On-the-Fly Emptiness Checks for Generalized Büchi Automata

Jean-Michel Couvreur[1] Alexandre Duret-Lutz[2]
Denis Poitrenaud[2]

[1]LaBRI, Université de Bordeaux 1, Talence, France

[2]LIP6, Université de Paris 6, France

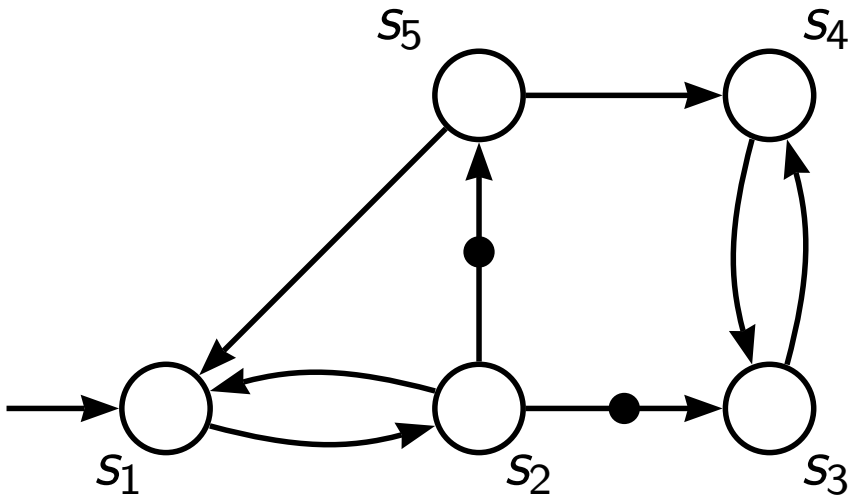
August 2005, 24th

Büchi Automata

A (transition-based) Büchi automaton has:

- A set of states, with a designated initial state,
- A set of transitions between states,
- A set of accepting transitions.

An infinite run of this automaton is accepting if it visits an accepting transition infinitely often.

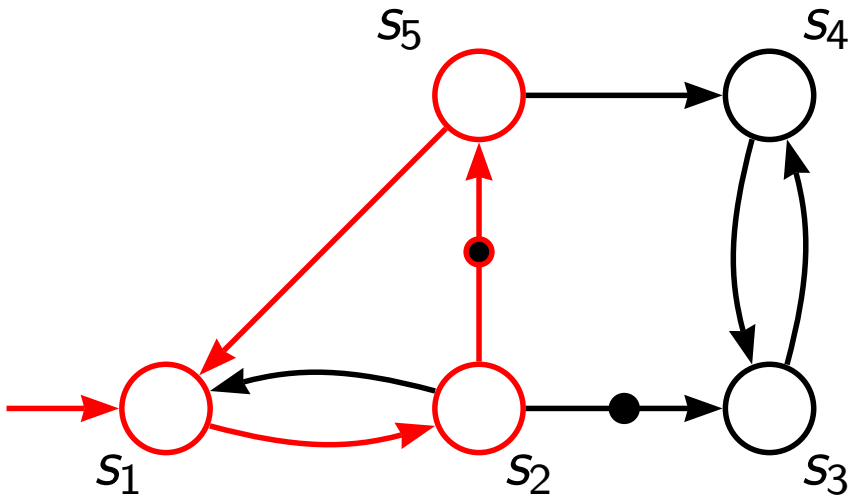


Büchi Automata

A (transition-based) Büchi automaton has:

- A set of states, with a designated initial state,
- A set of transitions between states,
- A set of accepting transitions.

An infinite run of this automaton is accepting if it visits an accepting transition infinitely often.



Emptiness Check

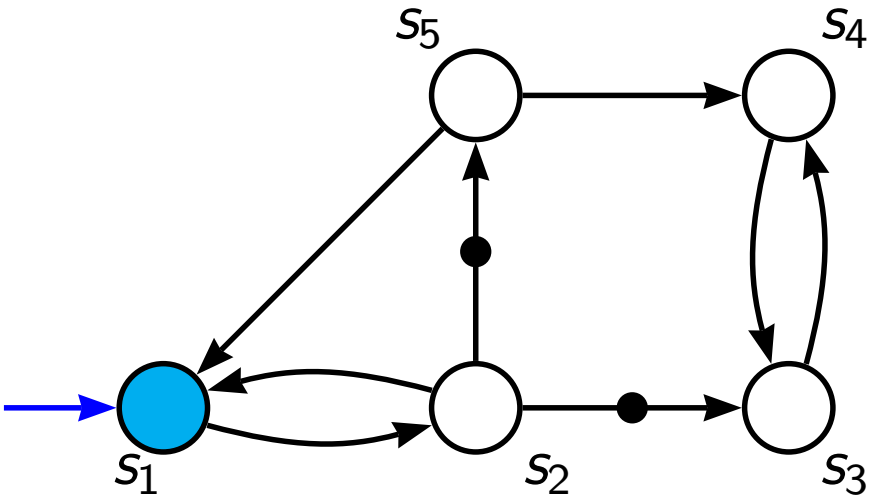
Emptiness Check = Does an automaton have no accepting run?

$\Longrightarrow$ Search for an accepting cycle reachable from the initial state.

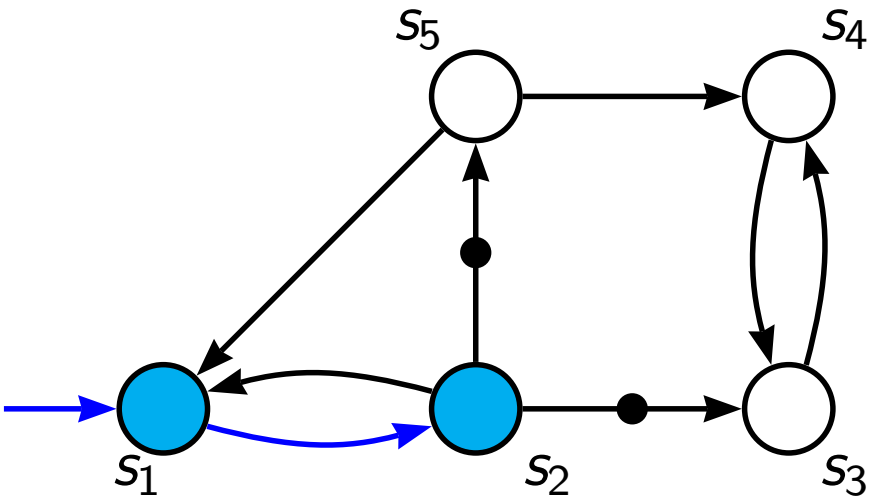
Emptiness Checks History

nested DFS	Courcoubetis et al.	'90
	Godefroid & Holzmann	'93
	Holzmann et al.	'96
	Gastin et al.	'04
	Schwoon & Esparza	'04

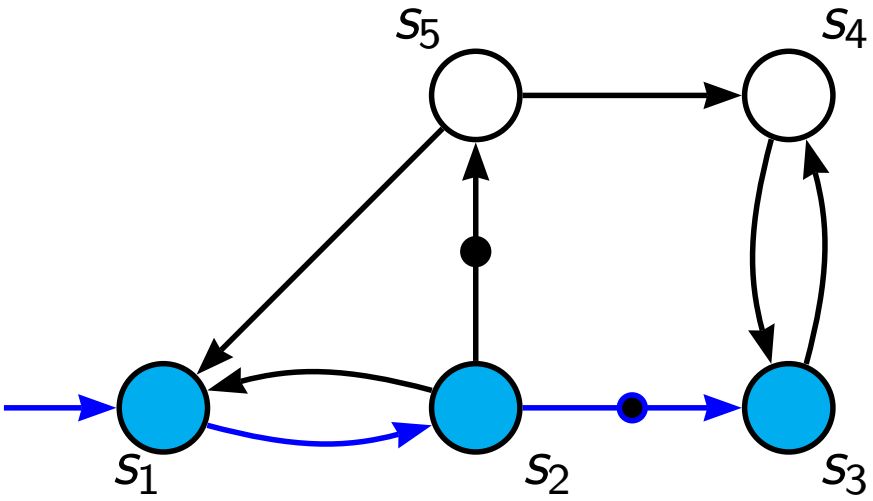
Nested DFS



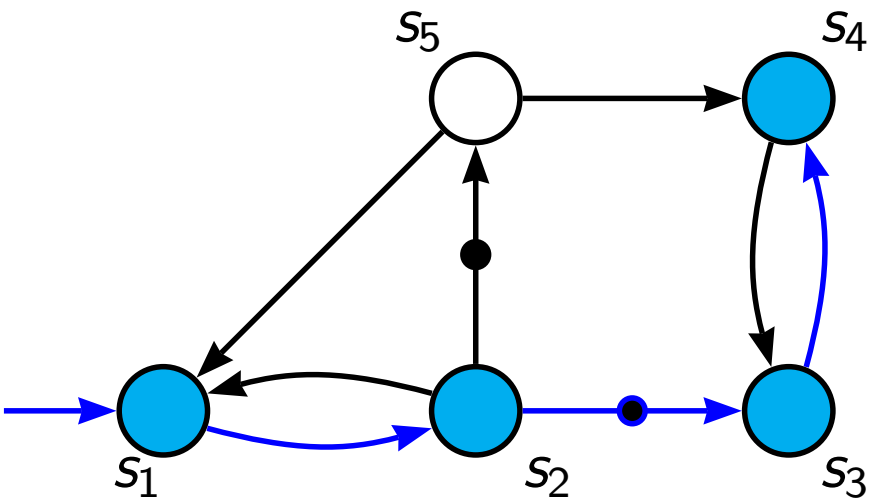
Nested DFS



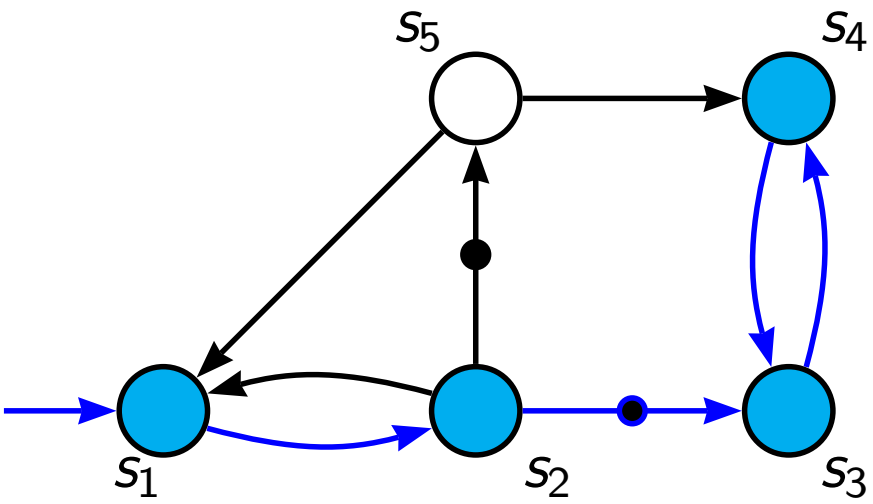
Nested DFS



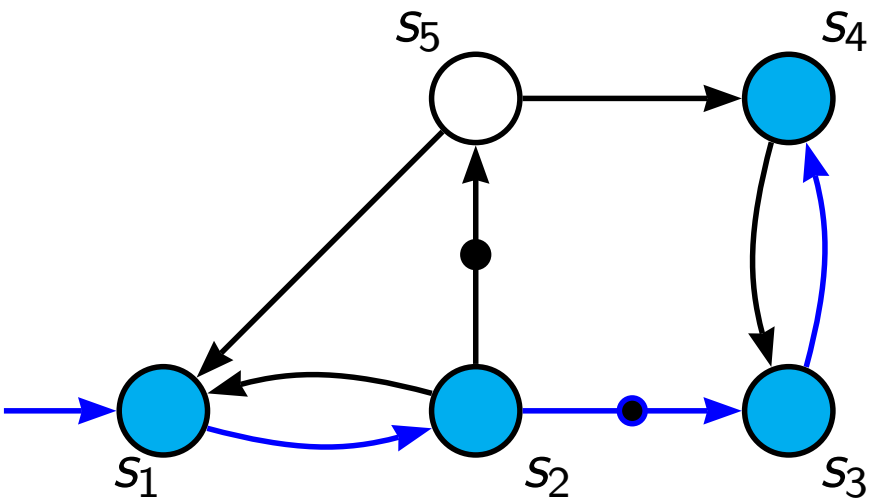
Nested DFS



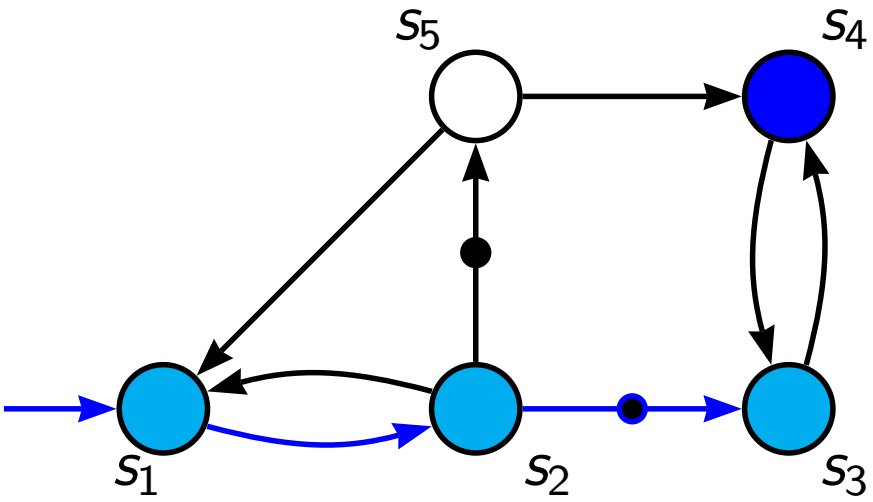
Nested DFS



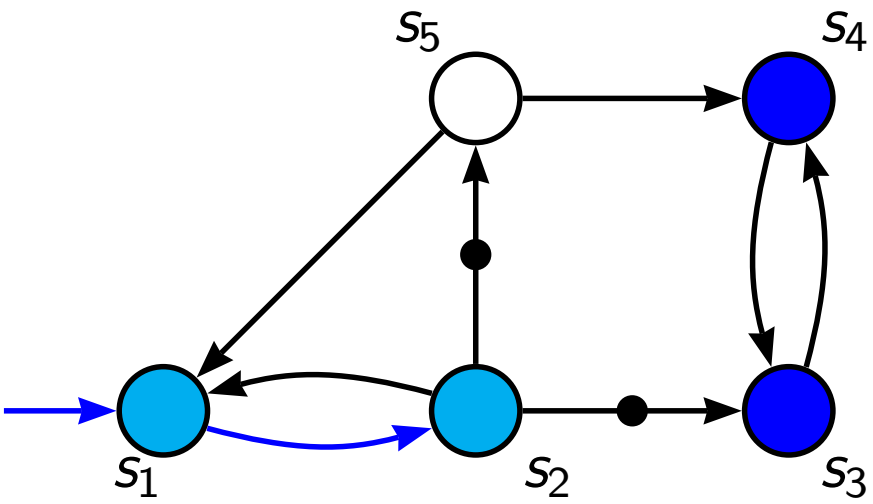
Nested DFS



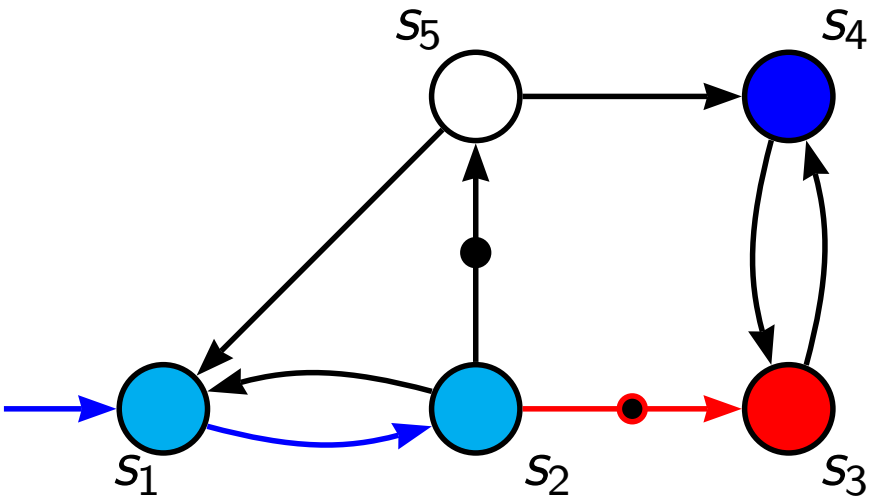
Nested DFS



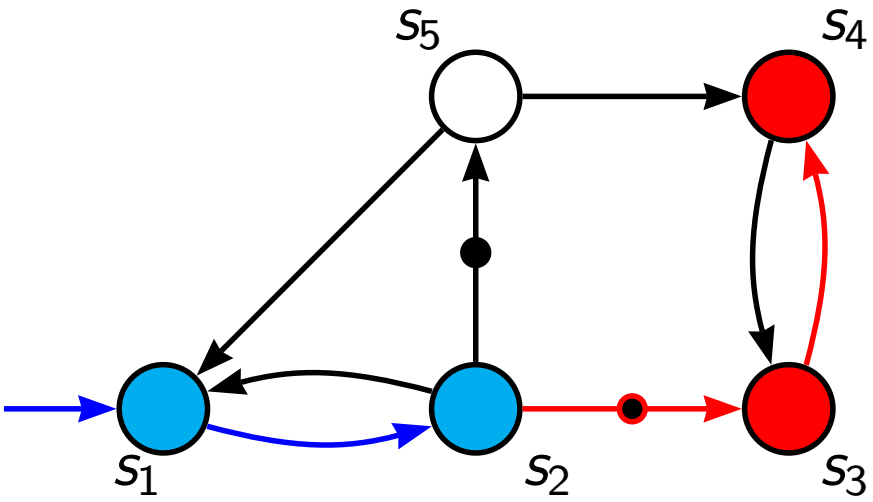
Nested DFS



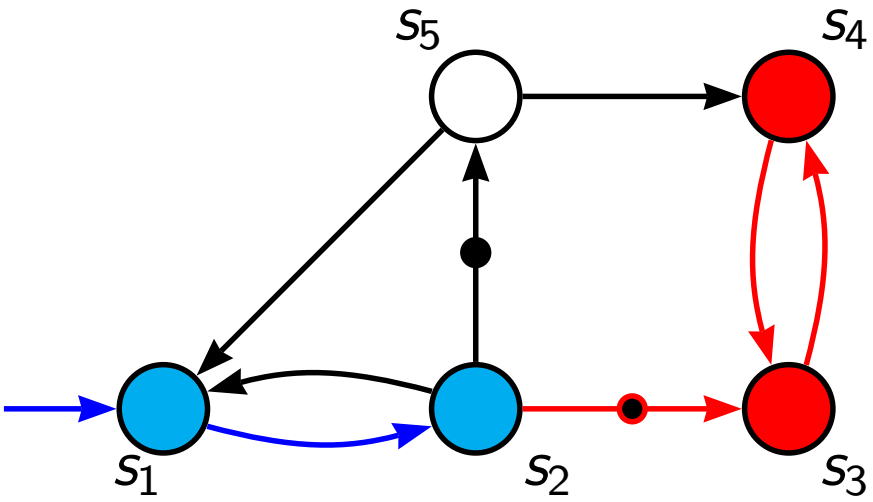
Nested DFS



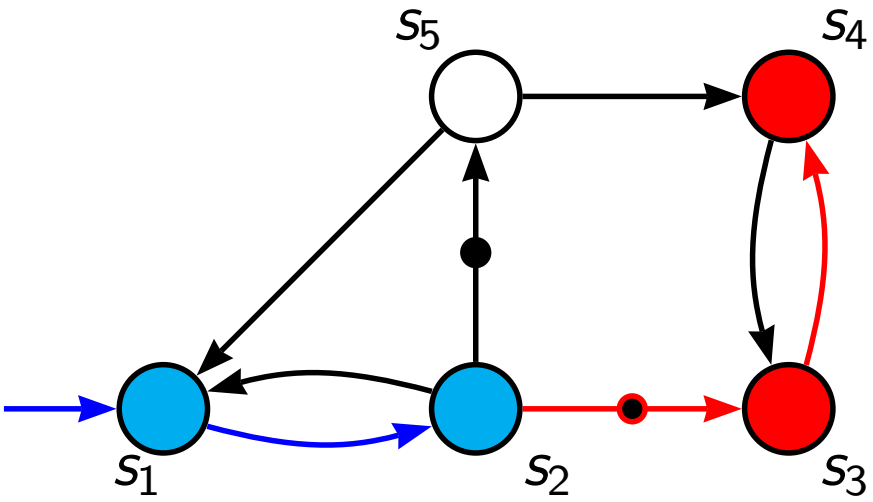
Nested DFS



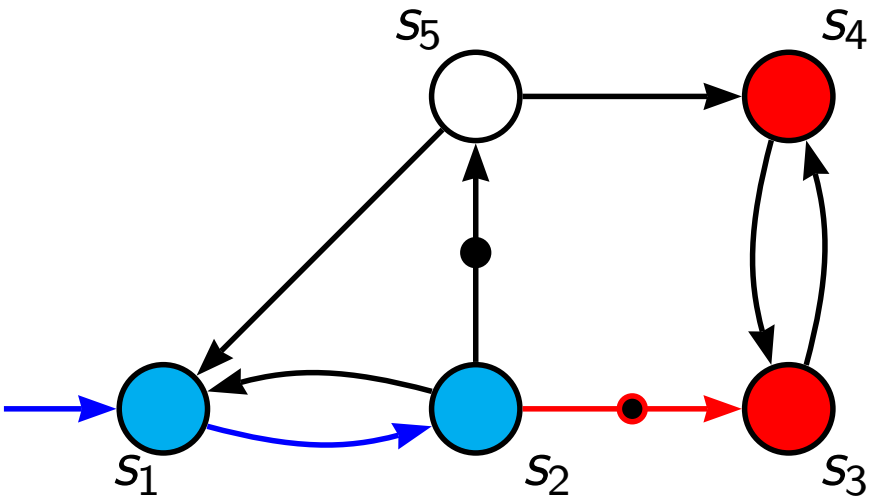
Nested DFS



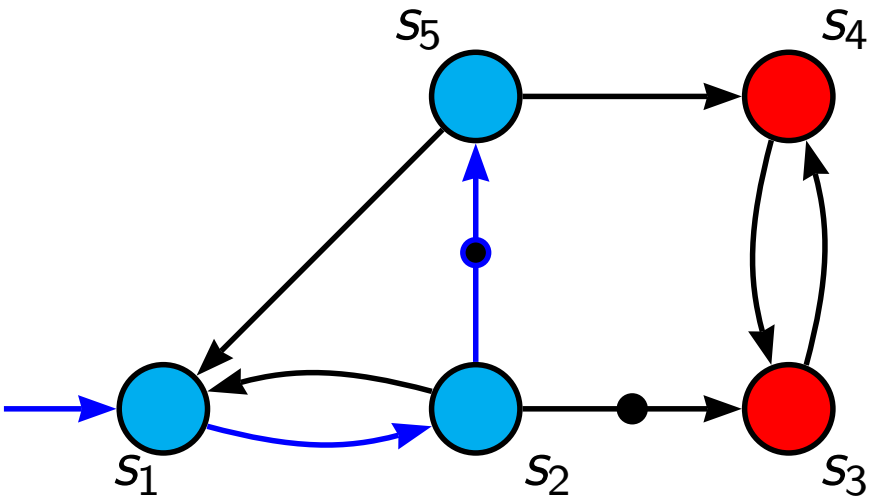
Nested DFS



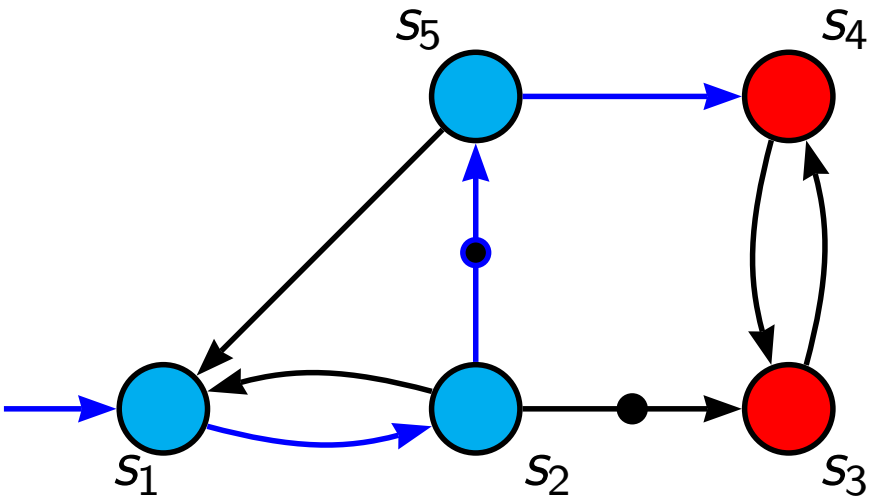
Nested DFS



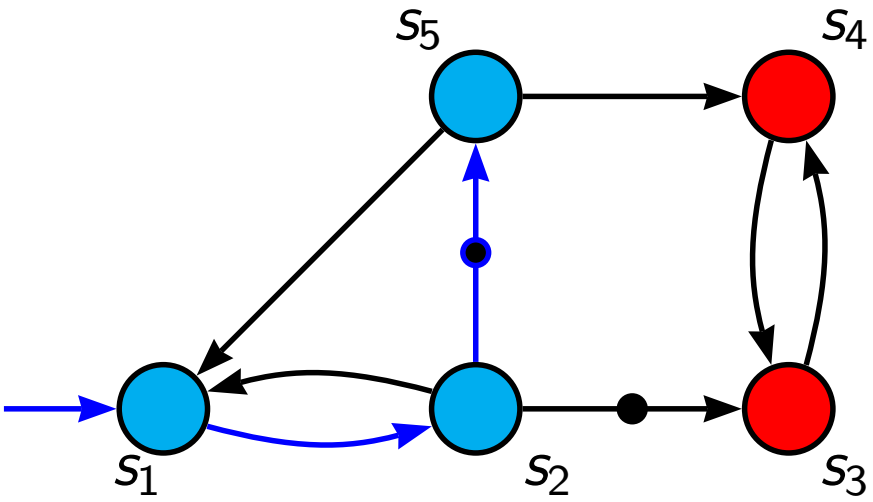
Nested DFS



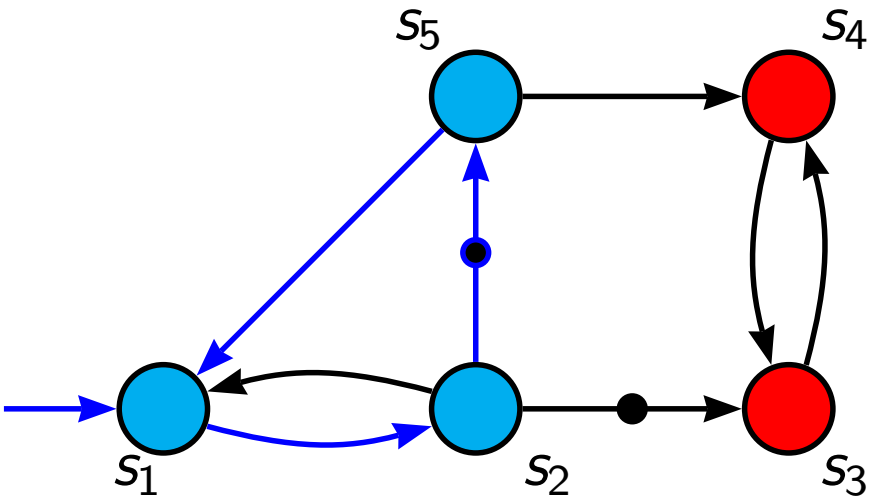
Nested DFS



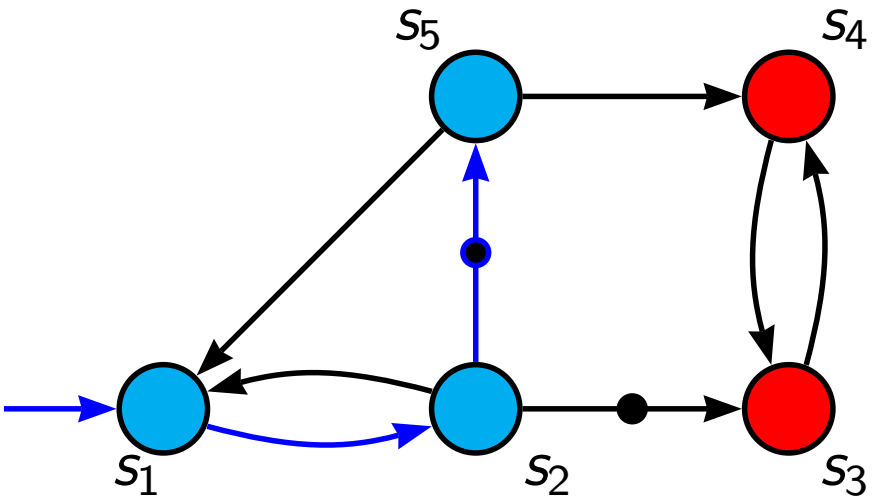
Nested DFS



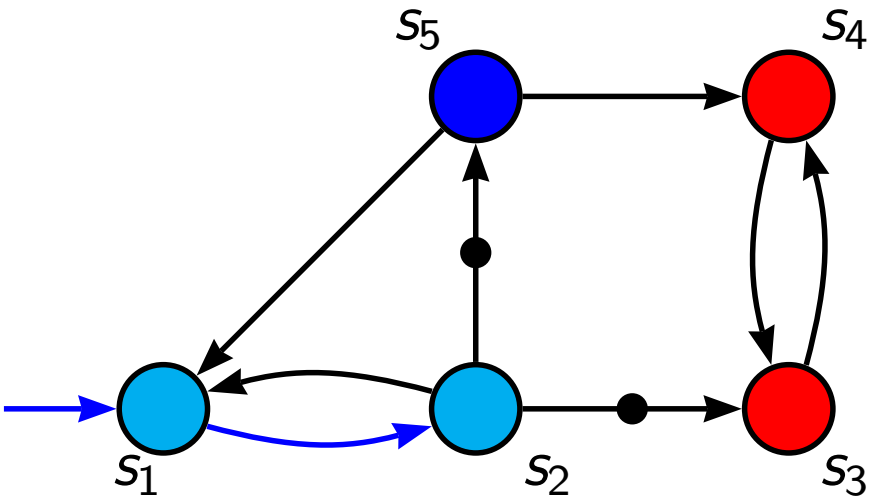
Nested DFS



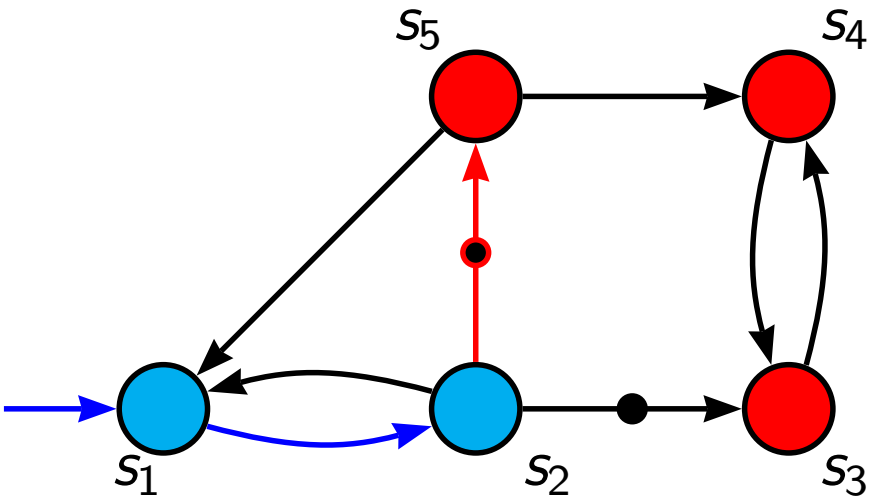
Nested DFS



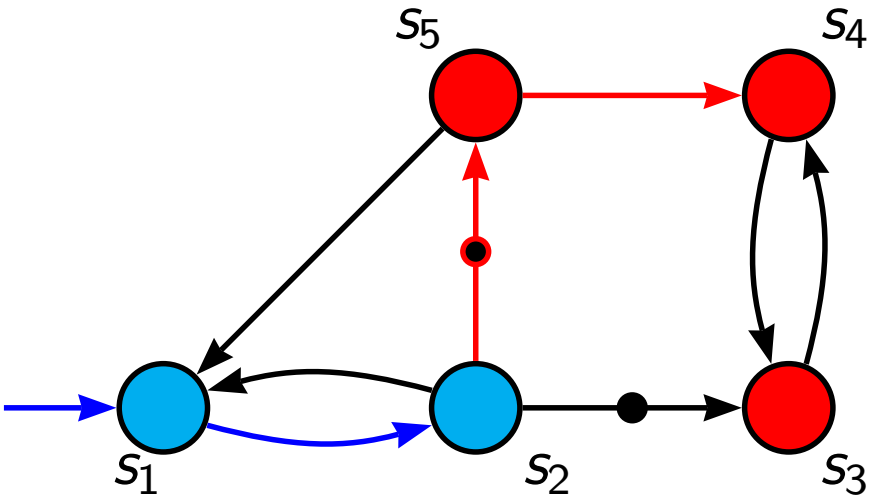
Nested DFS



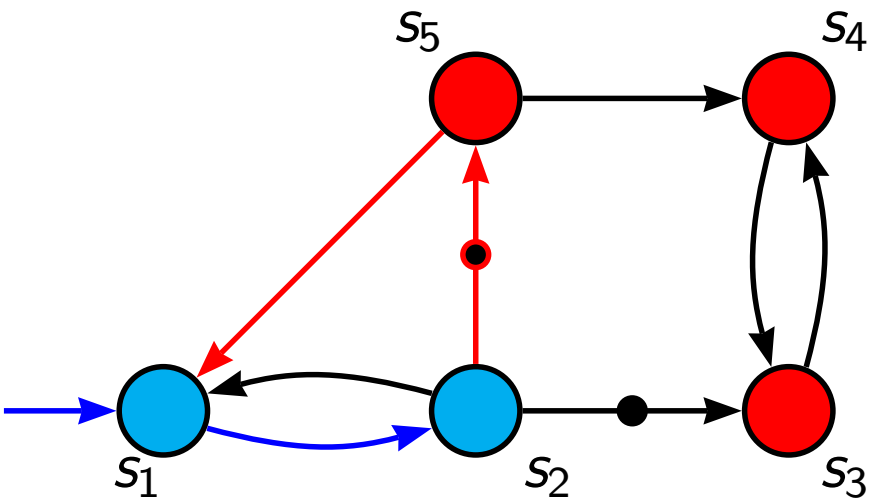
Nested DFS



Nested DFS

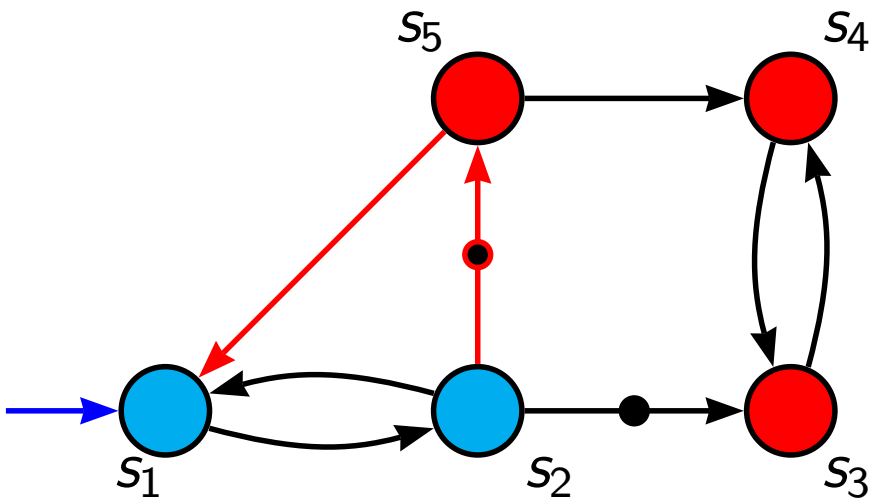


Nested DFS



Found!

Nested DFS



Found!

	entries in hash table	hash table size in bits	search stack depth	states traversed
upper bounds:	n	$n(s+2)$	n	$2n$

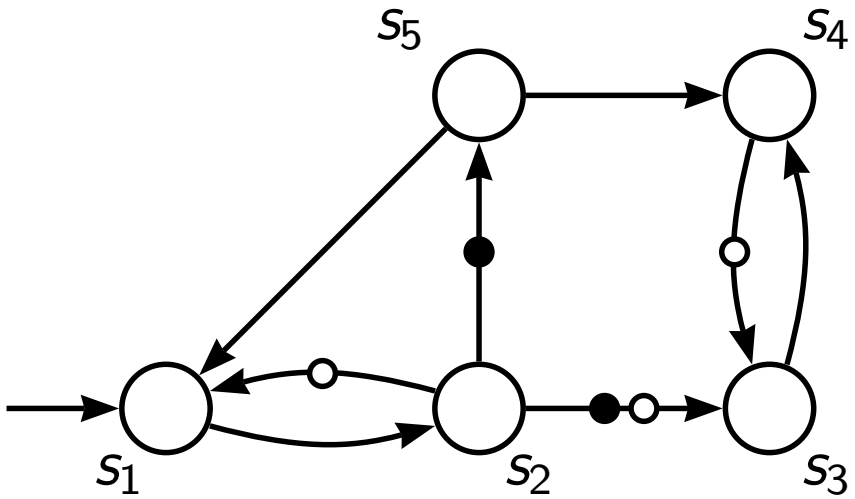
n = number of states; s = bits per state.

Generalized Büchi Automata

A Generalized (transition-based) Büchi automaton has:

- A set of states, with a designated initial state,
- A set of transitions between states,
- A set of accepting sets of transitions.

An infinite run of this automaton is accepting if it visits a transition from each accepting set infinitely often.

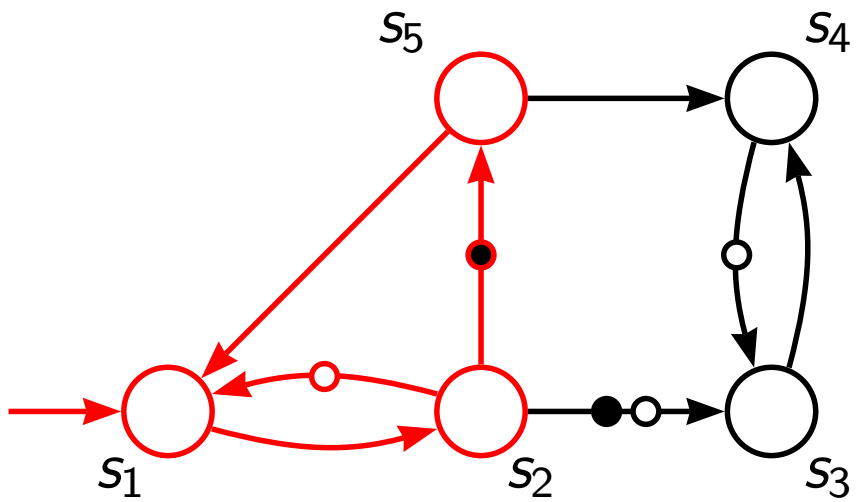


Generalized Büchi Automata

A Generalized (transition-based) Büchi automaton has:

- A set of states, with a designated initial state,
- A set of transitions between states,
- A set of accepting sets of transitions.

An infinite run of this automaton is accepting if it visits a transition from each accepting set infinitely often.



Degeneralization

A generalized automaton with

- n states
- m acceptance conditions

can be degeneralized into an automaton with

- nm states at worst
- 1 acceptance condition

Nested DFS on Generalized Büchi Automata

entries in hash table	hash table size in bits	search stack depth	states traversed
n	$n \quad (s \quad + 2)$	n	$2n$

n states,

s bits per state.

Nested DFS on Generalized Büchi Automata

entries in hash table	hash table size in bits	search stack depth	states traversed
nm	$nm(s_d + 2)$	nm	$2nm$

n states, m acceptance conditions,
s_d bits per degeneralized state.

Nested DFS on Generalized Büchi Automata

entries in hash table	hash table size in bits	search stack depth	states traversed
nm	$nm(s_d + 2)$	nm	$2nm$
n	$n(s_g + 2m)$	nm	$2nm$

n states, m acceptance conditions,
s_d bits per degeneralized state, s_g bits per generalized state ($s_g \leq s_d$).

Emptiness Checks History

nested DFS	Courcoubetis et al.	'90
	Godefroid & Holzmann	'93
	Holzmann et al.	'96
	Gastin et al.	'04
	Schwoon & Esparza	'04

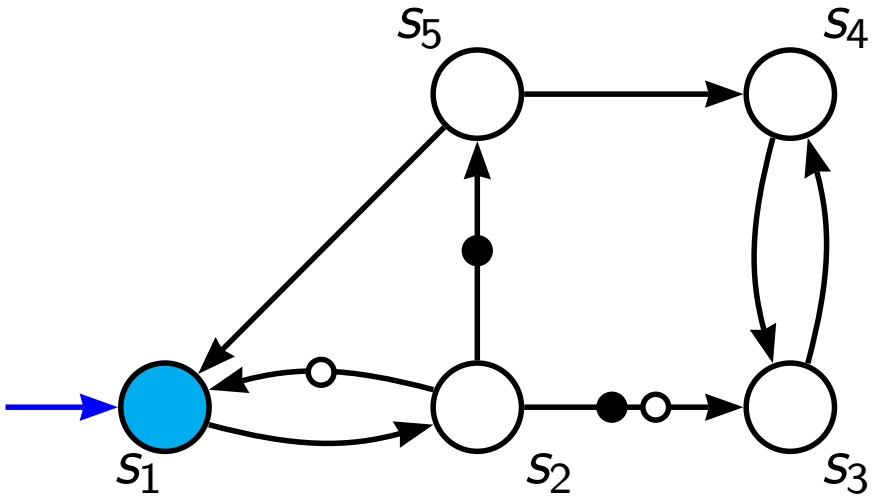
Emptiness Checks History

	degeneralized		generalized
nested DFS	Courcoubetis et al.	'90	
	Godefroid & Holzmann	'93	
	Holzmann et al.	'96	
	Gastin et al.	'04	
	Schwoon & Esparza	'04	

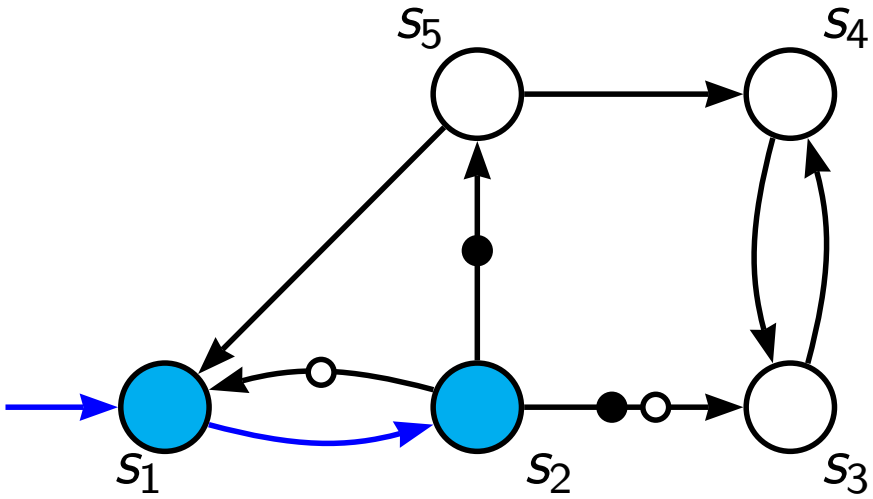
Emptiness Checks History

	degeneralized		generalized	
nested DFS	Courcoubetis et al.	'90		
	Godefroid & Holzmann	'93		
	Holzmann et al.	'96	Tauriainen	'03
	Gastin et al.	'04		
	Schwoon & Esparza	'04		

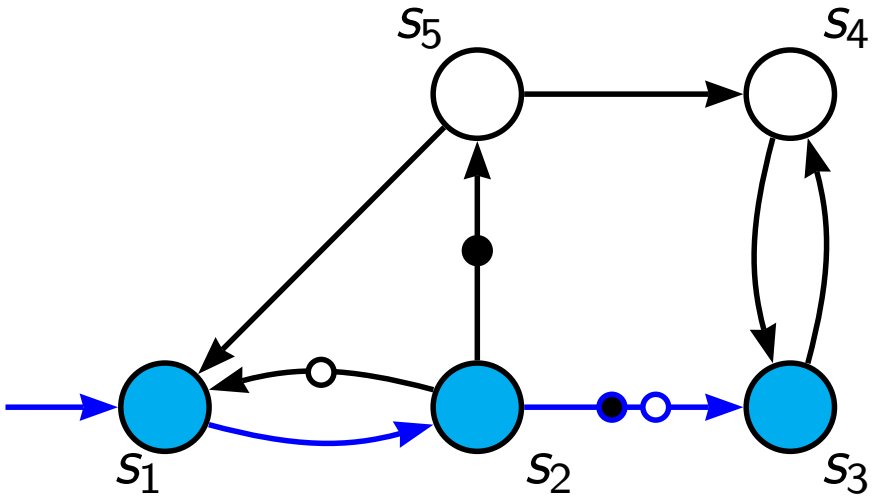
Generalized Nested DFS (Tauriainen'03)



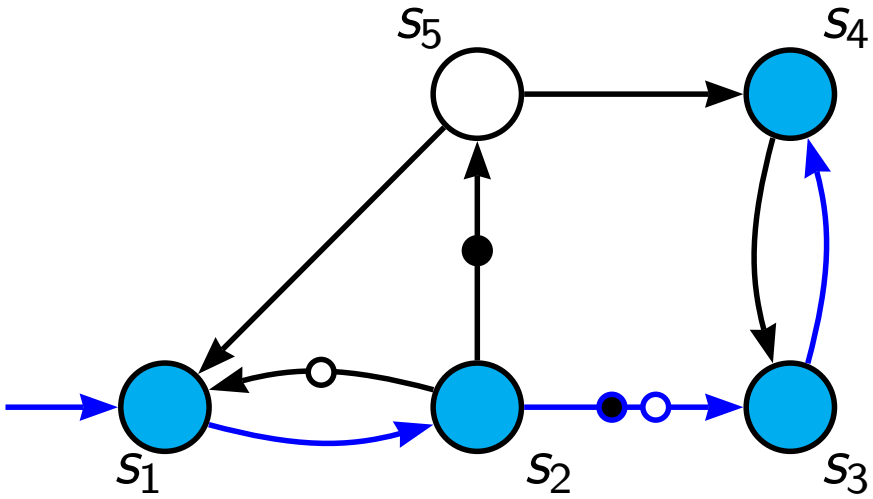
Generalized Nested DFS (Tauriainen'03)



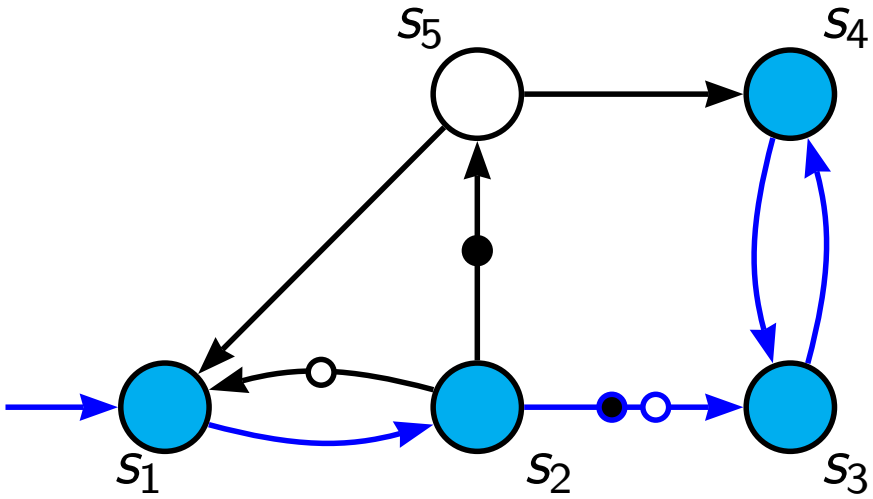
Generalized Nested DFS (Tauriainen'03)



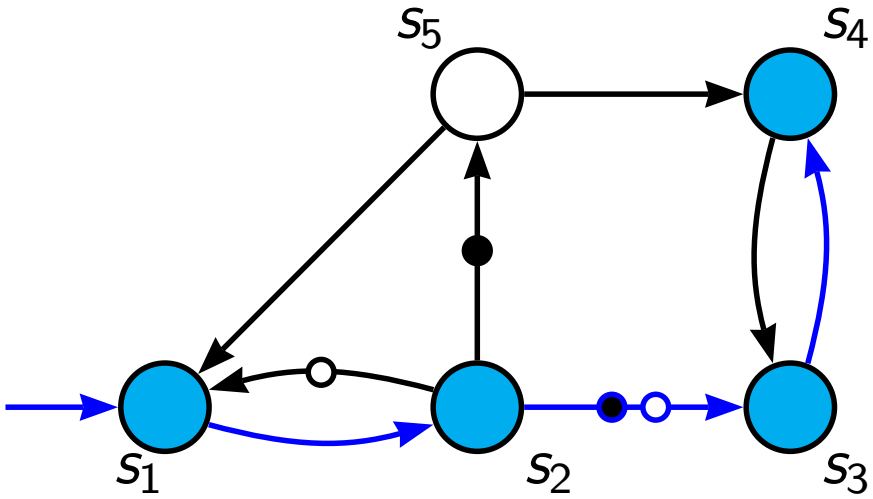
Generalized Nested DFS (Tauriainen'03)



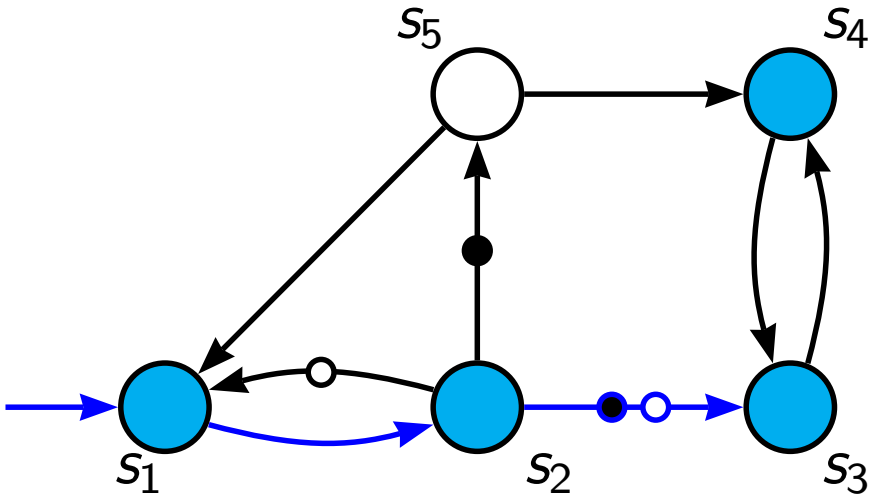
Generalized Nested DFS (Tauriainen'03)



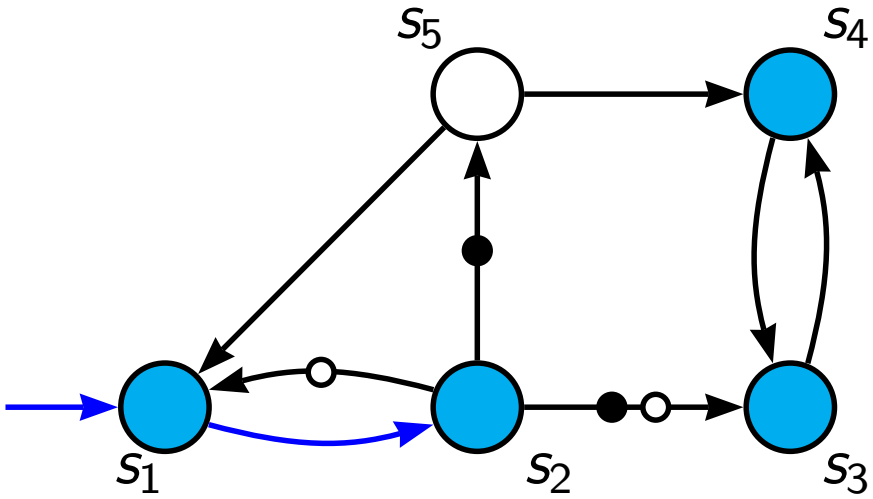
Generalized Nested DFS (Tauriainen'03)



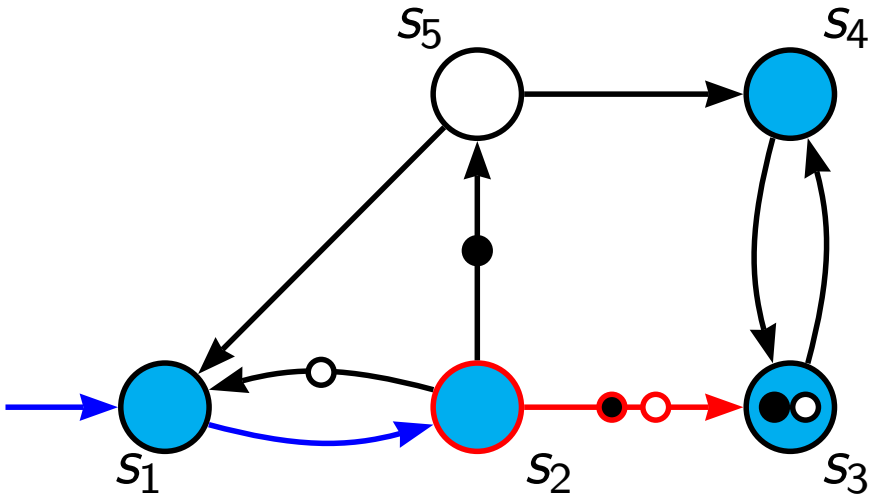
Generalized Nested DFS (Tauriainen'03)



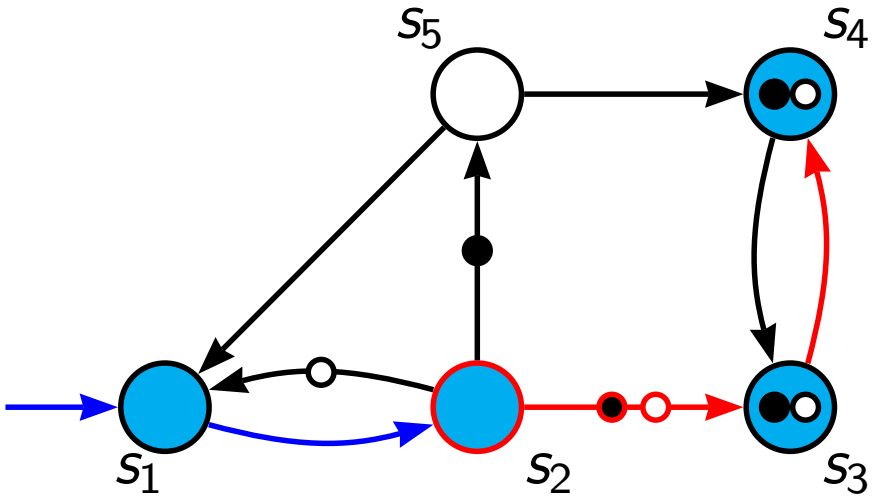
Generalized Nested DFS (Tauriainen'03)



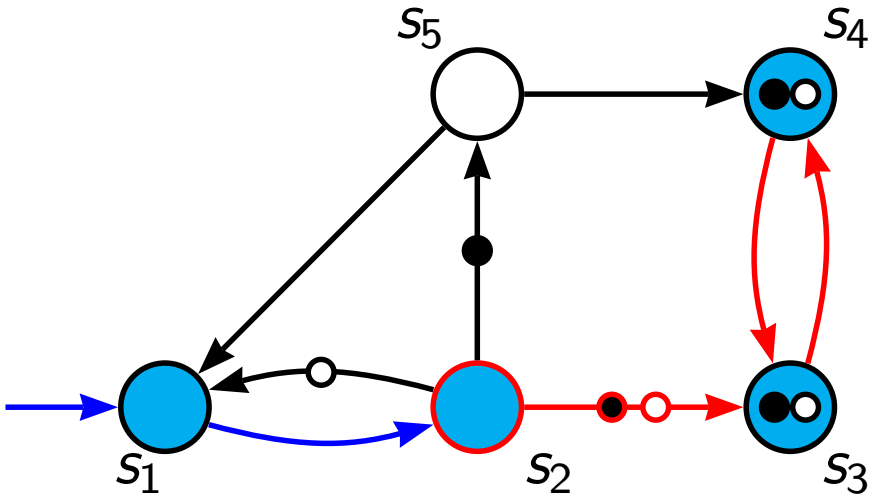
Generalized Nested DFS (Tauriainen'03)



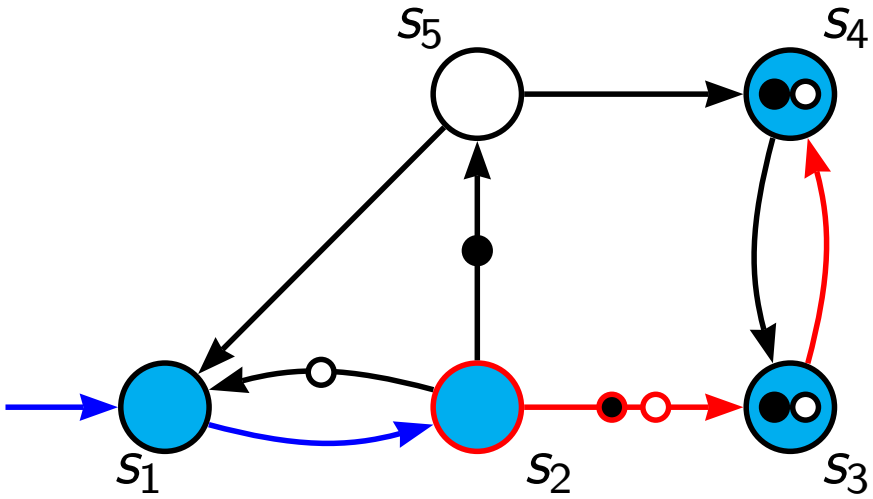
Generalized Nested DFS (Tauriainen'03)



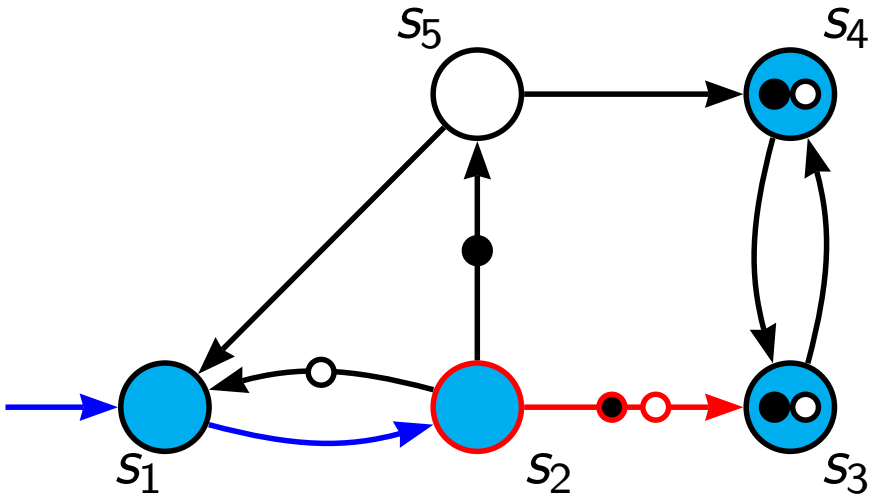
Generalized Nested DFS (Tauriainen'03)



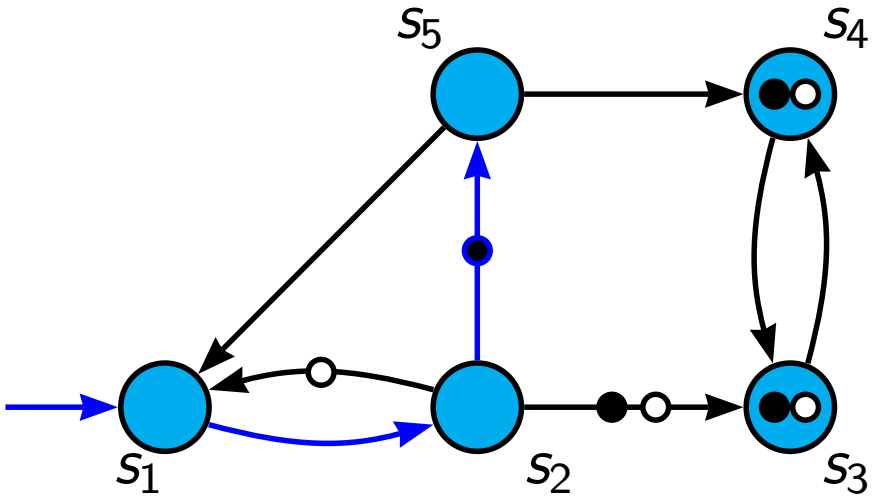
Generalized Nested DFS (Tauriainen'03)



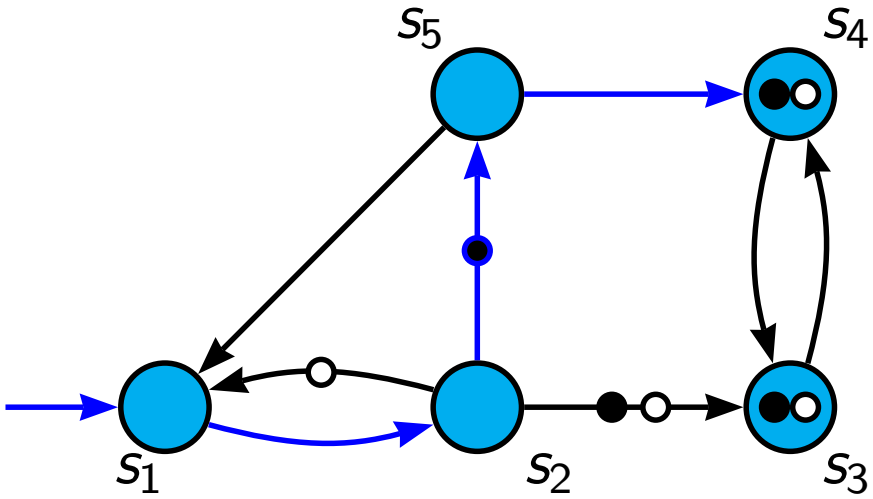
Generalized Nested DFS (Tauriainen'03)



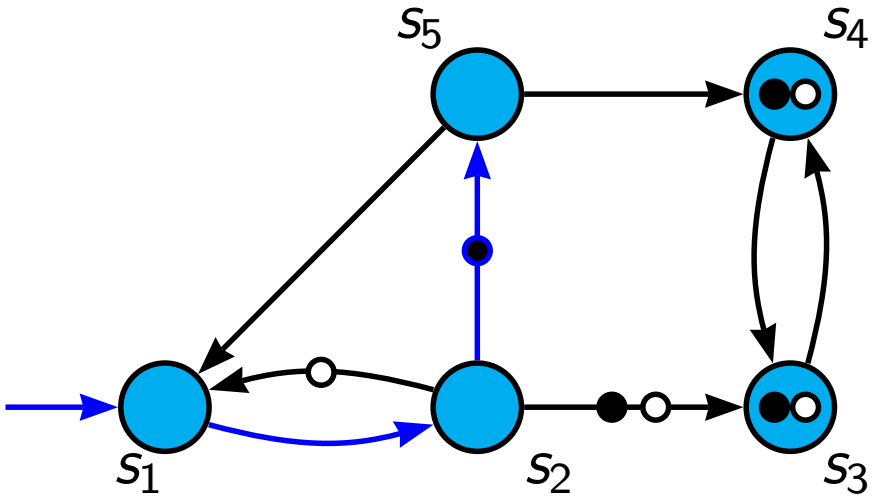
Generalized Nested DFS (Tauriainen'03)



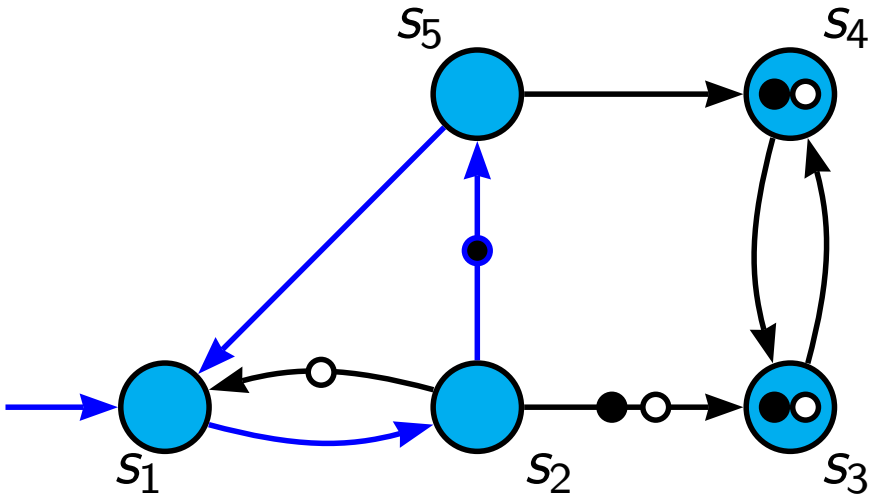
Generalized Nested DFS (Tauriainen'03)



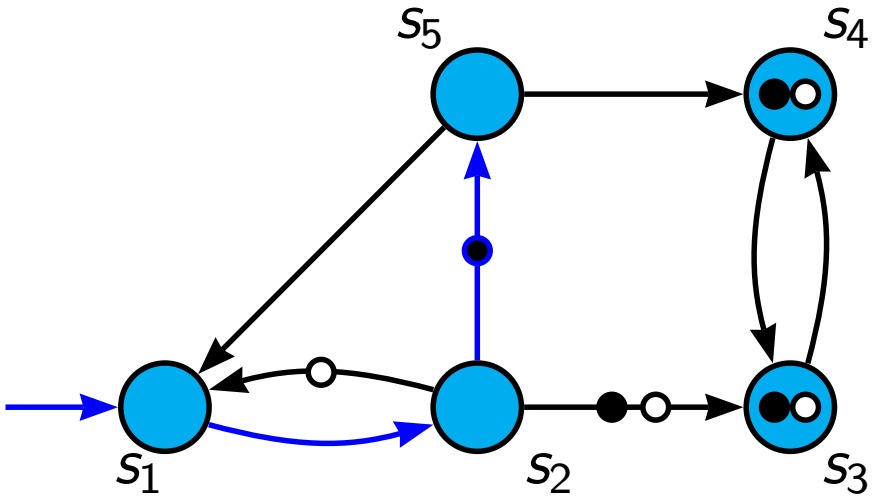
Generalized Nested DFS (Tauriainen'03)



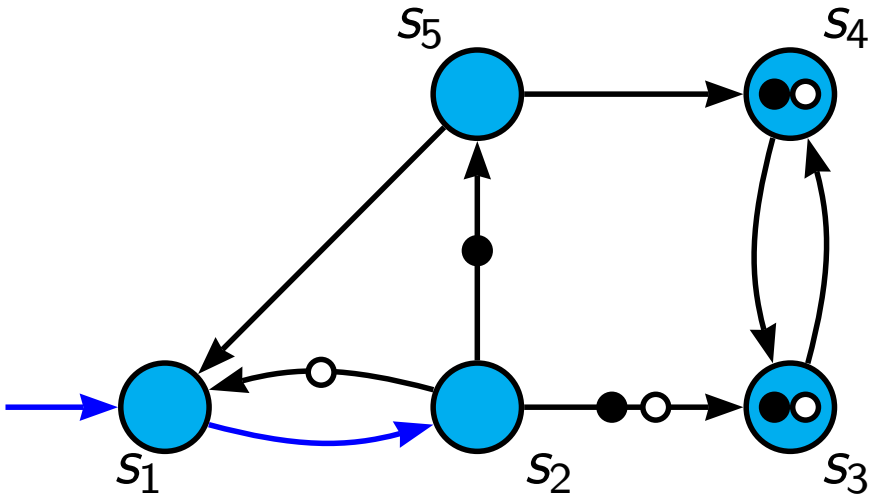
Generalized Nested DFS (Tauriainen'03)



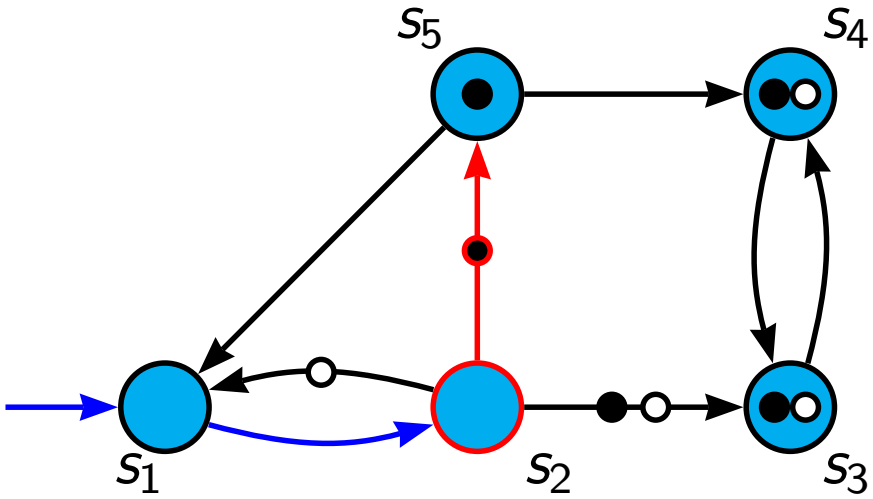
Generalized Nested DFS (Tauriainen'03)



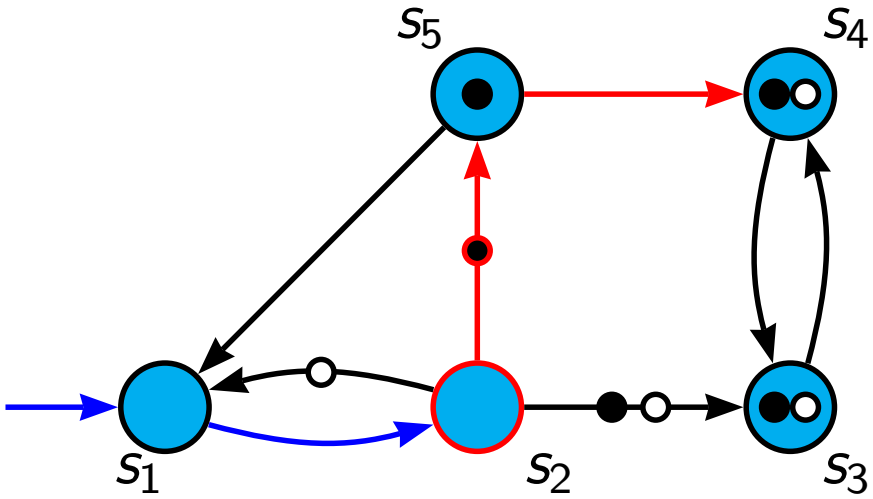
Generalized Nested DFS (Tauriainen'03)



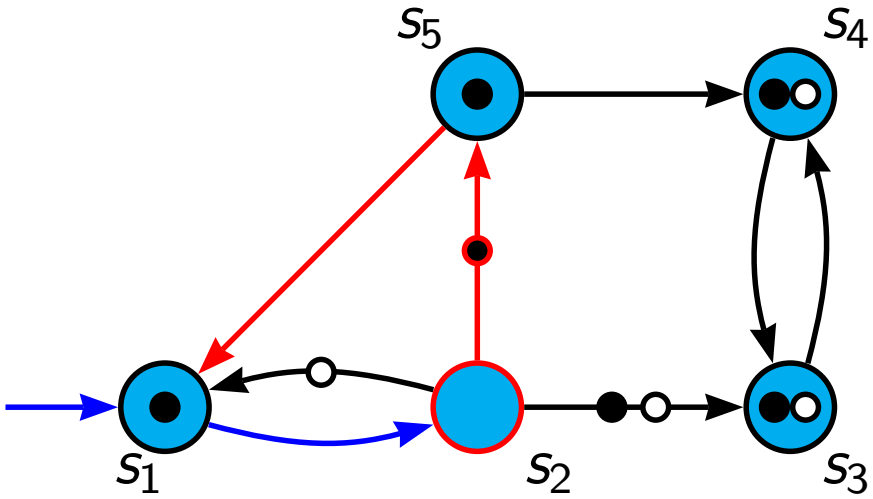
Generalized Nested DFS (Tauriainen'03)



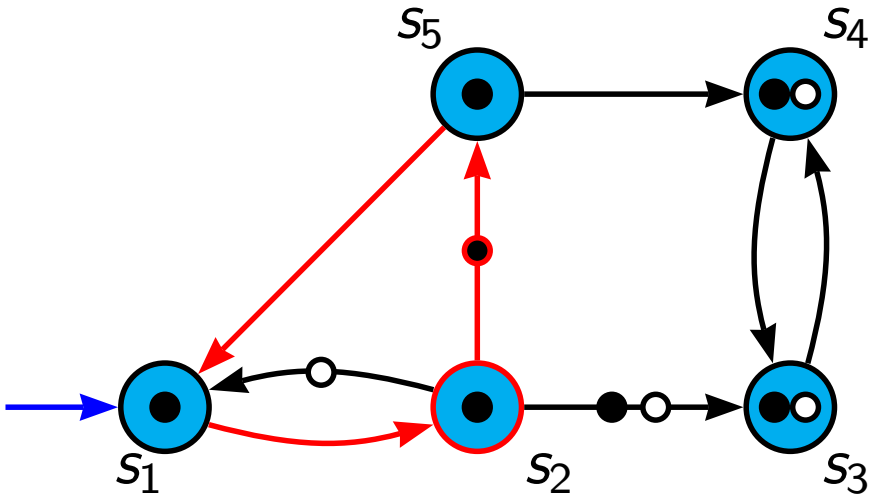
Generalized Nested DFS (Tauriainen'03)



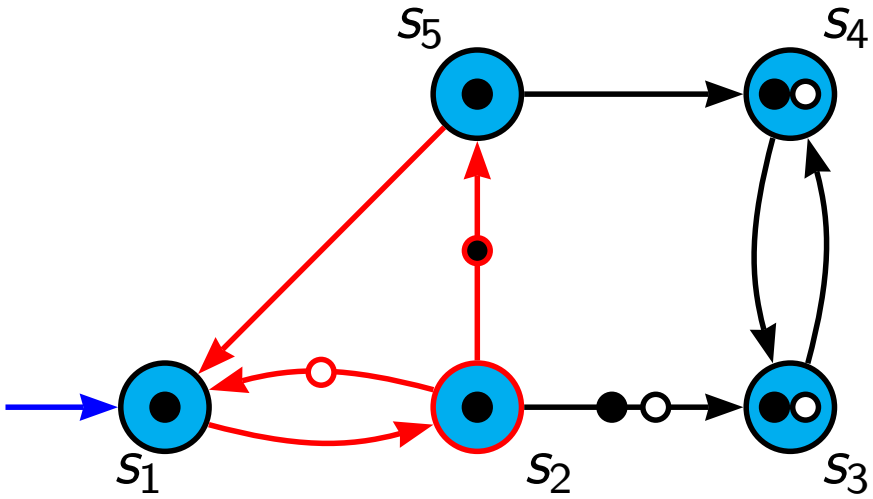
Generalized Nested DFS (Tauriainen'03)



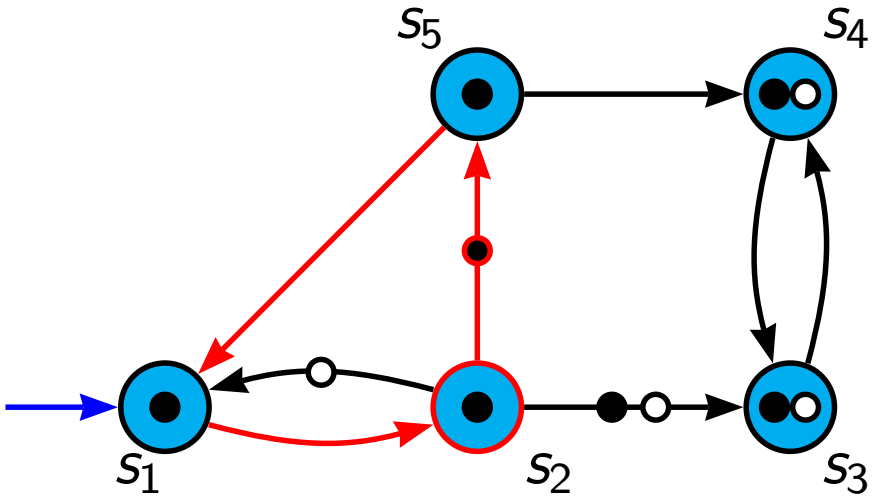
Generalized Nested DFS (Tauriainen'03)



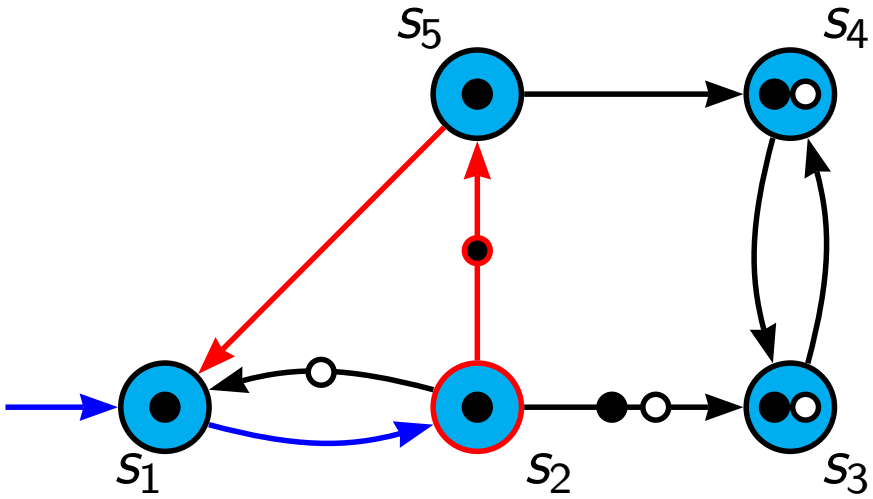
Generalized Nested DFS (Tauriainen'03)



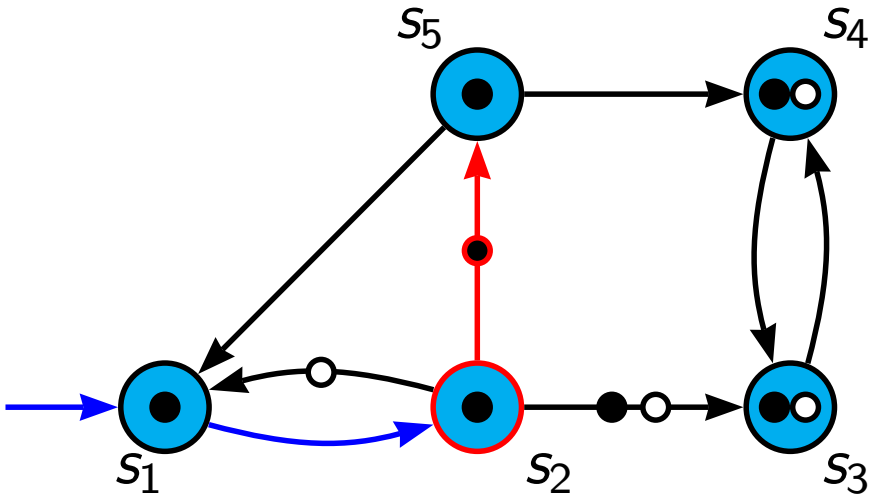
Generalized Nested DFS (Tauriainen'03)



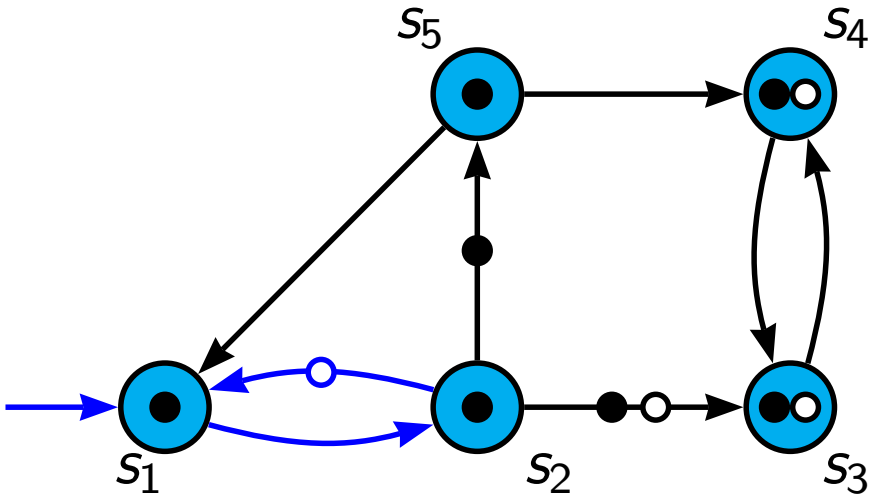
Generalized Nested DFS (Tauriainen'03)



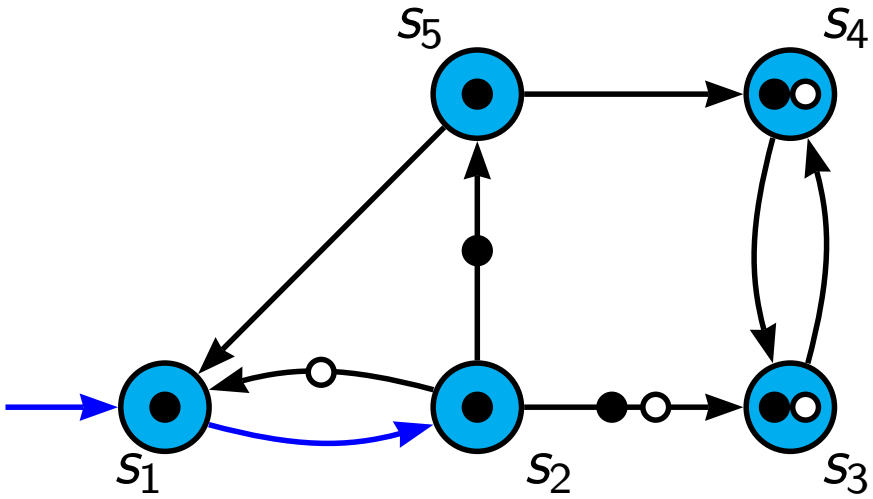
Generalized Nested DFS (Tauriainen'03)



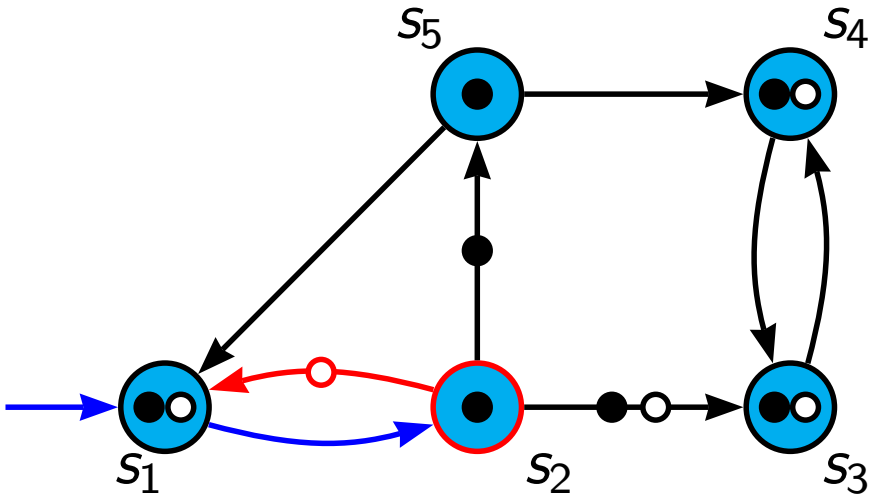
Generalized Nested DFS (Tauriainen'03)



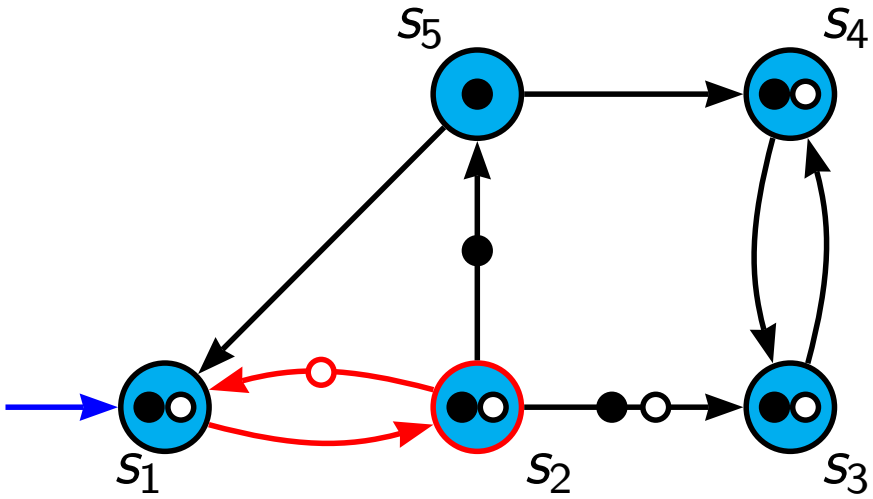
Generalized Nested DFS (Tauriainen'03)



Generalized Nested DFS (Tauriainen'03)

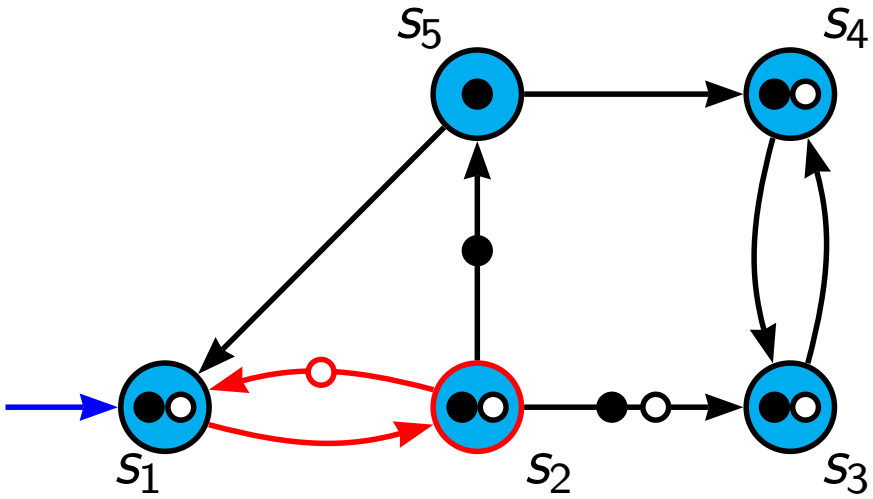


Generalized Nested DFS (Tauriainen'03)



Found!

Generalized Nested DFS (Tauriainen'03)



Found!

	entries in hash table	hash table size in bits	search stack depth	states traversed
degen+NDFS	n	$n(s_g + 2m)$	nm	$2nm$
gen. NDFS	n	$n(s_g + m)$	$2n$	$n(m+1)$

Emptiness Checks History

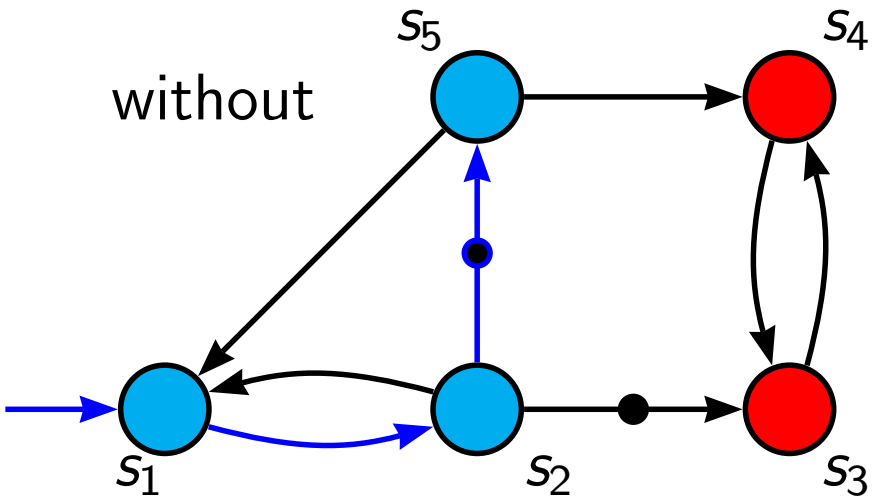
	degeneralized		generalized	
nested DFS	Courcoubetis et al.	'90		
	Godefroid & Holzmann	'93		
	Holzmann et al.	'96	Tauriainen	'03
	Gastin et al.	'04		
	Schwoon & Esparza	'04		

Emptiness Checks History

	degeneralized		generalized	
nested DFS	Courcoubetis et al.	'90		
	Godefroid & Holzmann	'93		
	Holzmann et al.	'96	Tauriainen	'03
	Gastin et al.	'04		
	Schwoon & Esparza	'04	Couvreur et al.	'05

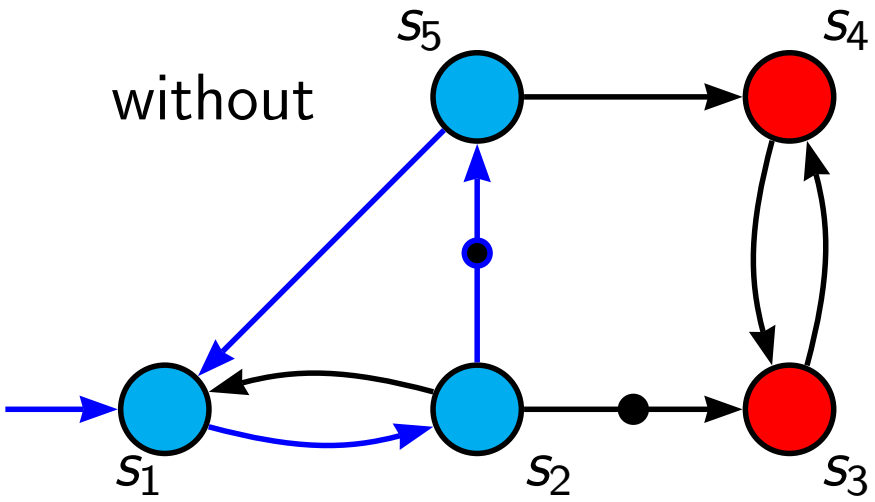
Our Generalized NDFS

- Merge more recent optimizations of Gastin et al. ('04) and Schwoon & Esparza ('04) into Taurainen's algorithm.
- Introduce another optimization: weighted blue stack.



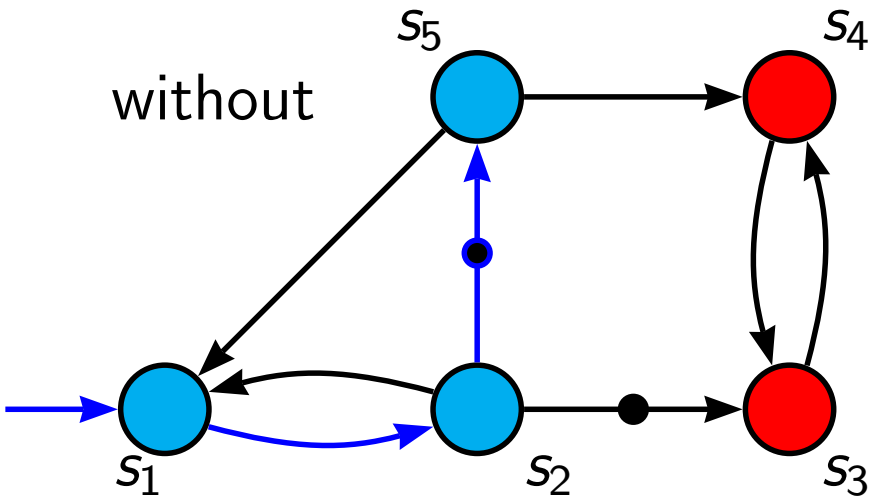
Our Generalized NDFS

- Merge more recent optimizations of Gastin et al. ('04) and Schwoon & Esparza ('04) into Taurainen's algorithm.
- Introduce another optimization: weighted blue stack.



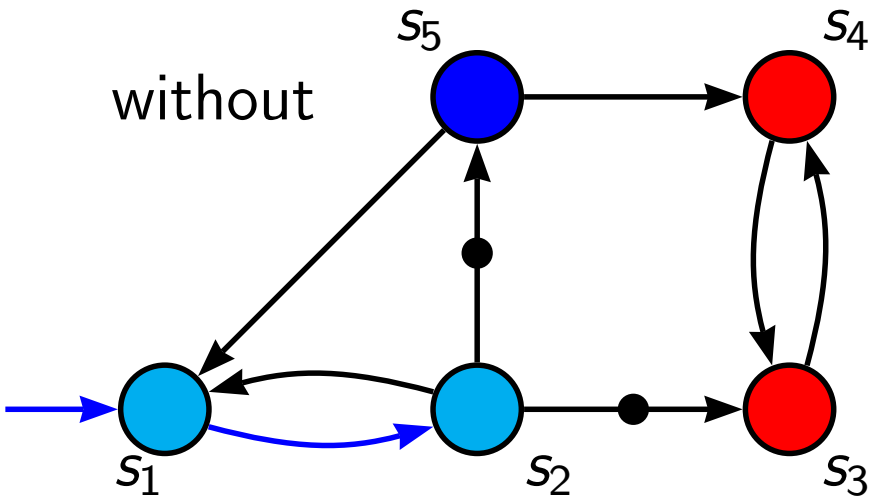
Our Generalized NDFS

- Merge more recent optimizations of Gastin et al. ('04) and Schwoon & Esparza ('04) into Taurainen's algorithm.
- Introduce another optimization: weighted blue stack.



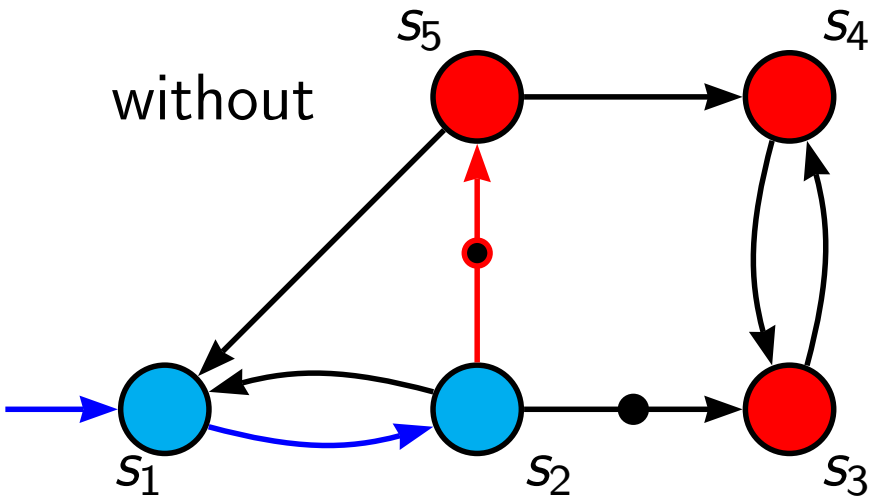
Our Generalized NDFS

- Merge more recent optimizations of Gastin et al. ('04) and Schwoon & Esparza ('04) into Taurainen's algorithm.
- Introduce another optimization: weighted blue stack.



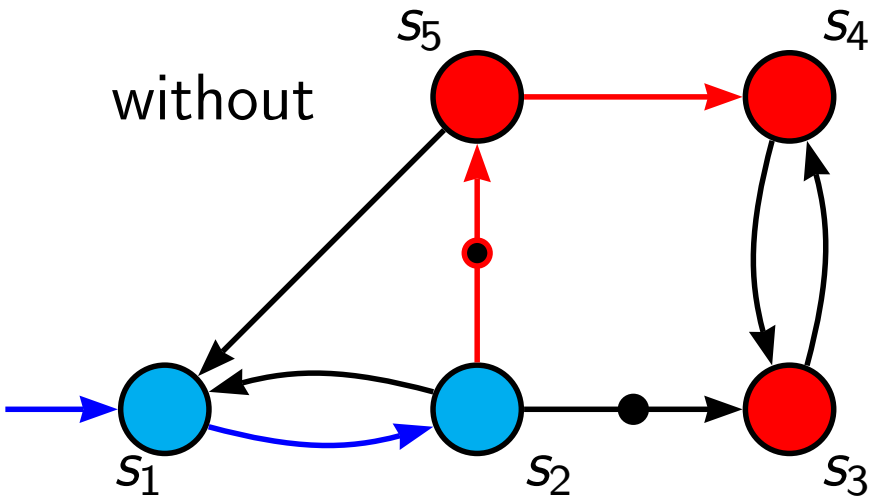
Our Generalized NDFS

- Merge more recent optimizations of Gastin et al. ('04) and Schwoon & Esparza ('04) into Taurainen's algorithm.
- Introduce another optimization: weighted blue stack.



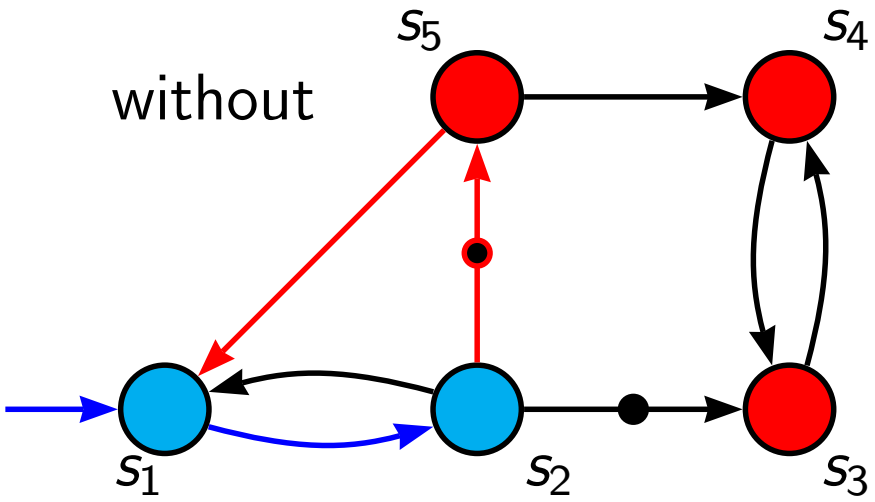
Our Generalized NDFS

- Merge more recent optimizations of Gastin et al. ('04) and Schwoon & Esparza ('04) into Taurainen's algorithm.
- Introduce another optimization: weighted blue stack.



Our Generalized NDFS

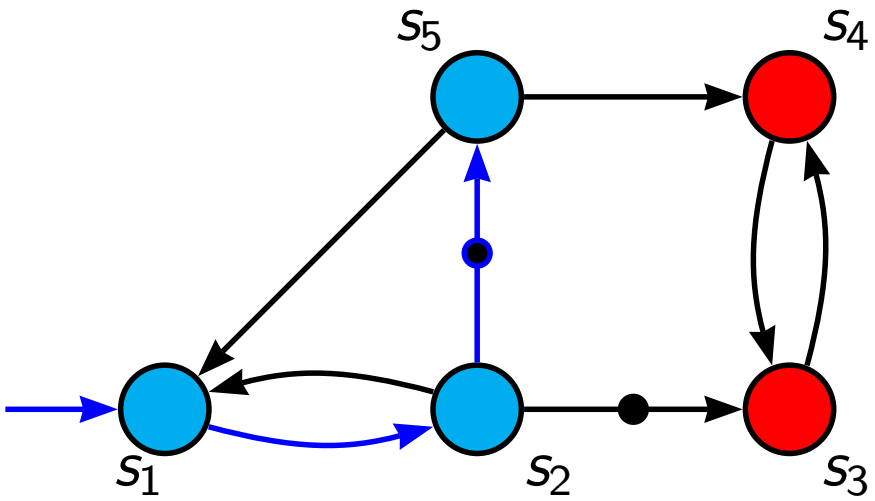
- Merge more recent optimizations of Gastin et al. ('04) and Schwoon & Esparza ('04) into Taurainen's algorithm.
- Introduce another optimization: weighted blue stack.



Found!

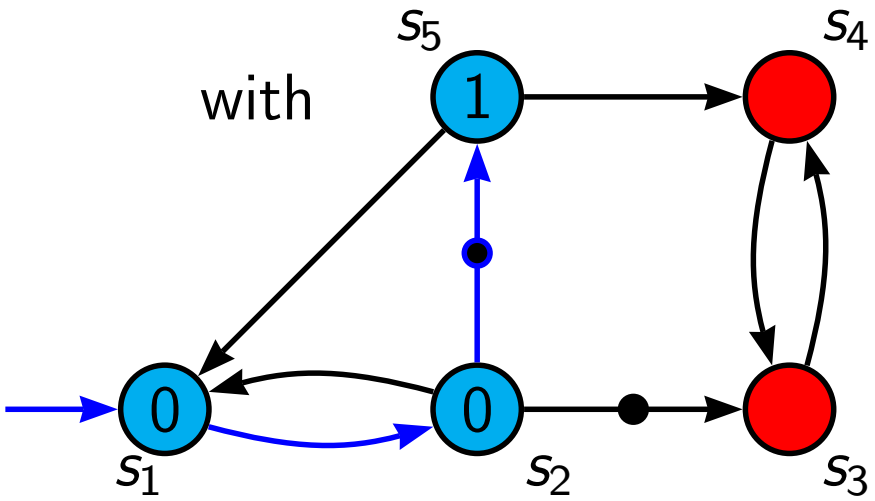
Our Generalized NDFS

- Merge more recent optimizations of Gastin et al. ('04) and Schwoon & Esparza ('04) into Taurainen's algorithm.
- Introduce another optimization: weighted blue stack.



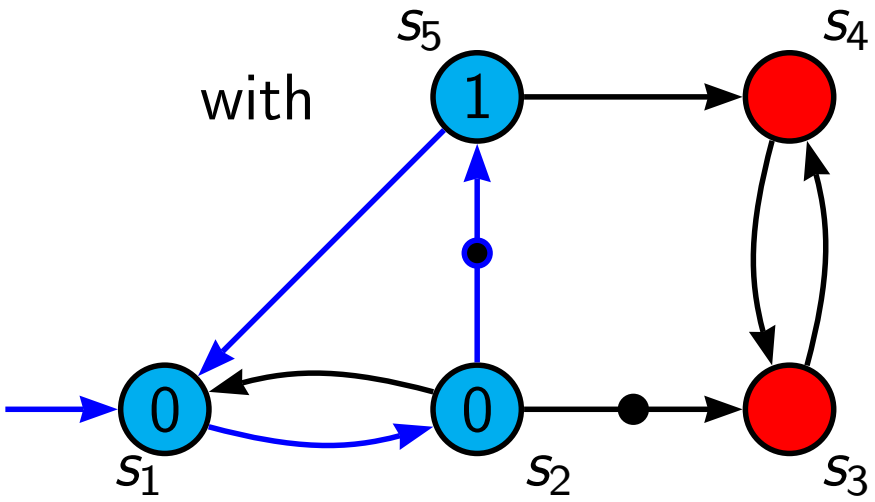
Our Generalized NDFS

- Merge more recent optimizations of Gastin et al. ('04) and Schwoon & Esparza ('04) into Taurainen's algorithm.
- Introduce another optimization: weighted blue stack.



Our Generalized NDFS

- Merge more recent optimizations of Gastin et al. ('04) and Schwoon & Esparza ('04) into Taurainen's algorithm.
- Introduce another optimization: weighted blue stack.



Found!

Generalizable with m counters per state in the blue search.

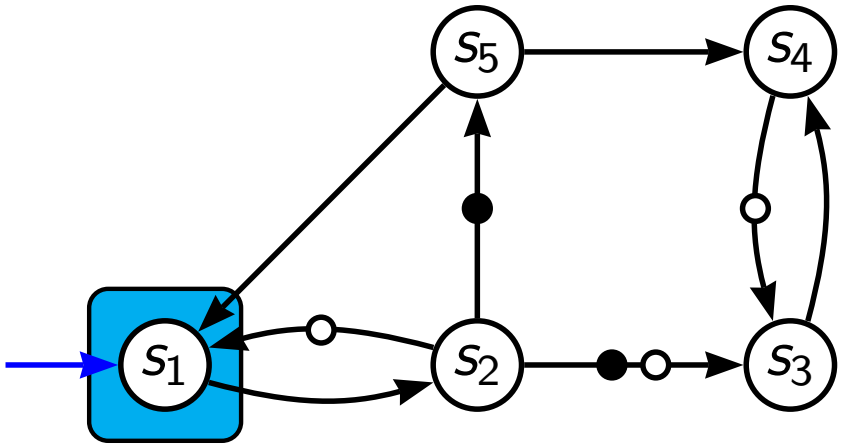
Emptiness Checks History

	degeneralized		generalized	
nested DFS	Courcoubetis et al.	'90		
	Godefroid & Holzmann	'93		
	Holzmann et al.	'96	Tauriainen	'03
	Gastin et al.	'04		
	Schwoon & Esparza	'04	Couvreur et al.	'05

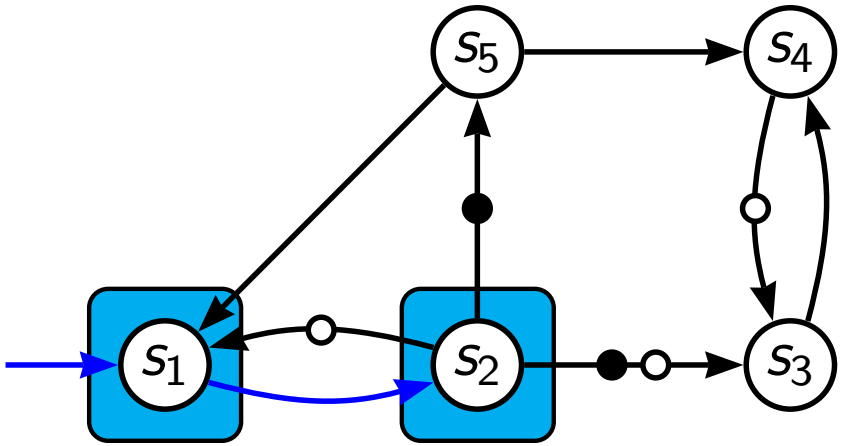
Emptiness Checks History

	degeneralized		generalized	
nested DFS	Courcoubetis et al.	'90		
	Godefroid & Holzmann	'93		
	Holzmann et al.	'96	Tauriainen	'03
	Gastin et al.	'04		
	Schwoon & Esparza	'04	Couvreur et al.	'05
SCC	Lichtenstein & Pnueli	'83	Couvreur	'99

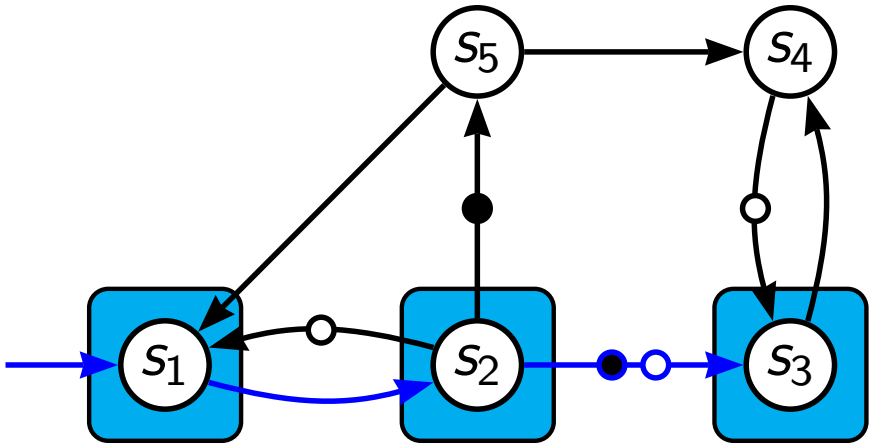
SCC-Based Emptiness Check



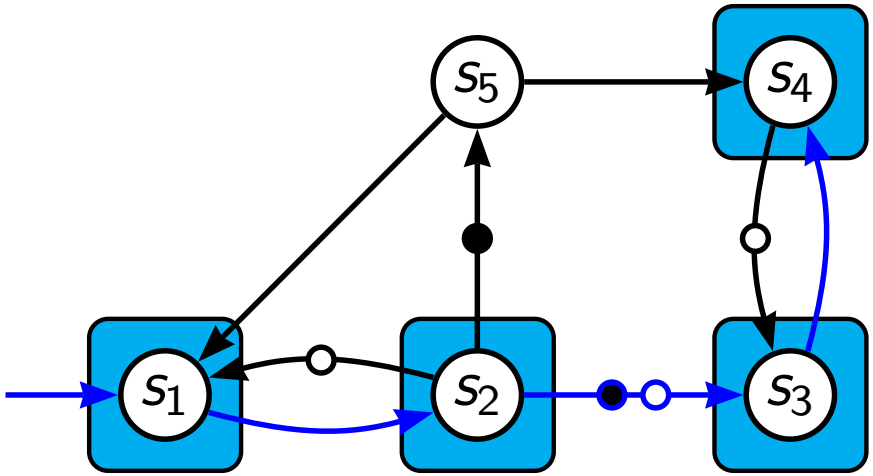
SCC-Based Emptiness Check



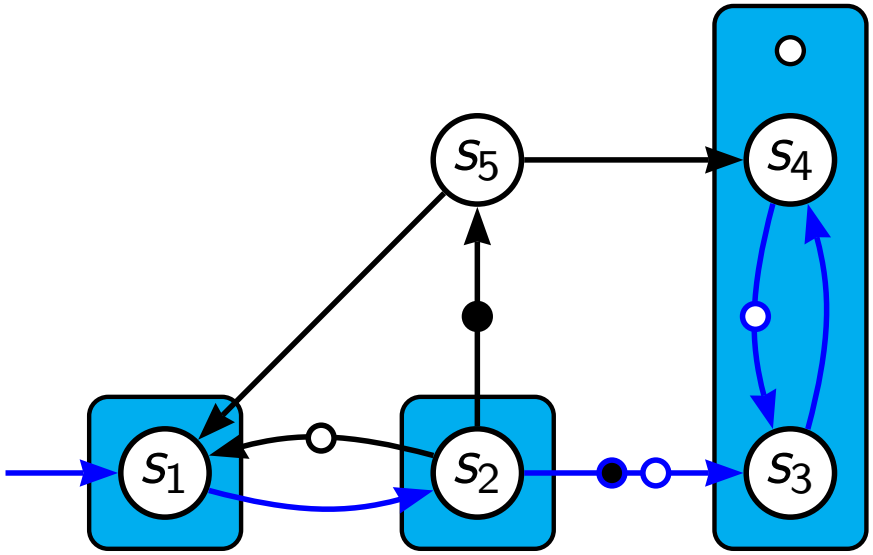
SCC-Based Emptiness Check



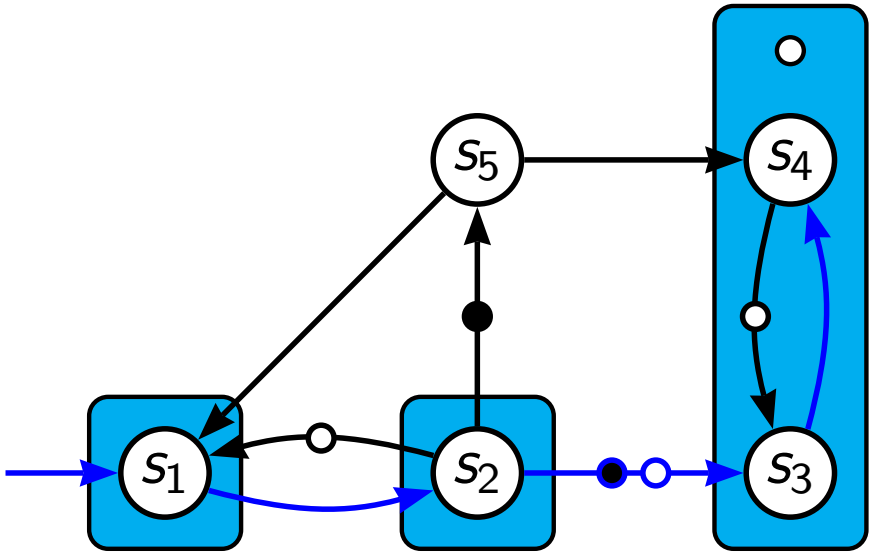
SCC-Based Emptiness Check



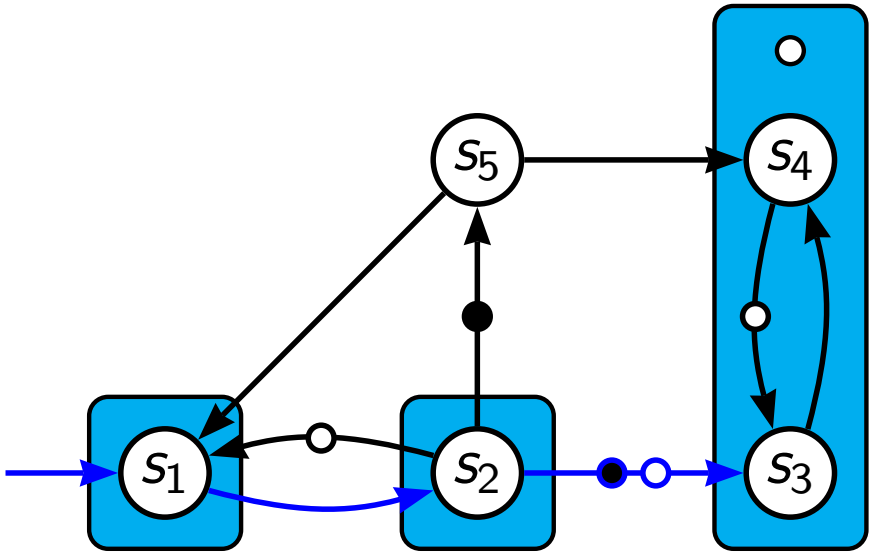
SCC-Based Emptiness Check



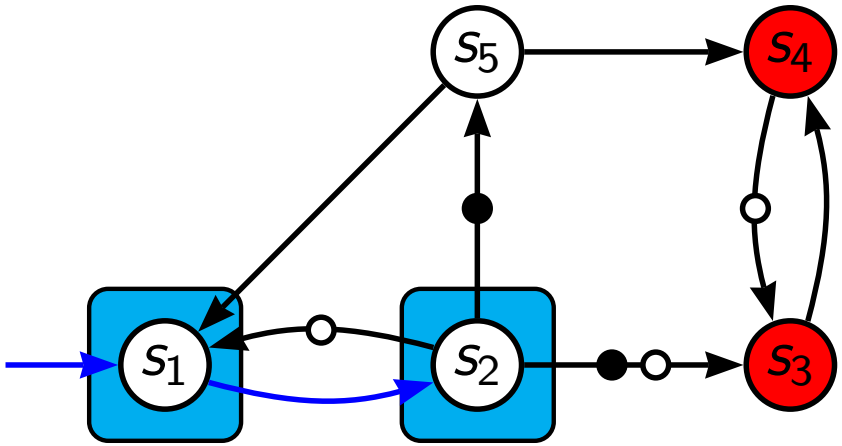
SCC-Based Emptiness Check



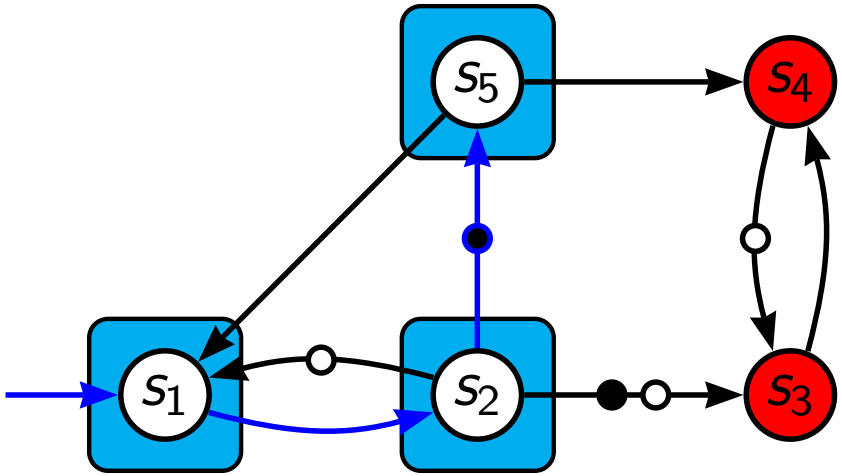
SCC-Based Emptiness Check



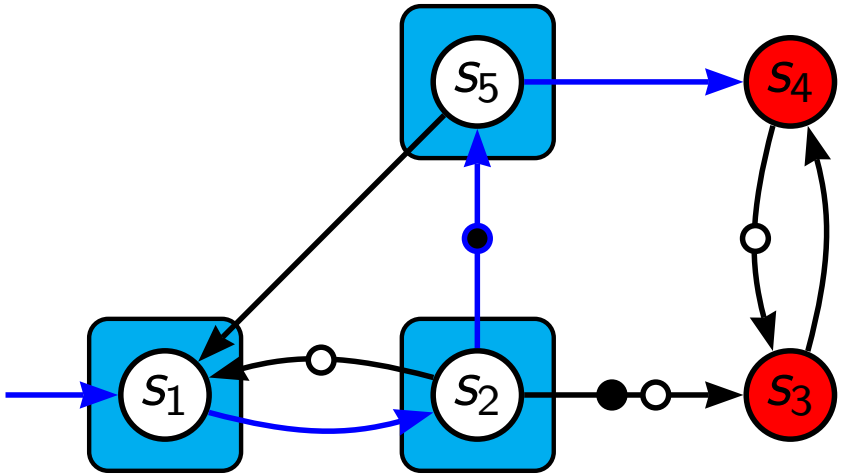
SCC-Based Emptiness Check



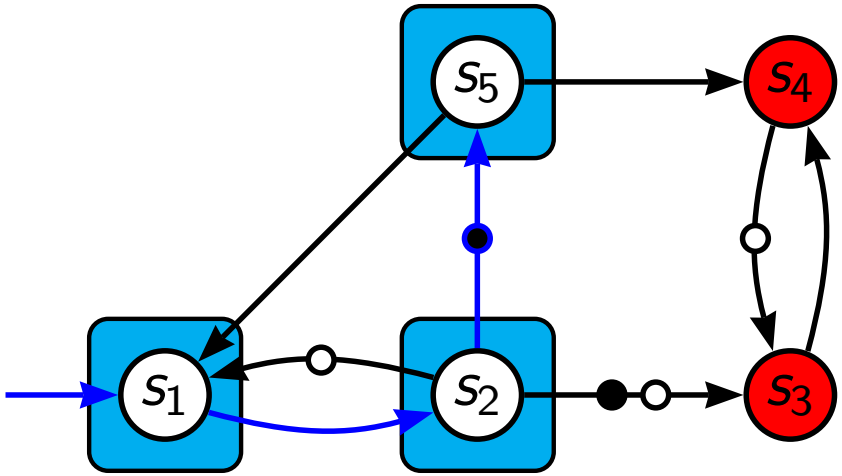
SCC-Based Emptiness Check



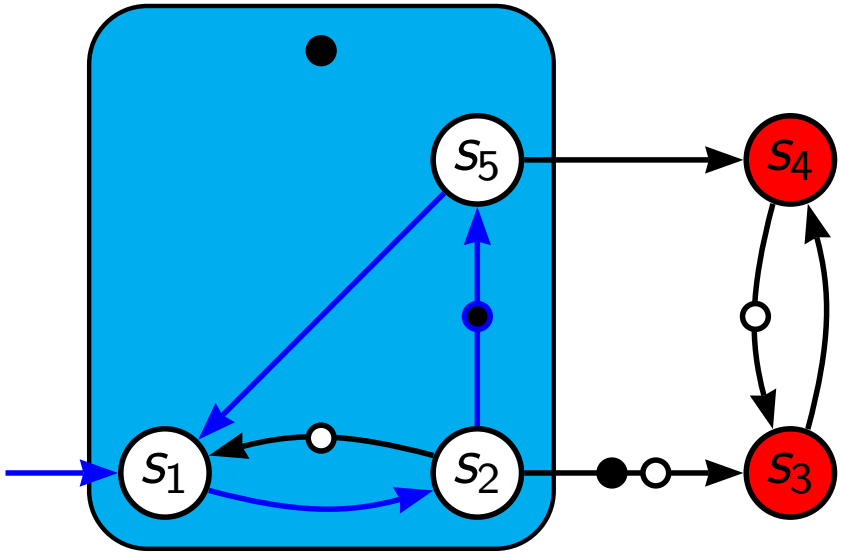
SCC-Based Emptiness Check



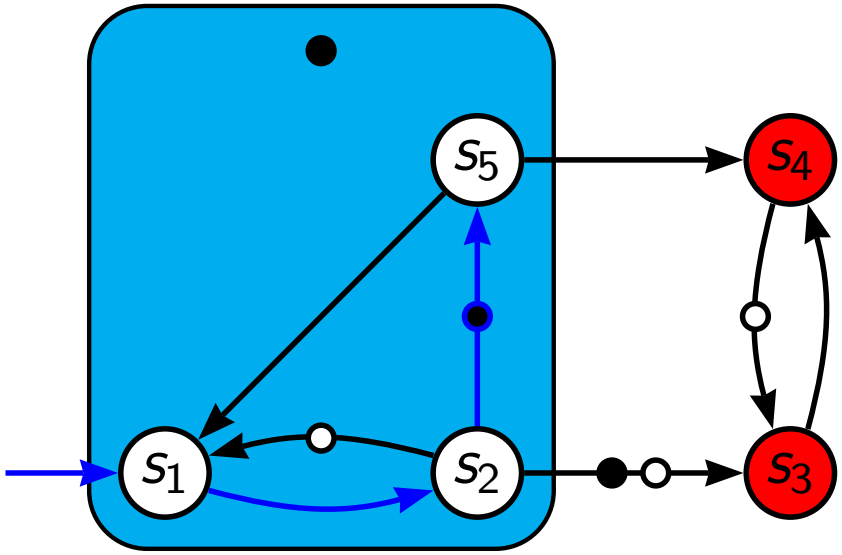
SCC-Based Emptiness Check



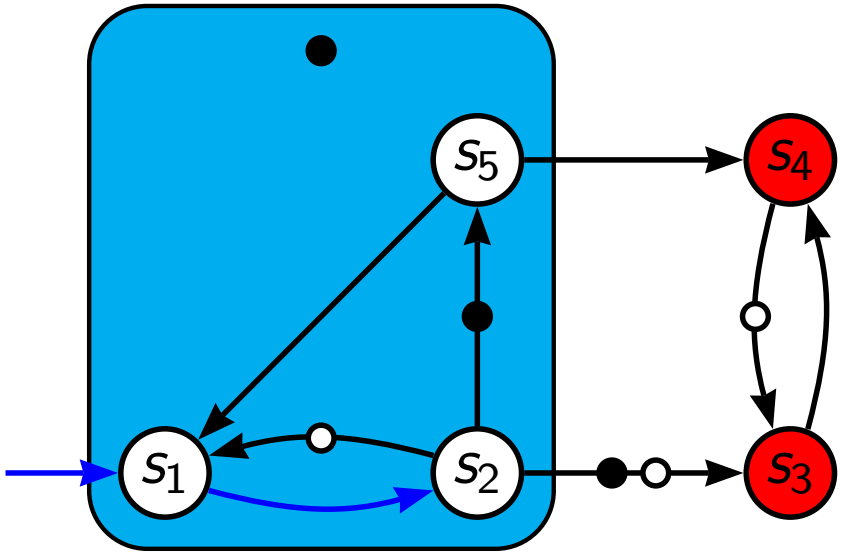
SCC-Based Emptiness Check



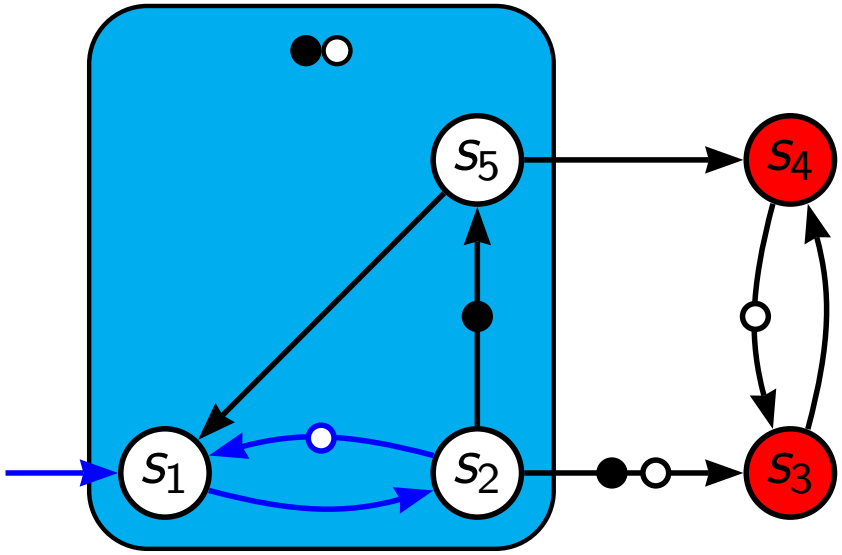
SCC-Based Emptiness Check



SCC-Based Emptiness Check

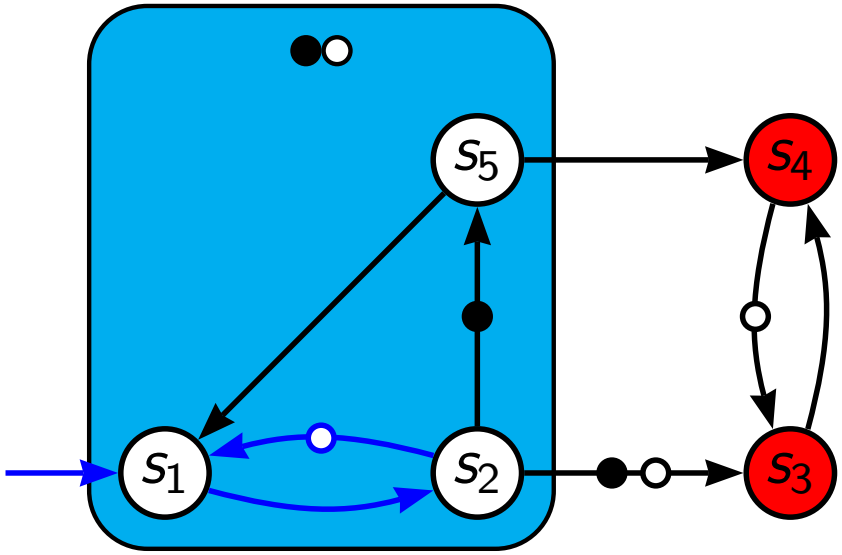


SCC-Based Emptiness Check



Found!

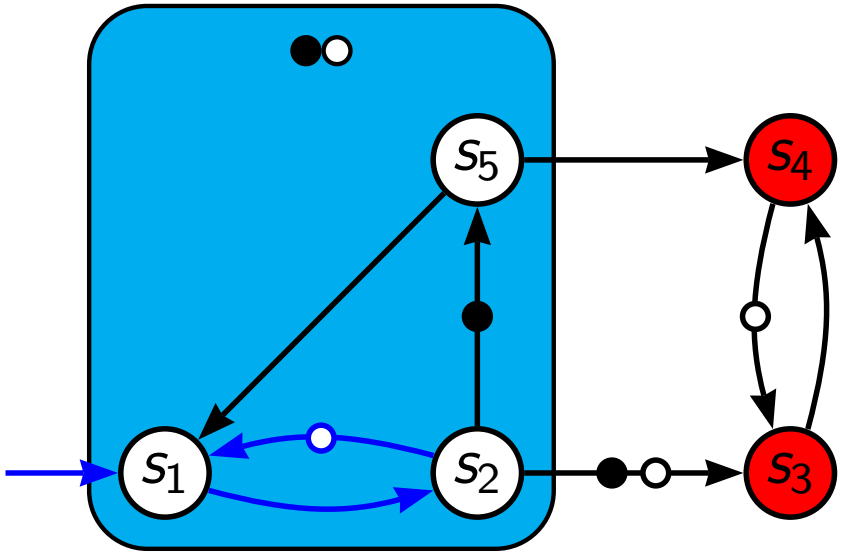
SCC-Based Emptiness Check



Found!

entries in hash table	hash table size in bits	search stack depth	states traversed
n	$n(s_g + \lg n)$	n	$2n$

SCC-Based Emptiness Check



Found!

entries in hash table	hash table size in bits	search stack depth	states traversed
n	$n(s_g + \lg n)$	n	$2n$

Can be reduced to n if the search stack explicitly stores the states of each component (require more memory).

Emptiness Checks History

	degeneralized		generalized	
nested DFS	Courcoubetis et al.	'90		
	Godefroid & Holzmann	'93		
	Holzmann et al.	'96	Tauriainen	'03
	Gastin et al.	'04		
	Schwoon & Esparza	'04	Couvreur et al.	'05
SCC	Lichtenstein & Pnueli	'83	Couvreur	'99

Emptiness Checks History

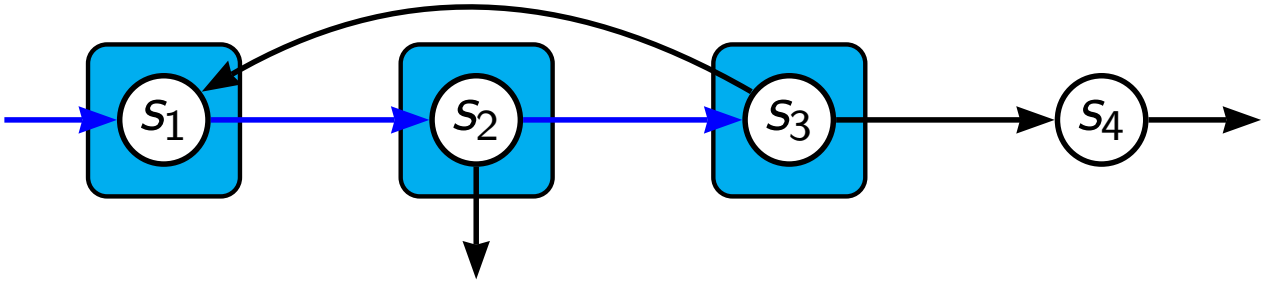
	degeneralized		generalized	
nested DFS	Courcoubetis et al.	'90		
	Godefroid & Holzmann	'93		
	Holzmann et al.	'96	Tauriainen	'03
	Gastin et al.	'04		
	Schwoon & Esparza	'04	Couvreur et al.	'05
SCC	Lichtenstein & Pnueli	'83	Couvreur	'99
	Geldenhuys & Valmari	'04	Geldenhuys & Valmari	'05

Emptiness Checks History

	degeneralized		generalized	
nested DFS	Courcoubetis et al.	'90		
	Godefroid & Holzmann	'93		
	Holzmann et al.	'96	Tauriainen	'03
	Gastin et al.	'04		
	Schwoon & Esparza	'04	Couvreur et al.	'05
SCC	Lichtenstein & Pnueli	'83	Couvreur	'99
	Geldenhuys & Valmari	'04	Geldenhuys & Valmari	'05
			Couvreur et al.	'05

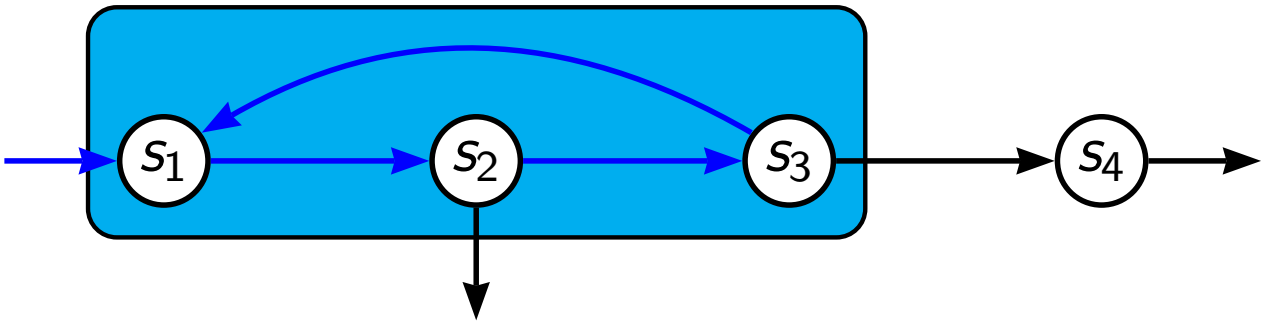
Two Heuristics for SCCs

- H1: visit transitions that go to visited states first.



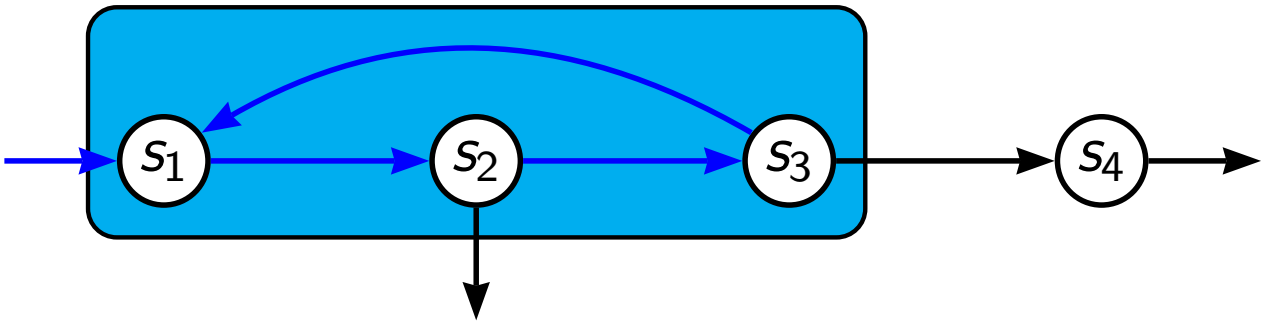
Two Heuristics for SCCs

- H1: visit transitions that go to visited states first.



Two Heuristics for SCCs

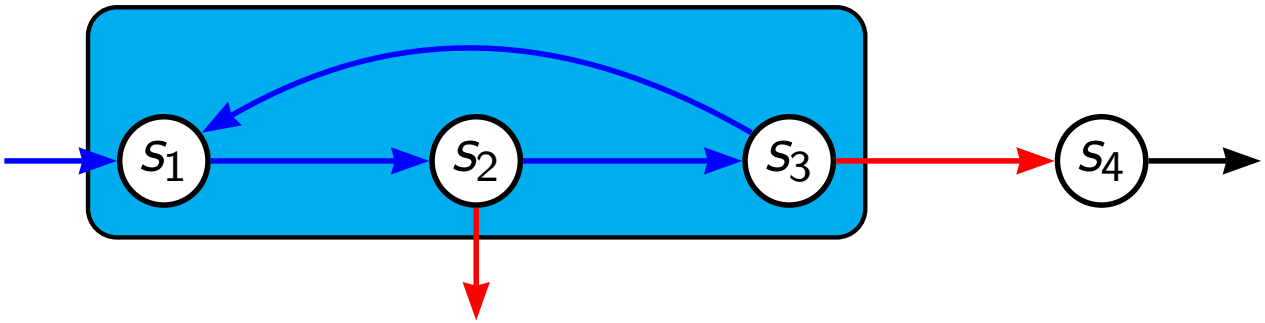
- H1: visit transitions that go to visited states first.



- H2: H1 + consider the DFS in term of SCC when choosing a successor.

Two Heuristics for SCCs

- H1: visit transitions that go to visited states first.



- H2: H1 + consider the DFS in term of SCC when choosing a successor.

Benchmarks

- Upper bounds easy to have.
- Objective : evaluate all these algorithms on the average, on non-empty automata.

	degeneralized		generalized	
nested DFS	Courcoubetis et al.	'90		
	Godefroid & Holzmann	'93		
	Holzmann et al.	'96	Tauriainen	'03
	Gastin et al.	'04		
	Schwoon & Esparza	'04	Couvreur et al.	'05
SCC	Lichtenstein & Pnueli	'83	Couvreur	'99
	Geldenhuys & Valmari	'04	Geldenhuys & Valmari	'05
			Couvreur et al.	'05

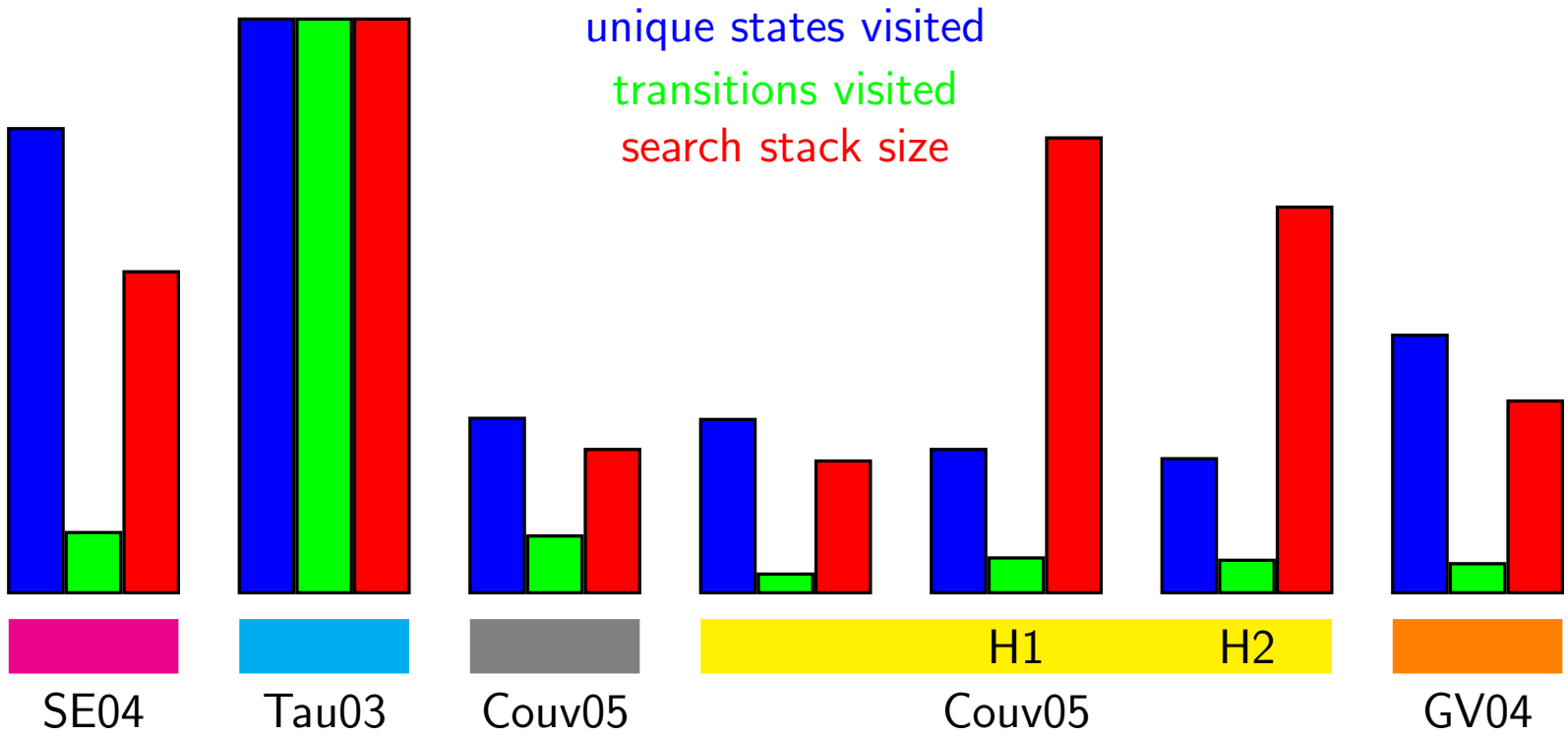
Benchmarks

- Upper bounds easy to have.
- Objective : evaluate all these algorithms on the average, on non-empty automata.

	degeneralized		generalized	
nested DFS	Courcoubetis et al.	'90		
	Godefroid & Holzmann	'93		
	Holzmann et al.	'96	Tauriainen	'03
	Gastin et al.	'04		
	Schwoon & Esparza	'04	Couvreur et al.	'05
SCC	Lichtenstein & Pnueli	'83	Couvreur	'99
	Geldenhuys & Valmari	'04	Geldenhuys & Valmari	'05
			Couvreur et al.	'05

Benchmarks

nested DFS	Schwoon & Esparza	'04	Tauriainen Couvreur et al.	'03 '05
SCC	Geldenhuys & Valmari	'04	Couvreur et al.	'05



Conclusions

- generalized vs. non-generalized:
  - generalized algorithms require less memory
  - generalized algorithms produce more meaningful counterexamples
  - weak fairness expressible using generalized conditions
  - non-generalized NDFSs produce counterexamples directly
- NDFS vs. SCC algorithms:
  - SCC algorithms check emptiness faster
  - SCC algorithms scale to generalized conditions and fairness conditions easily
  - NDFSs require less memory
- All these algorithms and the benchmark framework are implemented in our model-checking library:

  `http://spot.lip6.fr`