Emptiness of Powerset Büchi Automata using Inclusions Tests

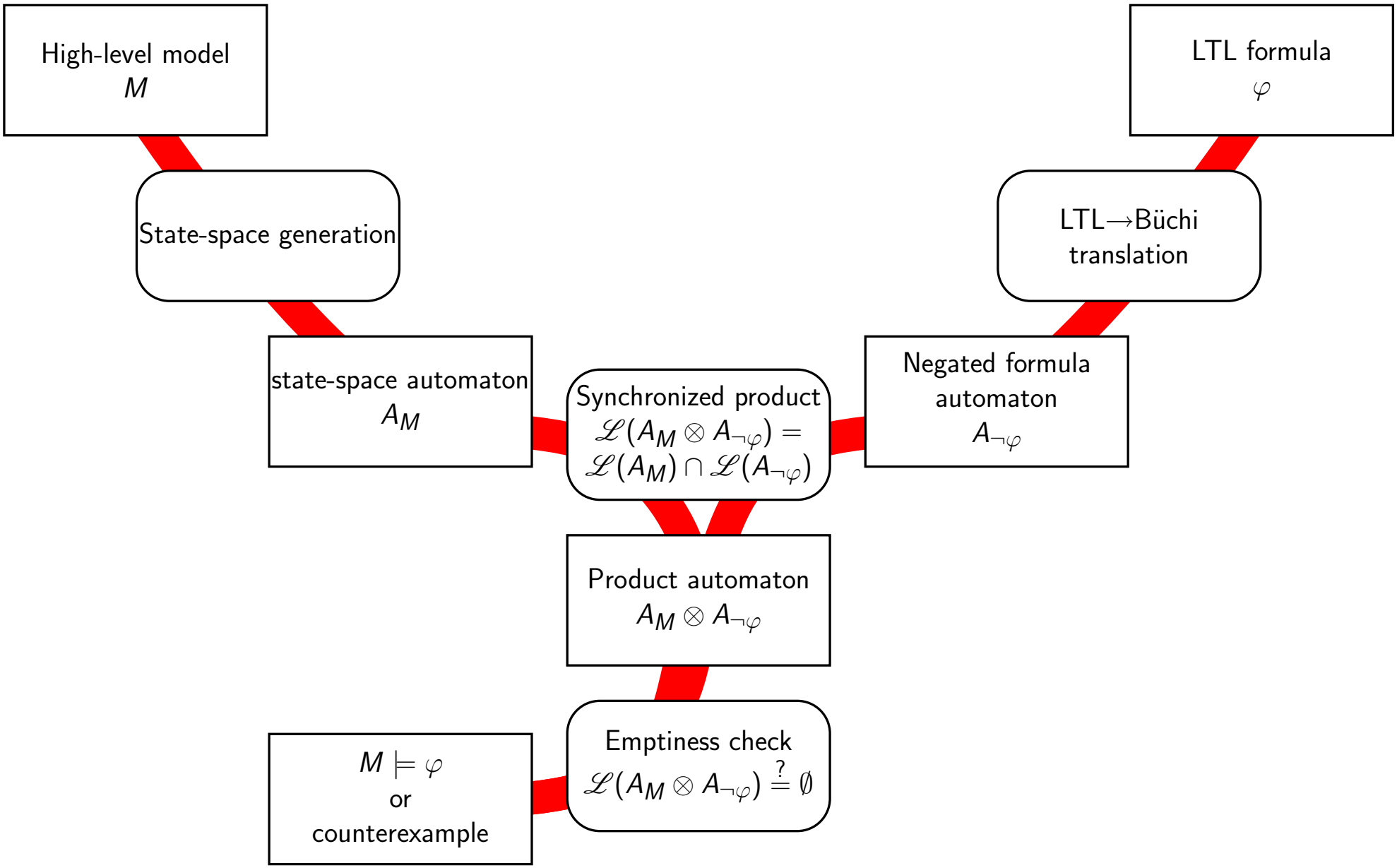
Souheib Baarir Alexandre Duret-Lutz

LIP6, Université de Paris 6, France

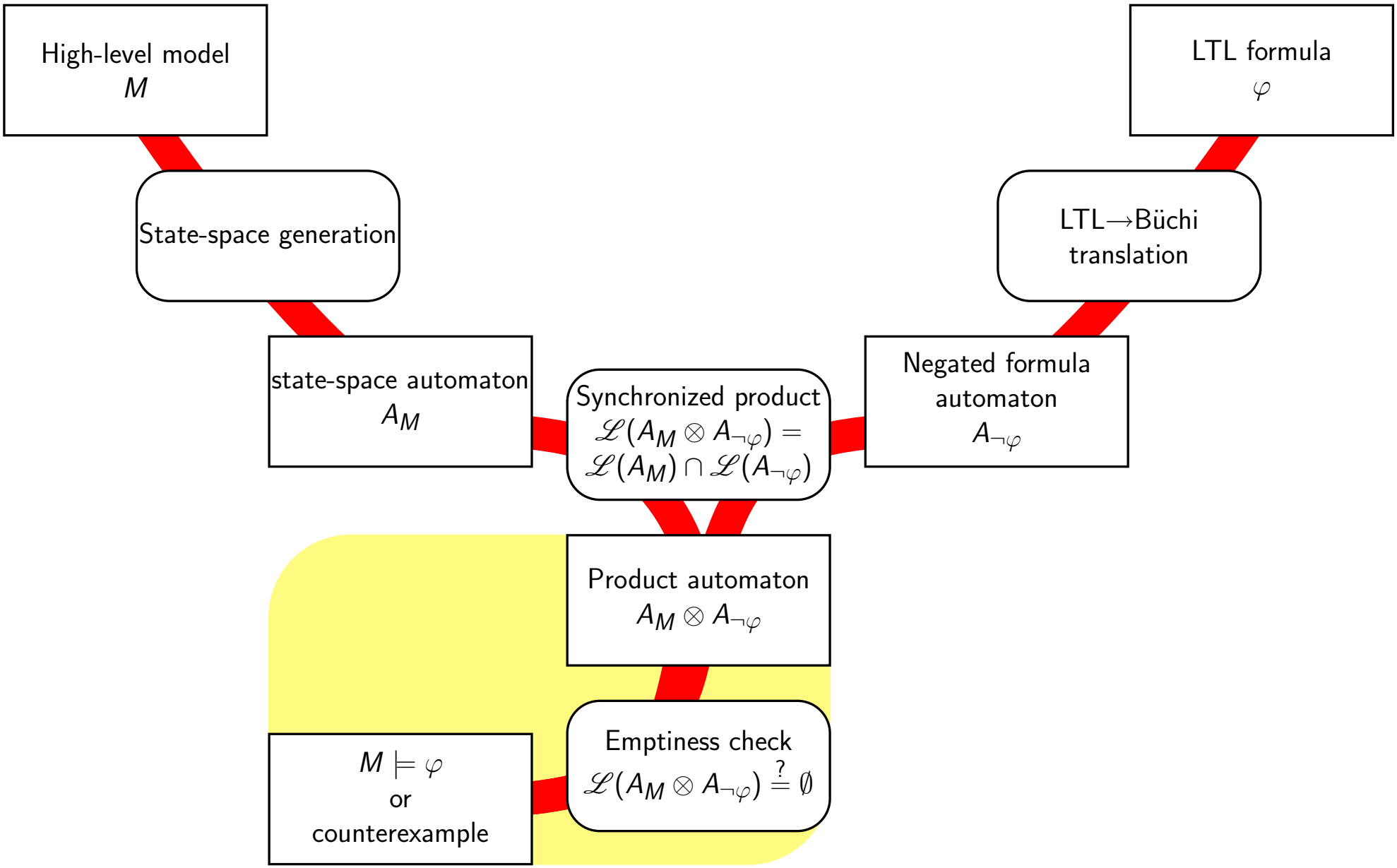
12th July 2007

alexandre.duret-lutz@lip6.fr

Automata-Theoretic Approach to Model Checking



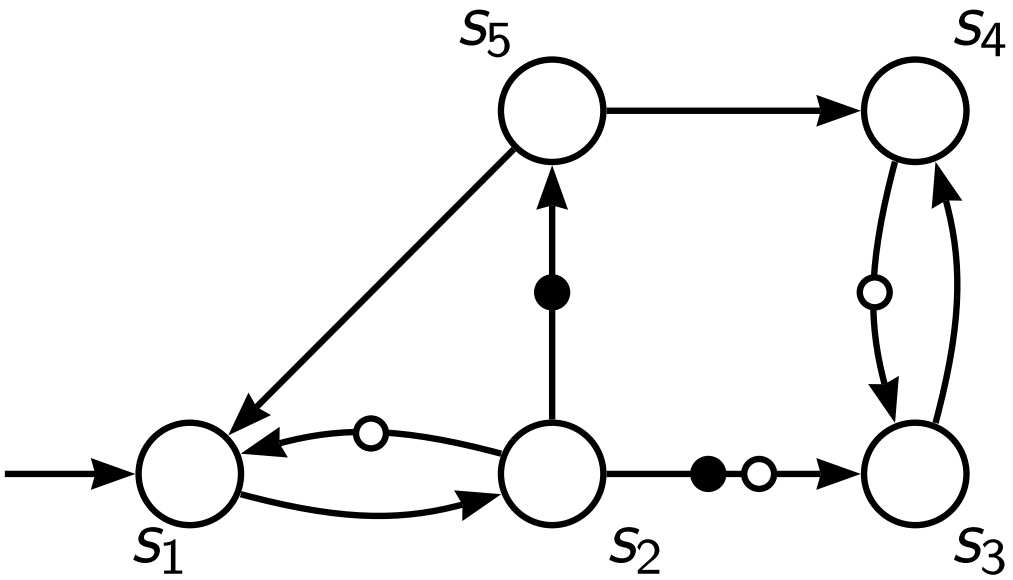
Automata-Theoretic Approach to Model Checking



Emptiness Checks for Generalized Büchi Automata

Transition-based Generalized Büchi Automata

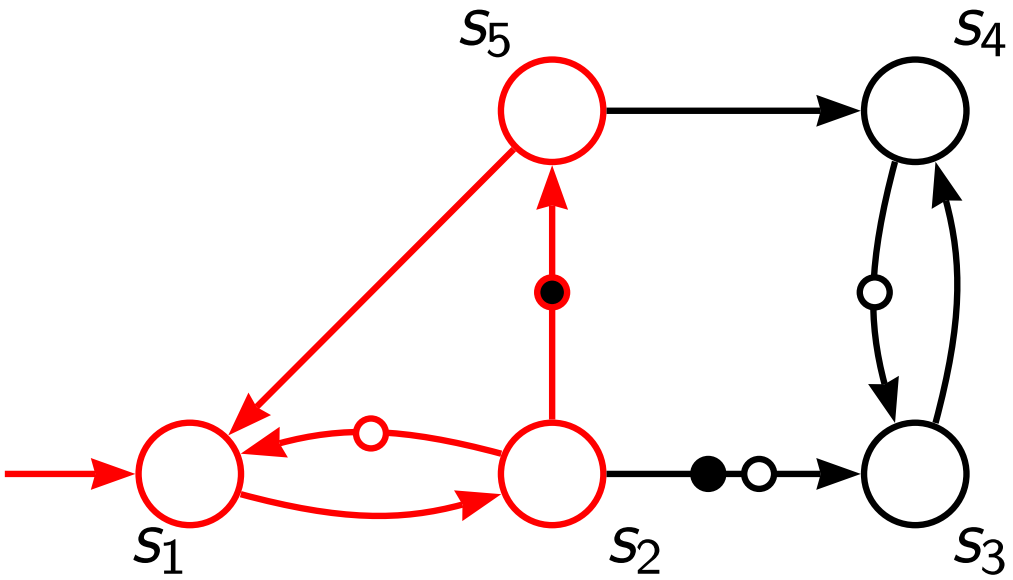
An infinite run of this automaton is accepting if it visits a transition from each accepting condition ($\bullet, \circ, \ldots$) infinitely often.



Emptiness Check = Does an automaton have no accepting run?
$\Longrightarrow$ Search for an accepting cycle reachable from the initial state.

Transition-based Generalized Büchi Automata

An infinite run of this automaton is accepting if it visits a transition from each accepting condition ($\bullet, \circ, \ldots$) infinitely often.

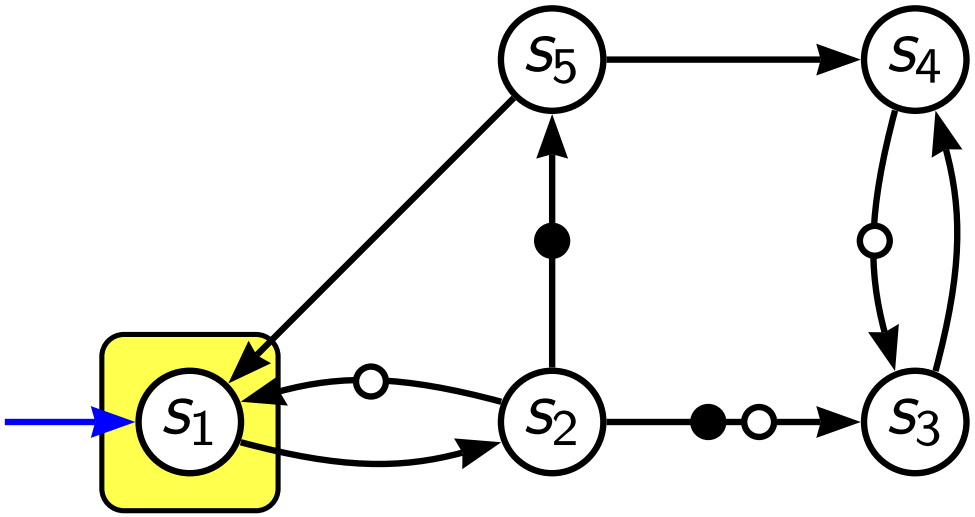


Emptiness Check = Does an automaton have no accepting run?
$\Longrightarrow$ Search for an accepting cycle reachable from the initial state.

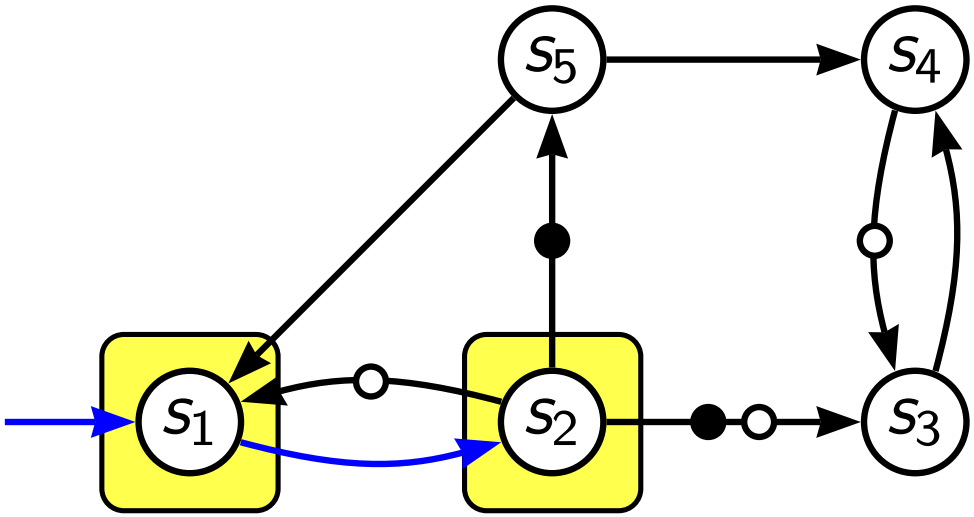
Emptiness Checks

	degeneralized (**one** acceptance condition)	generalized (m acceptance conditions)
Nested DFS	DFS for acc. transitions + Nested DFS to find cycle • 2 bits per state • immediate counterexamples • require degeneralization (size $\times m$) • slow	DFS for acc. transitions + **several** Nested DFS • $\log_2(m+1)$ bits per state • visit states several times • slow
SCC	(pointless)	Compute SCCs on the fly, abort on accepting SCC • fastest • visit states once • indifferent to m • SCC information useful • one integer per state

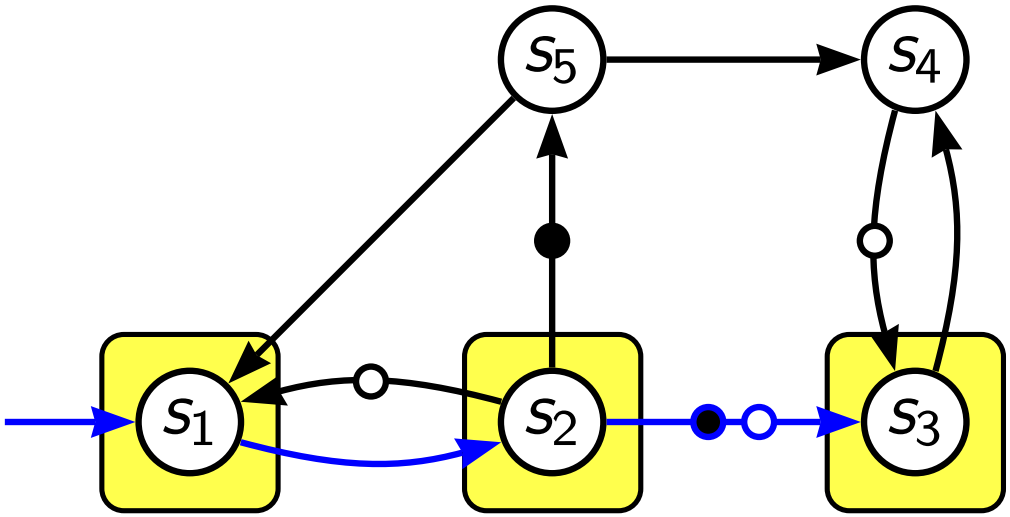
Couvreur's SCC-Based Emptiness Check



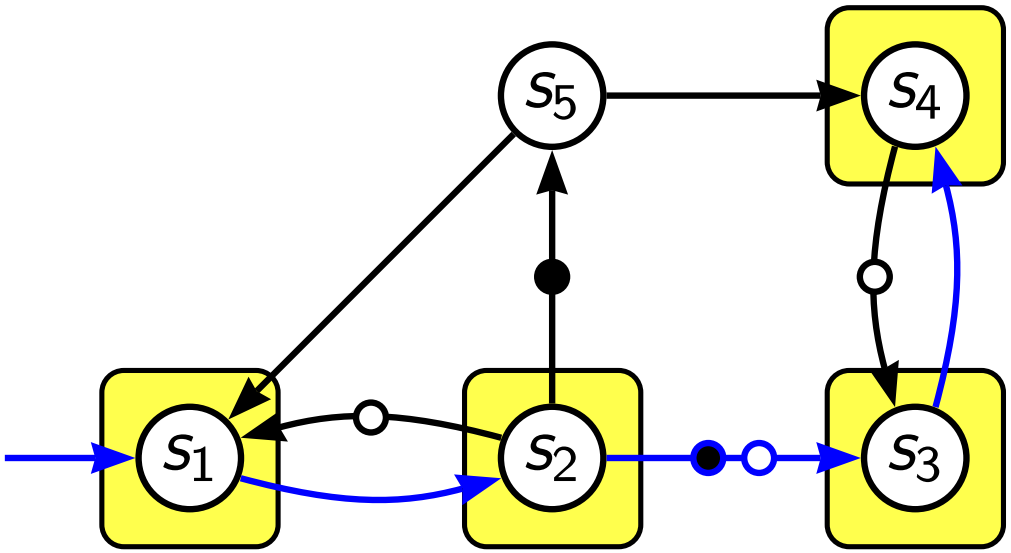
Couvreur's SCC-Based Emptiness Check



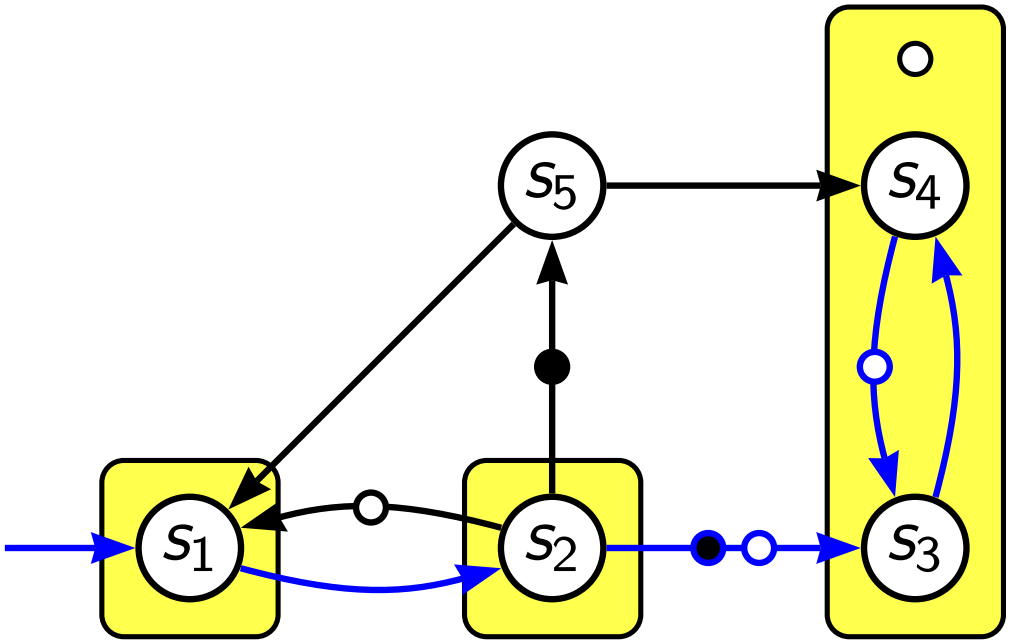
Couvreur's SCC-Based Emptiness Check



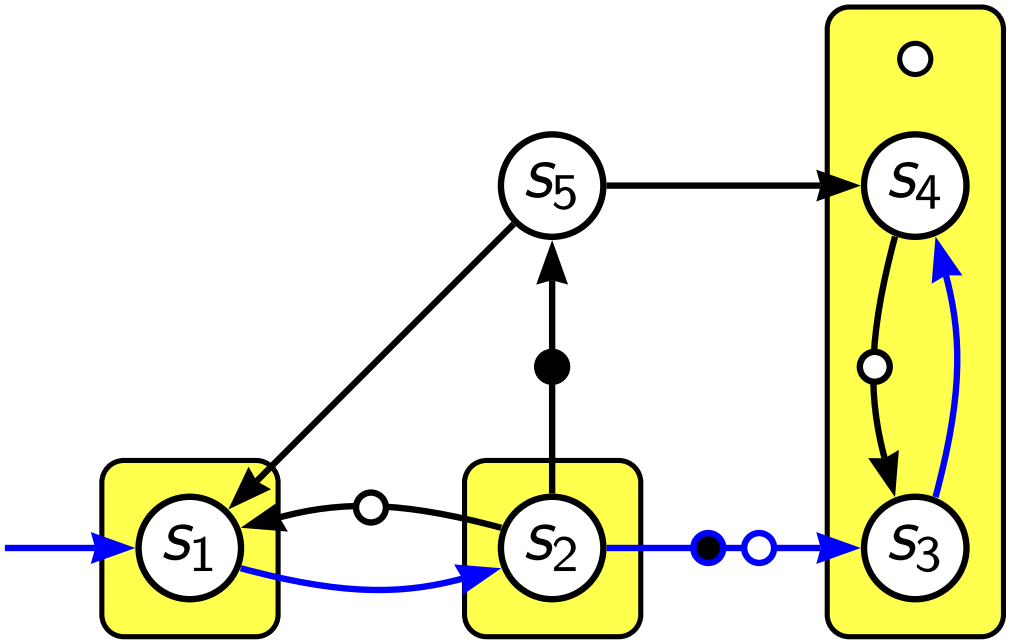
Couvreur's SCC-Based Emptiness Check



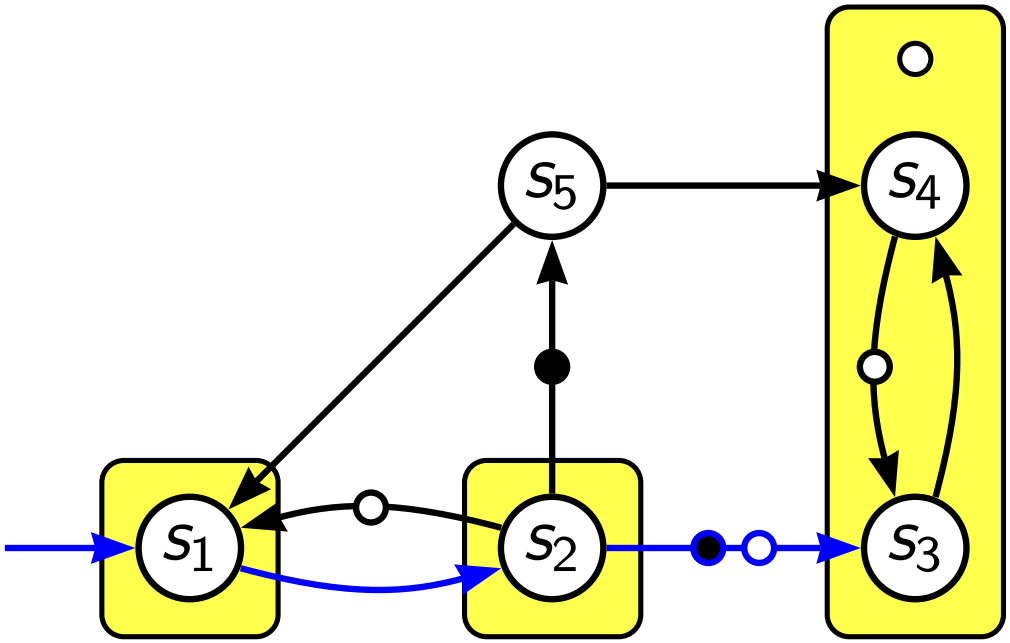
Couvreur's SCC-Based Emptiness Check



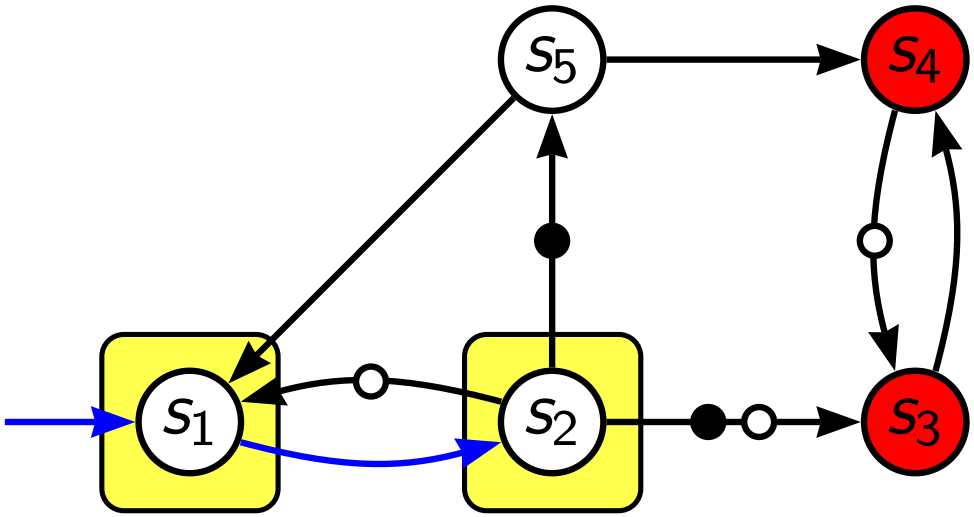
Couvreur's SCC-Based Emptiness Check



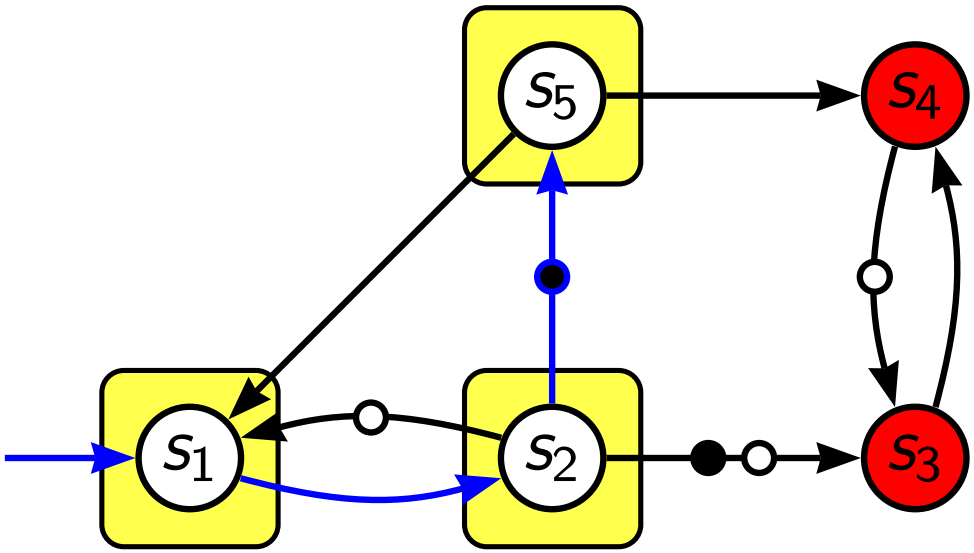
Couvreur's SCC-Based Emptiness Check



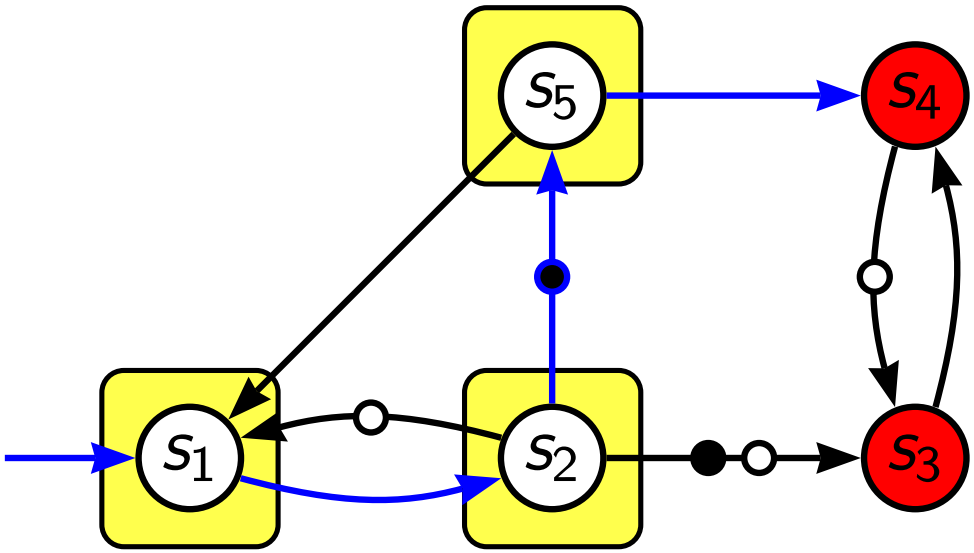
Couvreur's SCC-Based Emptiness Check



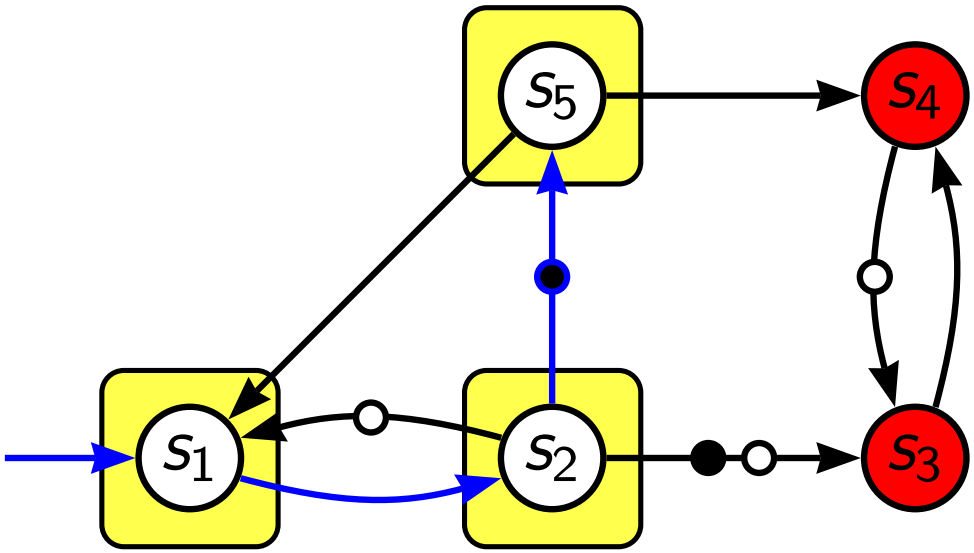
Couvreur's SCC-Based Emptiness Check



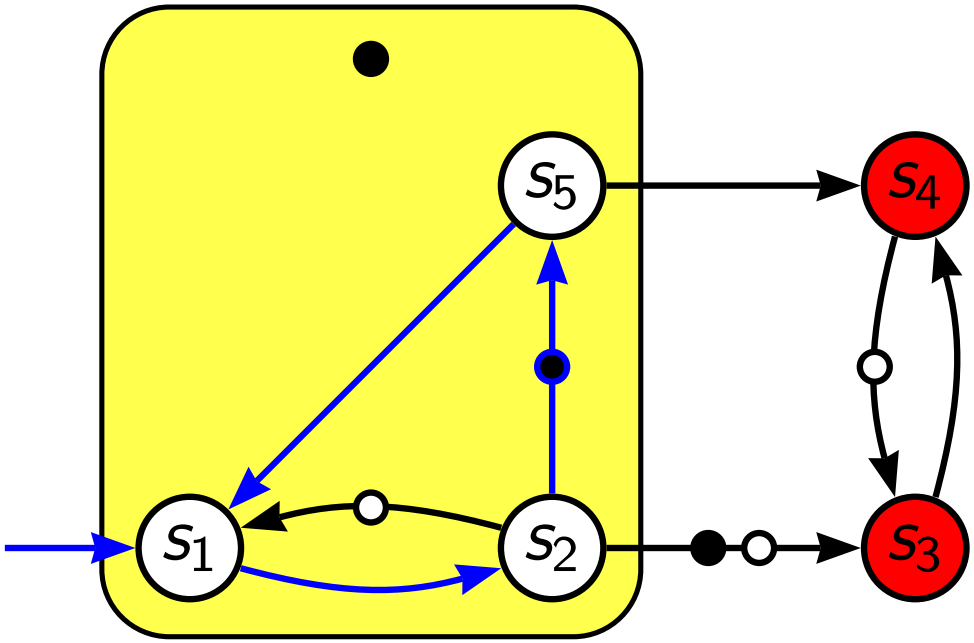
Couvreur's SCC-Based Emptiness Check



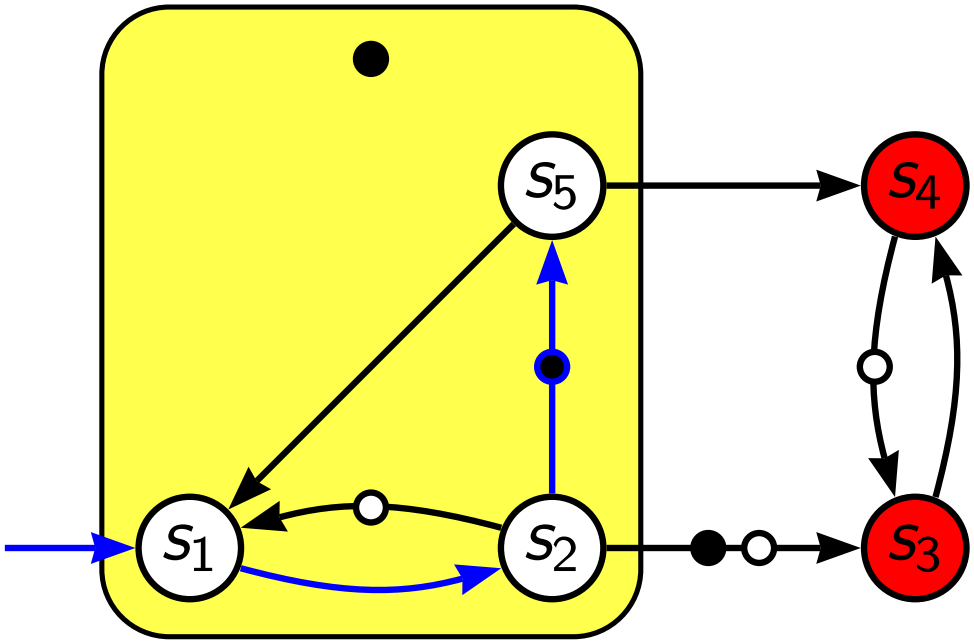
Couvreur's SCC-Based Emptiness Check



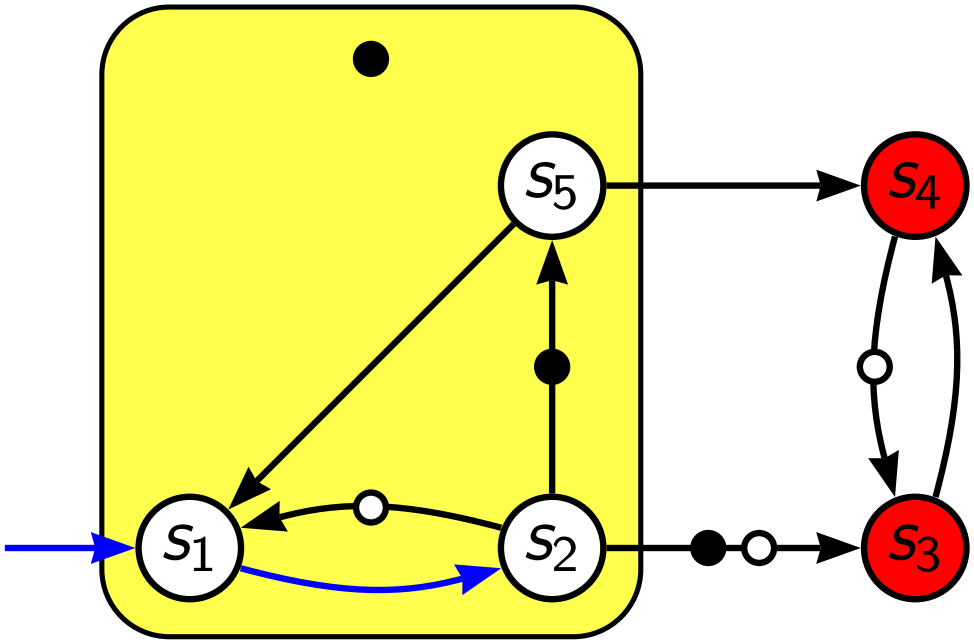
Couvreur's SCC-Based Emptiness Check



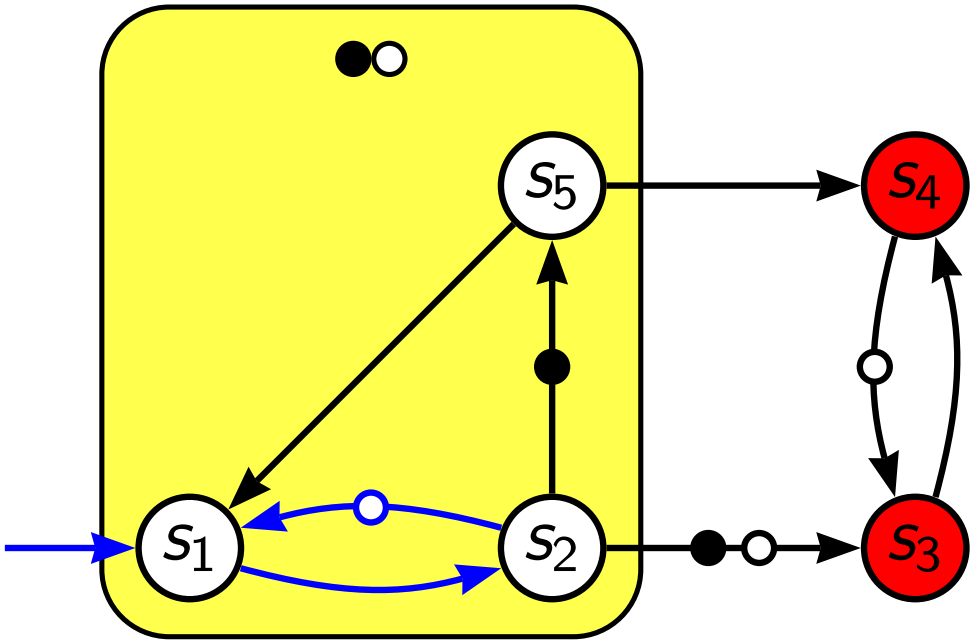
Couvreur's SCC-Based Emptiness Check



Couvreur's SCC-Based Emptiness Check



Couvreur's SCC-Based Emptiness Check



Found!

Emptiness Check for GBA, with Inclusion Checks

Context

State space reduction using symmetries:

- Global Symmetries
  (e.g., client/server system with many identical clients)

- Partial Symmetries [Haddad, Ilié, Ajami: FORTE'00]
  (e.g., two clients have different access priorities to the server)

Context

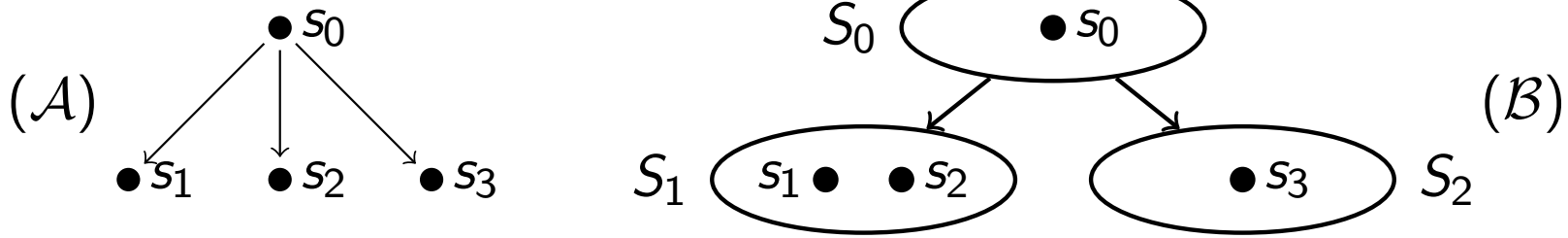
State space reduction using symmetries:

- Global Symmetries
  (e.g., client/server system with many identical clients)
  - Define equivalence classes of states.
  - Reduction straightforward.
  - But globally symmetric state spaces are uncommon.
- Partial Symmetries [Haddad, Ilié, Ajami: FORTE'00]
  (e.g., two clients have different access priorities to the server)

Context

State space reduction using symmetries:

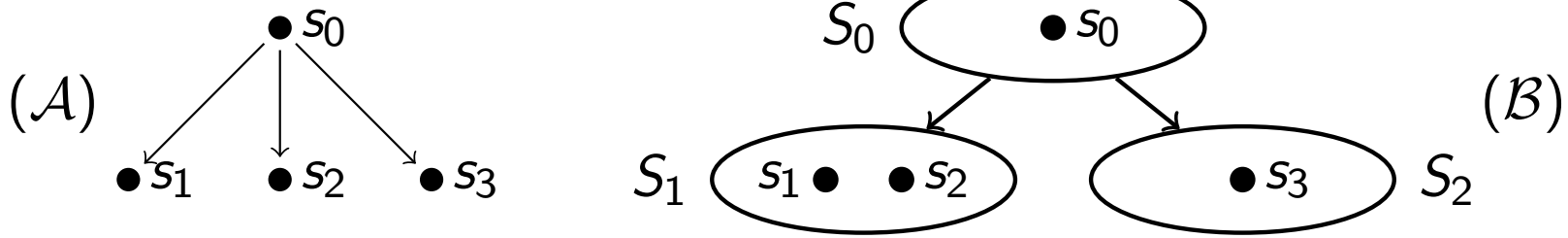
- Global Symmetries
  (e.g., client/server system with many identical clients)
  - Define equivalence classes of states.
  - Reduction straightforward.
  - But globally symmetric state spaces are uncommon.
- Partial Symmetries [Haddad, Ilié, Ajami: FORTE'00]
  (e.g., two clients have different access priorities to the server)
  - Exploits symmetries locally.
  - Partitions the successors of a state:



Context

State space reduction using symmetries:

- Global Symmetries
  (e.g., client/server system with many identical clients)
  - Define equivalence classes of states.
  - Reduction straightforward.
  - But globally symmetric state spaces are uncommon.
- Partial Symmetries [Haddad, Ilié, Ajami: FORTE'00]
  (e.g., two clients have different access priorities to the server)
  - Exploits symmetries locally.
  - Partitions the successors of a state:



  - Bounds of the "reduction": $|\mathcal{B}| \leq 2^{|\mathcal{A}|}$...

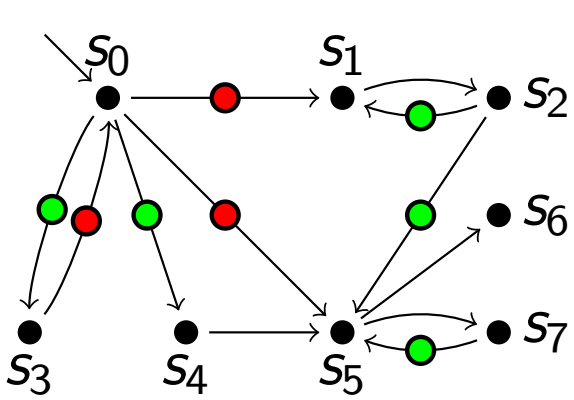
Benchmark

			basic			p. sym.		
model	n		st.	tr.	T	st.	tr.	T
WCS3	28	nonempty prod.	28	80	0.05	25	58	0.06
WCS4	28							
WCS5	28							
PO22	18							
PO23	18							
PO32	22							
WCS3	22	empty products	99	279	0.06	94	255	0.13
WCS4	22							
WCS5	22							
PO22	32							
PO23	32							
PO32	28							

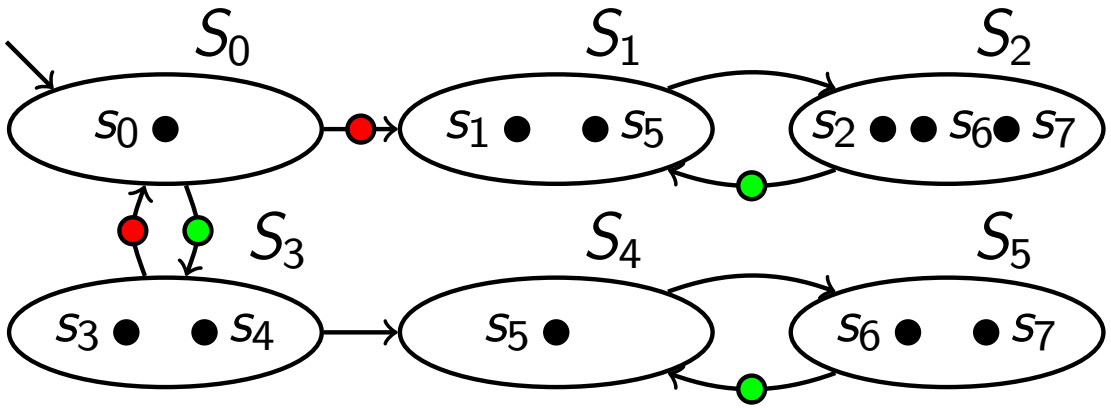
Benchmark

			basic			p. sym.		
model	n		st.	tr.	T	st.	tr.	T
WCS3	28	nonempty prod.	28	80	0.05	25	58	0.06
WCS4	28		78	250	0.06	77	202	0.14
WCS5	28		290	979	0.13	416	1309	2.76
PO22	18		252	431	0.13	690	1307	0.95
PO23	18		292	511	0.17	770	1441	1.48
PO32	22		1173	2235	0.65	2184	4730	7.40
WCS3	22	empty products	99	279	0.06	94	255	0.13
WCS4	22		434	1485	0.13	602	2063	1.60
WCS5	22		1889	7430	0.60	4224	17744	144
PO22	32		2484	5482	0.91	866	1817	2.16
PO23	32		3253	7200	1.56	952	2030	3.68
PO32	28		4617	10651	2.48	1334	2848	4.97

Generalization, from the Emptiness Check POV



$(\mathcal{A})$



$(\mathcal{B})$

Any method that can construct such a $(\mathcal{B})$ instead of $(\mathcal{A})$ so that

- $\forall s \multimap s', \forall S \ni s, \exists S' \ni s'$ such that $S \multimap S'$
- $\forall S \multimap S', \forall s' \in S', \exists s \in S$ such that $s \multimap s'$

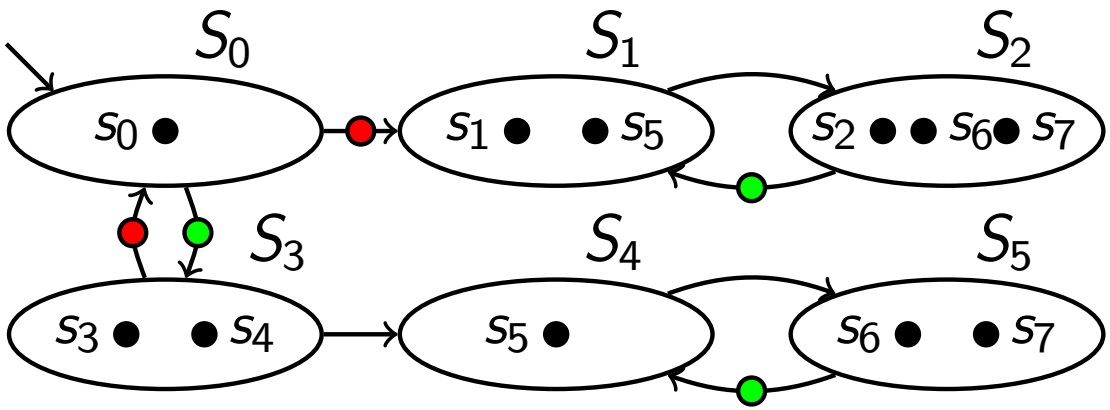
will guarantee emptiness check equivalence

$$\mathrm{Acc}(\mathcal{A}) = \emptyset \iff \mathrm{Acc}(\mathcal{B}) = \emptyset$$

Inclusion and Dead States

$$S \supseteq S' \quad \Longrightarrow \quad \Big(\mathrm{Acc}(\mathcal{B}[S]) = \emptyset \implies \mathrm{Acc}(\mathcal{B}[S']) = \emptyset\Big)$$

$(\mathcal{B})$



Benchmark

			basic			p. sym.			p. sym. + DSI		
model	n		st.	tr.	T	st.	tr.	T	st.	tr.	T
WCS3	28	nonempty prod.	28	80	0.05	25	58	0.06	26	51	0.06
WCS4	28	nonempty prod.	78	250	0.06	77	202	0.14	66	176	0.17
WCS5	28	nonempty prod.	290	979	0.13	416	1309	2.76	294	1118	6.44
PO22	18	nonempty prod.	252	431	0.13	690	1307	0.95	738	1505	1.10
PO23	18	nonempty prod.	292	511	0.17	770	1441	1.48	750	1550	1.69
PO32	22	nonempty prod.	1173	2235	0.65	2184	4730	7.40	1400	3031	3.75
WCS3	22	empty products	99	279	0.06	94	255	0.13	91	250	0.14
WCS4	22	empty products	434	1485	0.13	602	2063	1.60	568	1980	2.17
WCS5	22	empty products	1889	7430	0.60	4224	17744	144	3905	16719	107
PO22	32	empty products	2484	5482	0.91	866	1817	2.16	864	1814	2.14
PO23	32	empty products	3253	7200	1.56	952	2030	3.68	868	1830	3.08
PO32	28	empty products	4617	10651	2.48	1334	2848	4.97	1294	2784	4.78

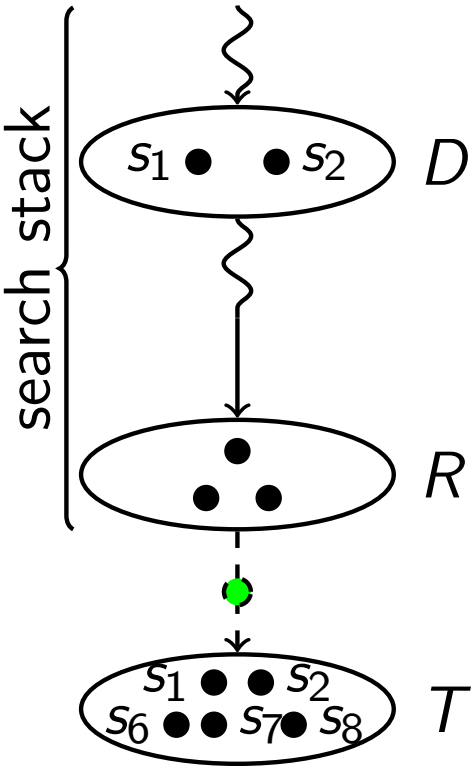
Benchmark

			basic			p. sym.			p. sym. + DSI		
model	n		st.	tr.	T	st.	tr.	T	st.	tr.	T
WCS3	28	nonempty prod.	28	80	0.05	25	58	0.06	26	51	0.06
WCS4	28		78	250	0.06	77	202	0.14	66	176	0.17
WCS5	28		290	979	0.13	416	1309	2.76	294	1118	6.44
PO22	18		252	431	0.13	690	1307	0.95	738	1505	1.10
PO23	18		292	511	0.17	770	1441	1.48	750	1550	1.69
PO32	22		1173	2235	0.65	2184	4730	7.40	1400	3031	3.75
WCS3	22	empty products	99	279	0.06	94	255	0.13	91	250	0.14
WCS4	22		434	1485	0.13	602	2063	1.60	568	1980	2.17
WCS5	22		1889	7430	0.60	4224	17744	144	3905	16719	107
PO22	32		2484	5482	0.91	866	1817	2.16	864	1814	2.14
PO23	32		3253	7200	1.56	952	2030	3.68	868	1830	3.08
PO32	28		4617	10651	2.48	1334	2848	4.97	1294	2784	4.78

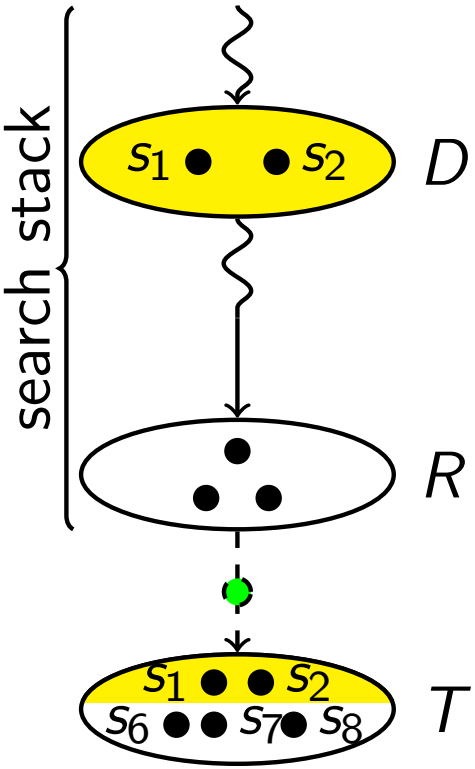
Benchmark

			basic			p. sym.			p. sym. + DSI		
model	n		st.	tr.	T	st.	tr.	T	st.	tr.	T
WCS3	28	nonempty prod.	28	80	0.05	25	58	0.06	26	51	0.06
WCS4	28	nonempty prod.	78	250	0.06	77	202	0.14	66	176	0.17
WCS5	28	nonempty prod.	290	979	0.13	416	1309	2.76	294	1118	6.44
PO22	18	nonempty prod.	252	431	0.13	690	1307	0.95	738	1505	1.10
PO23	18	nonempty prod.	292	511	0.17	770	1441	1.48	750	1550	1.69
PO32	22	nonempty prod.	1173	2235	0.65	2184	4730	7.40	1400	3031	3.75
WCS3	22	empty products	99	279	0.06	94	255	0.13	91	250	0.14
WCS4	22	empty products	434	1485	0.13	602	2063	1.60	568	1980	2.17
WCS5	22	empty products	1889	7430	0.60	4224	17744	144	3905	16719	107
PO22	32	empty products	2484	5482	0.91	866	1817	2.16	864	1814	2.14
PO23	32	empty products	3253	7200	1.56	952	2030	3.68	868	1830	3.08
PO32	28	empty products	4617	10651	2.48	1334	2848	4.97	1294	2784	4.78

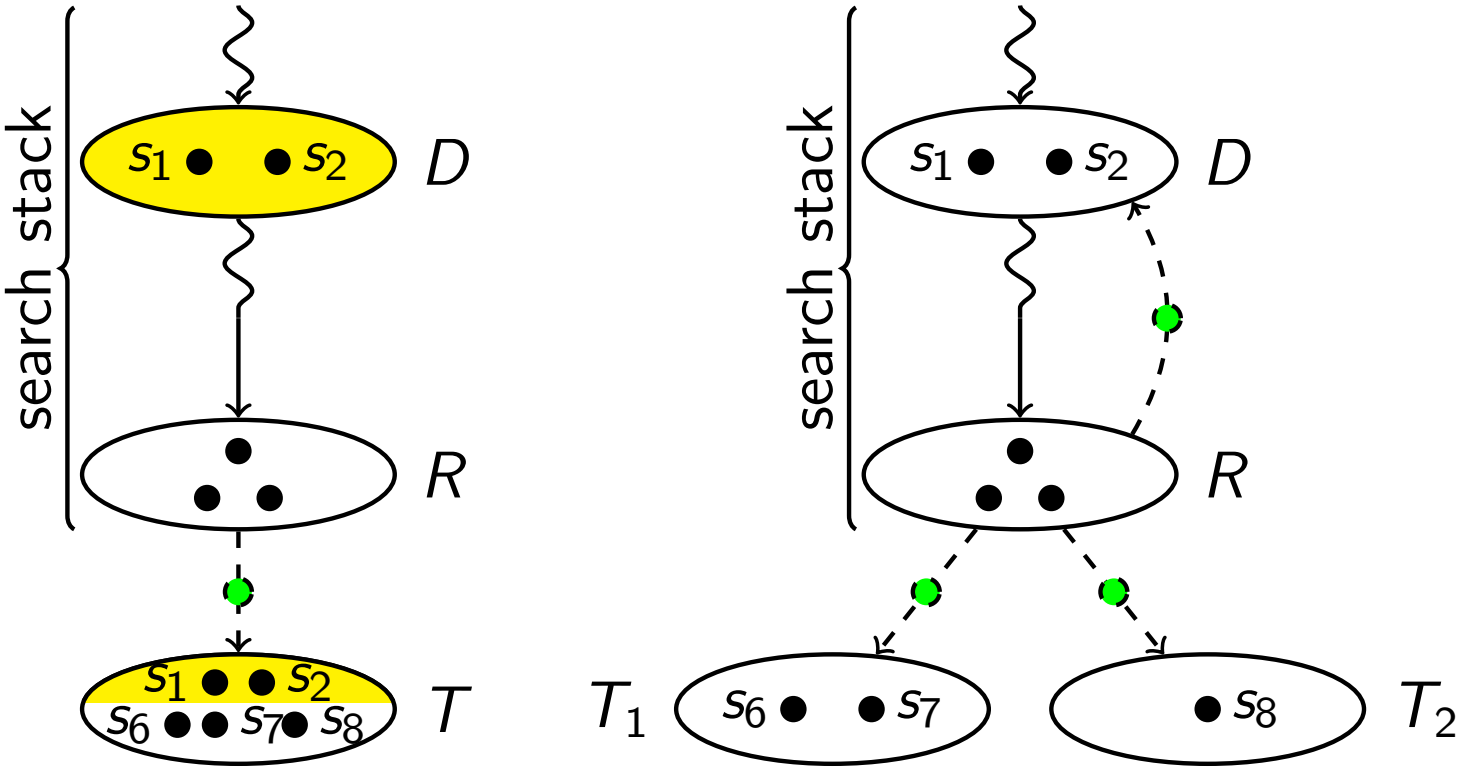
Inclusion and Search Stack States



Inclusion and Search Stack States



Inclusion and Search Stack States



- Directs the on-the-fly construction of the state space.
- Explicits a loop in the stack: good to abort early
- Tries to reuse existing states: a heuristic attempt to fight $2^{|\mathcal{A}|}$

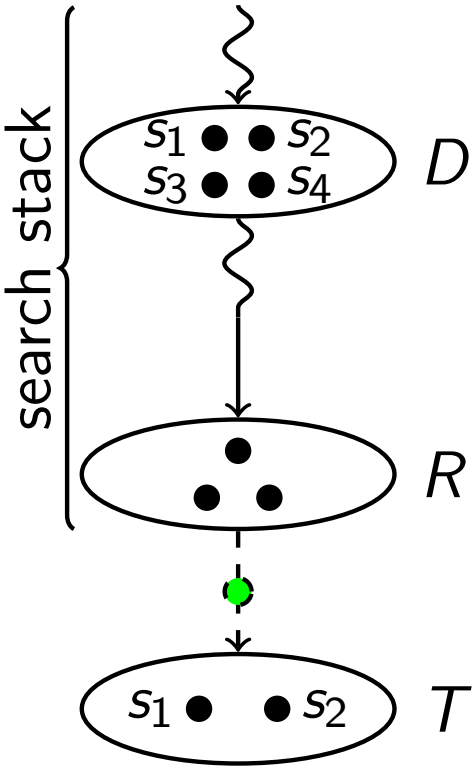
Benchmark

			basic			p. sym. + DSI			p.sym.+DSI+Incl		
model	n		st.	tr.	T	st.	tr.	T	st.	tr.	T
WCS3	28	nonempty prod.	28	80	0.05	26	51	0.06	24	45	0.05
WCS4	28	nonempty prod.	78	250	0.06	66	176	0.17	43	96	0.10
WCS5	28	nonempty prod.	290	979	0.13	294	1118	6.44	106	287	0.95
PO22	18	nonempty prod.	252	431	0.13	738	1505	1.10	738	1505	1.10
PO23	18	nonempty prod.	292	511	0.17	750	1550	1.69	750	1550	1.69
PO32	22	nonempty prod.	1173	2235	0.65	1400	3031	3.75	1400	3031	3.75
WCS3	22	empty products	99	279	0.06	91	250	0.14	73	194	0.13
WCS4	22	empty products	434	1485	0.13	568	1980	2.17	297	940	1.07
WCS5	22	empty products	1889	7430	0.60	3905	16719	107	1370	4815	23.7
PO22	32	empty products	2484	5482	0.91	864	1814	2.14	864	1814	2.14
PO23	32	empty products	3253	7200	1.56	868	1830	3.08	868	1831	3.09
PO32	28	empty products	4617	10651	2.48	1294	2784	4.78	1294	2784	4.78

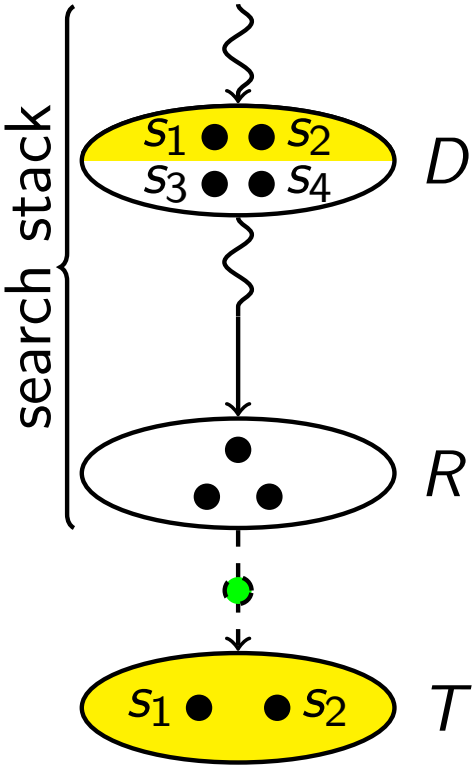
Benchmark

			basic			p. sym. + DSI			p.sym.+DSI+Incl		
model	n		st.	tr.	T	st.	tr.	T	st.	tr.	T
WCS3	28	nonempty prod.	28	80	0.05	26	51	0.06	24	45	0.05
WCS4	28	nonempty prod.	78	250	0.06	66	176	0.17	43	96	0.10
WCS5	28	nonempty prod.	290	979	0.13	294	1118	6.44	106	287	0.95
PO22	18	nonempty prod.	252	431	0.13	738	1505	1.10	738	1505	1.10
PO23	18	nonempty prod.	292	511	0.17	750	1550	1.69	750	1550	1.69
PO32	22	nonempty prod.	1173	2235	0.65	1400	3031	3.75	1400	3031	3.75
WCS3	22	empty products	99	279	0.06	91	250	0.14	73	194	0.13
WCS4	22	empty products	434	1485	0.13	568	1980	2.17	297	940	1.07
WCS5	22	empty products	1889	7430	0.60	3905	16719	107	1370	4815	23.7
PO22	32	empty products	2484	5482	0.91	864	1814	2.14	864	1814	2.14
PO23	32	empty products	3253	7200	1.56	868	1830	3.08	868	1831	3.09
PO32	28	empty products	4617	10651	2.48	1294	2784	4.78	1294	2784	4.78

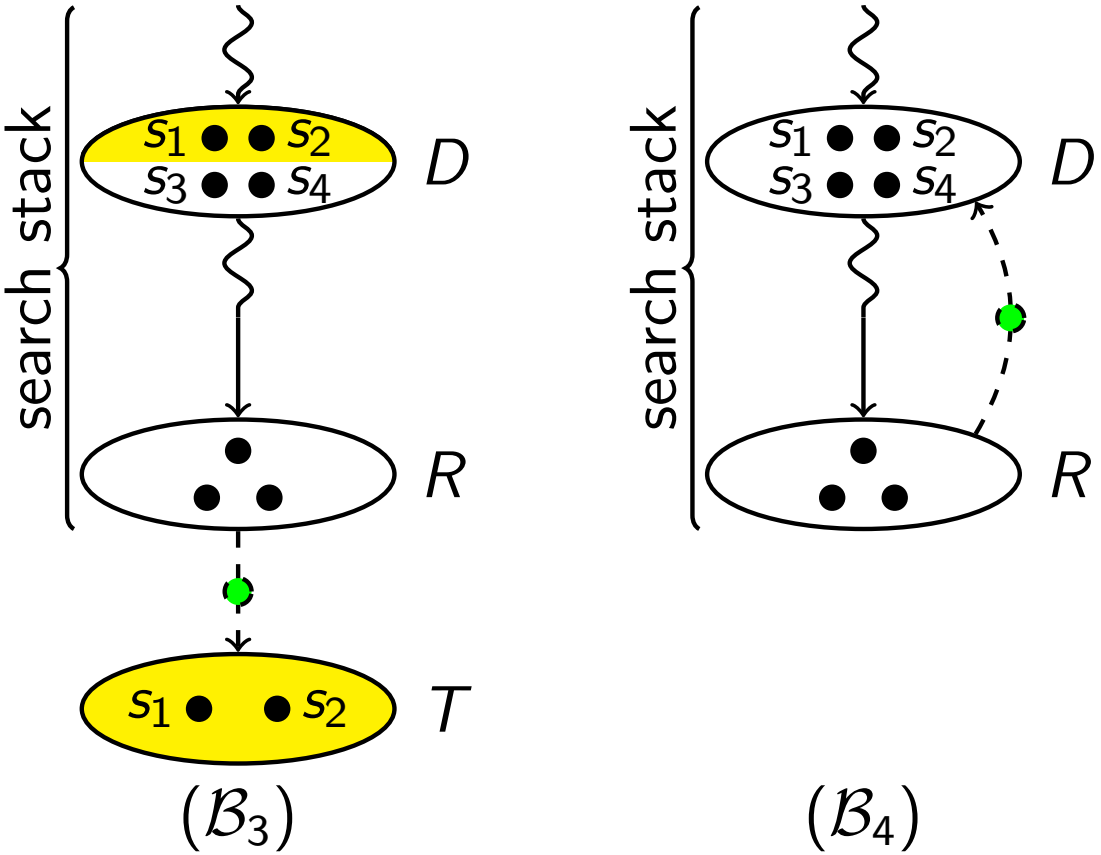
Reverse Inclusion and Search Stack States



Reverse Inclusion and Search Stack States



Reverse Inclusion and Search Stack States



- In this case we only have $\mathrm{Acc}(\mathcal{B}_4) = \emptyset \implies \mathrm{Acc}(\mathcal{B}_3) = \emptyset$
- The emptiness check returns "*empty*" or "*I don't know*"

Benchmark

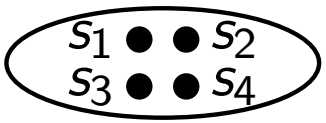
			basic			p.sym.+DSI+Incl			p.sym.+DST+RIncl		
model	n		st.	tr.	T	st.	tr.	T	st.	tr.	T
WCS3	28	nonempty prod.	28	80	0.05	24	45	0.05	21	39	0.05
WCS4	28		78	250	0.06	43	96	0.10	29	57	0.06
WCS5	28		290	979	0.13	106	287	0.95	39	82	0.07
PO22	18		252	431	0.13	738	1505	1.10	738	1505	1.10
PO23	18		292	511	0.17	750	1550	1.69	750	1550	1.69
PO32	22		1173	2235	0.65	1400	3031	3.75	1392	2982	3.71
WCS3	22	empty products	99	279	0.06	73	194	0.13	30	70	0.07
WCS4	22		434	1485	0.13	297	940	1.07	64	177	0.15
WCS5	22		1889	7430	0.60	1370	4815	23.7	136	428	0.46
PO22	32		2484	5482	0.91	865	1814	2.14	864	1813	2.14
PO23	32		3253	7200	1.56	868	1831	3.09	868	1830	3.08
PO32	28		4617	10651	2.48	1294	2784	4.78	1294	2783	4.79

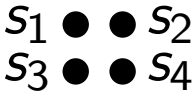
Benchmark

			basic			p.sym.+DSI+Incl			p.sym.+DST+RIncl		
model	n		st.	tr.	T	st.	tr.	T	st.	tr.	T
WCS3	28	nonempty prod.	28	80	0.05	24	45	0.05	21	39	0.05
WCS4	28		78	250	0.06	43	96	0.10	29	57	0.06
WCS5	28		290	979	0.13	106	287	0.95	39	82	0.07
PO22	18		252	431	0.13	738	1505	1.10	738	1505	1.10
PO23	18		292	511	0.17	750	1550	1.69	750	1550	1.69
PO32	22		1173	2235	0.65	1400	3031	3.75	1392	2982	3.71
WCS3	22	empty products	99	279	0.06	73	194	0.13	30	70	0.07
WCS4	22		434	1485	0.13	297	940	1.07	64	177	0.15
WCS5	22		1889	7430	0.60	1370	4815	23.7	136	428	0.46
PO22	32		2484	5482	0.91	865	1814	2.14	864	1813	2.14
PO23	32		3253	7200	1.56	868	1831	3.09	868	1830	3.08
PO32	28		4617	10651	2.48	1294	2784	4.78	1294	2783	4.79

Summary and Perspectives

Requirements:

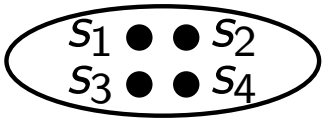
- A representation of 

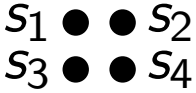
smaller than that of 

- A construction that guarantees
  - $\forall s \longrightarrow s', \forall S \ni s, \exists S' \ni s'$ such that $S \longrightarrow S'$
  - $\forall S \longrightarrow S', \forall s' \in S', \exists s \in S$ such that $s \longrightarrow s'$
- A representation of states that allows inclusion checks
- Fast lookup of candidates for inclusion checks
- A decomposition operation (optional)

Summary and Perspectives

Requirements:

- A representation of 

smaller than that of 

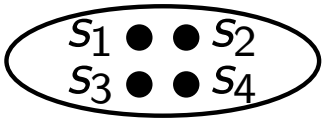
- A construction that guarantees
  - $\forall s \longrightarrow s', \forall S \ni s, \exists S' \ni s'$ such that $S \longrightarrow S'$
  - $\forall S \longrightarrow S', \forall s' \in S', \exists s \in S$ such that $s \longrightarrow s'$
- A representation of states that allows inclusion checks
- Fast lookup of candidates for inclusion checks
- A decomposition operation (optional)

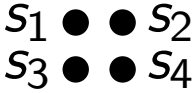
Two new emptiness checks:

- Correct, with inclusion in the stack
- Approximative, with the reverse inclusion

Summary and Perspectives

Requirements:

- A representation of 

smaller than that of 

- A construction that guarantees
  - $\forall s \longrightarrow s', \forall S \ni s, \exists S' \ni s'$ such that $S \longrightarrow S'$
  - $\forall S \longrightarrow S', \forall s' \in S', \exists s \in S$ such that $s \longrightarrow s'$
- A representation of states that allows inclusion checks
- Fast lookup of candidates for inclusion checks
- A decomposition operation (optional)

Two new emptiness checks:

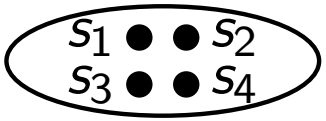
- Correct, with inclusion in the stack
- Approximative, with the reverse inclusion

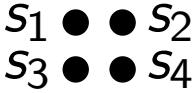
What next?

- Both inclusions could be combined.
- Investigate other state-space reductions with "set" states.
- Conditions probably too strong.

Summary and Perspectives

Requirements:

- A representation of 

smaller than that of 

- A construction that guarantees
  - $\forall s \longrightarrow s', \forall S \ni s, \exists S' \ni s'$ such that $S \longrightarrow S'$
  - $\forall S \longrightarrow S', \forall s' \in S', \exists s \in S$ such that $s \longrightarrow s'$
- A representation of states that allows inclusion checks
- Fast lookup of candidates for inclusion checks
- A decomposition operation (optional)

Two new emptiness checks:

- Correct, with inclusion in the stack
- Approximative, with the reverse inclusion

What next?

- Both inclusions could be combined.
- Investigate other state-space reductions with "set" states.
- Conditions probably too strong.