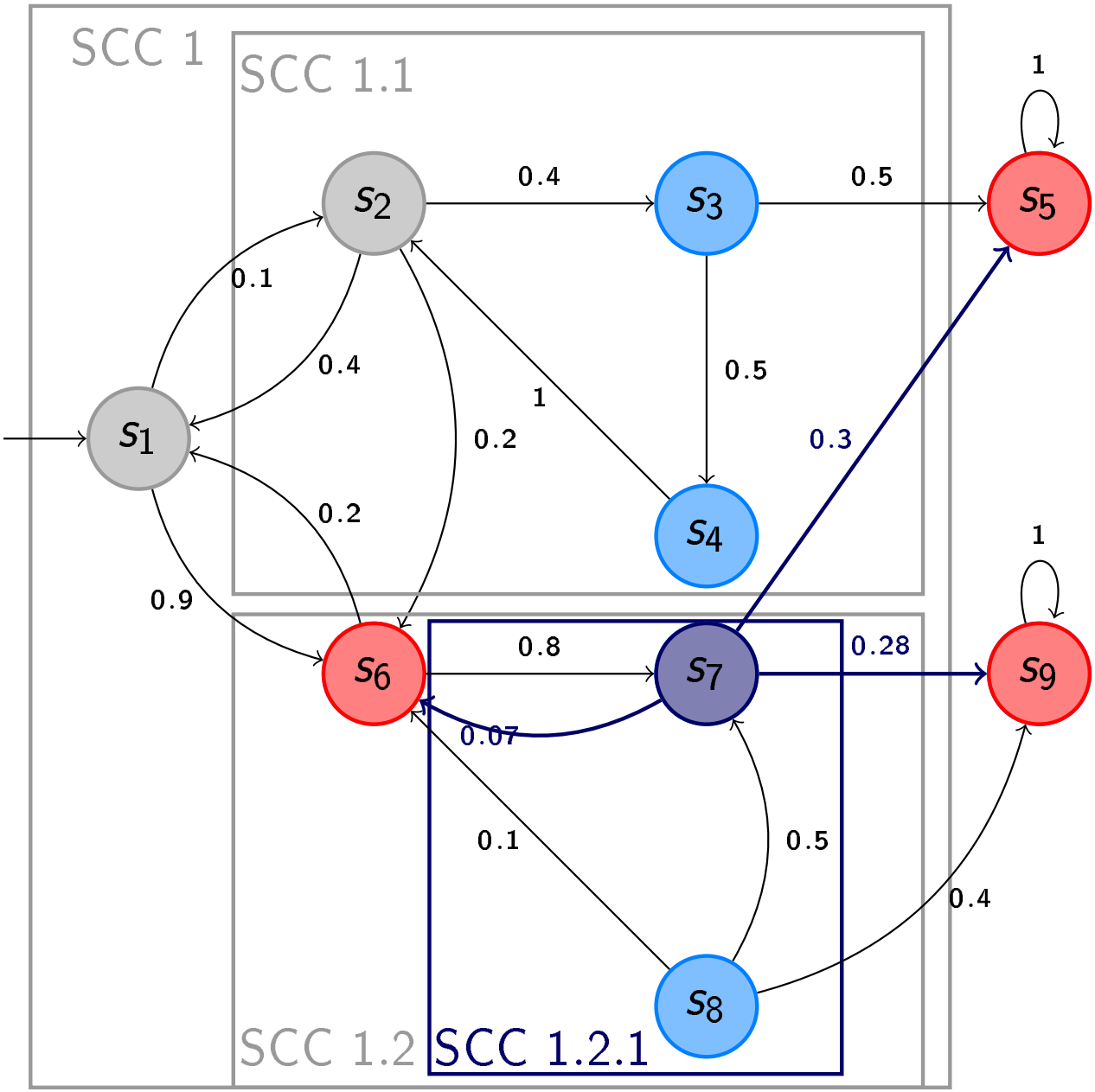
Example



SCC 1.2.1:

- No SCCs found!
- Compute probabilities for reaching output nodes from input node.

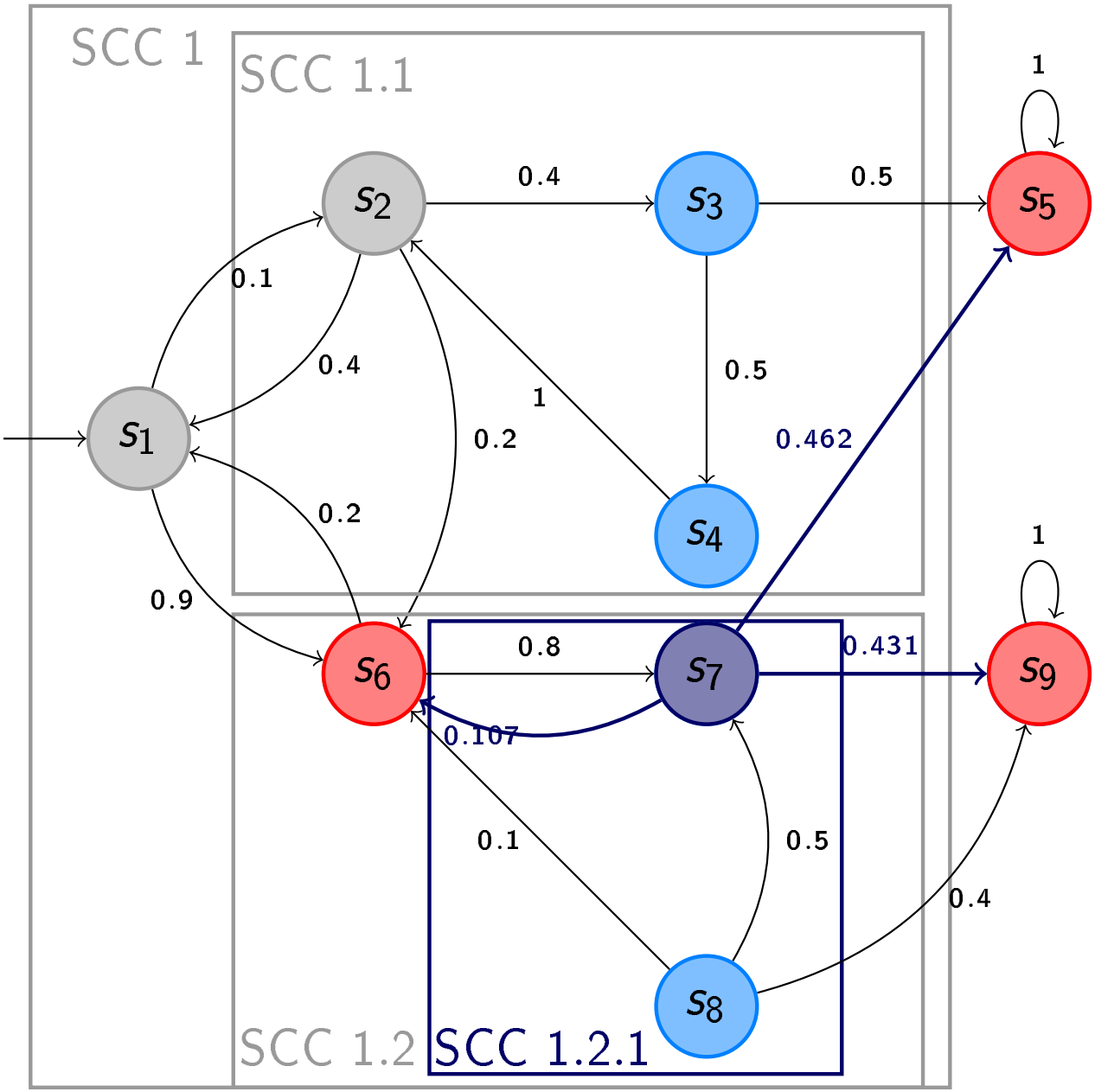
Example



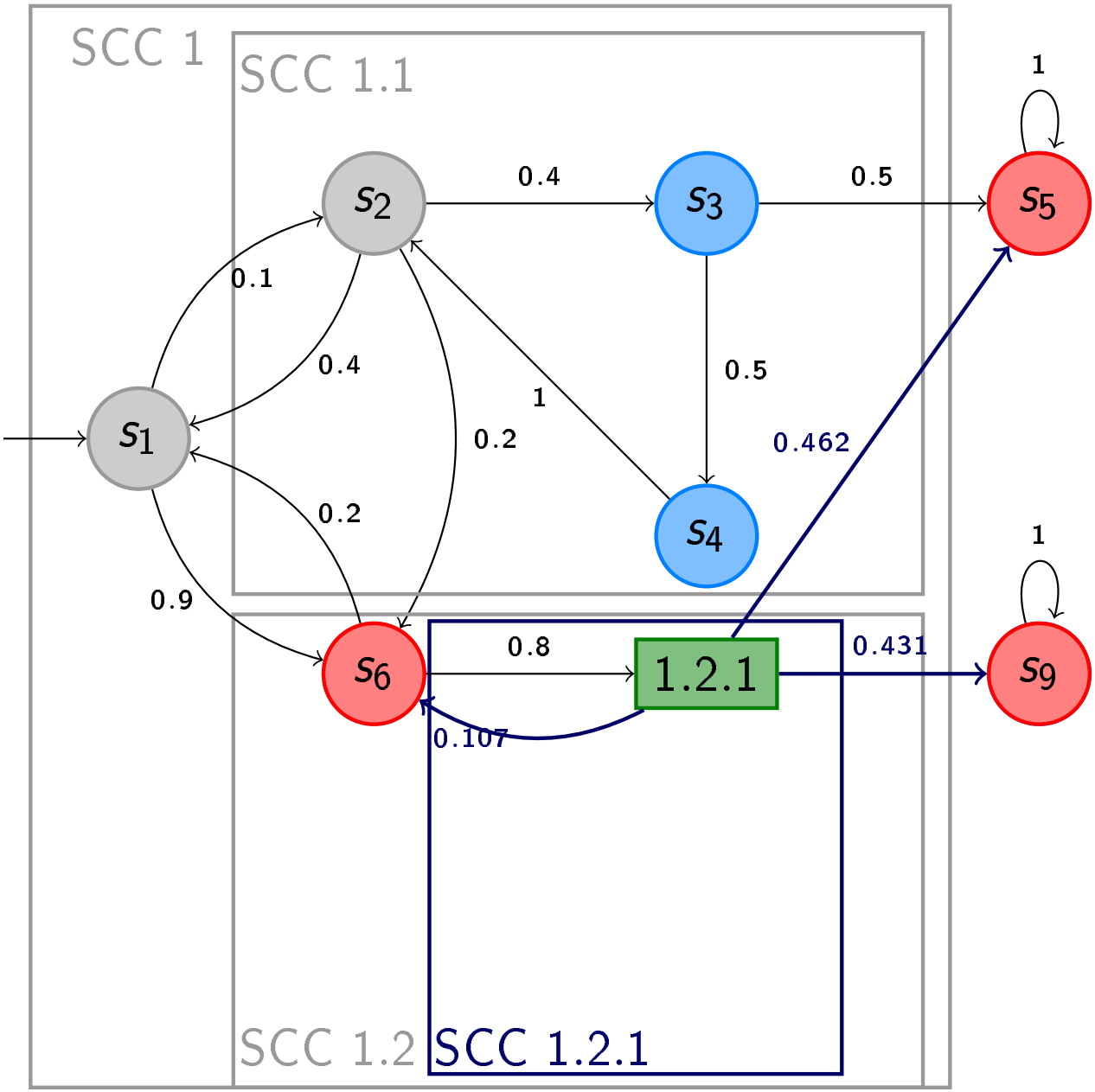
SCC 1.2.1:

- Scale probabilities.

Example


SCC 1
SCC 1.1
SCC 1.2
SCC 1.2.1
s_1
s_2
s_3
s_4
s_5
s_6
s_7
s_8
s_9
0.4
0.5
1
0.1
0.4
0.5
1
0.2
0.462
0.2
0.9
0.8
0.431
1
0.107
0.1
0.5
0.4

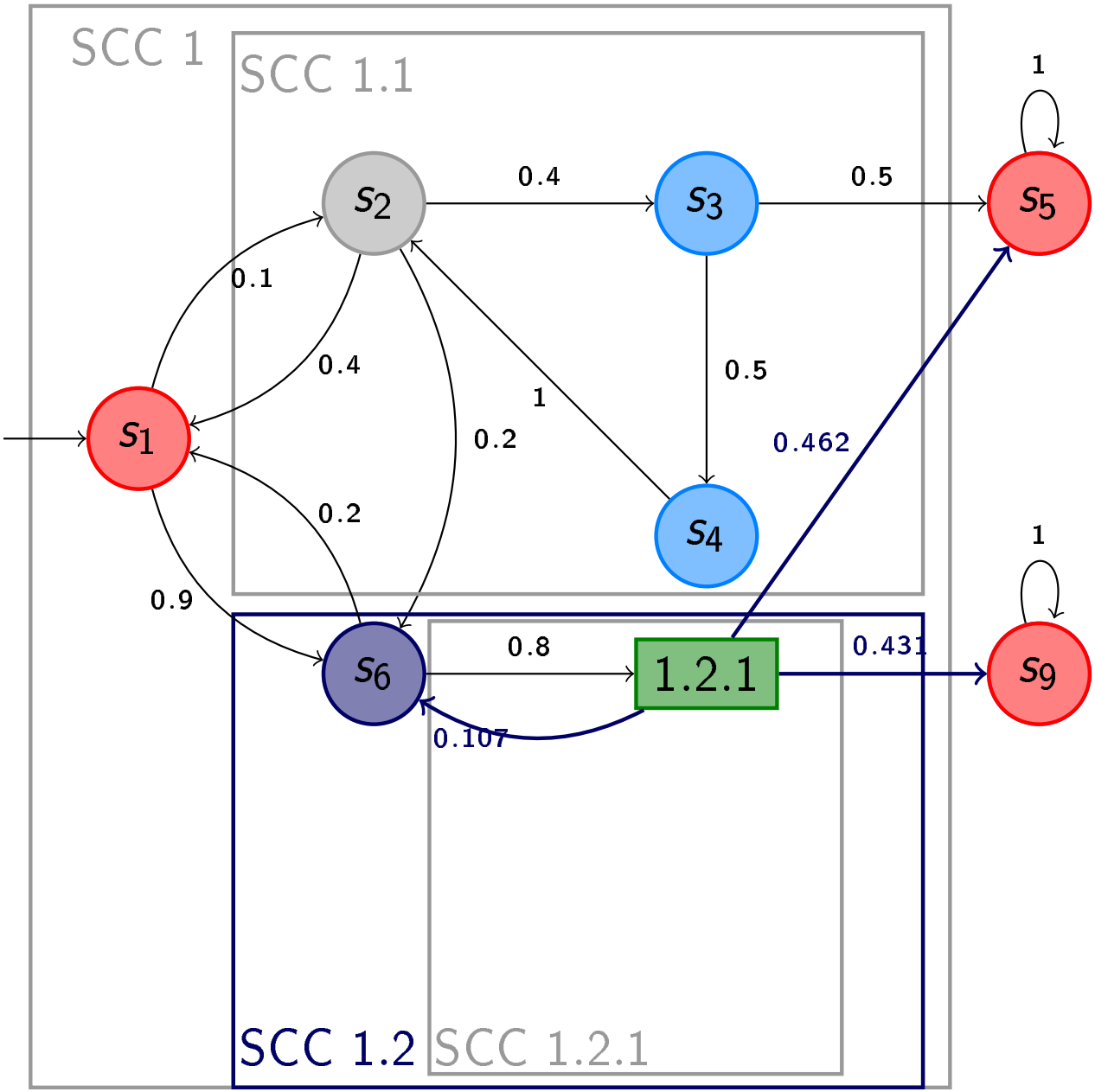
Example



SCC 1.2.1:

- Reduced to abstract node.

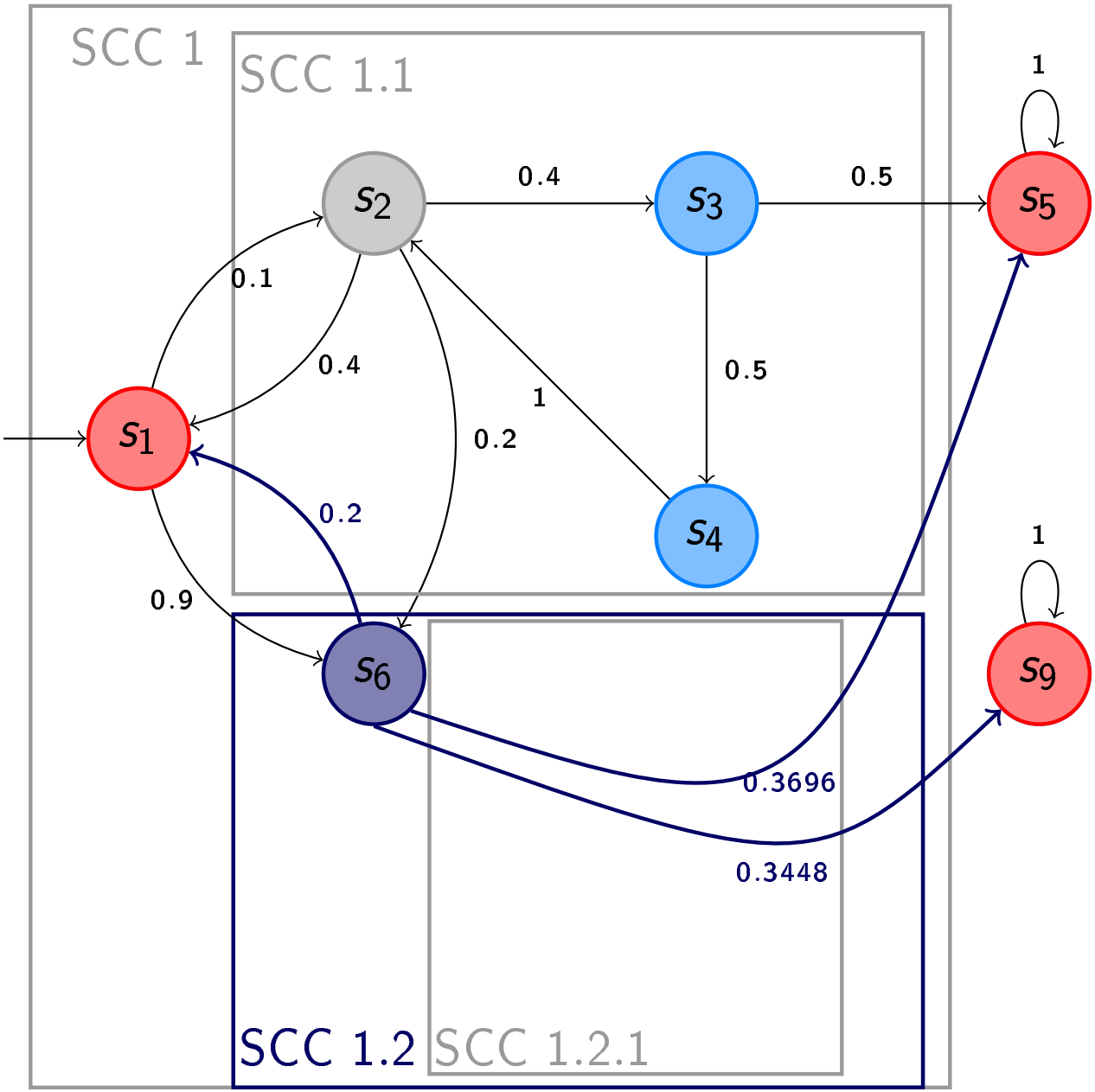
Example



SCC 1.2:

- Compute probabilities for reaching output nodes from input node.

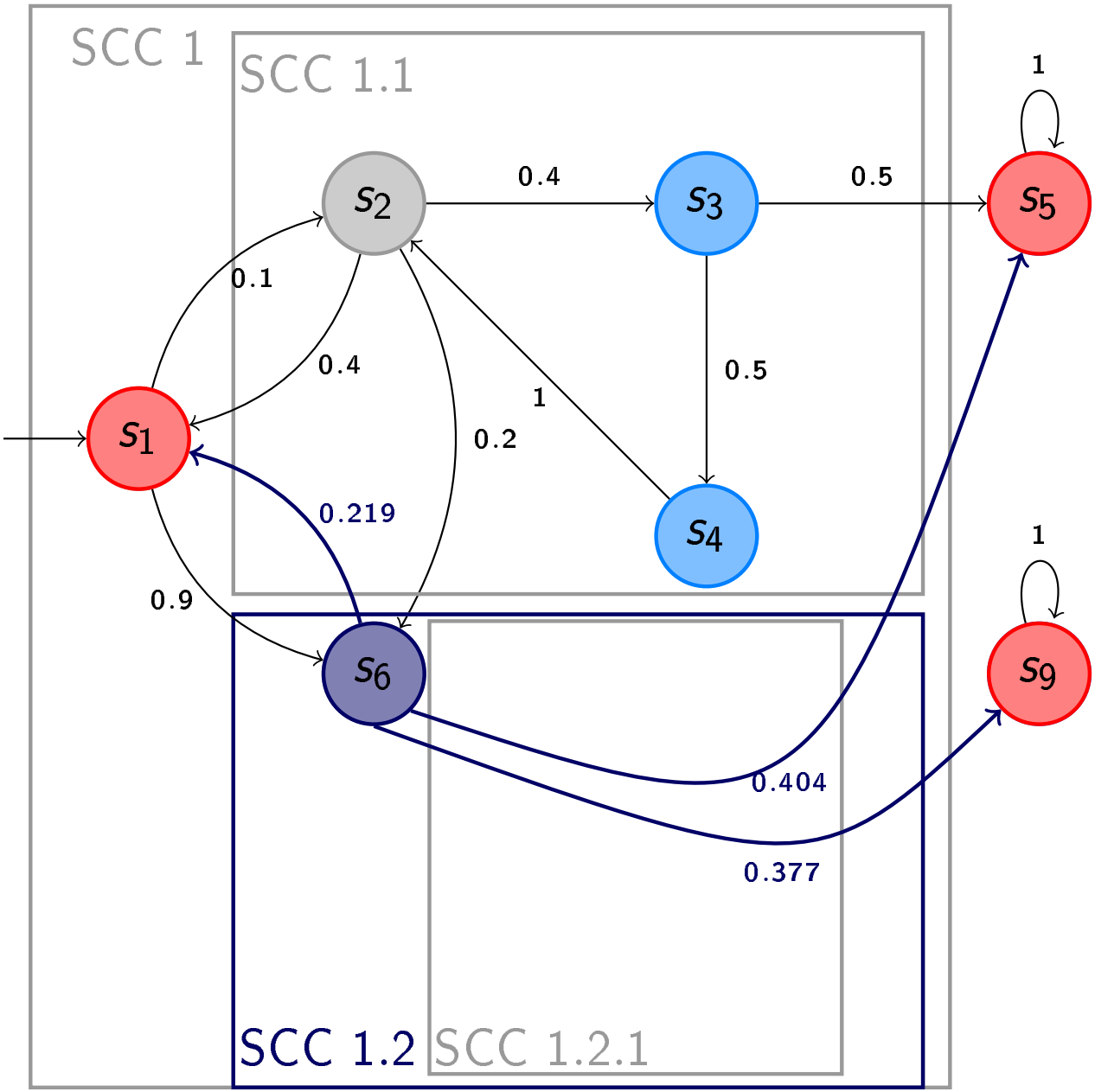
Example



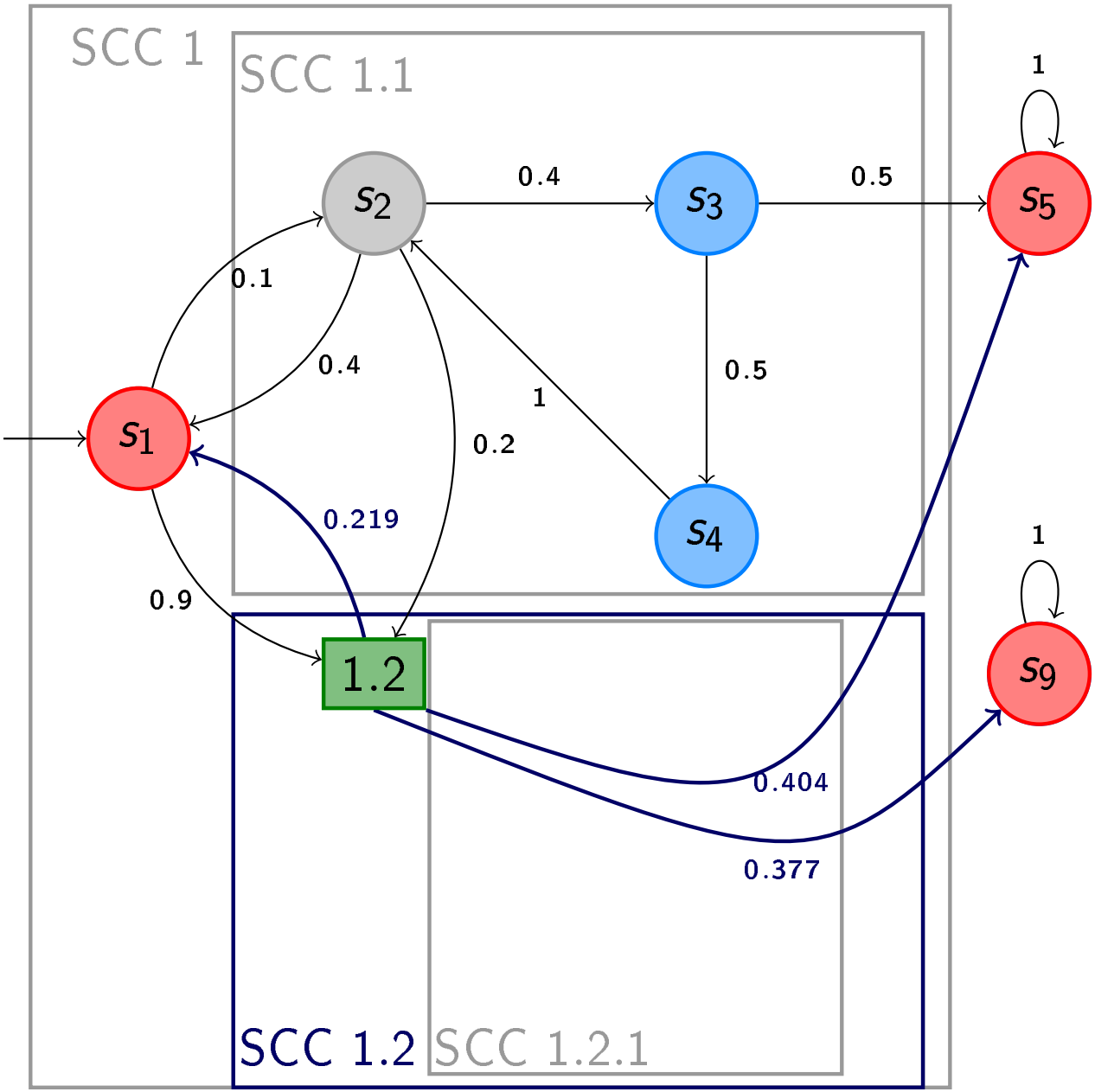
SCC 1.2:

- Scale probabilities.

Example


SCC 1
SCC 1.1
SCC 1.2
SCC 1.2.1
s_1
s_2
s_3
s_4
s_5
s_6
s_9
0.4
0.5
1
0.1
0.4
0.5
1
0.2
0.219
0.9
1
0.404
0.377

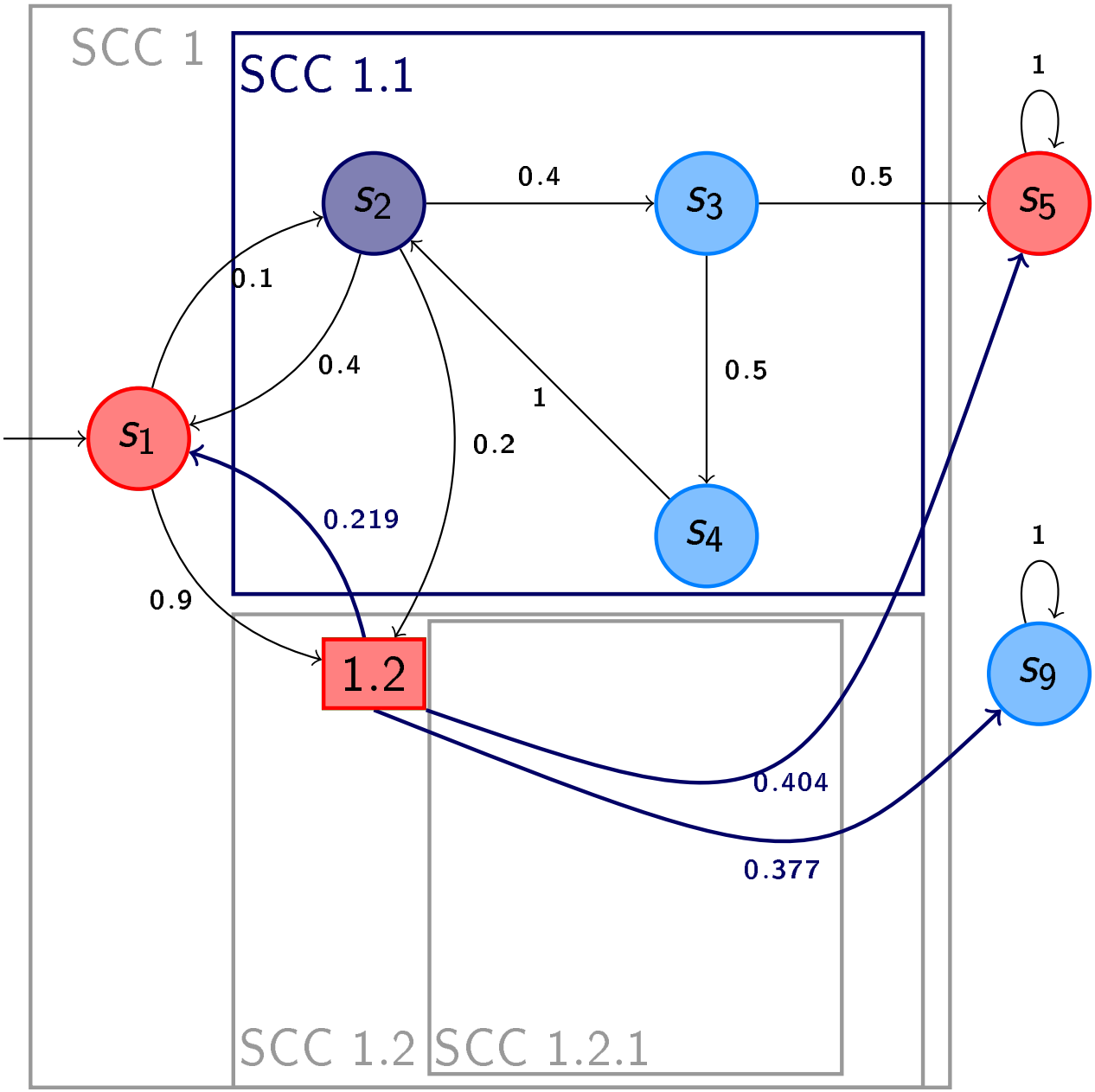
Example



SCC 1.2:

- Reduced to abstract node.

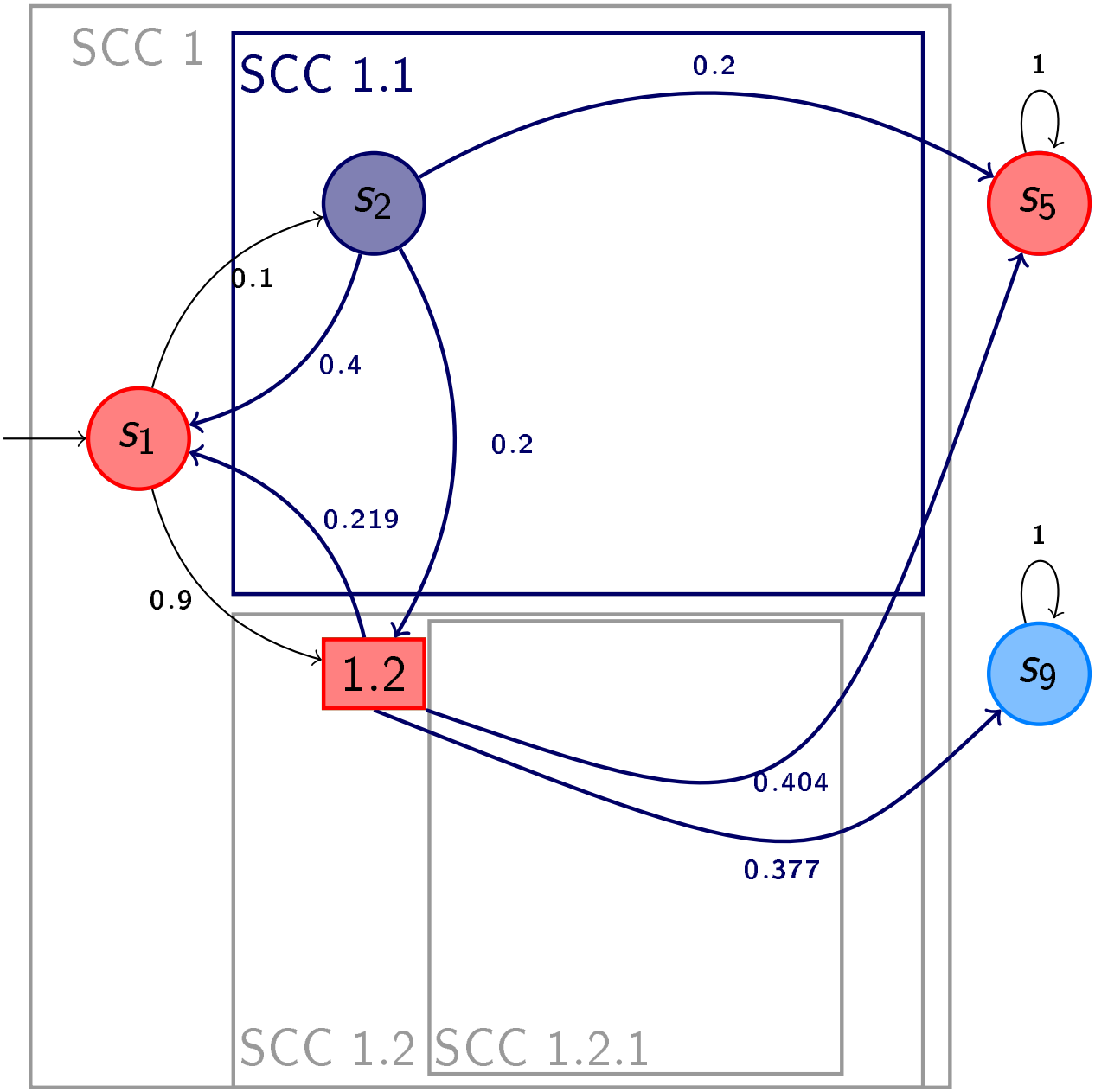
Example



SCC 1.1:

- Compute probabilities for reaching output nodes from input node.

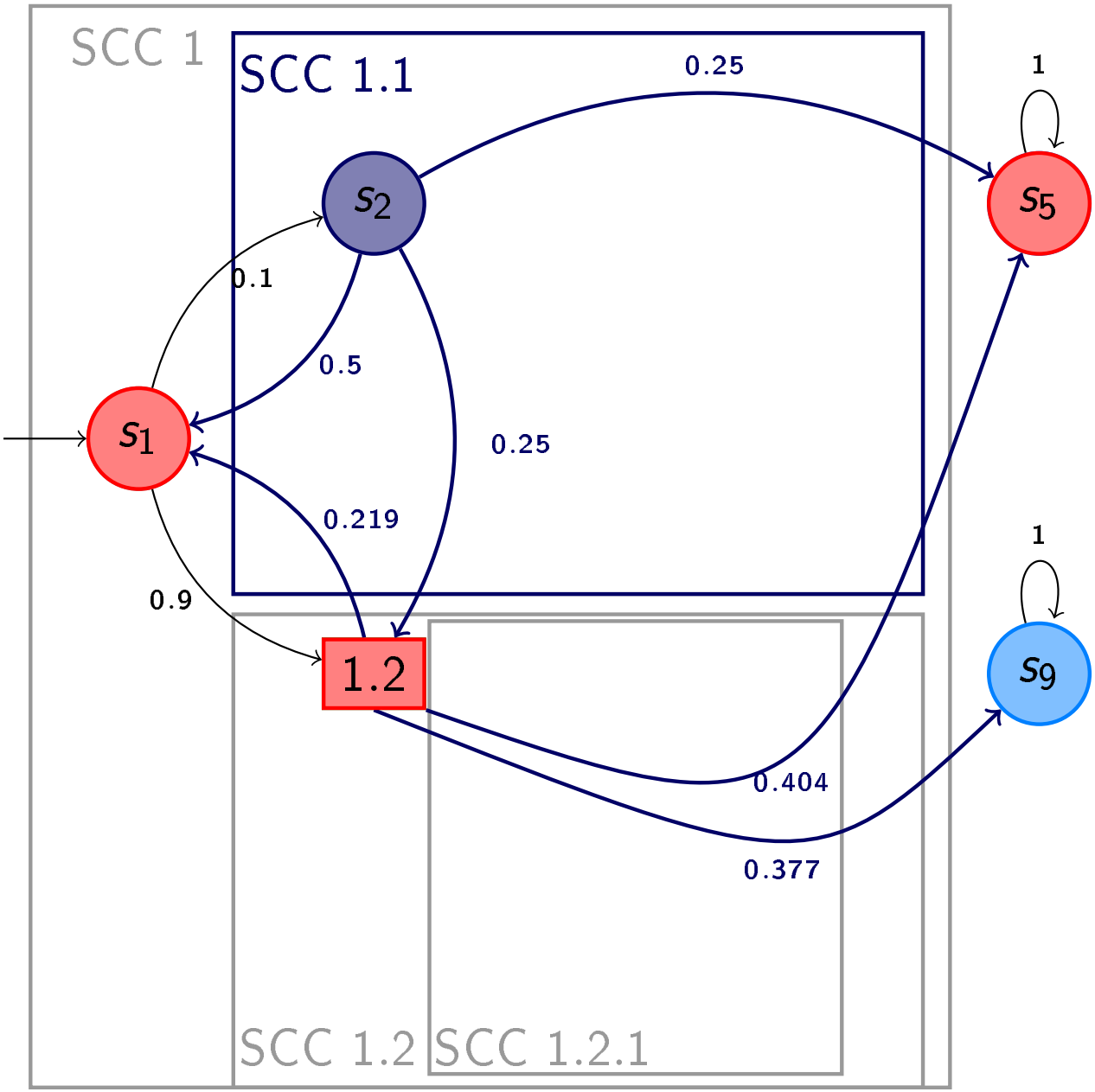
Example



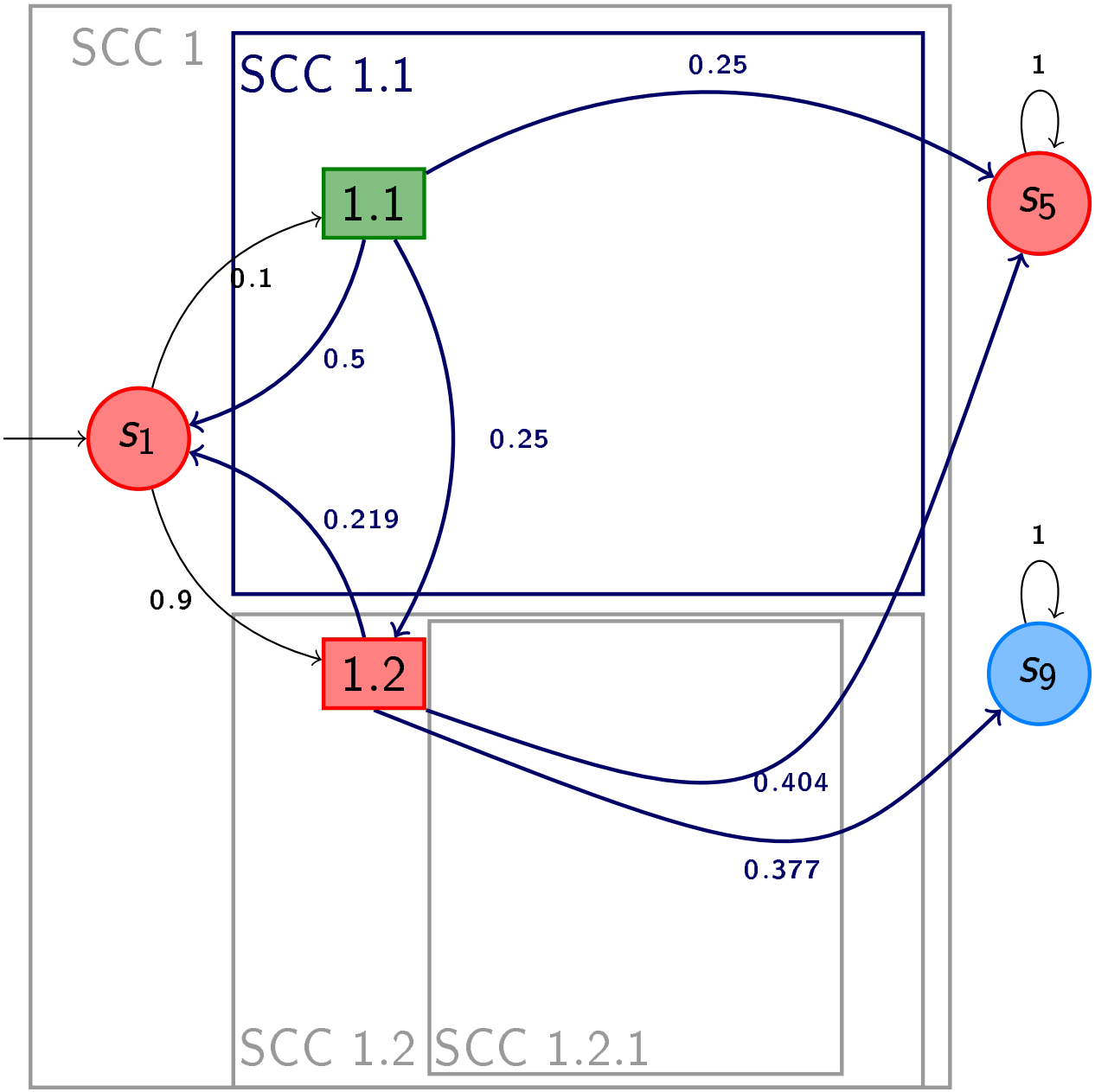
SCC 1.1:

- Scale probabilities.

Example


SCC 1
SCC 1.1
0.25
1
s_2
s_5
0.1
0.5
s_1
0.25
0.219
1
0.9
1.2
s_9
0.404
0.377
SCC 1.2
SCC 1.2.1

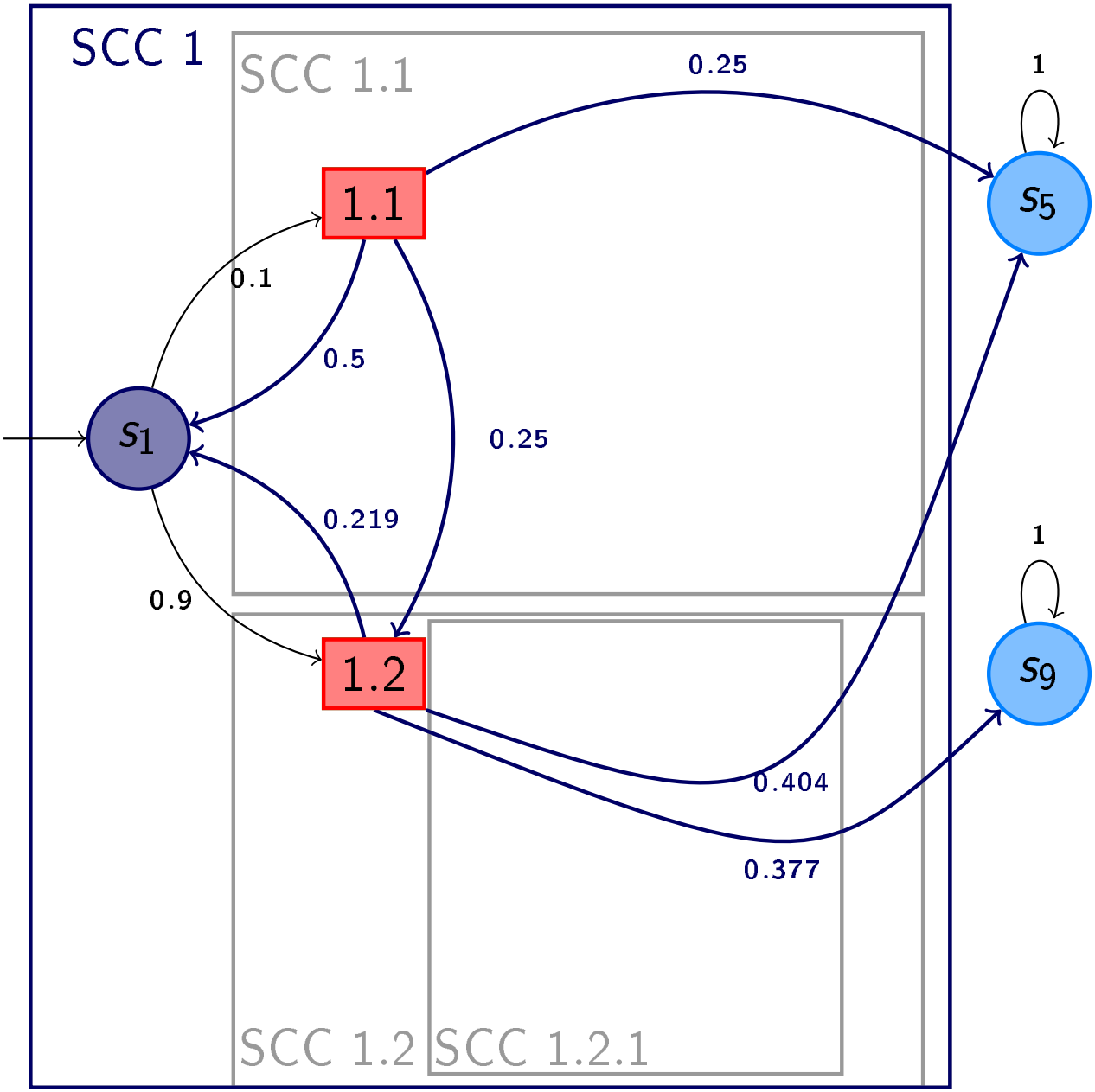
Example



SCC 1.1:

- Reduced to abstract node.

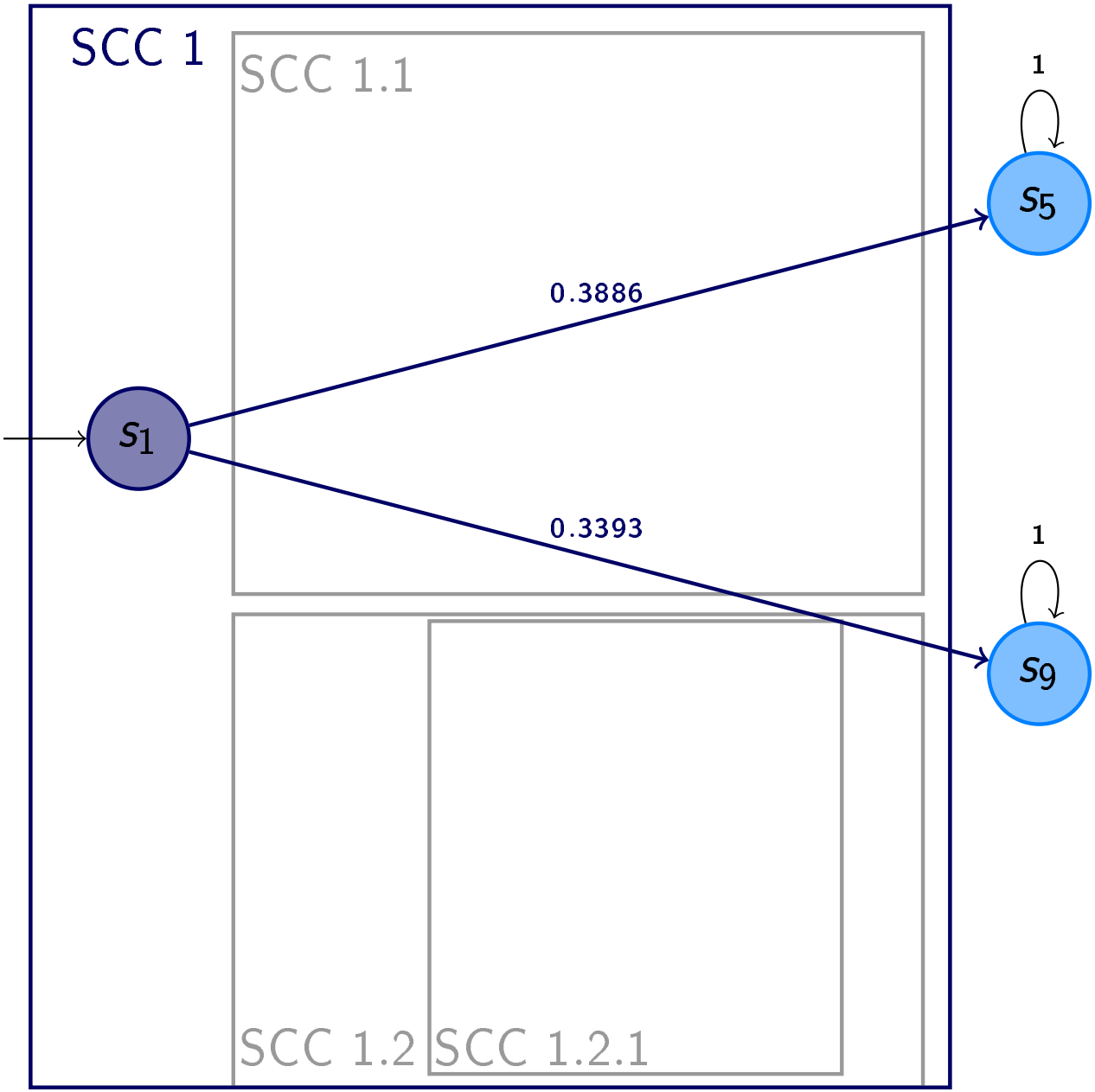
Example



SCC 1:

- Compute probabilities for reaching output nodes from input node.

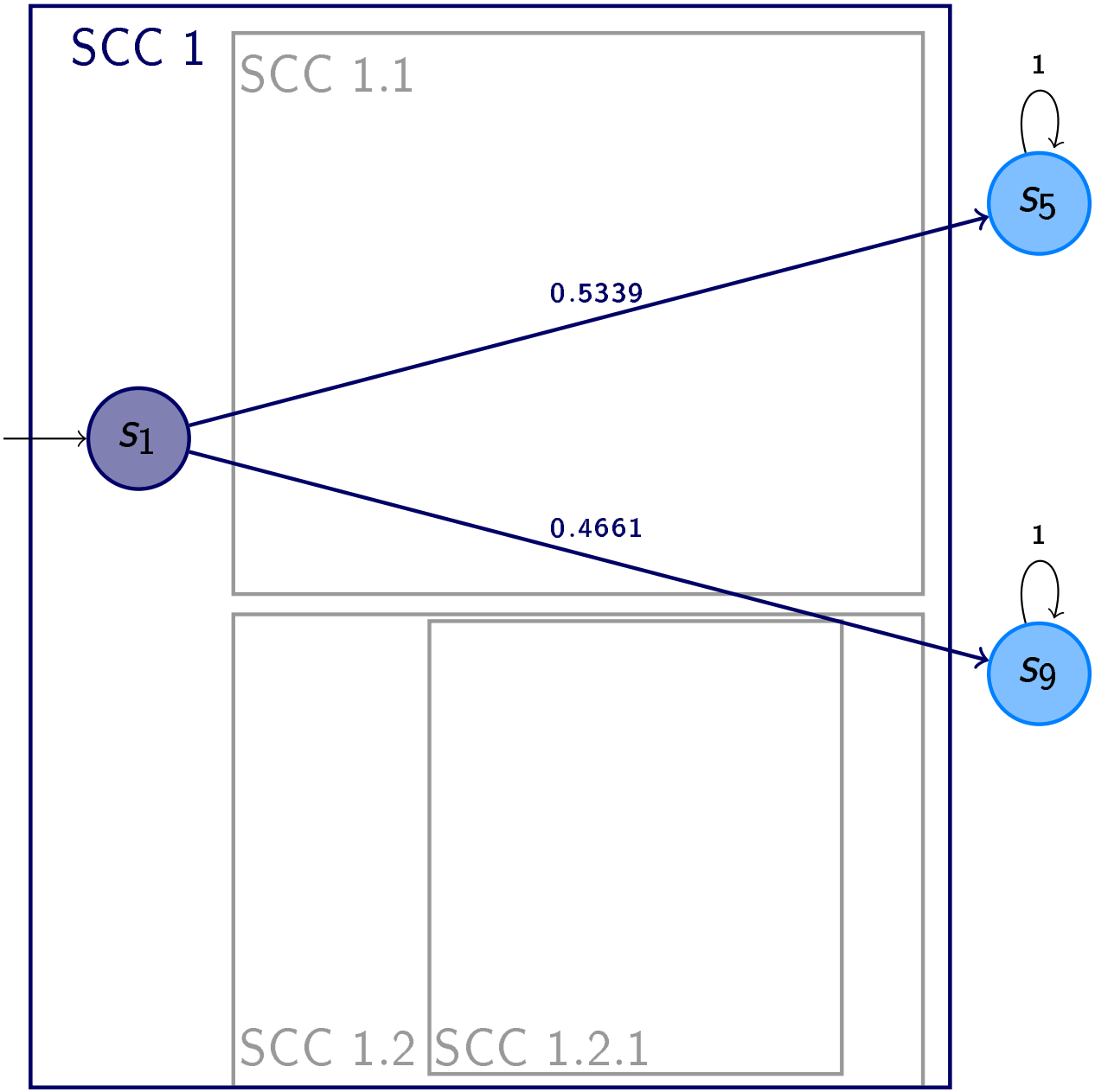
Example



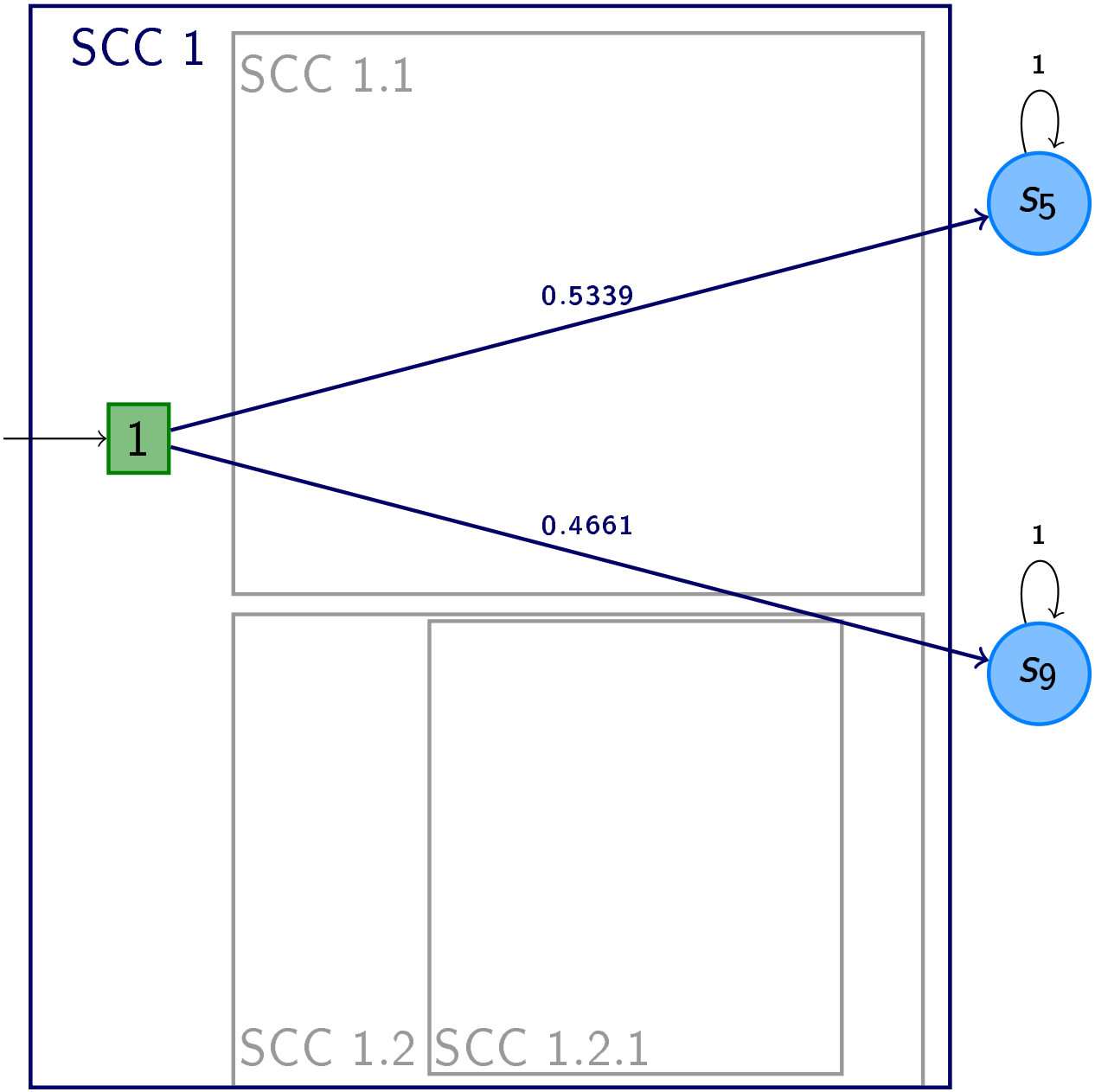
SCC 1:

- Scale probabilities.

Example



Example



SCC 1:

- Reduced to abstract node.
- Model Checking result for reachability

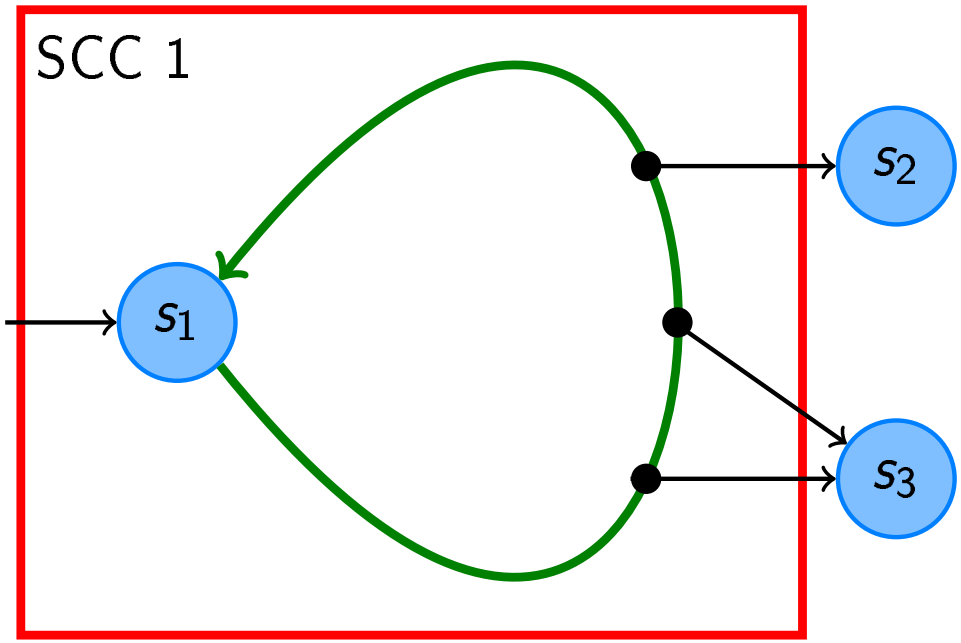
Recursive Algorithm for SCC-Abstraction

```
void SCC_Abstraction(MarkovChain G) {
 searchForSCCs(G);
 if (SCCs_found) {
  for all SCCs scc {
   ignoreInputNode(scc);
   SCC_Abstraction(scc);
   }
 }
 else {
  computeReachabilityProbabilitiesToOutputNodes();
  insertResults();
 }
}
```

Contents

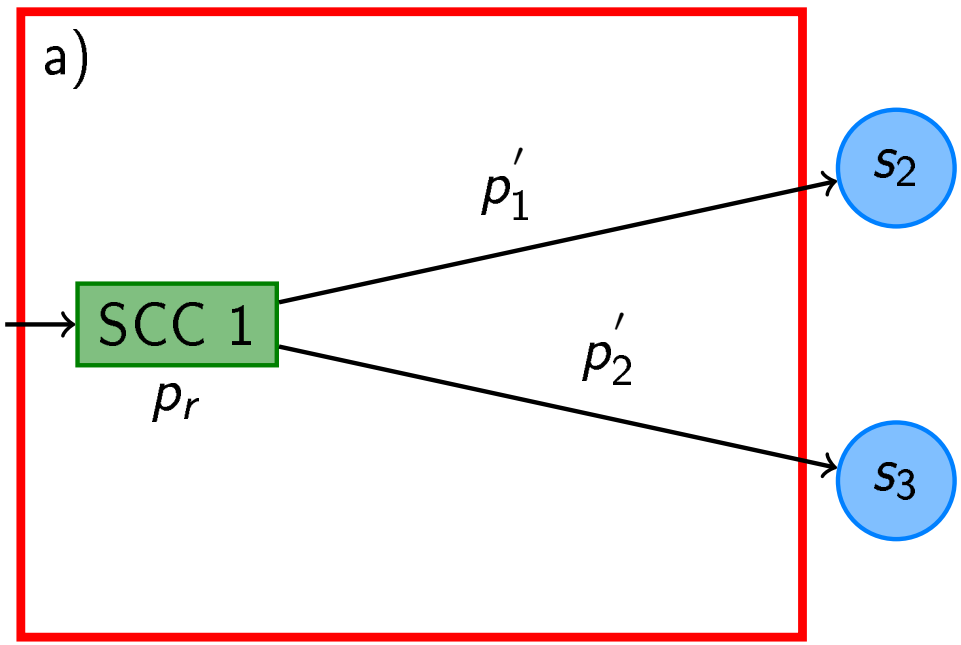
Towards Counterexamples

- Information of original Markov Chain is saved.
- SCCs are represented by abstract nodes.
- Abstract paths may contain both abstract and concrete nodes.



Towards Counterexamples

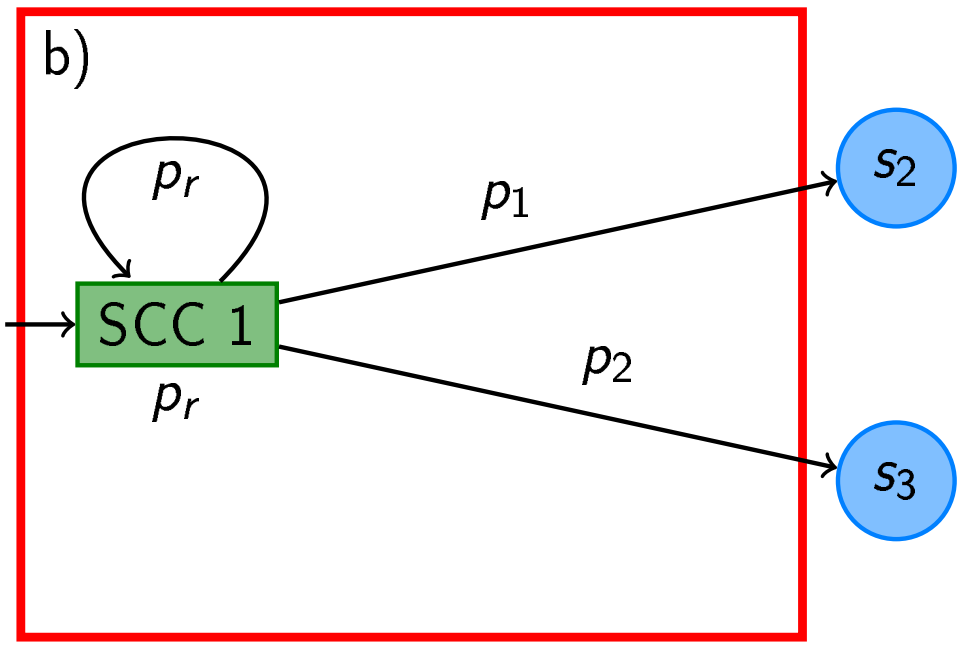
- Information of original Markov Chain is saved.
- SCCs are represented by abstract nodes.
- Abstract paths may contain both abstract and concrete nodes.
  a) Probability mass for all paths walking through SCC



- Previously calculated scaled outgoing probabilities p_1', p_2'
- Saved probability p_r for coming back to ingoing node

Towards Counterexamples

- Information of original Markov Chain is saved.
- SCCs are represented by abstract nodes.
- Abstract paths may contain both abstract and concrete nodes.
  a) Probability mass for all paths walking through SCC
  b) Probabilities for returning to ingoing node and leaving



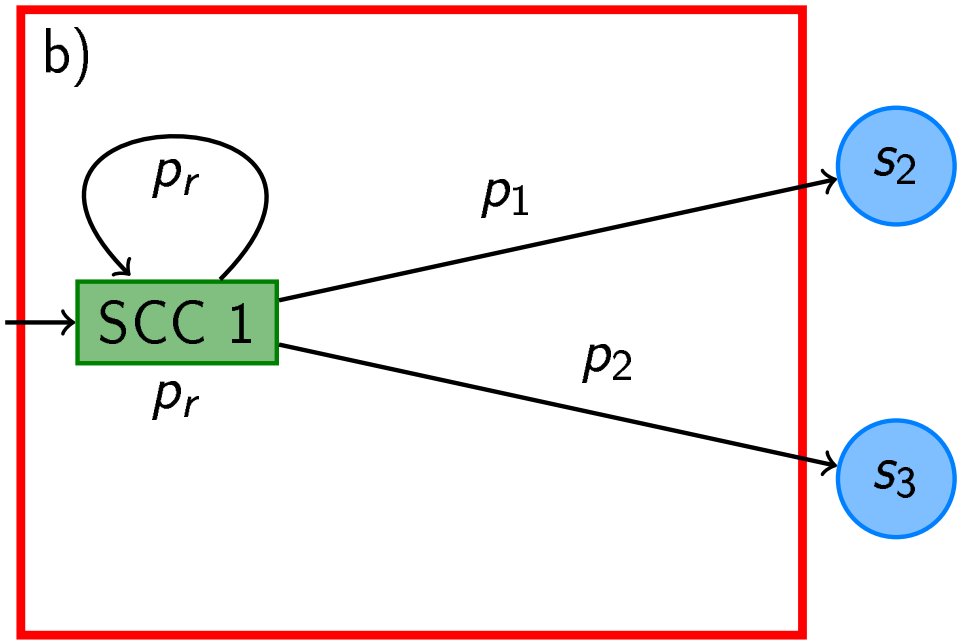
- Unscaled probabilities are computed

$$p_1 = (1 - p_r) \cdot p_1'$$

$$p_2 = (1 - p_r) \cdot p_2'$$

Towards Counterexamples

- Information of original Markov Chain is saved.
- SCCs are represented by abstract nodes.
- Abstract paths may contain both abstract and concrete nodes.
  a) Probability mass for all paths walking through SCC
  b) Probabilities for returning to ingoing node and leaving
- Interactive counterexample refinement

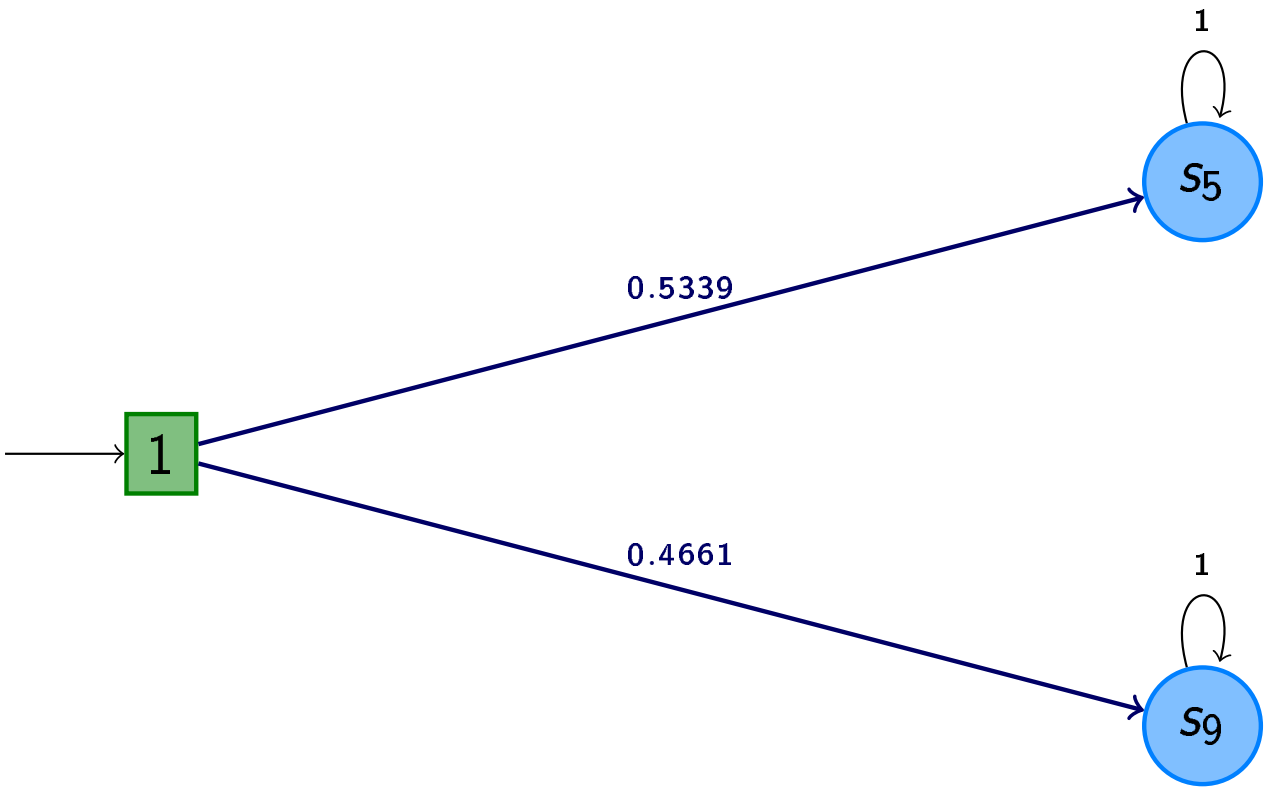


- Unscaled probabilities are computed

$$p_1 = (1 - p_r) \cdot p_1'$$

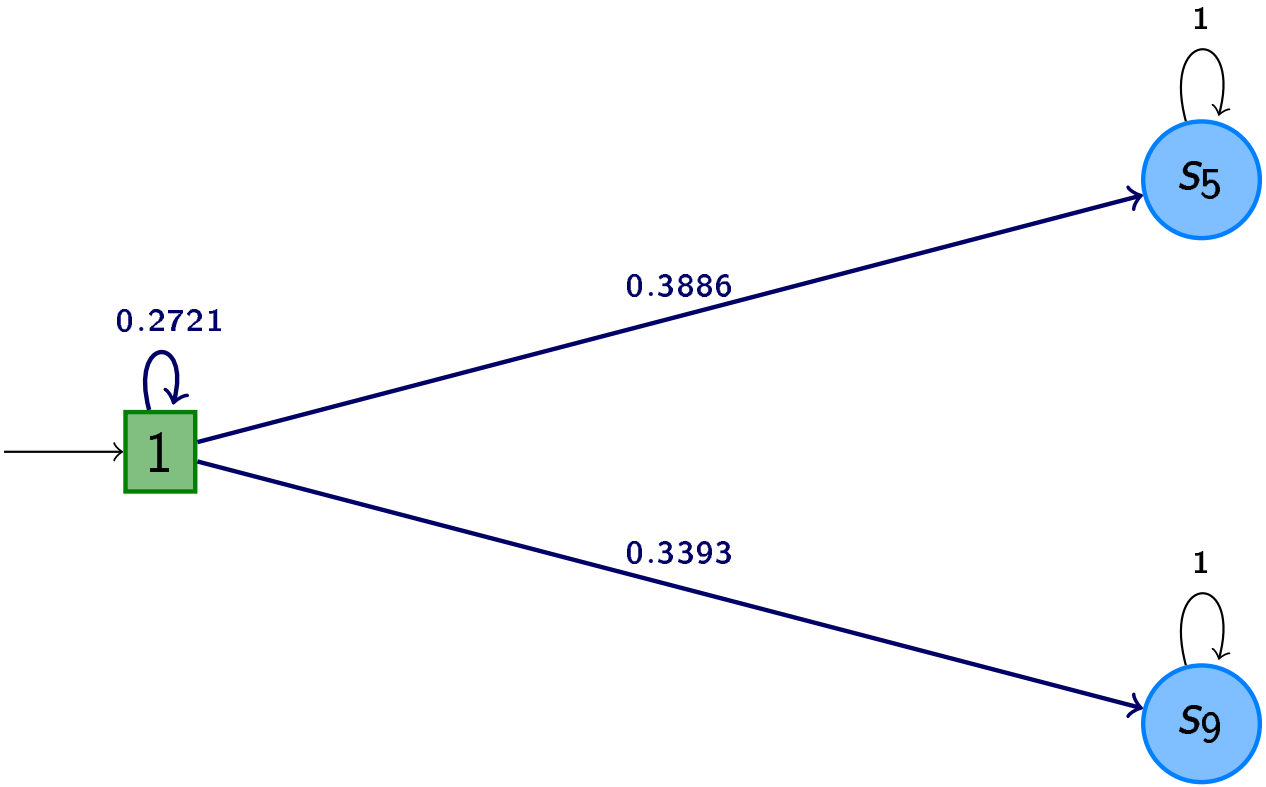
$$p_2 = (1 - p_r) \cdot p_2'$$

Interactive Counterexample Refinement - Example



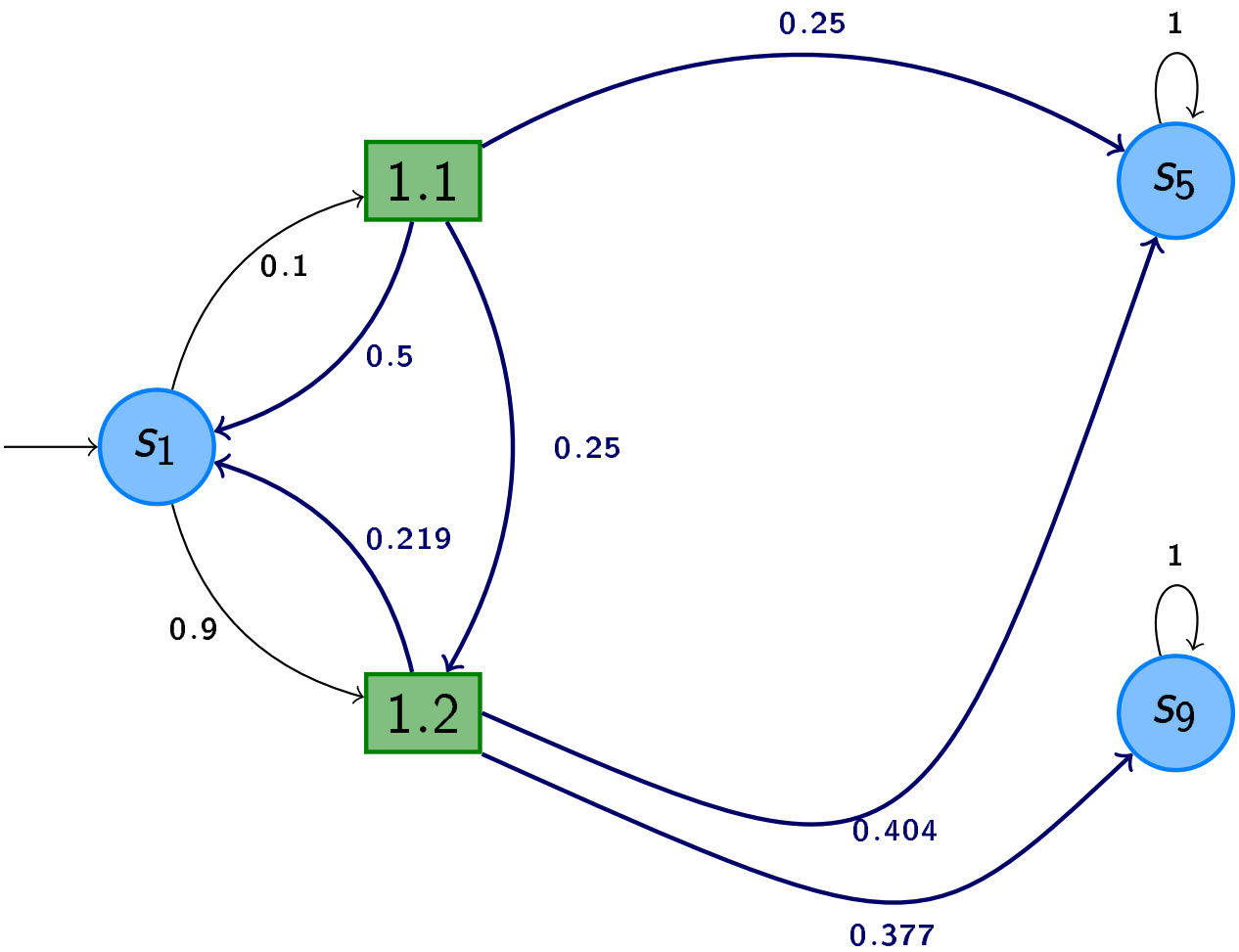
Property to refute:
Probability of reaching s_5 is less or equal than 0.3

Interactive Counterexample Refinement - Example



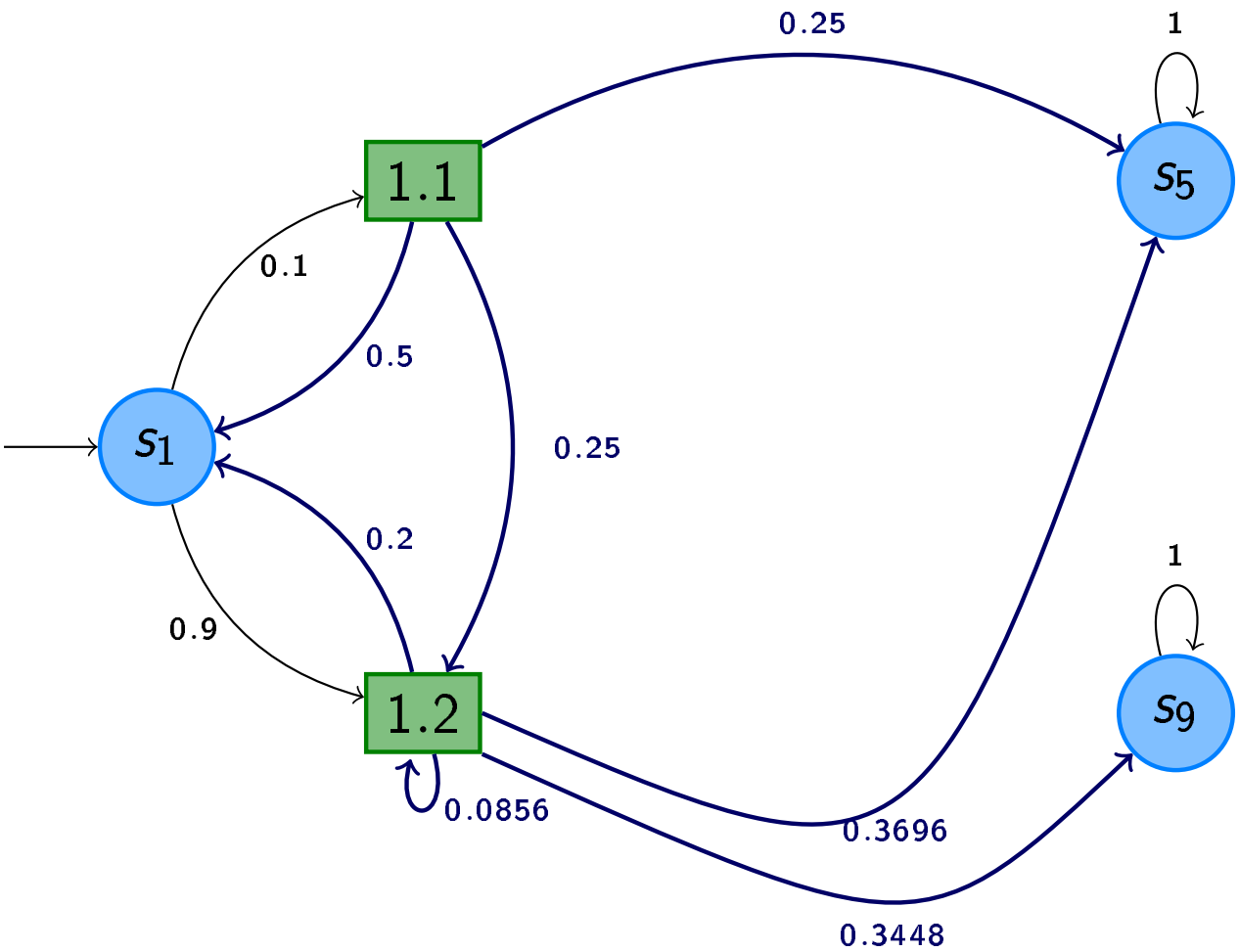
Returning to s_1 is not needed

Interactive Counterexample Refinement - Example



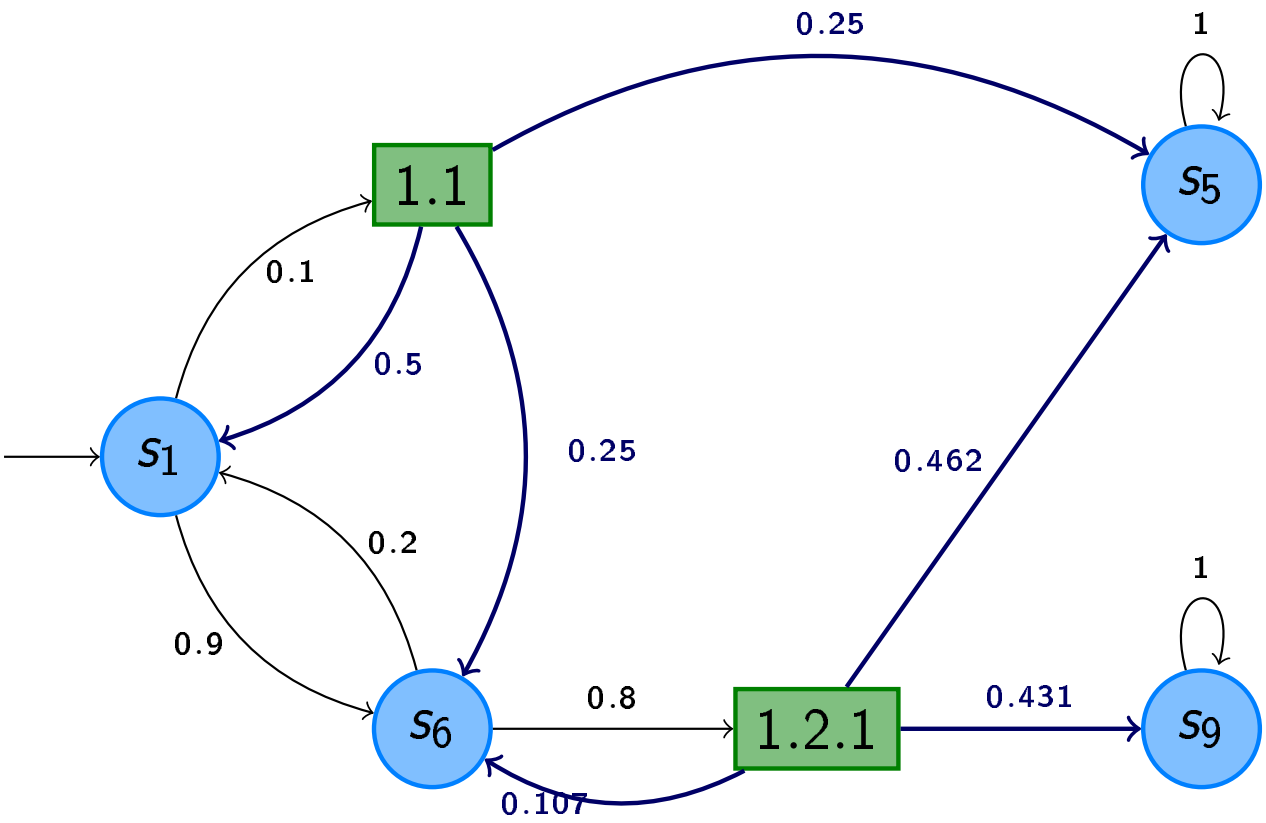
Visit of abstract node 1.1 and hidden SCC is not needed

Interactive Counterexample Refinement - Example



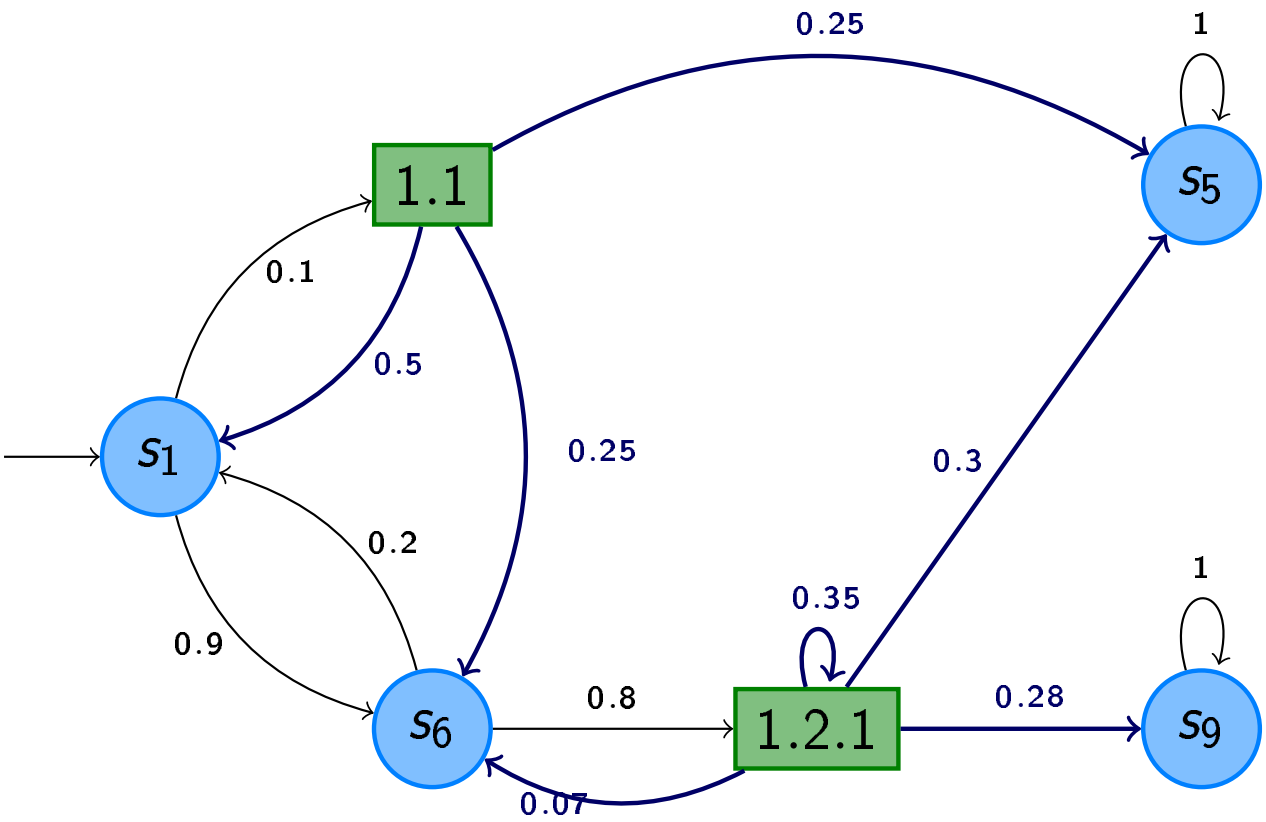
Returning to abstract node 1.2 is not needed

Interactive Counterexample Refinement - Example



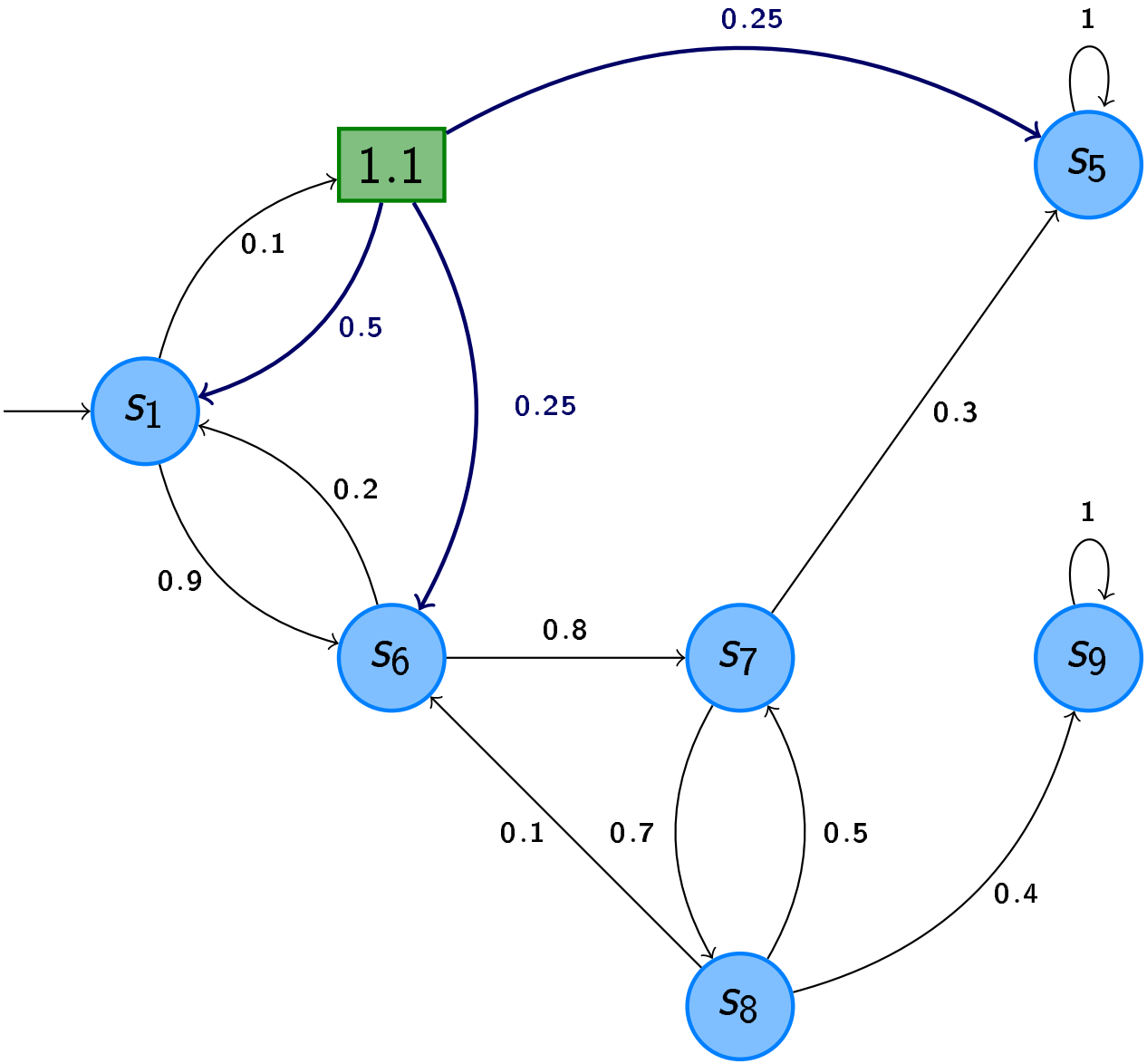
Abstract path
$s_1 \longrightarrow s_6 \longrightarrow 1.2.1 \longrightarrow s_5$
induces probability
mass of 0.33264

Interactive Counterexample Refinement - Example



At least one revisit of 1.2.1 is needed

Interactive Counterexample Refinement - Example



Pathes

$s_1 \rightarrow s_6 \rightarrow s_7 \rightarrow s_5$

$s_1 \rightarrow s_6 \rightarrow s_7 \rightarrow s_8 \rightarrow s_7 \rightarrow s_5$

$s_1 \rightarrow s_6 \rightarrow s_7 \rightarrow s_8 \rightarrow s_7 \rightarrow s_8 \rightarrow s_7 \rightarrow s_5$

induce probability mass 0.31806 and are therefore sufficient as counterexample.

Perspectives

SCC-Based Abstraction, Model Checking, and Interactive Counterexample Refinement

- Recursive SCC-based algorithm
- Abstract and concrete information is saved
- User can control structure and level of abstraction of counterexamples

Future work

- Automation of counterexample-search
- Structure-based counterexamples
- User-Interface
- Further case-studies