Combining Explicit and Symbolic Approaches for Better On-the-Fly LTL Model Checking

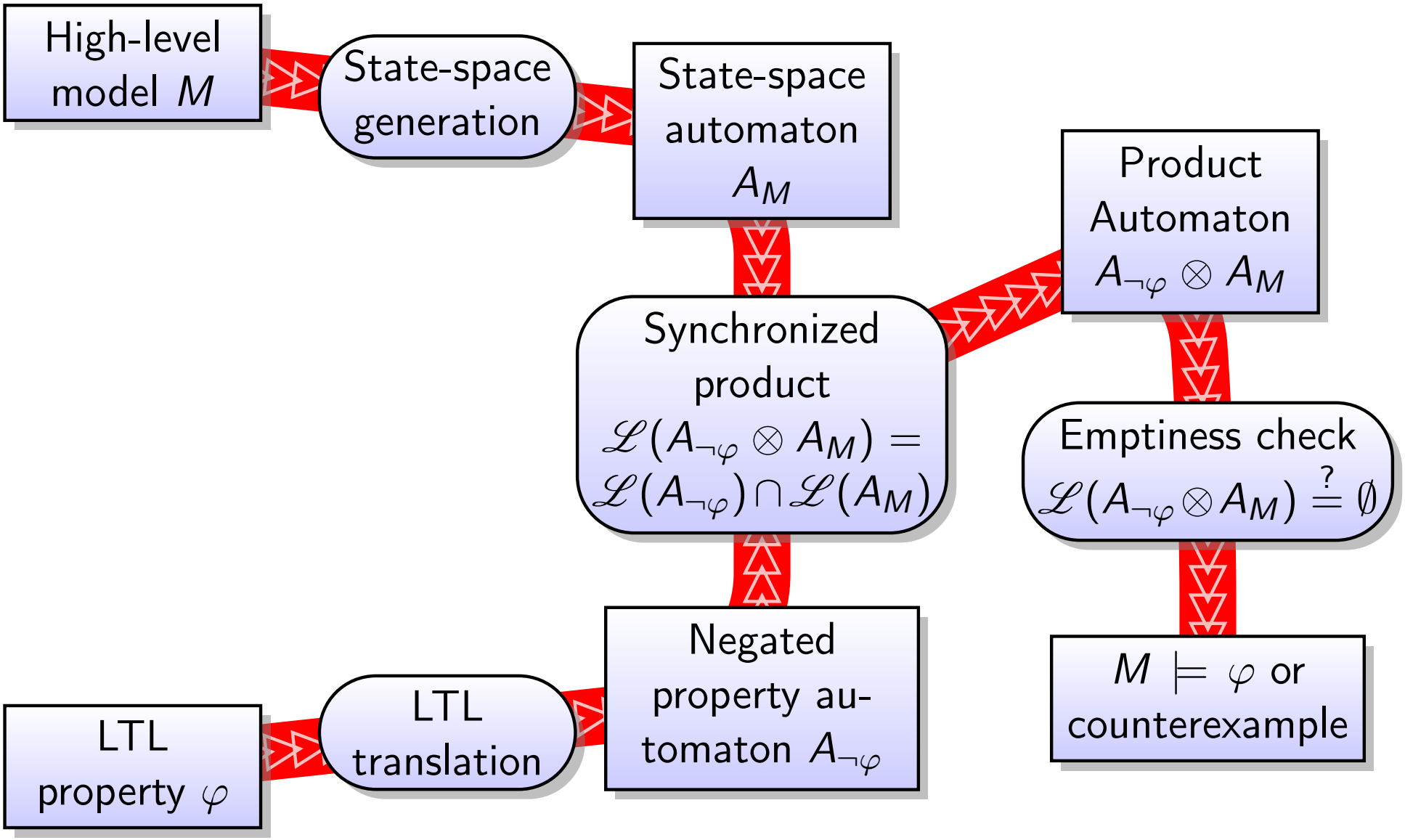
Kais Klai (LIPN/Paris 13)
Joint work with A. Duret-Lutz, D. Poitrenaud and Y. Thierry-Mieg

MeFoSyLoMa 4 Novembre 2011

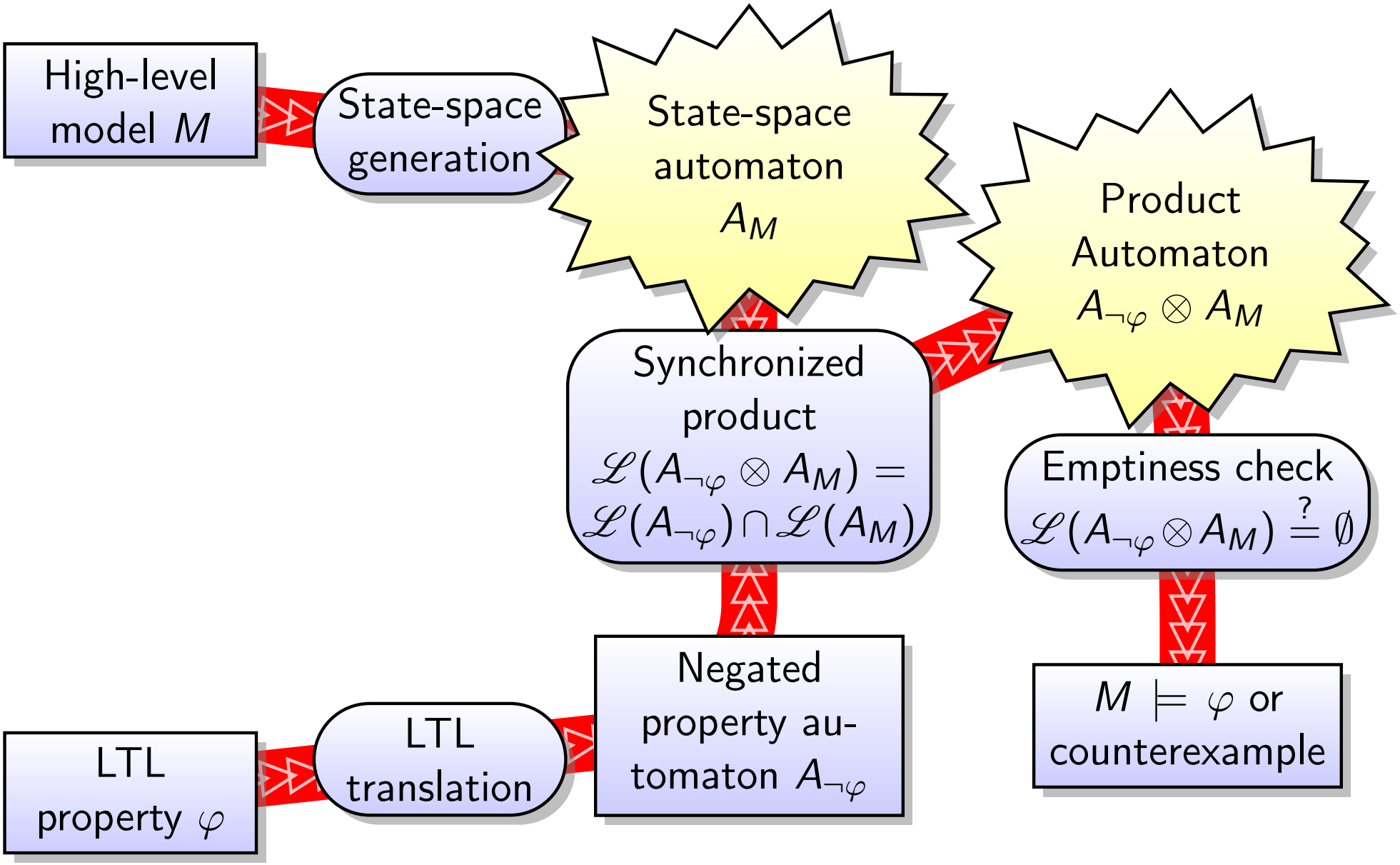
Outline

1. Automata-Theoretic Explicit LTL Model Checking
2. Explicit Approach
3. Fully Symbolic
4. Hybrid approaches
   - SOG
   - SOP
   - SLAP
   - SLAP-FST
5. Summary
6. Experiments
7. Results
8. Conclusion and Perspectives

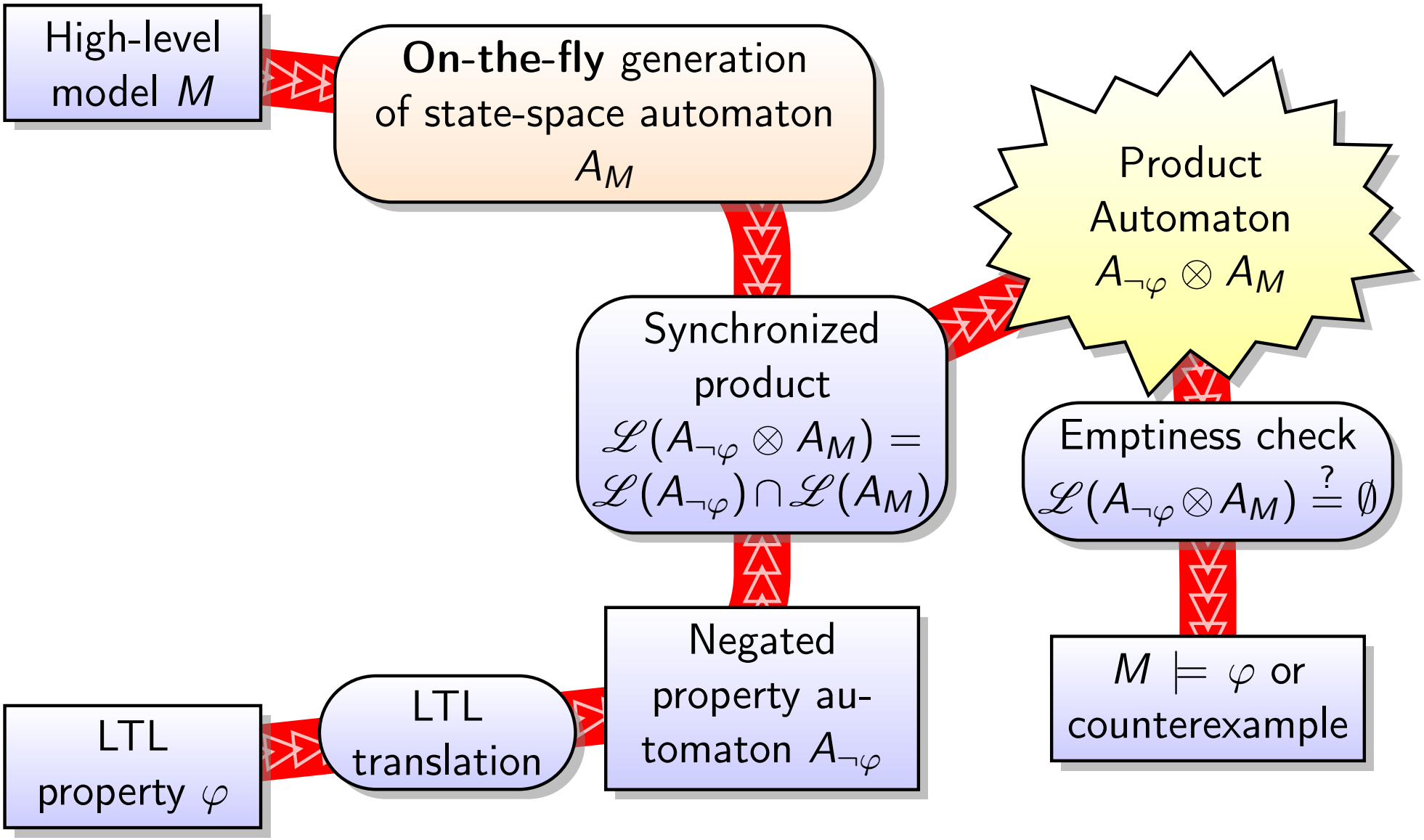
Automata-Theoretic Explicit LTL Model Checking



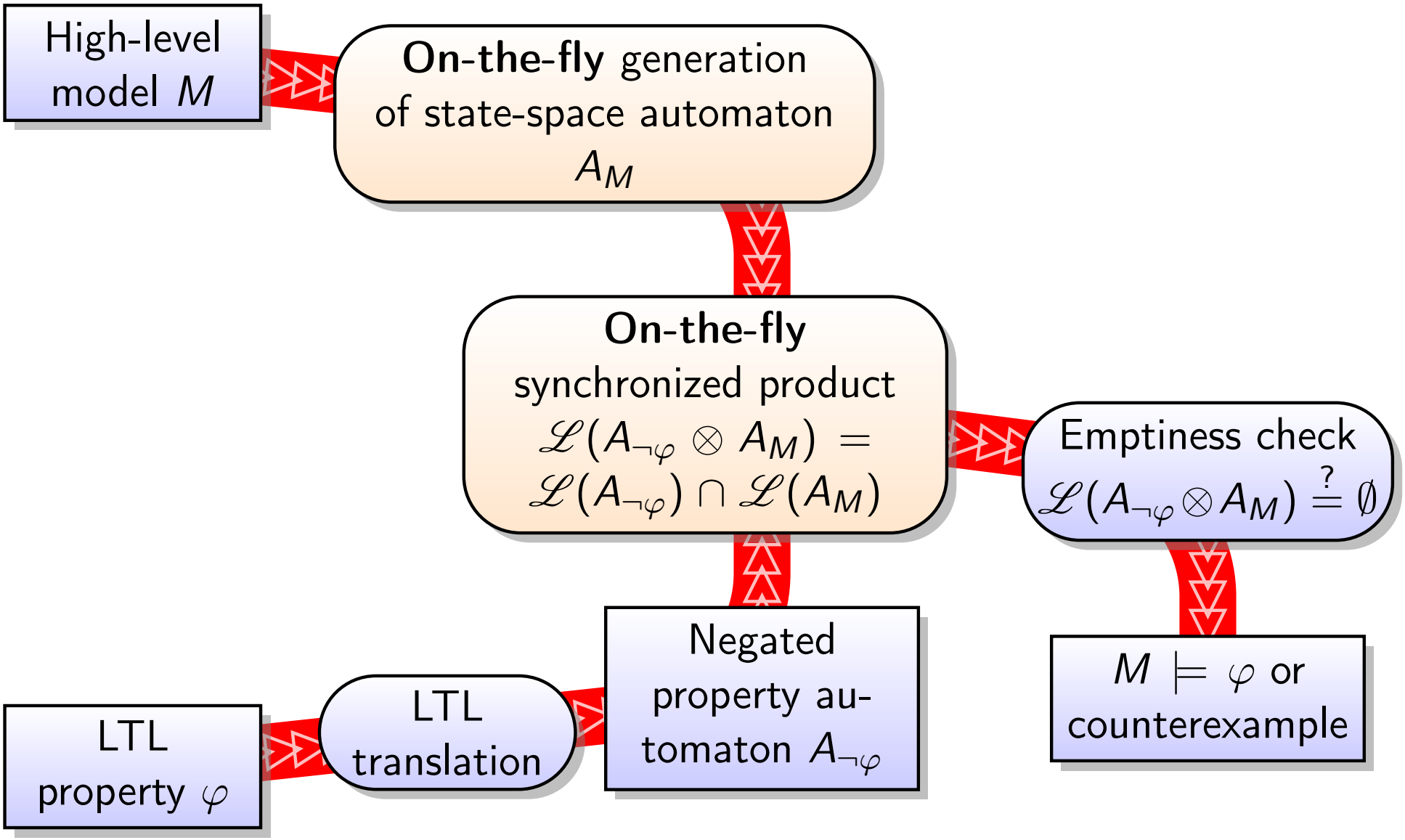
Automata-Theoretic Explicit LTL Model Checking



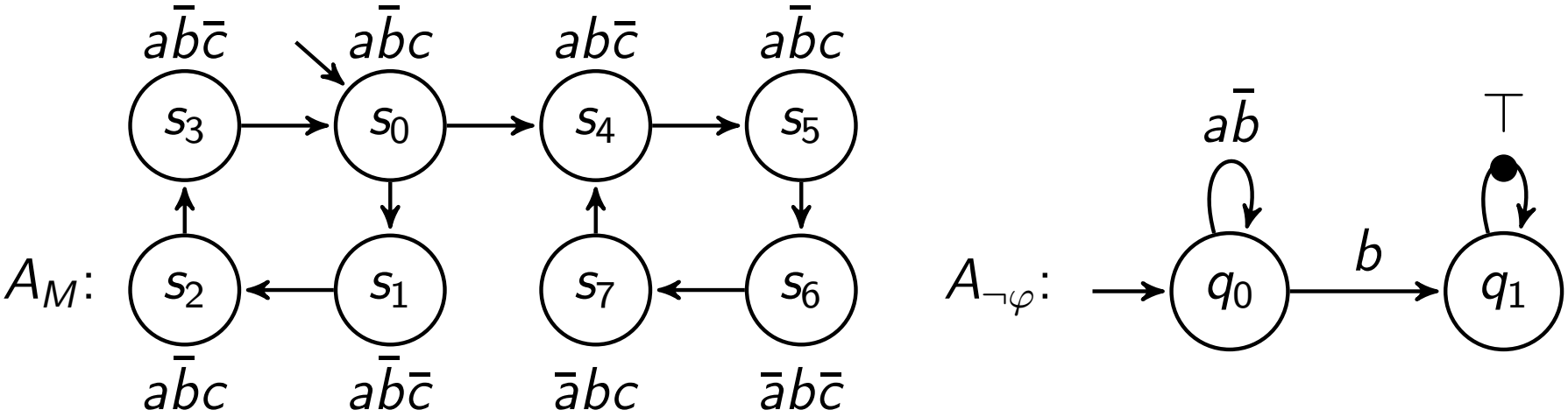
Automata-Theoretic Explicit LTL Model Checking



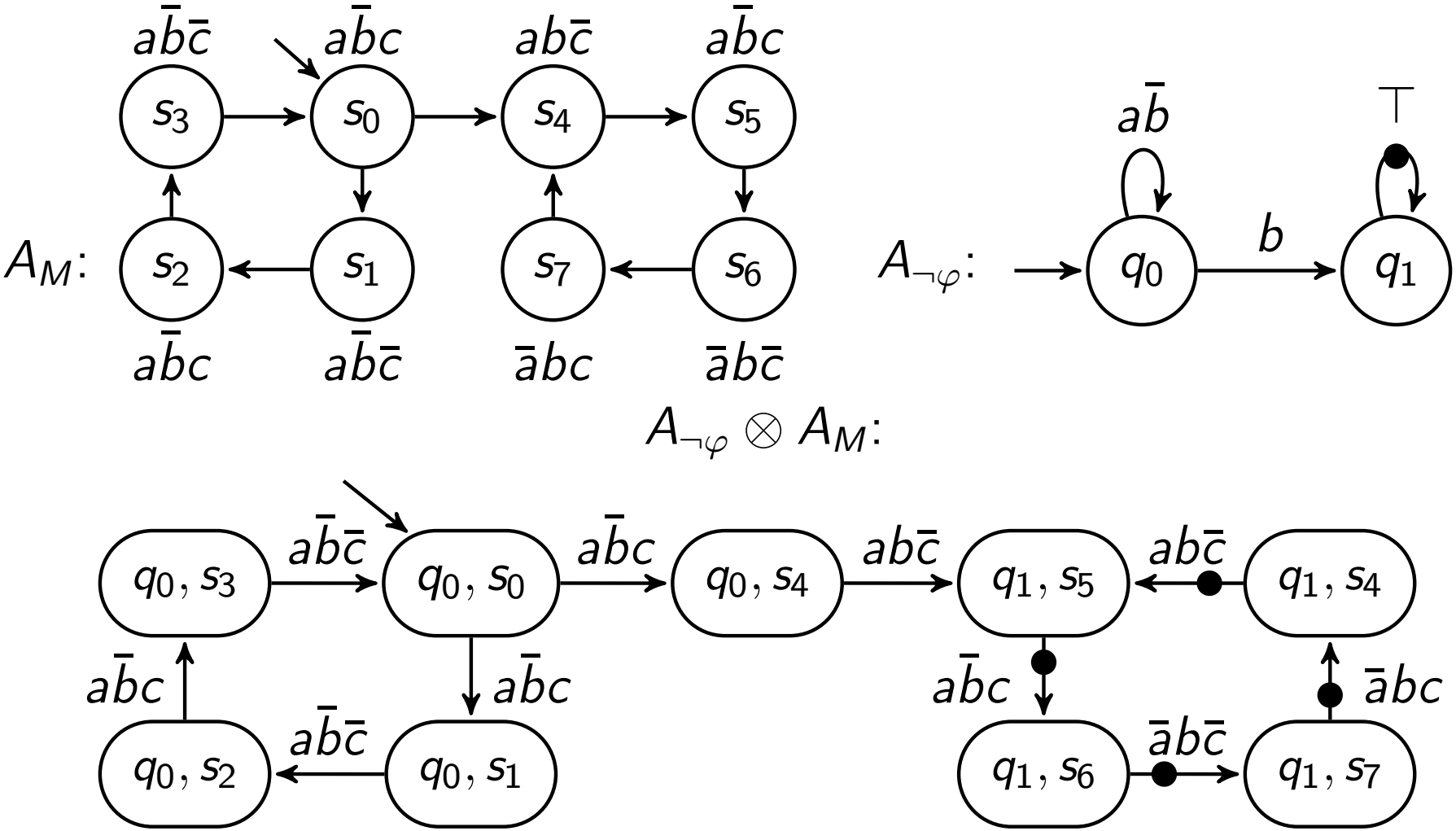
Automata-Theoretic Explicit LTL Model Checking



Explicit Approach



Explicit Approach



- Emptiness check = (on-the-fly) emptiness check of the product
- State explosion problem

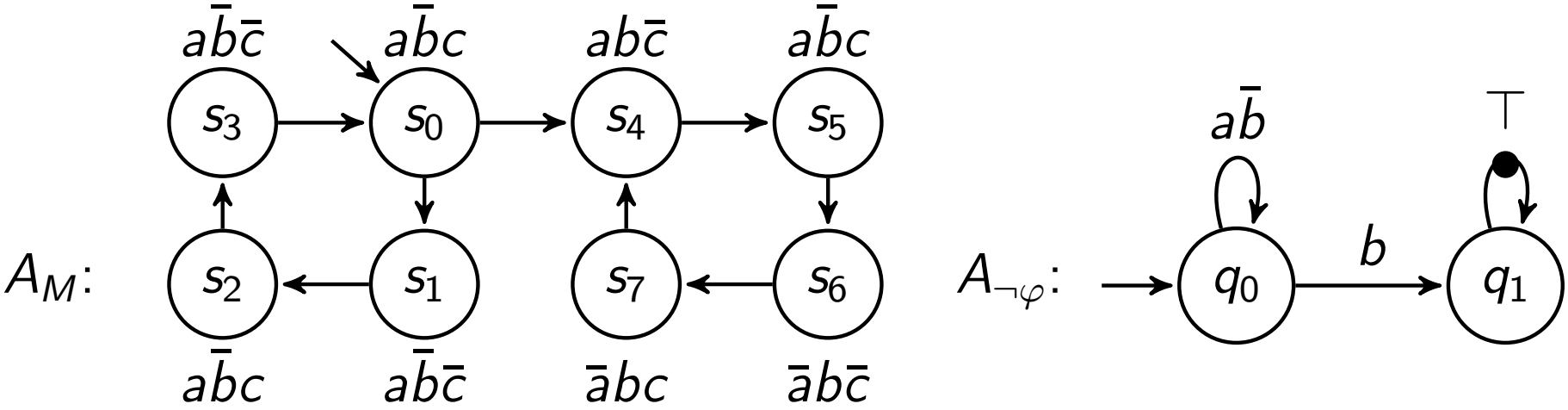
Fully Symbolic Approach (FS)

- Encode the Kripke structure and the property automaton using BDDs.
- Combine the two BDDs. (Symbolic product.)
- Use fixpoint computations to decide whether the product contains an accepting cycle.
- Two symbolic emptiness checks compared:
  - EL
  - OWCTY
- Emptiness check is not on-the-fly
- No suttering invariant reductions

Hybrid approaches

- Goal : combine explicit and symbolic approaches
- Get the best of both worlds
- Explicit graphs with symbolic nodes (aggregates)
- The system
  1. SOG: event-based logic (Phd Thesis Kais Klai 2003, S. Haddad, J-M. Ilié and K. Klai ATVA'04)
  2. SOG: state-based logic (K. Klai and D. Poitrenaud, PN'08)
- The product:
  - Symbolic Observation Product (SOP)
  - Self Loop Aggregation Product (SLAP)
  - SLAP - Fully Symbolic search in Terminal states (SLAP-FST)

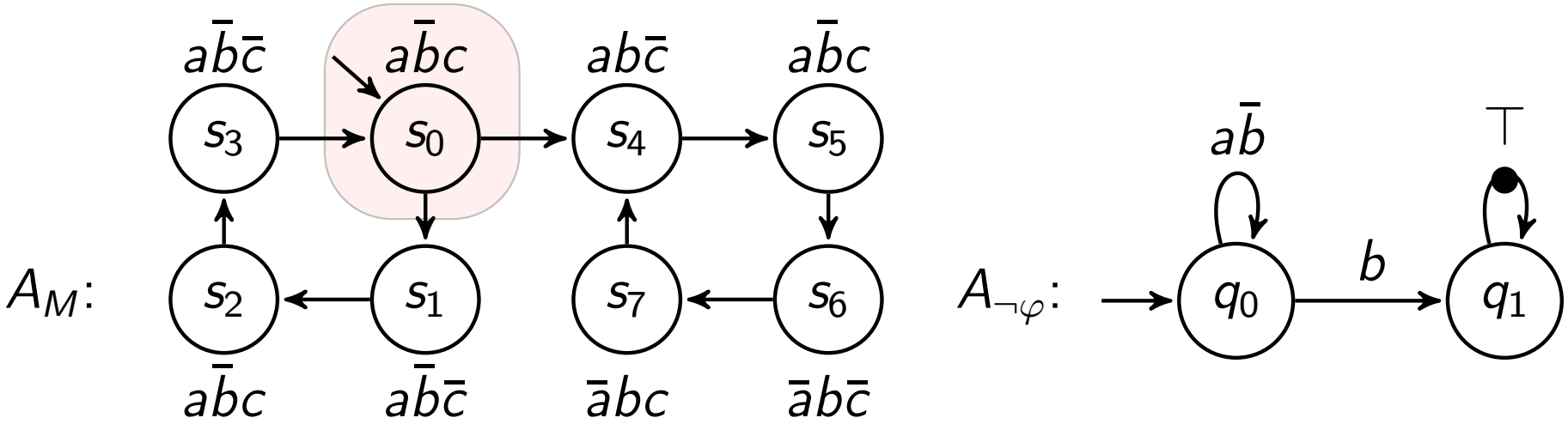
Symbolic Observation Graph (SOG)



Symbolic Observation **Graph**:

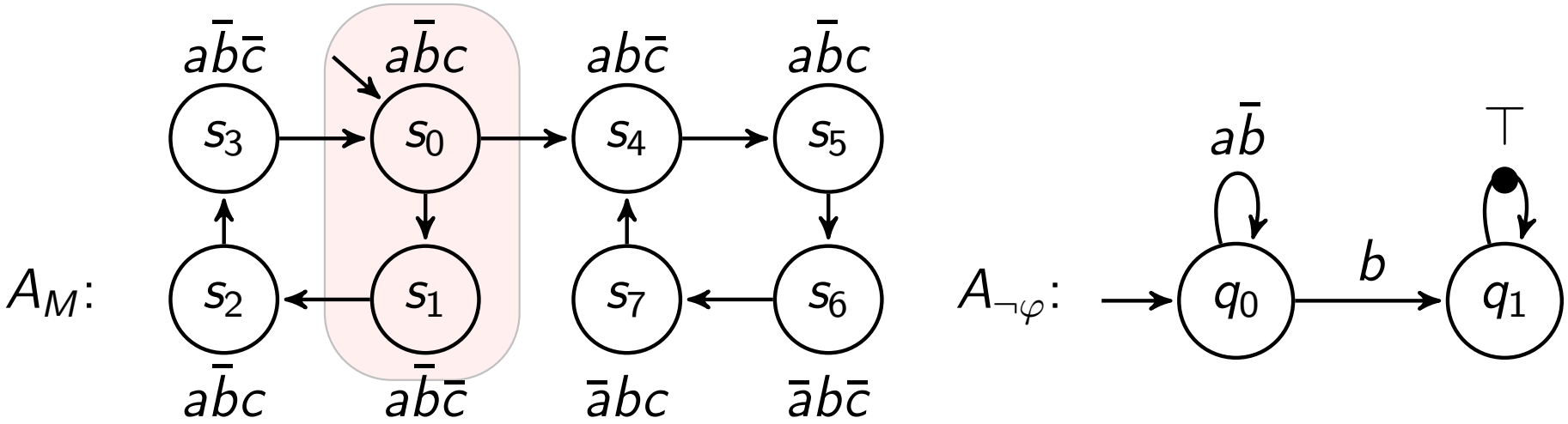
- For stuttering invariant properties (e.g., LTL$\setminus$ **X**)
- Ignore non-observable propositions
- Aggregate Kripke states with homogeneous labels
- Represent aggregates using BDDs

Symbolic Observation Graph (SOG)



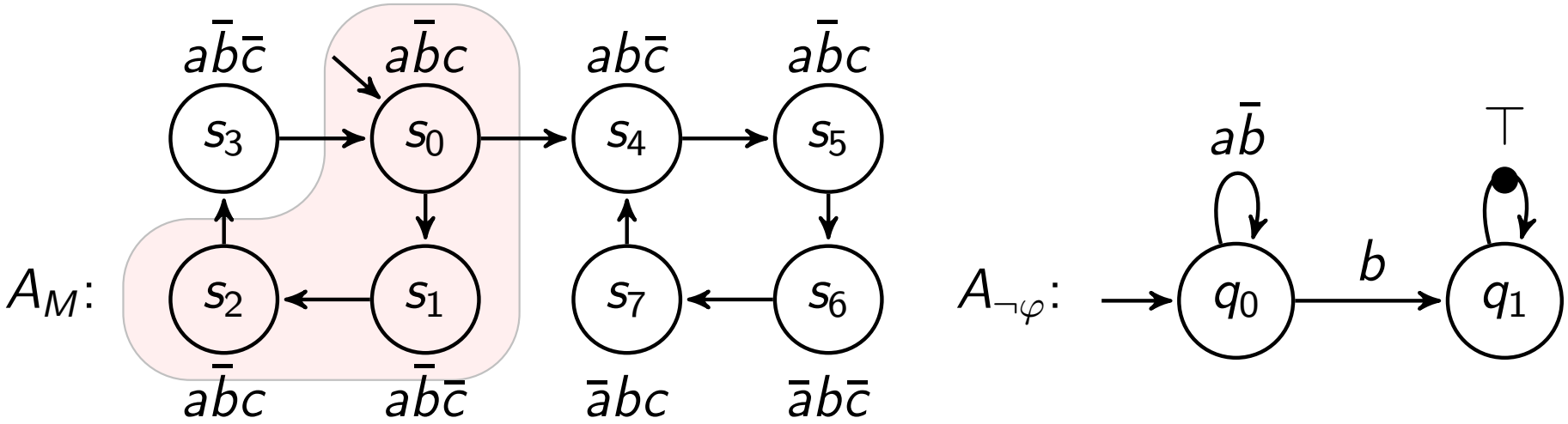
Symbolic Observation **Graph**:

Symbolic Observation Graph (SOG)



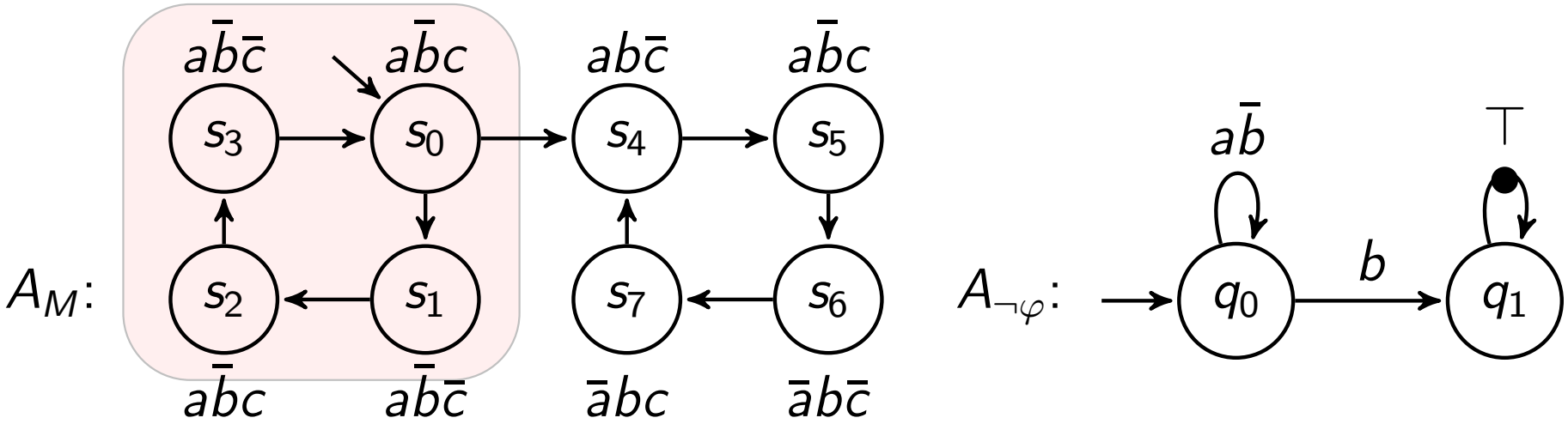
Symbolic Observation **Graph**:

Symbolic Observation Graph (SOG)



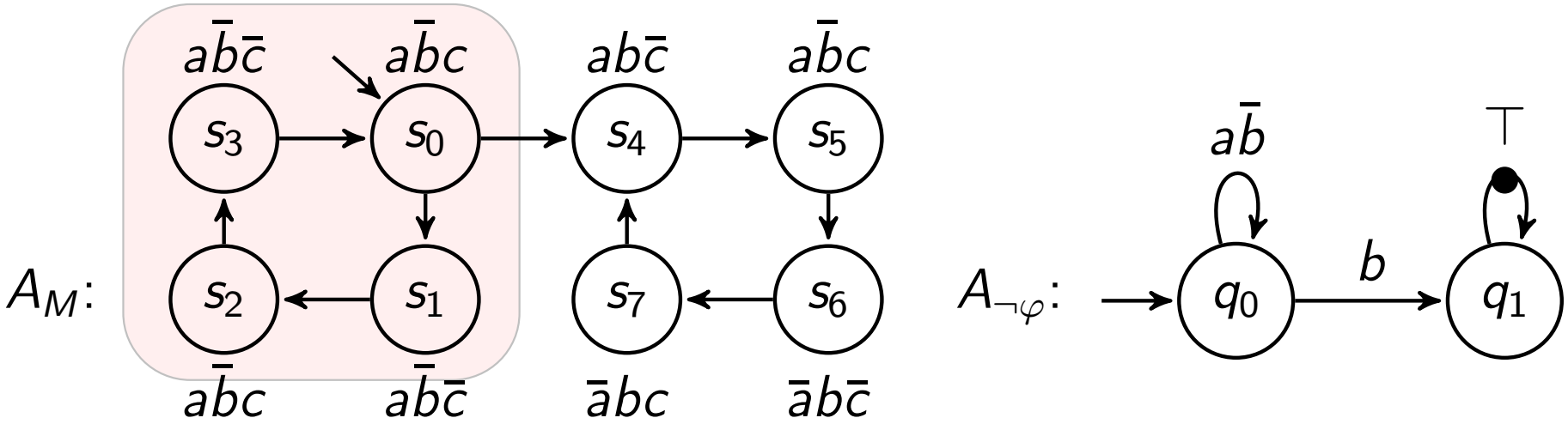
Symbolic Observation **Graph**:

Symbolic Observation Graph (SOG)

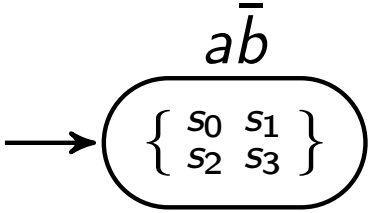


Symbolic Observation **Graph**:

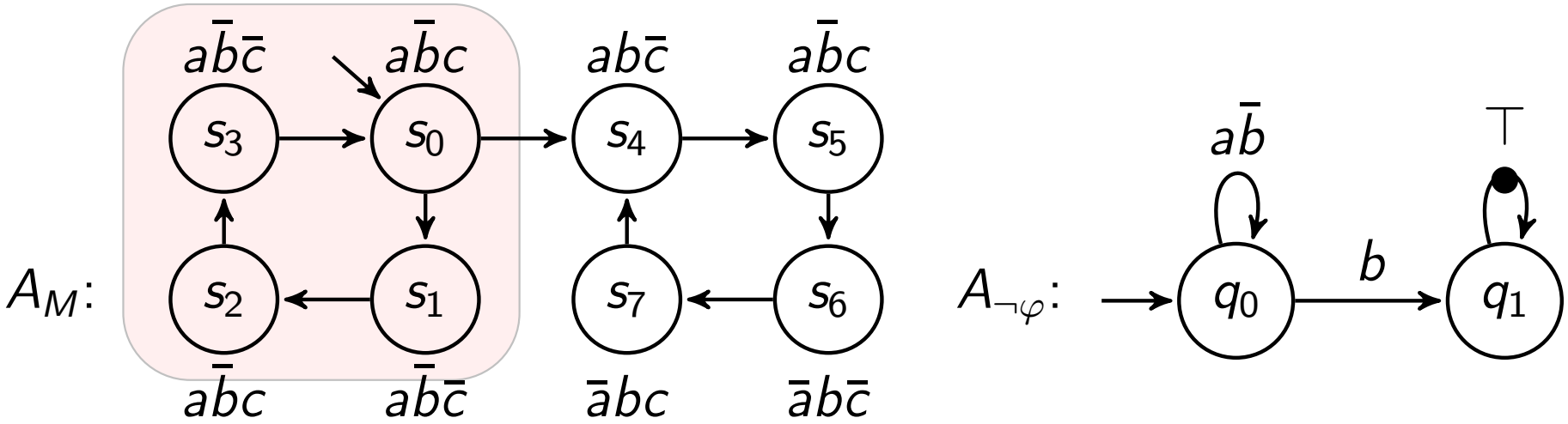
Symbolic Observation Graph (SOG)



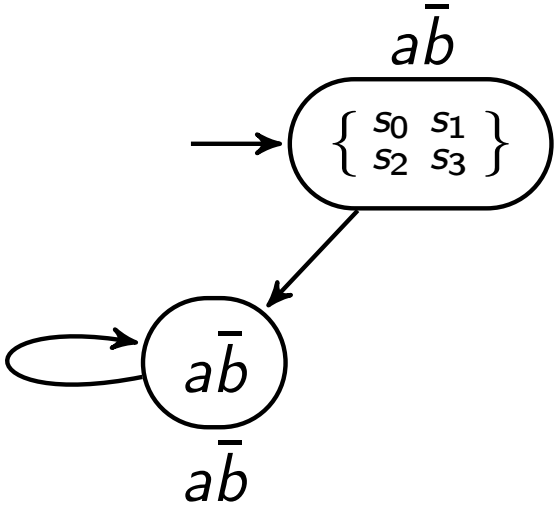
Symbolic Observation **Graph**:



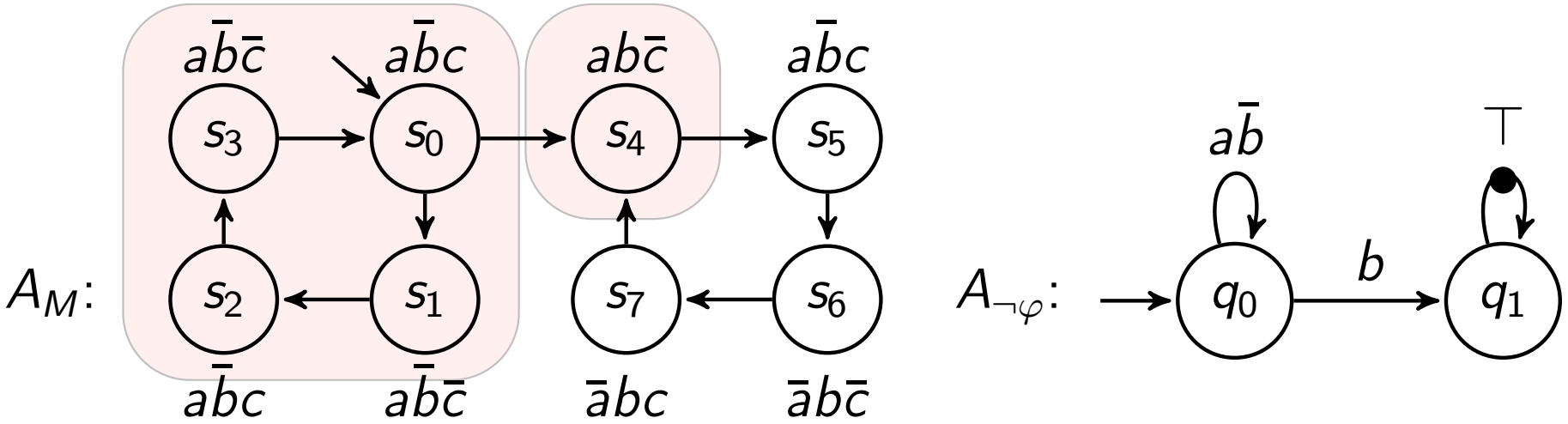
Symbolic Observation Graph (SOG)



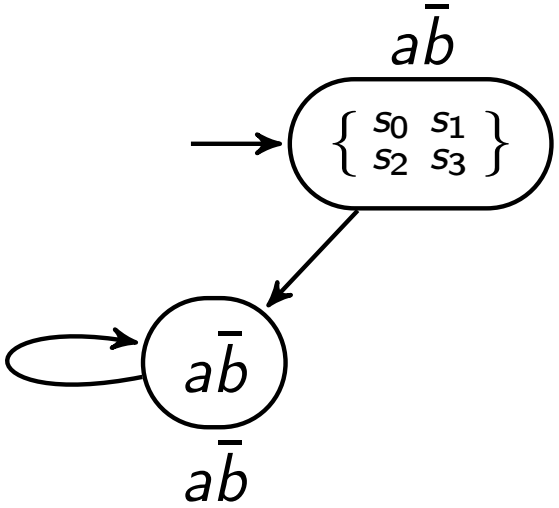
Symbolic Observation **Graph**:



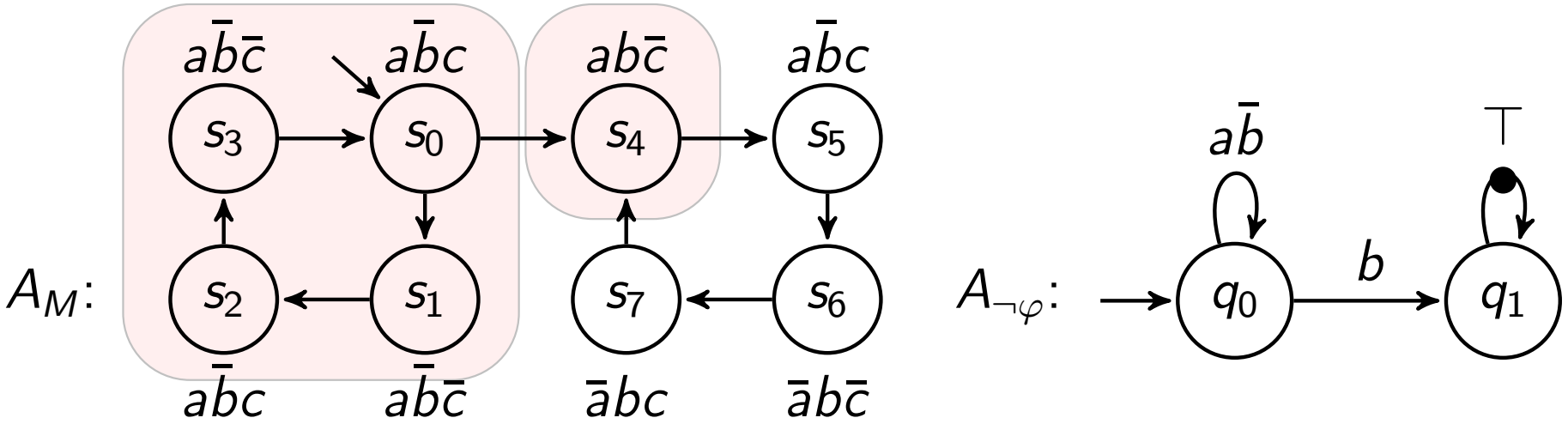
Symbolic Observation Graph (SOG)



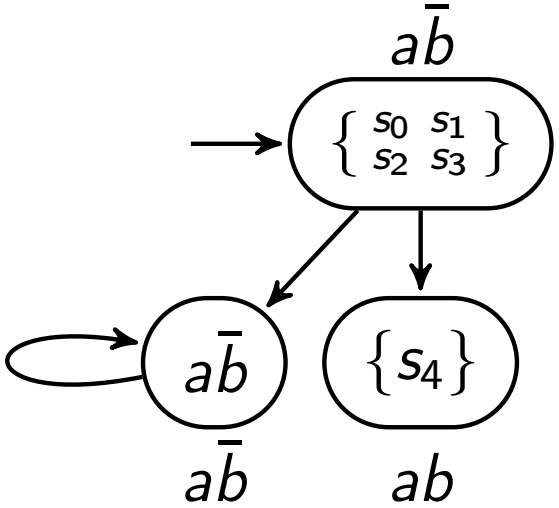
Symbolic Observation **Graph**:



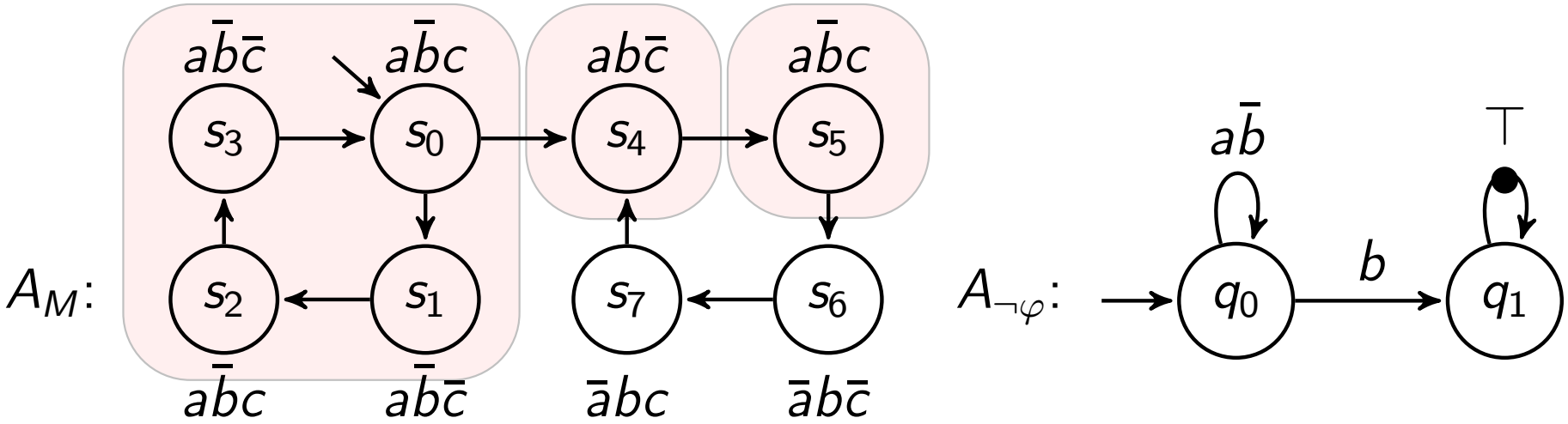
Symbolic Observation Graph (SOG)



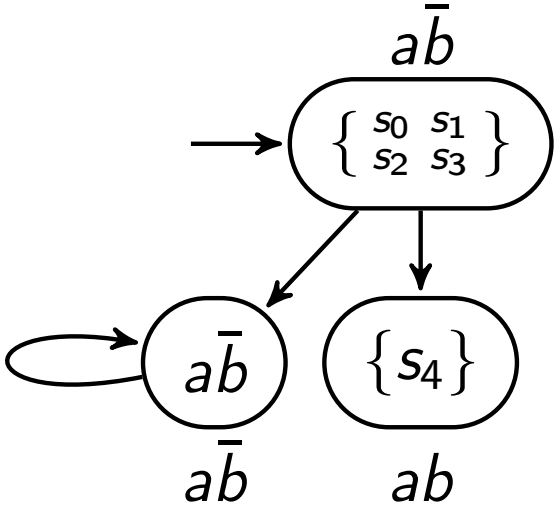
Symbolic Observation **Graph**:



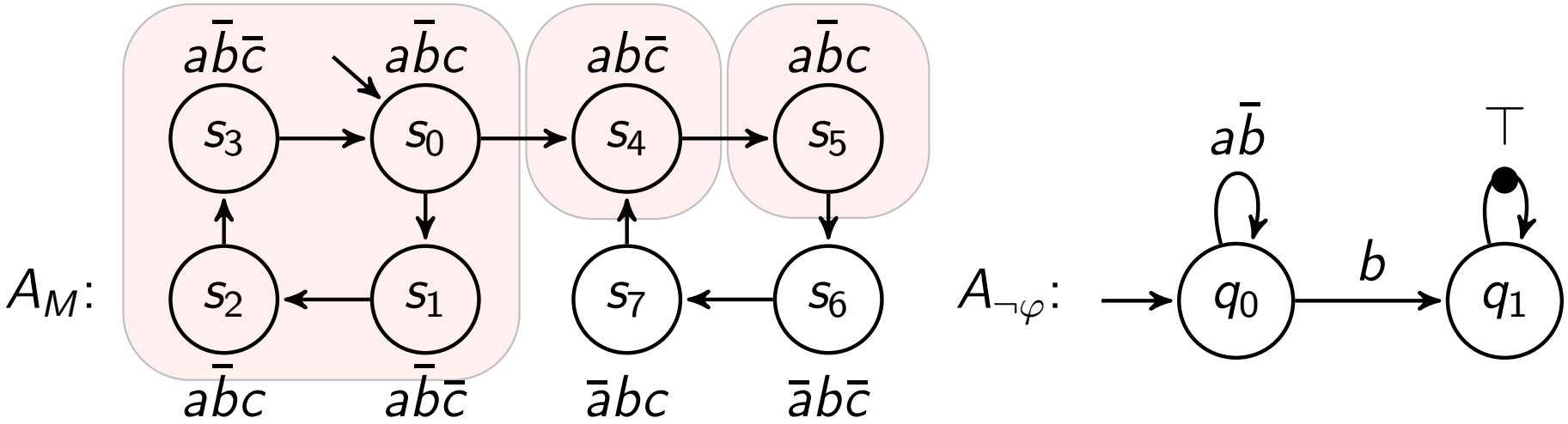
Symbolic Observation Graph (SOG)



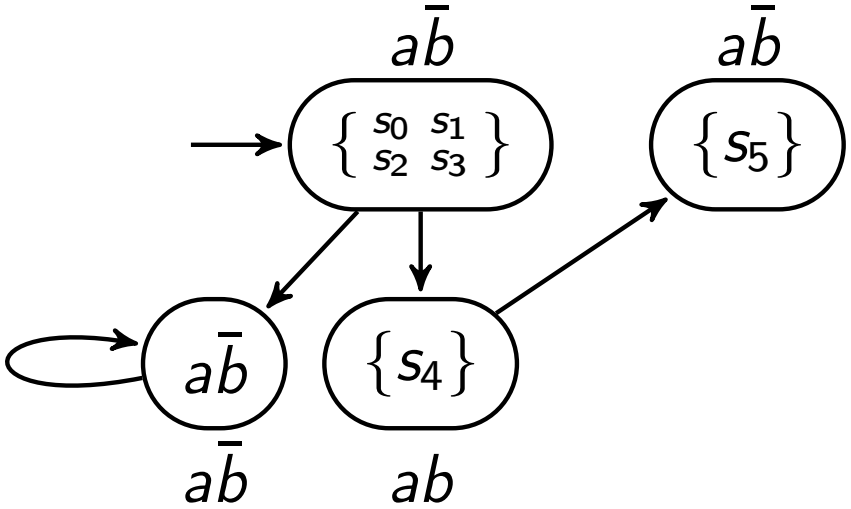
Symbolic Observation **Graph**:



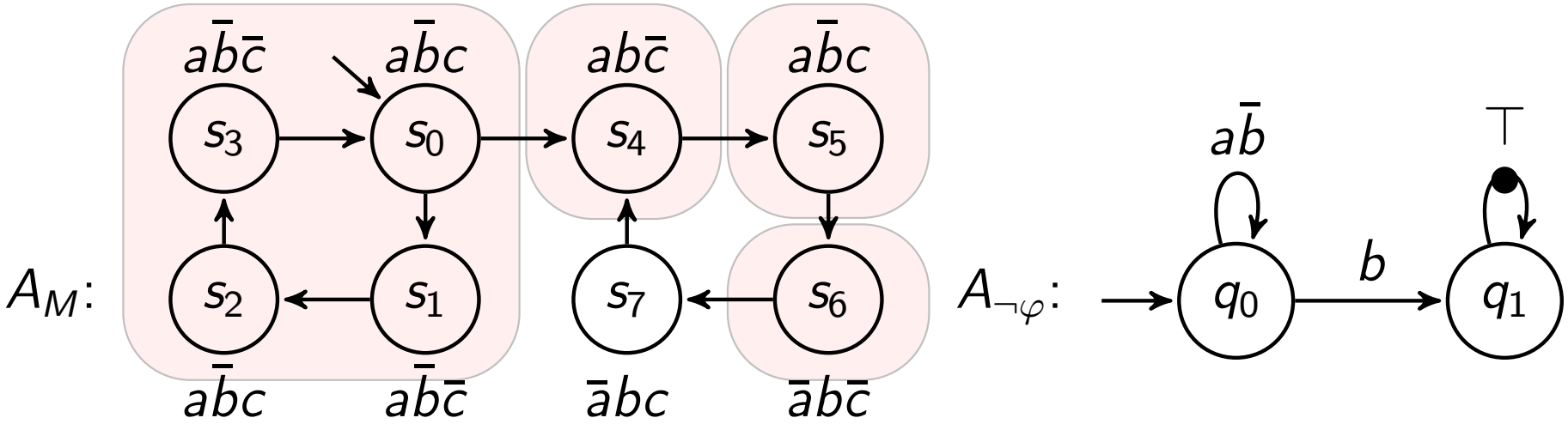
Symbolic Observation Graph (SOG)



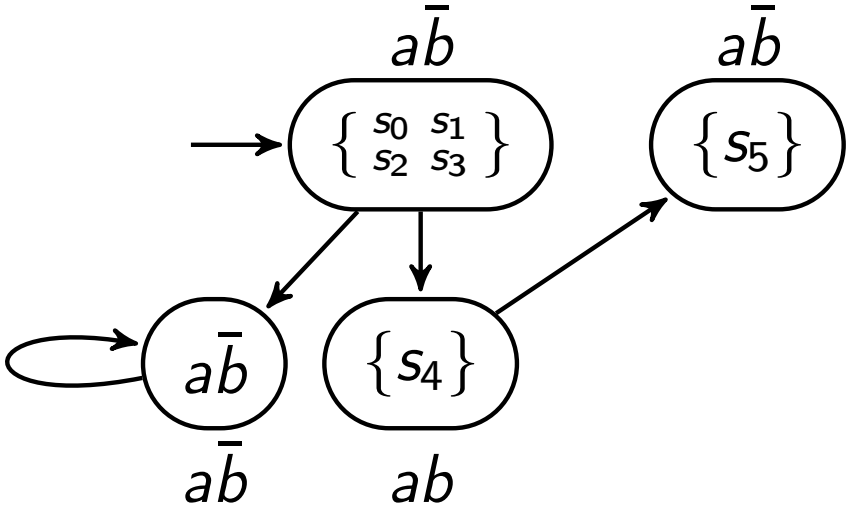
Symbolic Observation **Graph**:



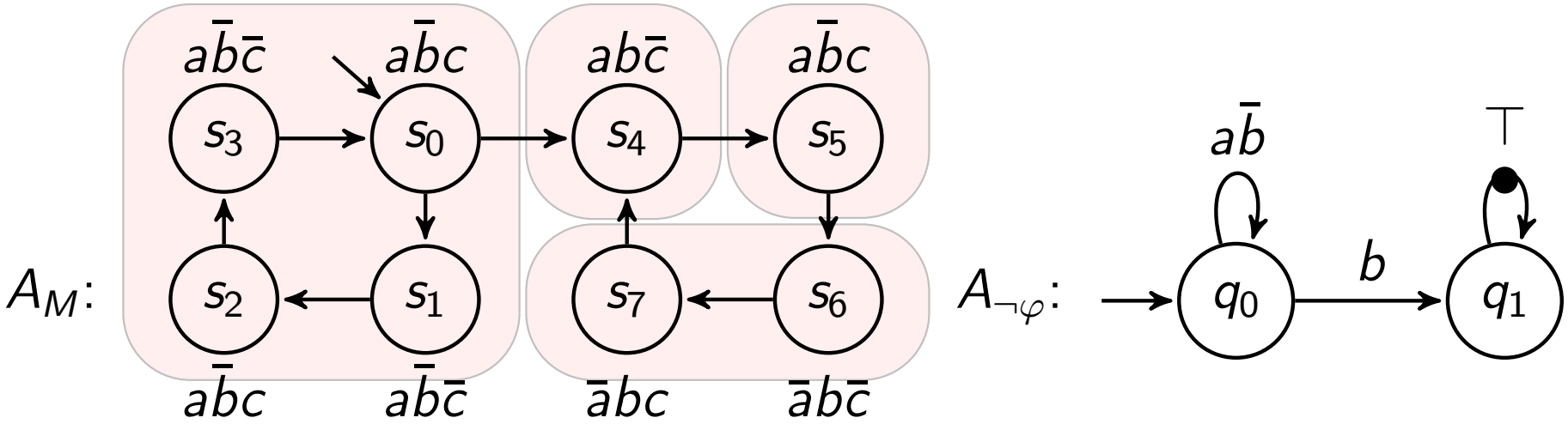
Symbolic Observation Graph (SOG)



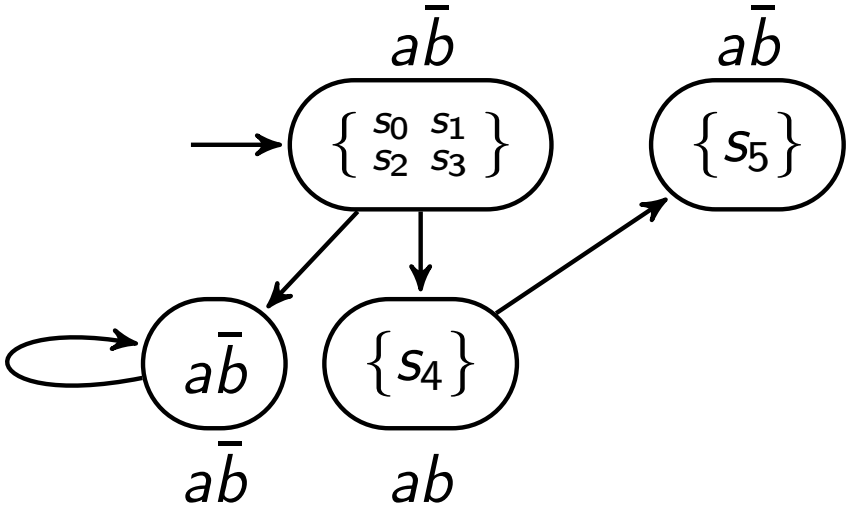
Symbolic Observation **Graph**:



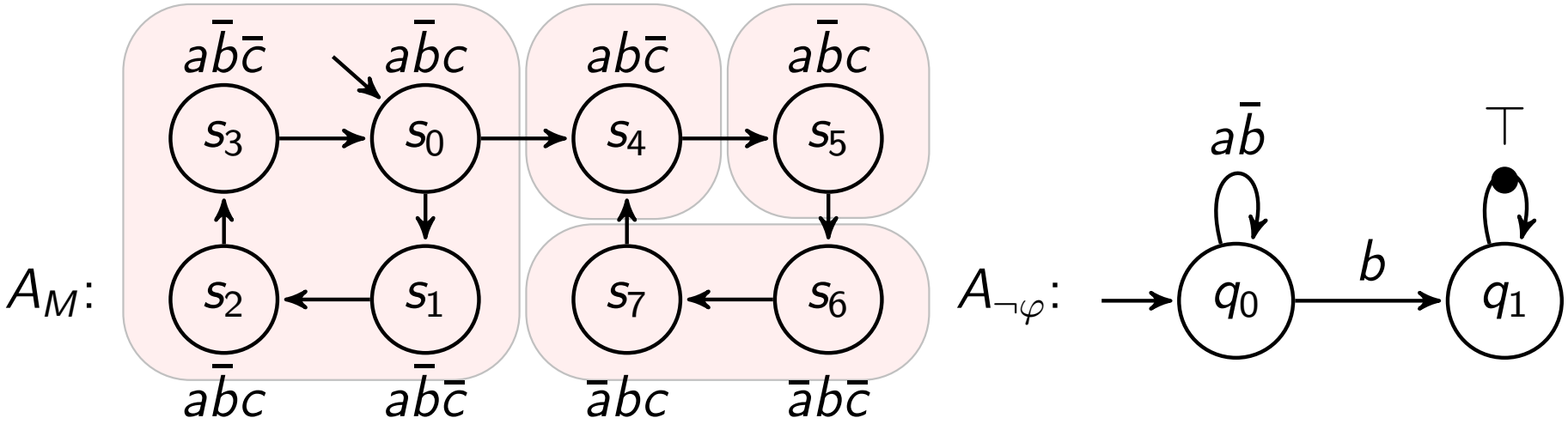
Symbolic Observation Graph (SOG)



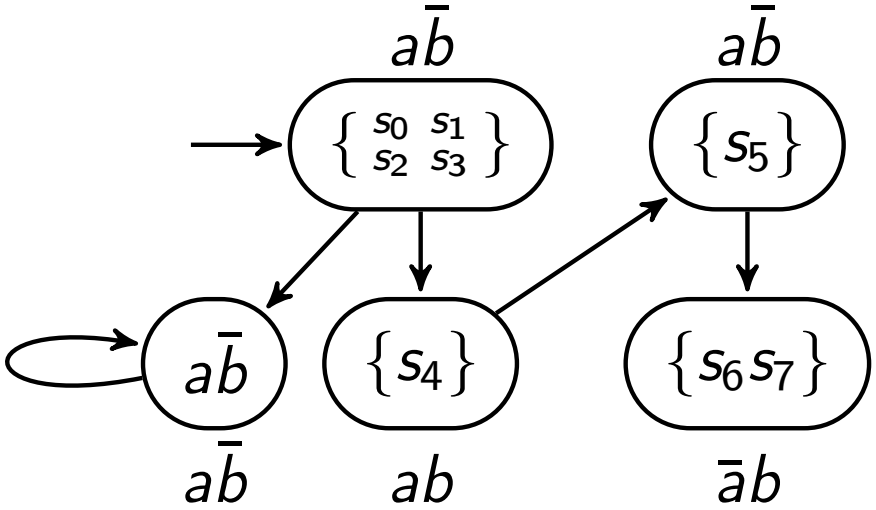
Symbolic Observation **Graph**:



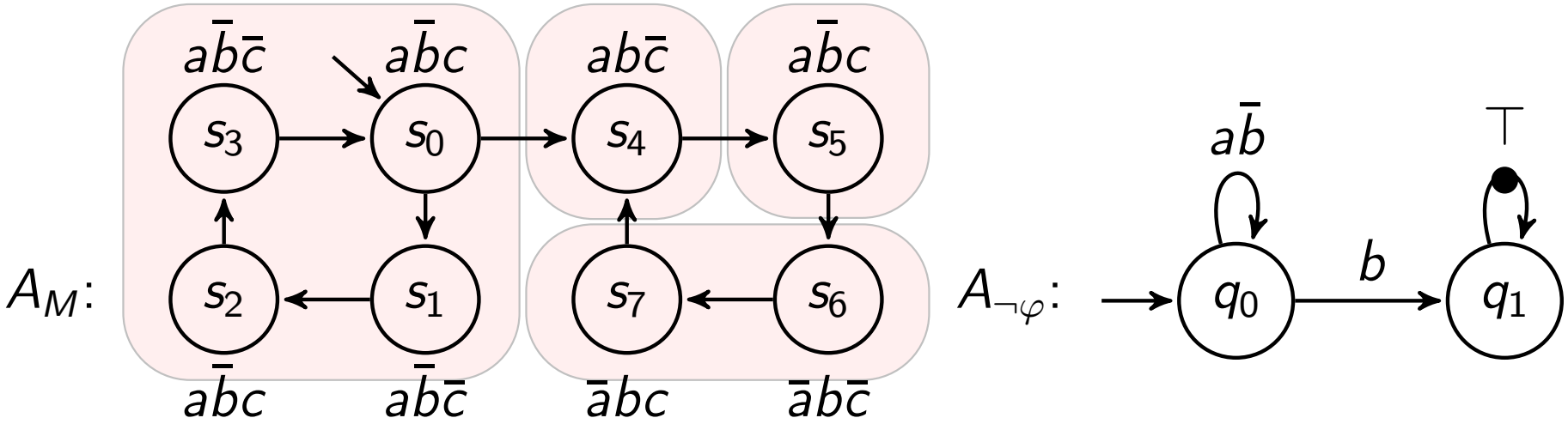
Symbolic Observation Graph (SOG)



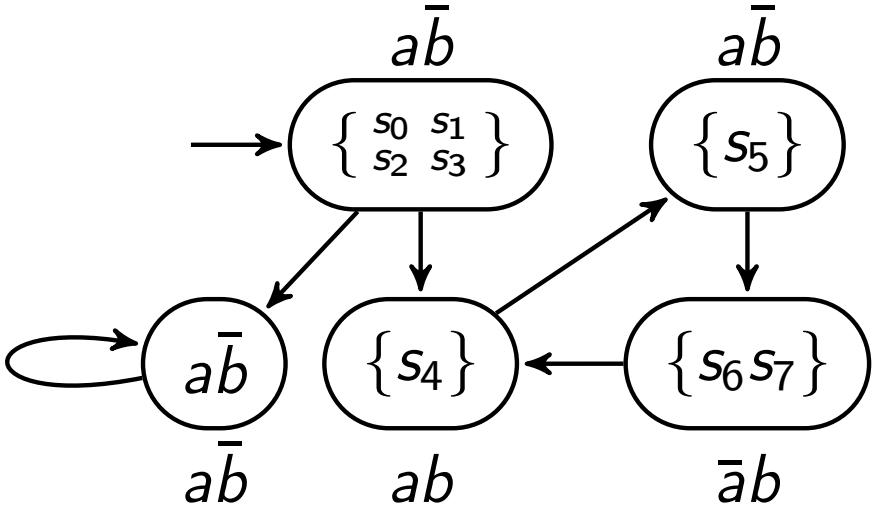
Symbolic Observation **Graph**:



Symbolic Observation Graph (SOG)

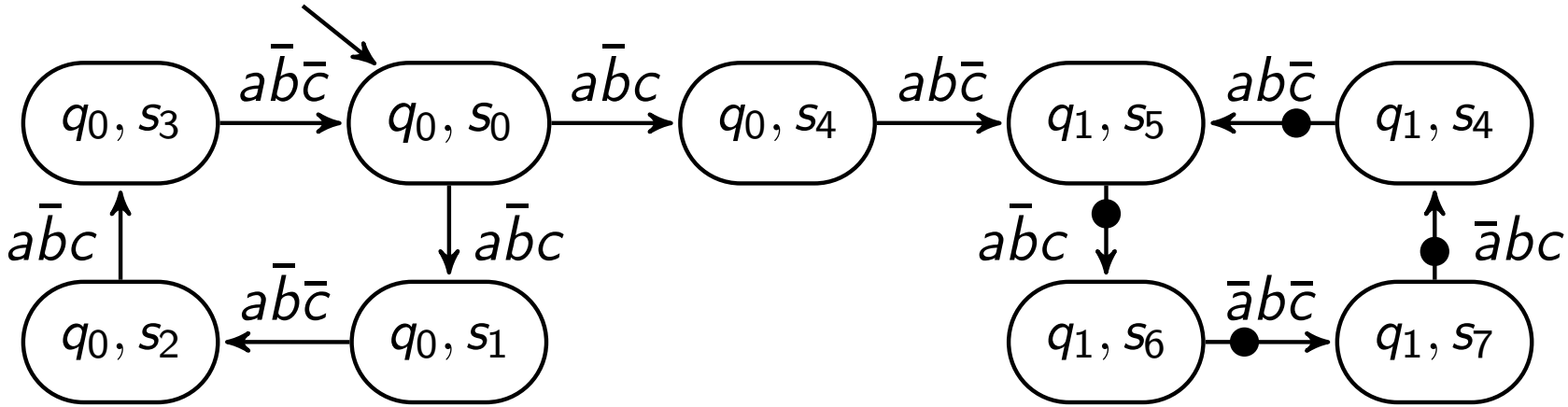


Symbolic Observation **Graph**:

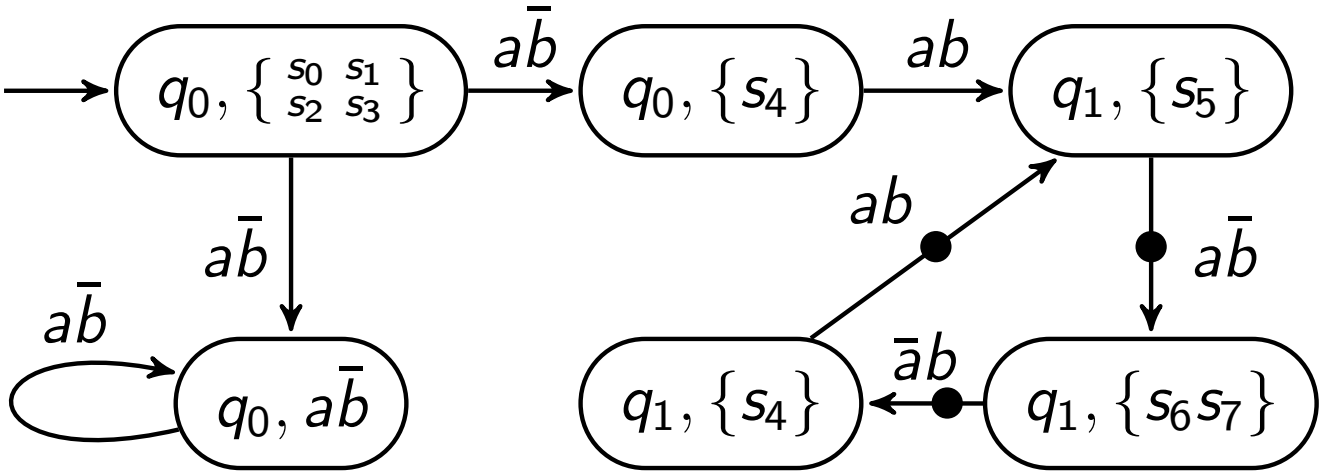


Product Sizes: Kripke vs. SOG

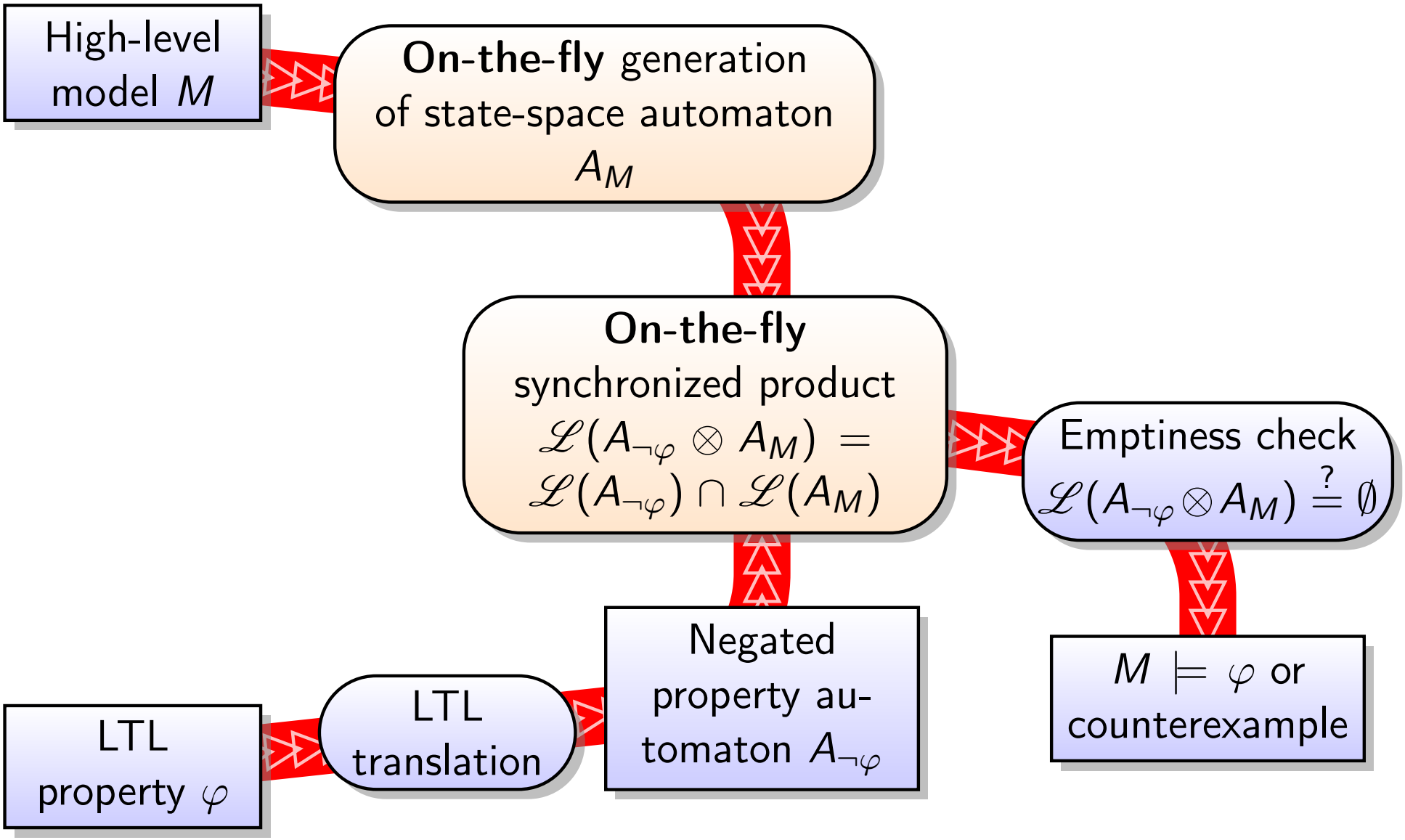
$A_{\neg\varphi} \otimes A_M$:



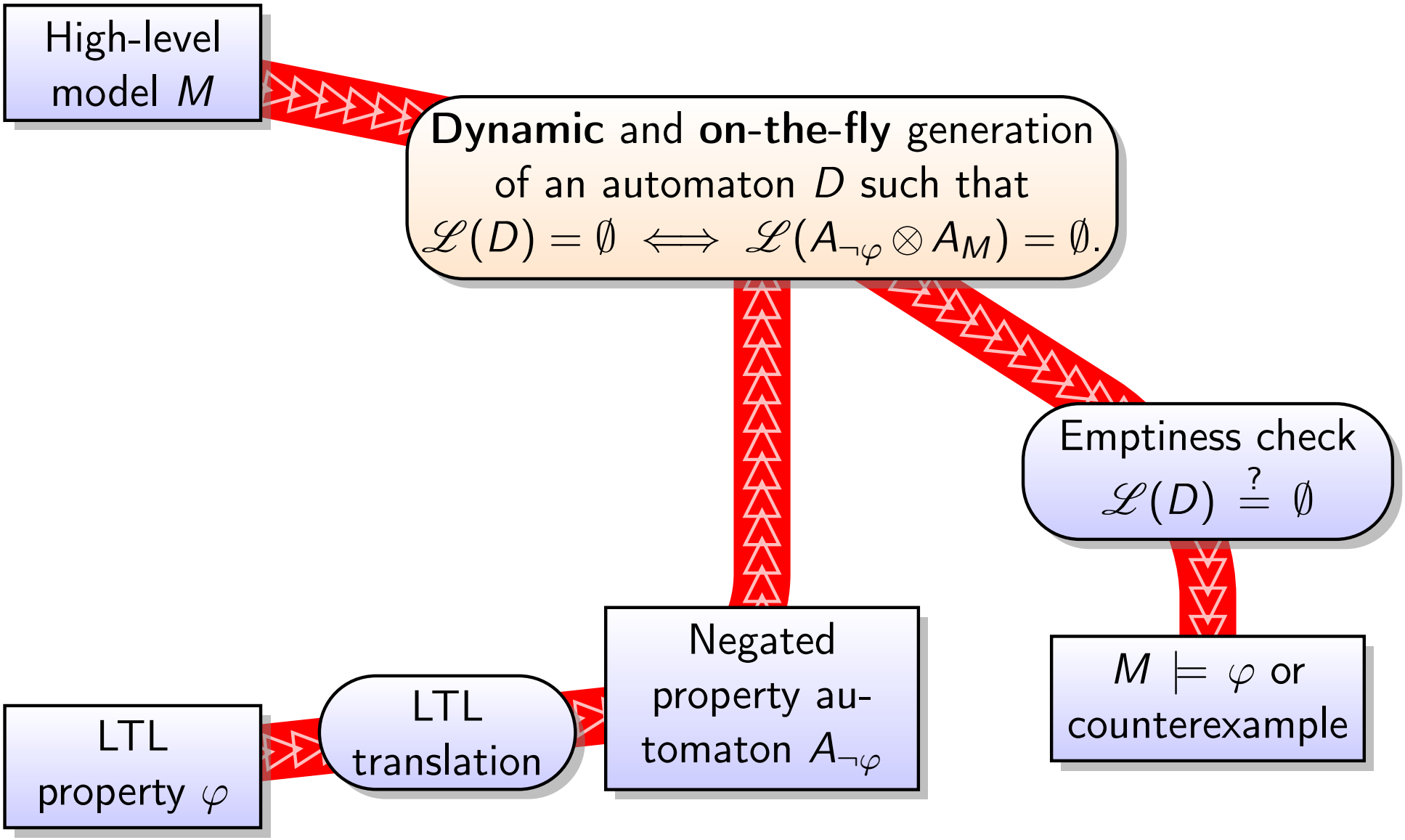
$A_{\neg\varphi} \otimes$ SOG:



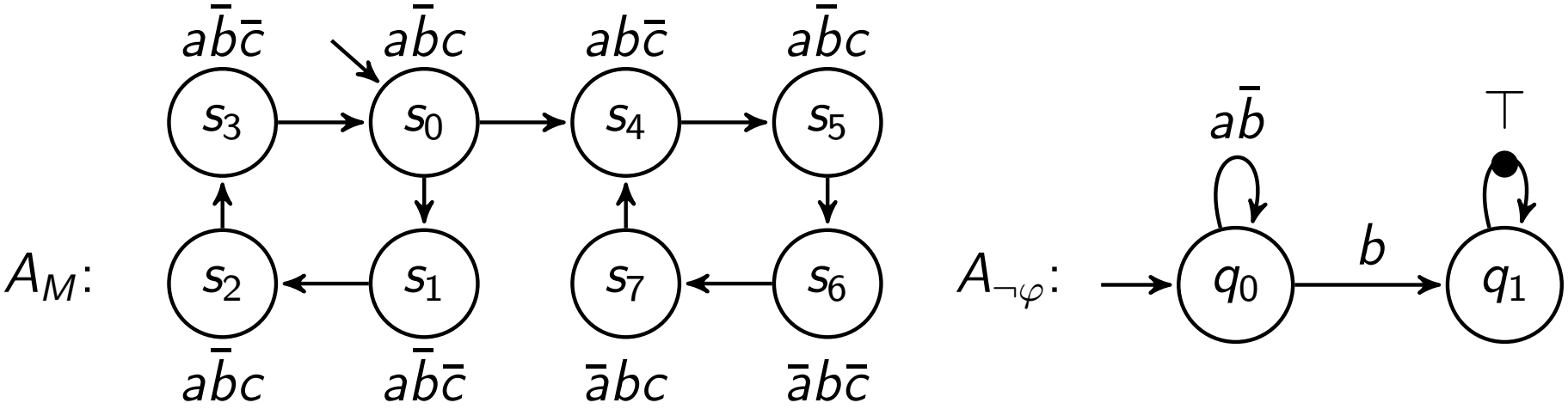
Building a Product Directly



Building a Product Directly



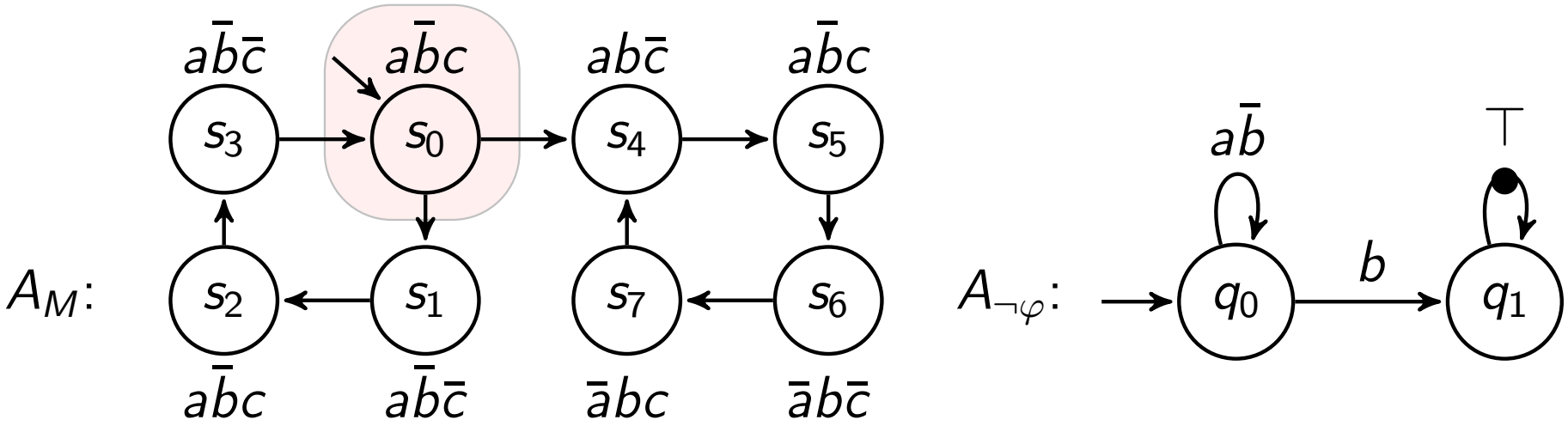
Symbolic Observation Product (SOP)



Symbolic Observation **Product**:

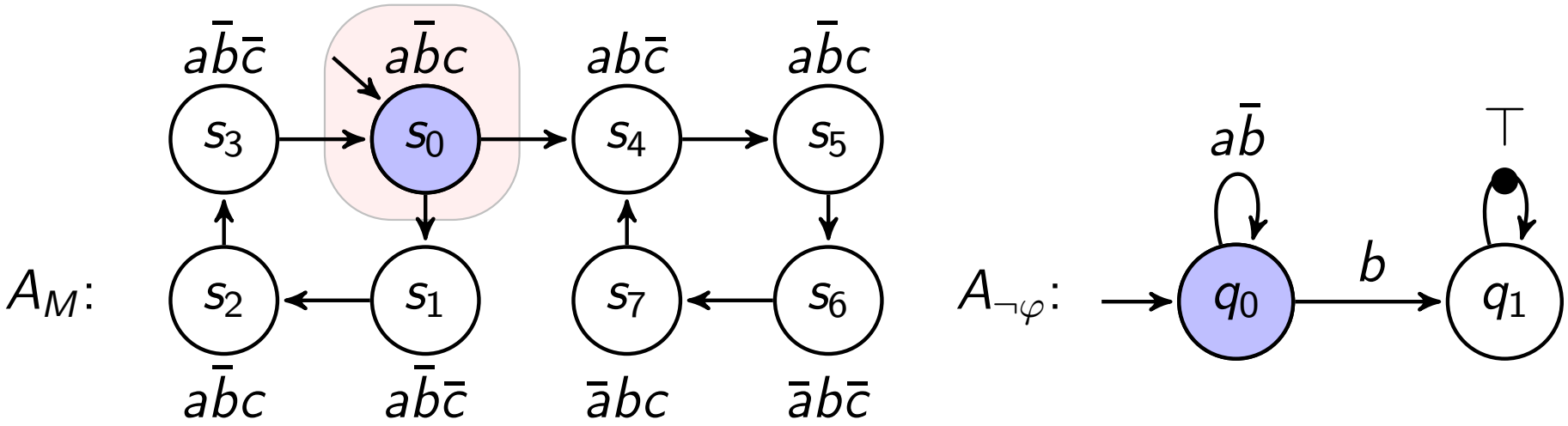
- A dynamic extension of the SOG
- Gather all states whose valuation do not change the propositions observed by the property automaton **at the current point**.

Symbolic Observation Product (SOP)



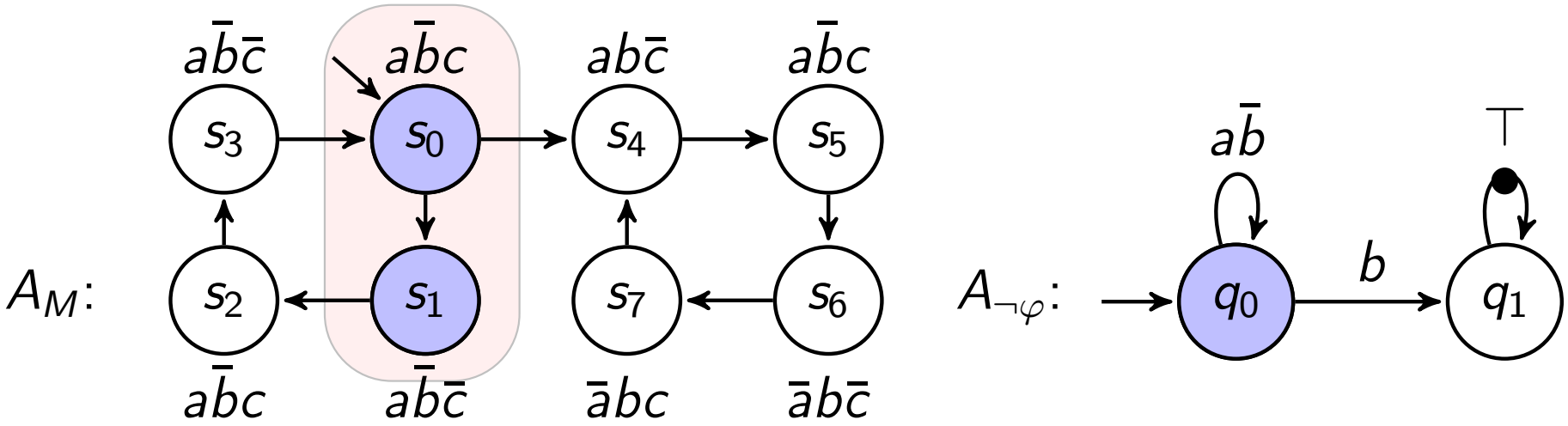
Symbolic Observation **Product**:

Symbolic Observation Product (SOP)



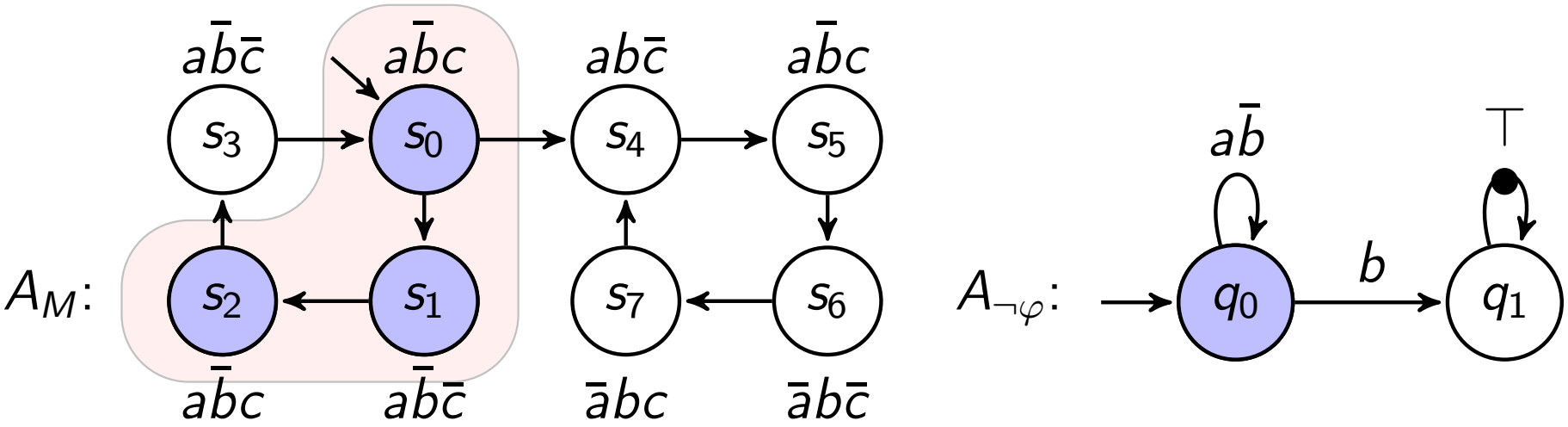
Symbolic Observation **Product**:

Symbolic Observation Product (SOP)



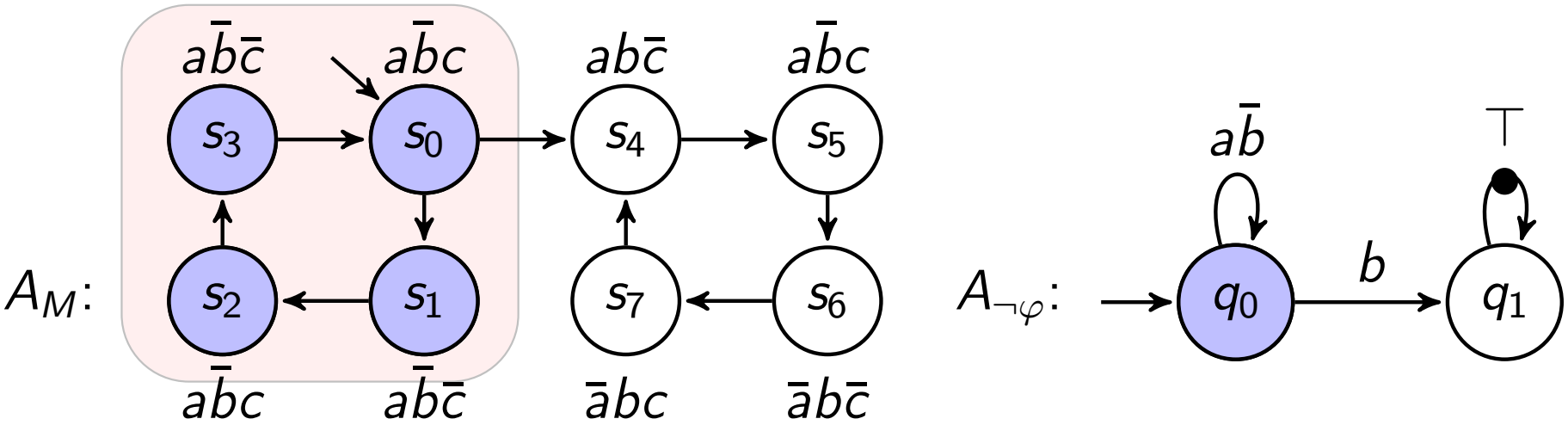
Symbolic Observation **Product**:

Symbolic Observation Product (SOP)



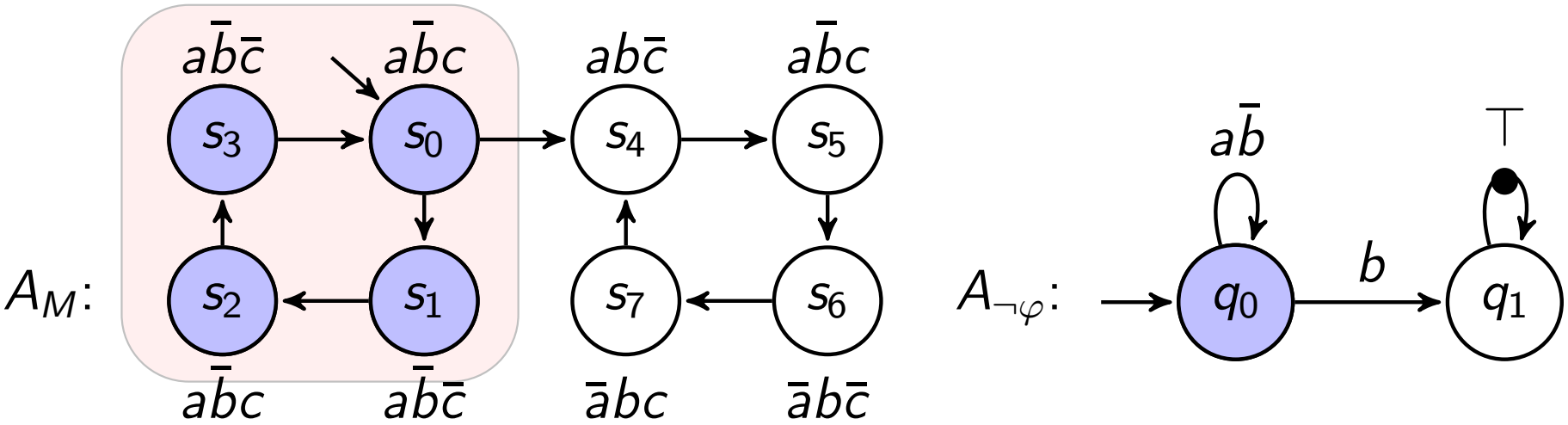
Symbolic Observation **Product**:

Symbolic Observation Product (SOP)

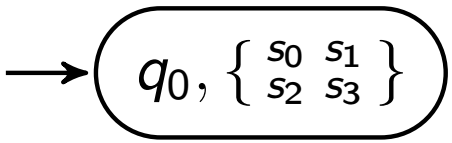


Symbolic Observation **Product**:

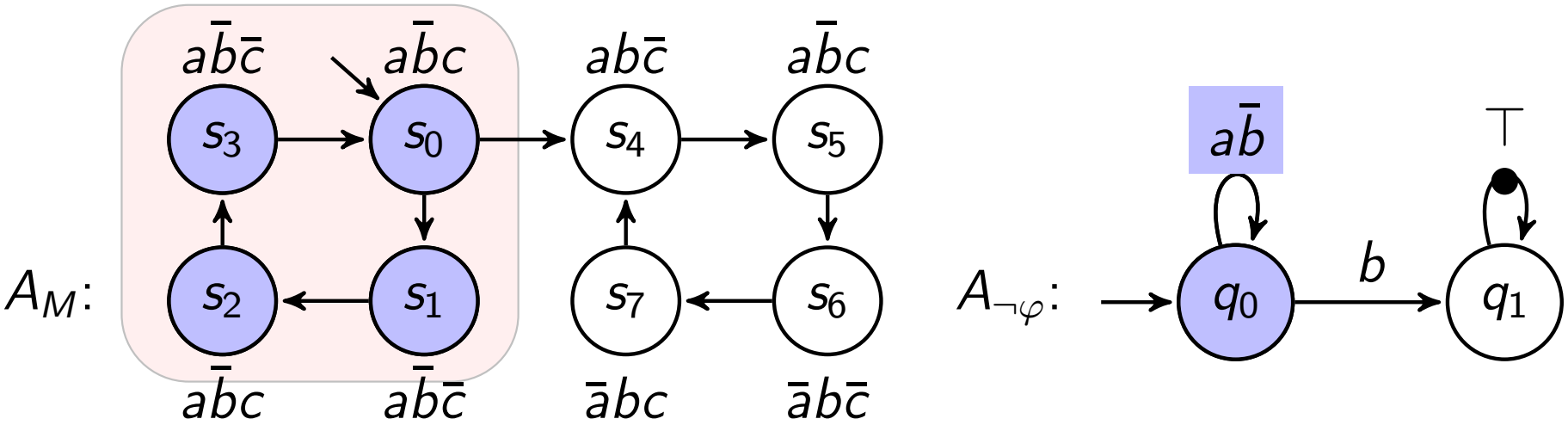
Symbolic Observation Product (SOP)



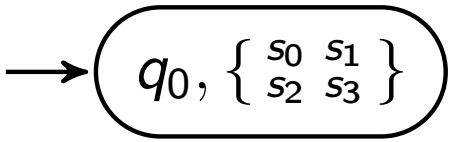
Symbolic Observation **Product**:



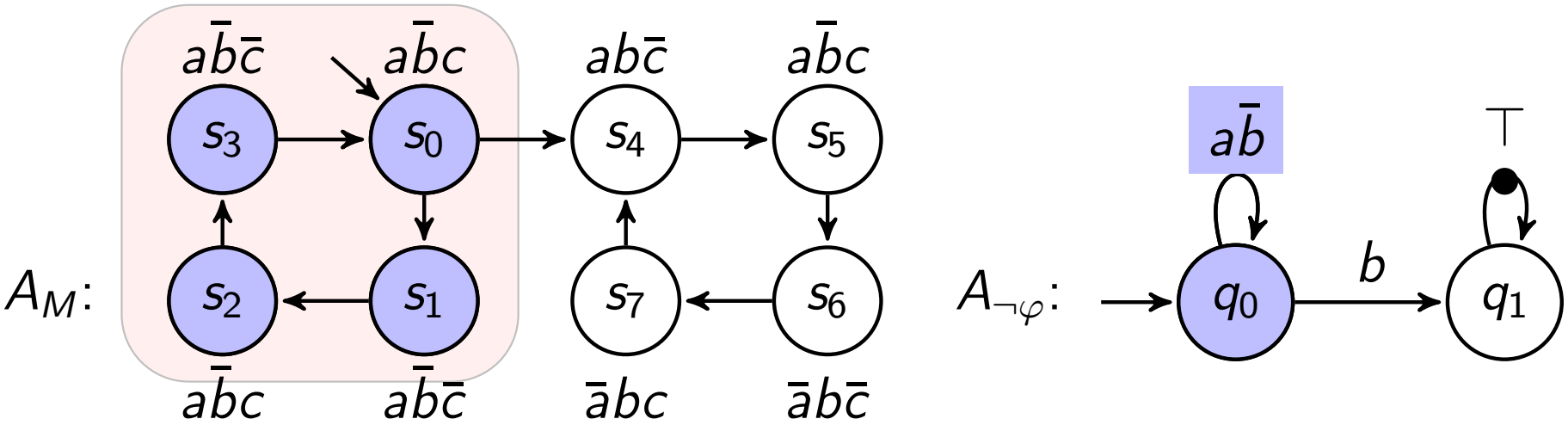
Symbolic Observation Product (SOP)



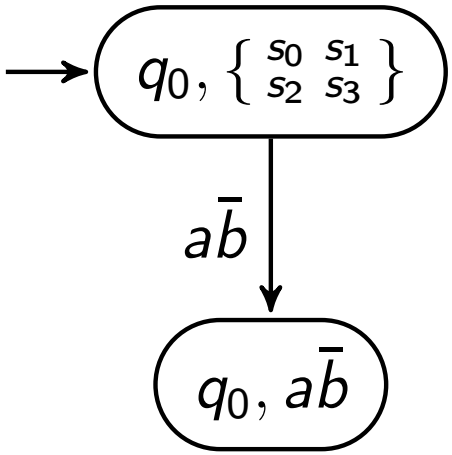
Symbolic Observation **Product**:



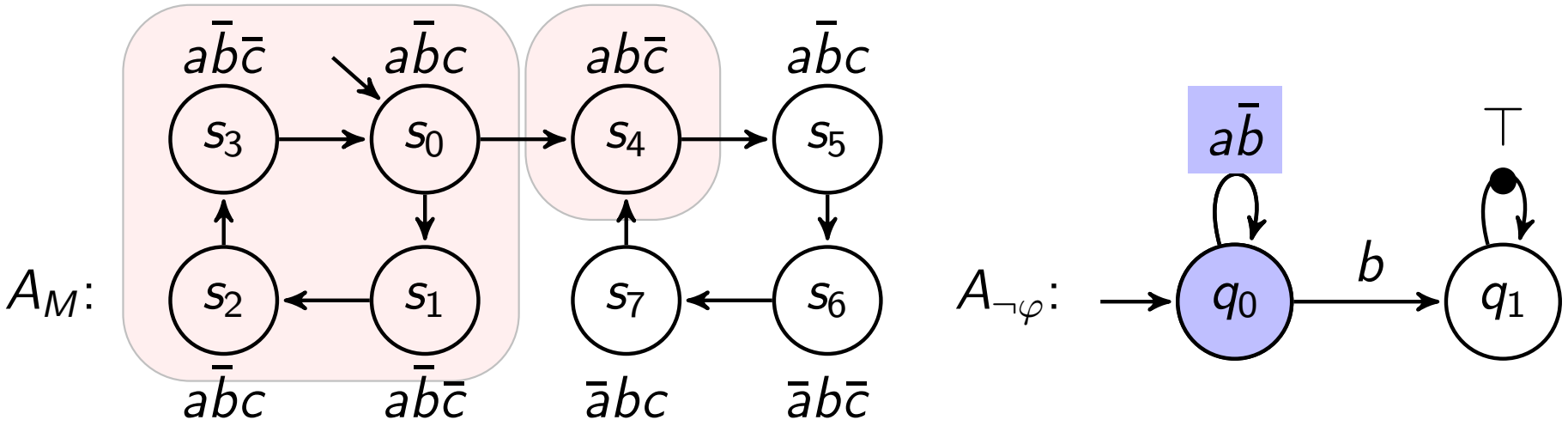
Symbolic Observation Product (SOP)



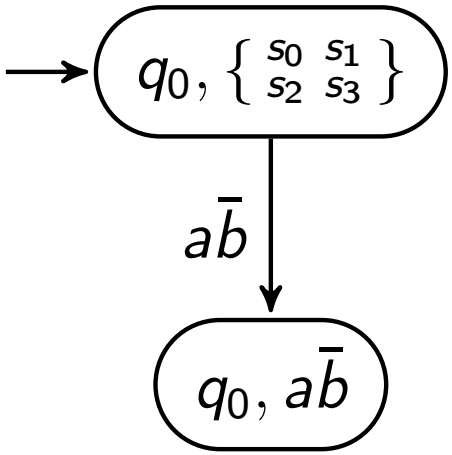
Symbolic Observation **Product**:



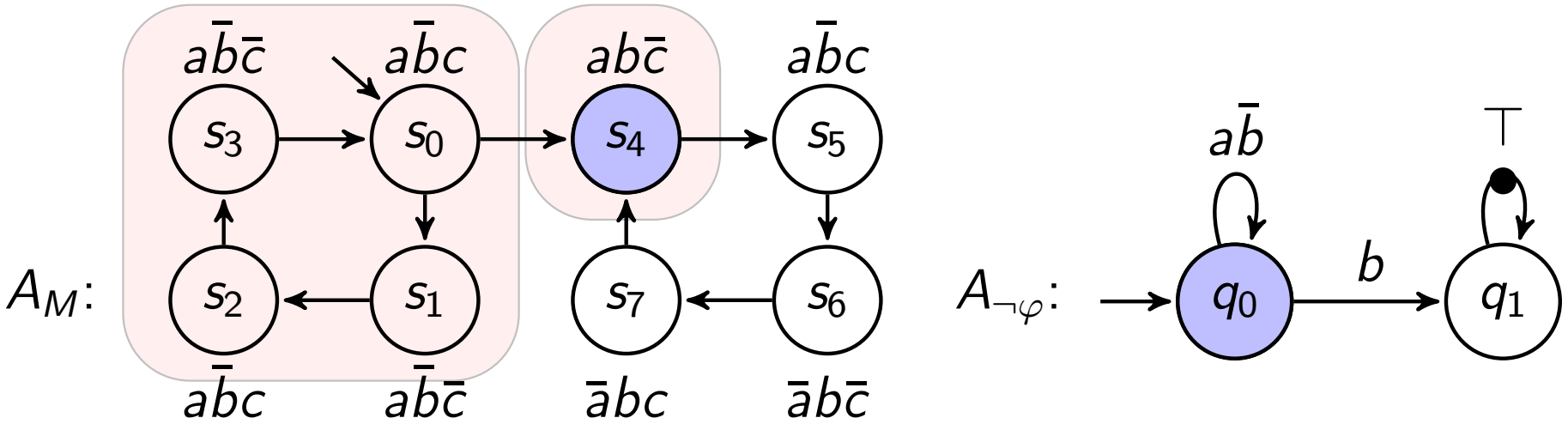
Symbolic Observation Product (SOP)



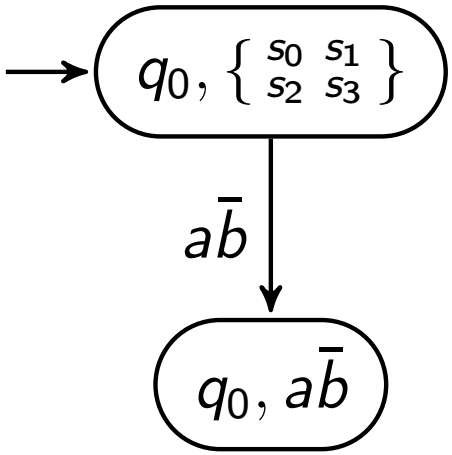
Symbolic Observation **Product**:



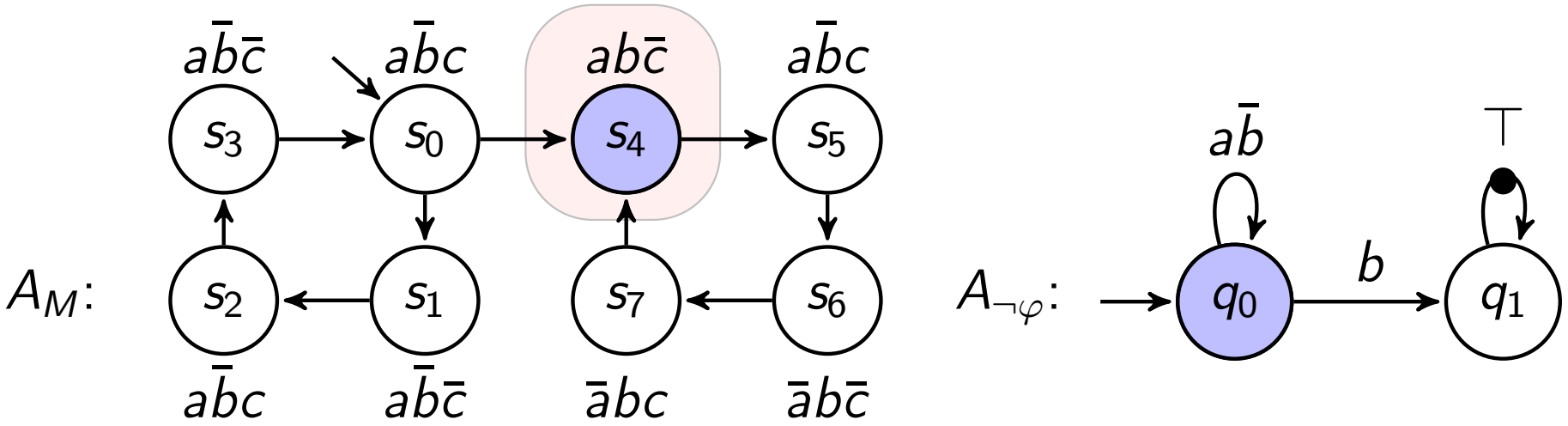
Symbolic Observation Product (SOP)



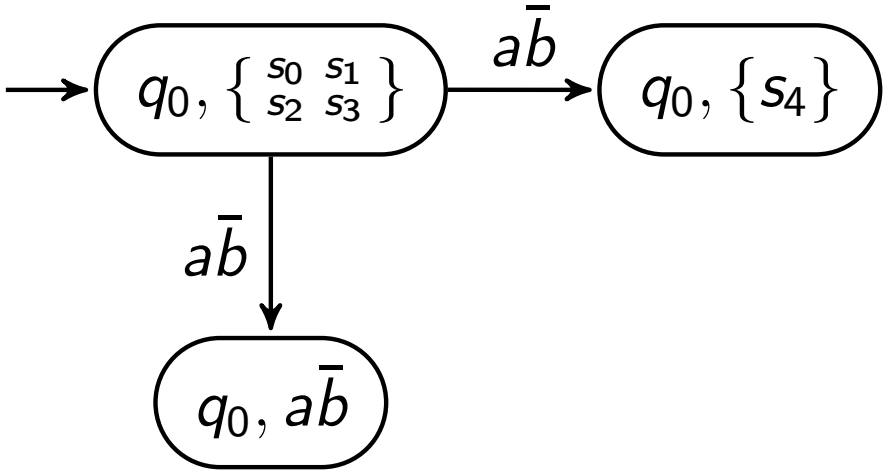
Symbolic Observation **Product**:



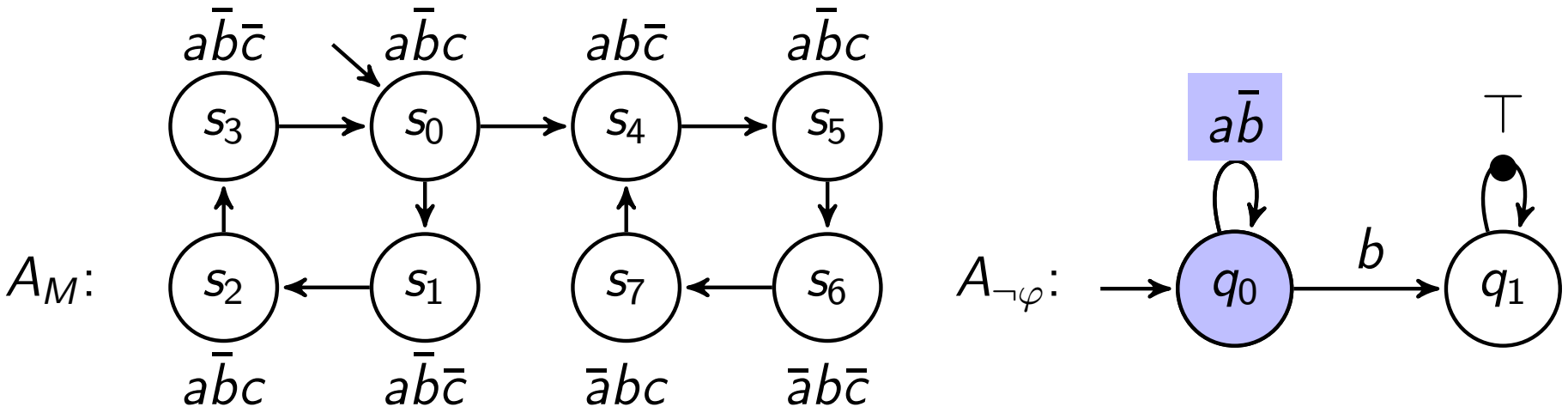
Symbolic Observation Product (SOP)



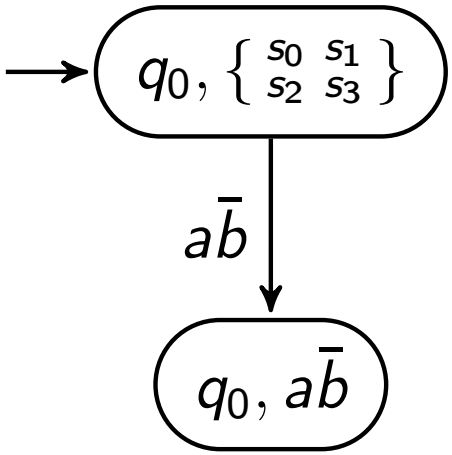
Symbolic Observation **Product**:



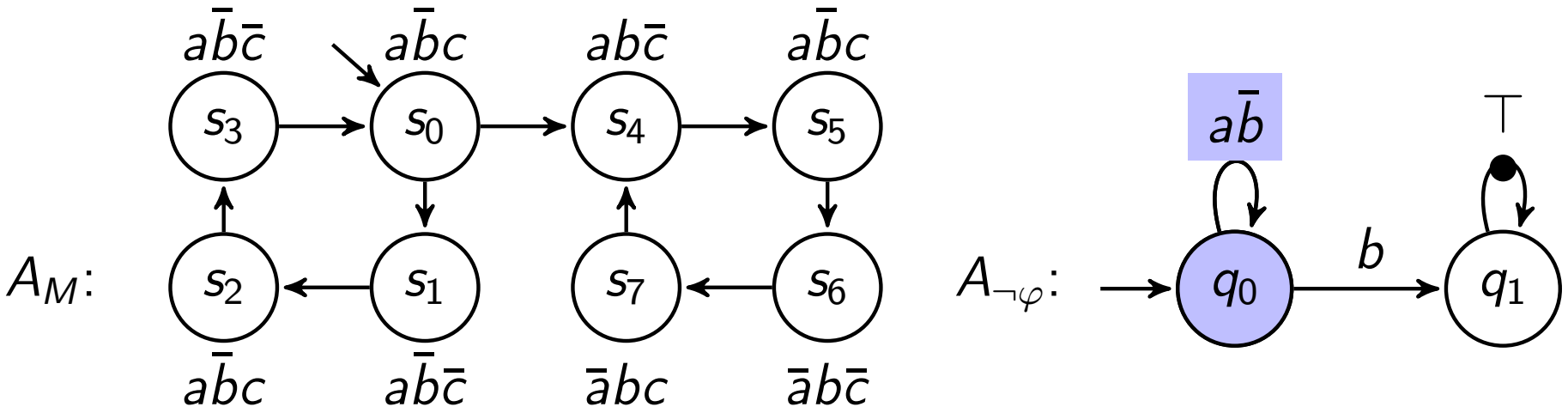
Symbolic Observation Product (SOP)



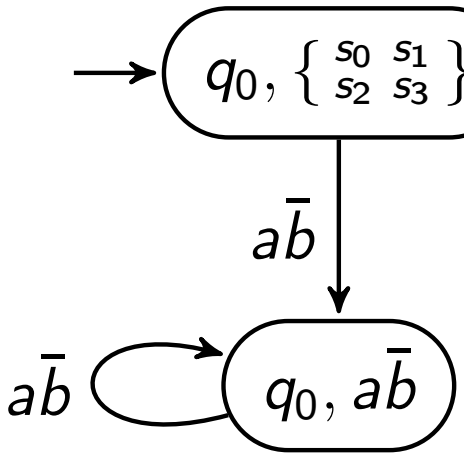
Symbolic Observation **Product**:



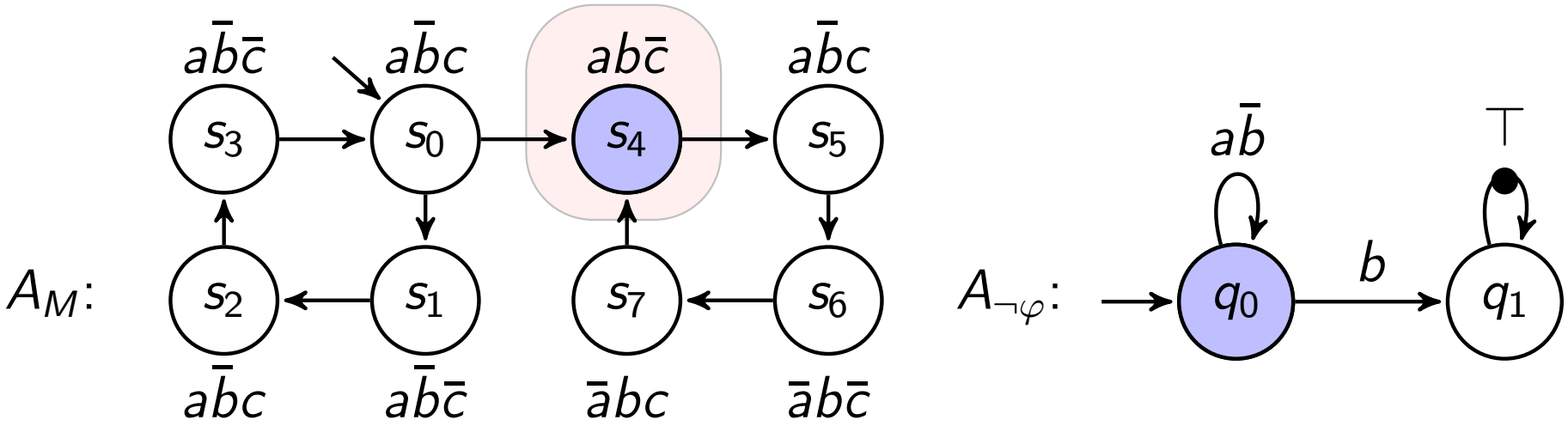
Symbolic Observation Product (SOP)



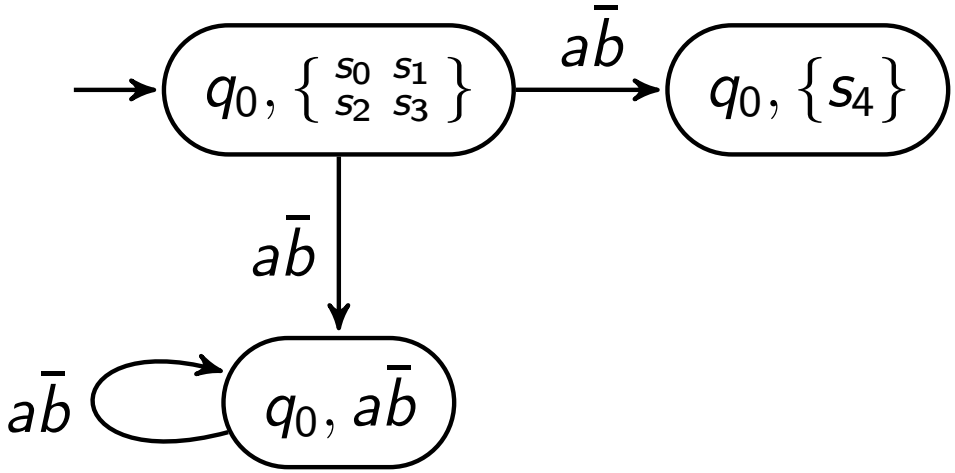
Symbolic Observation **Product**:



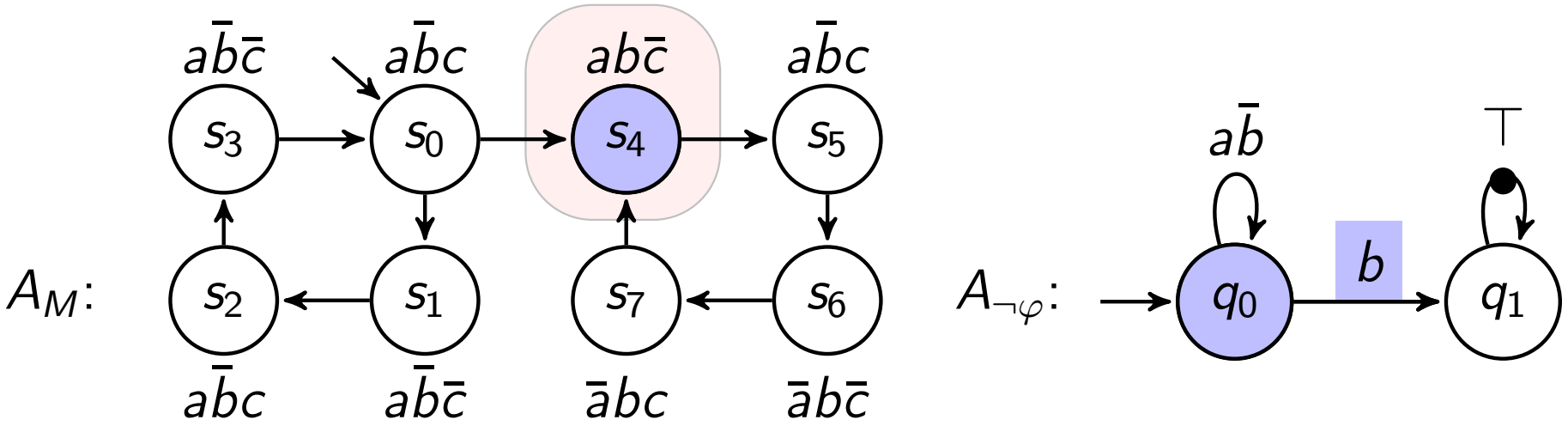
Symbolic Observation Product (SOP)



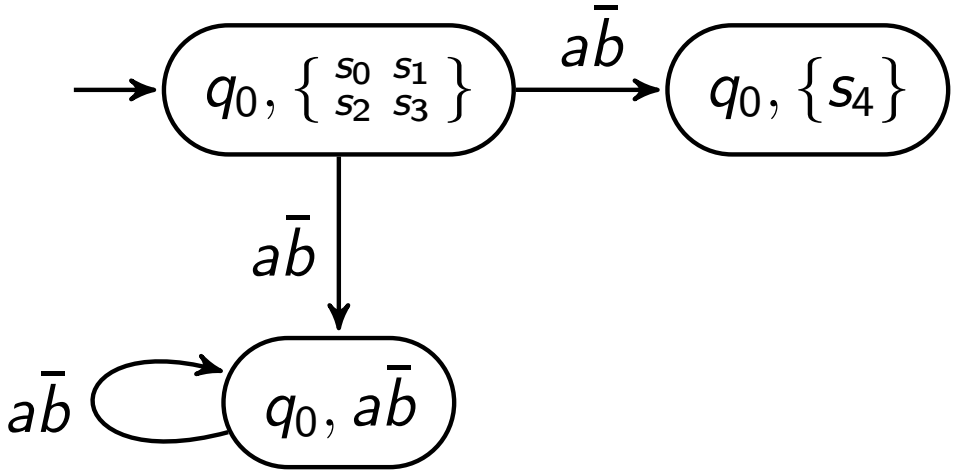
Symbolic Observation **Product**:



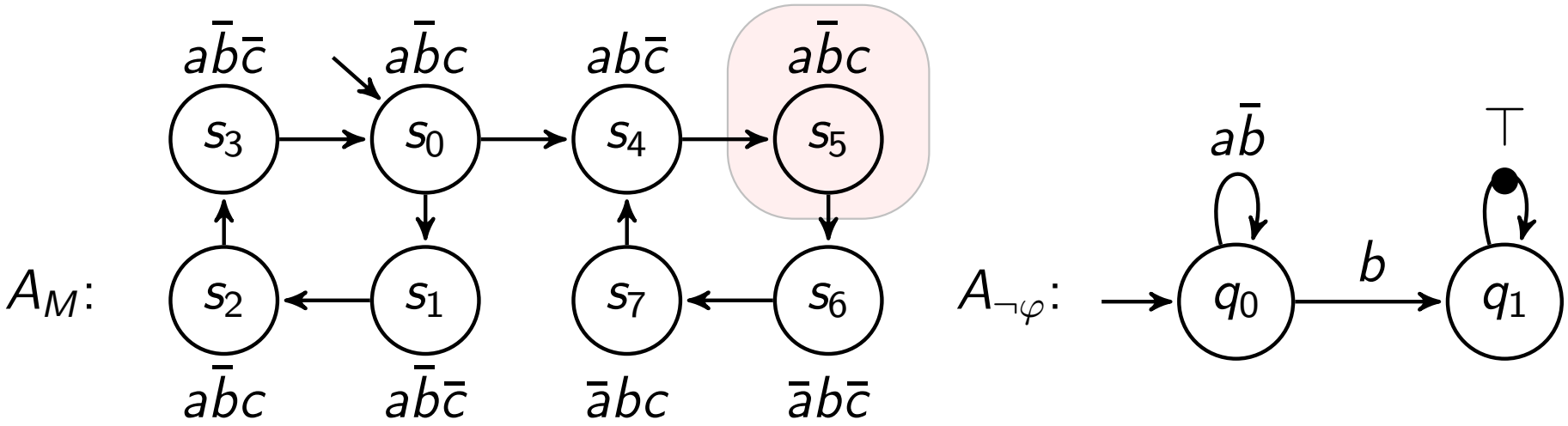
Symbolic Observation Product (SOP)



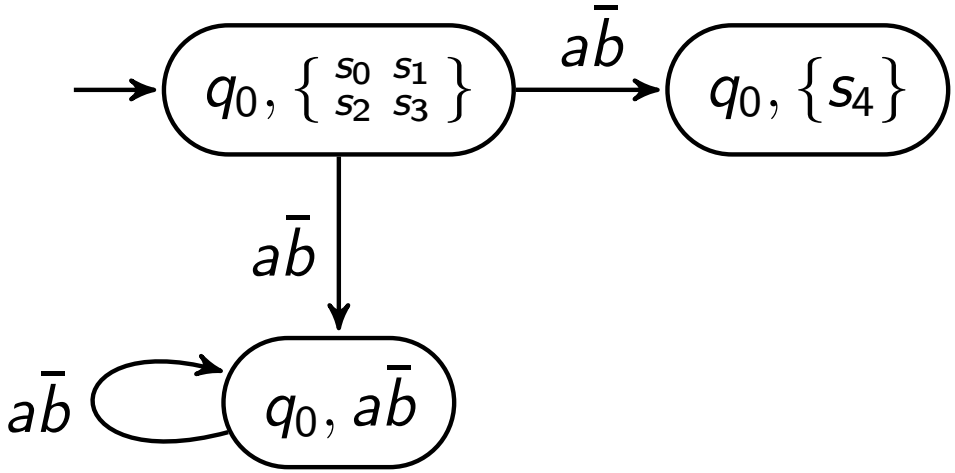
Symbolic Observation **Product**:



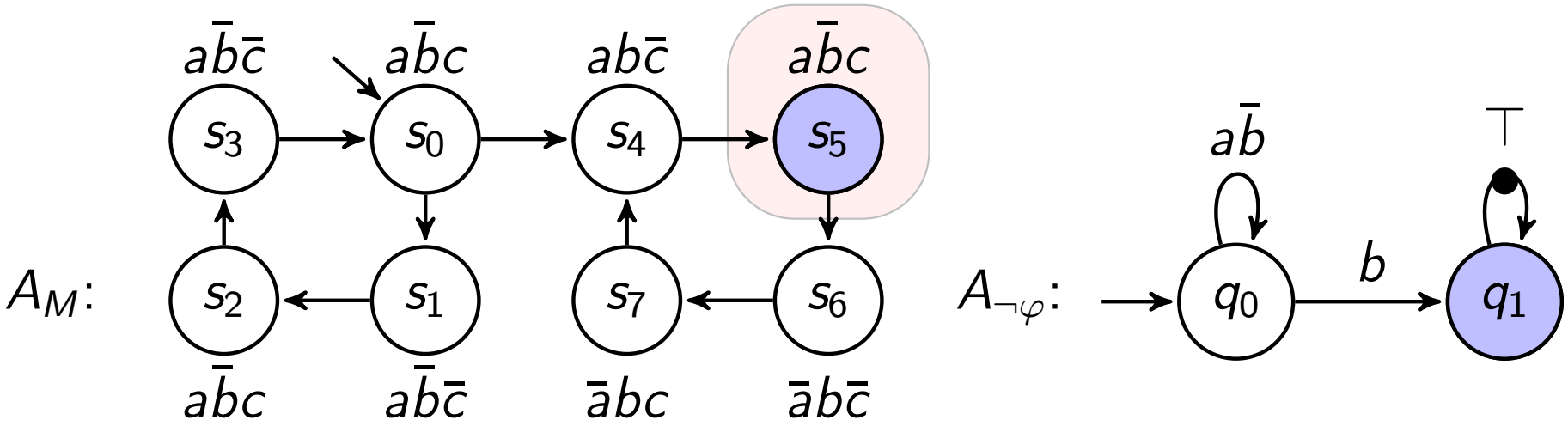
Symbolic Observation Product (SOP)



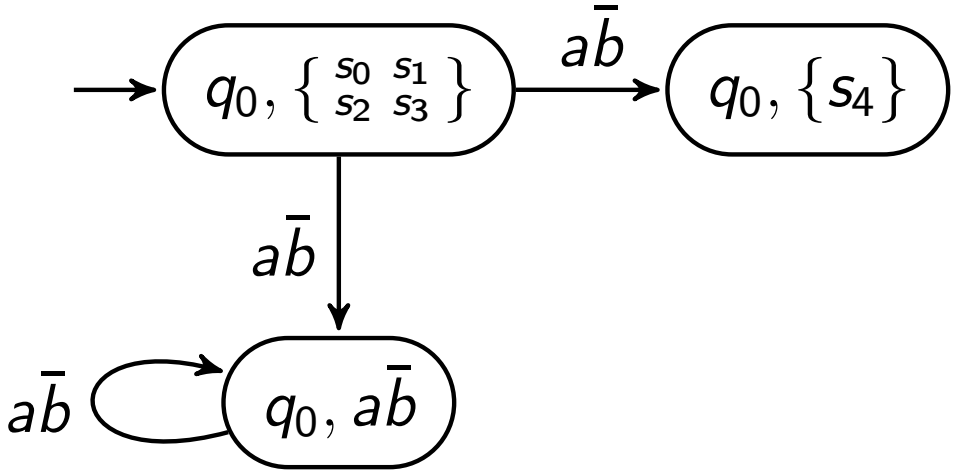
Symbolic Observation **Product**:



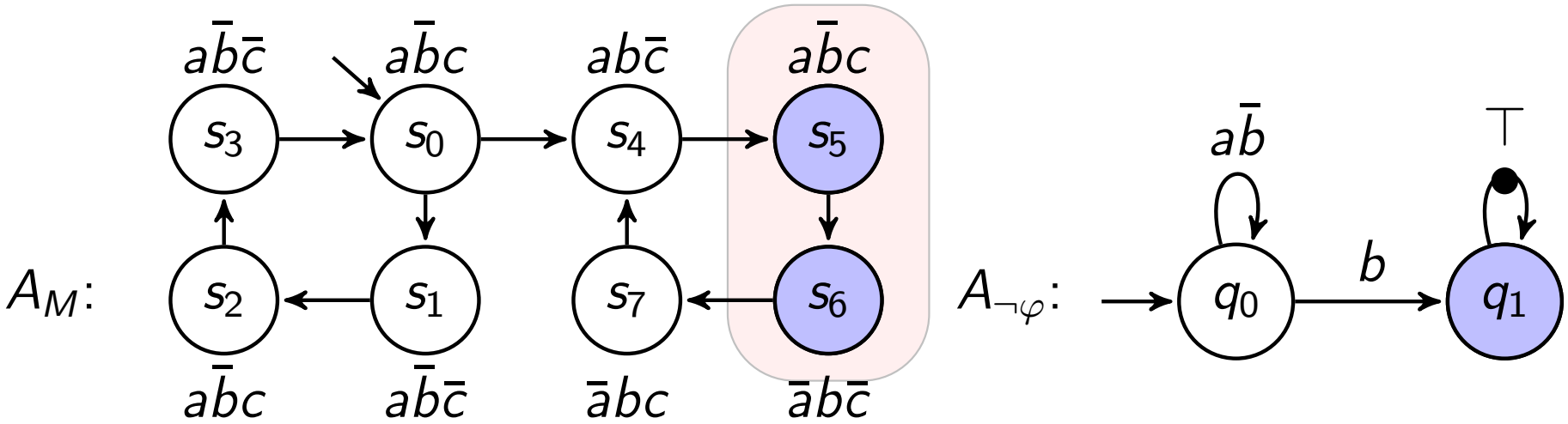
Symbolic Observation Product (SOP)



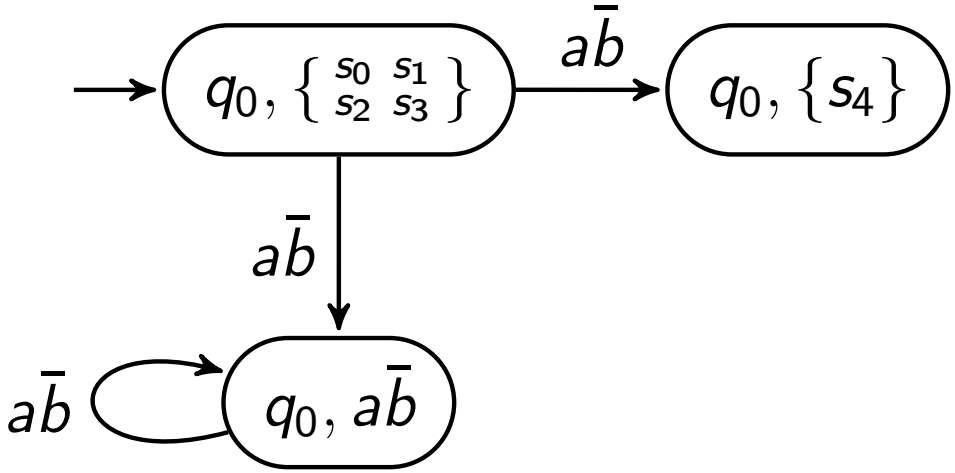
Symbolic Observation **Product**:



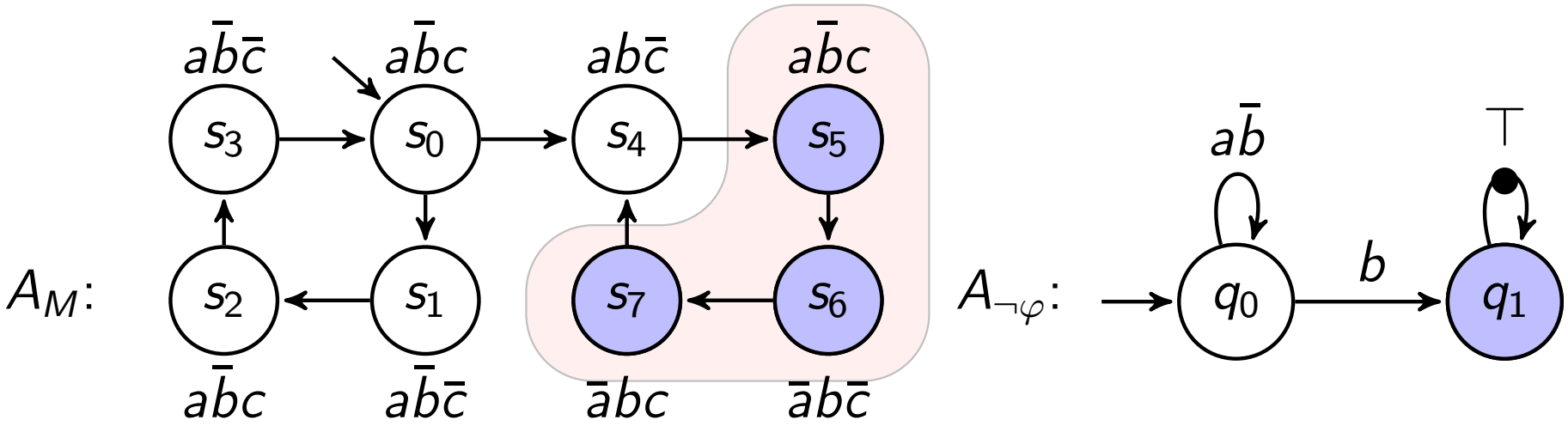
Symbolic Observation Product (SOP)



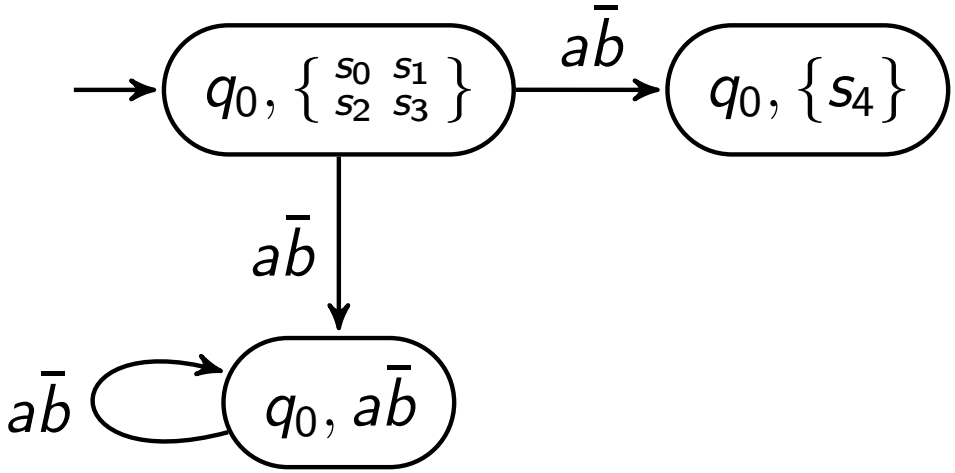
Symbolic Observation **Product**:



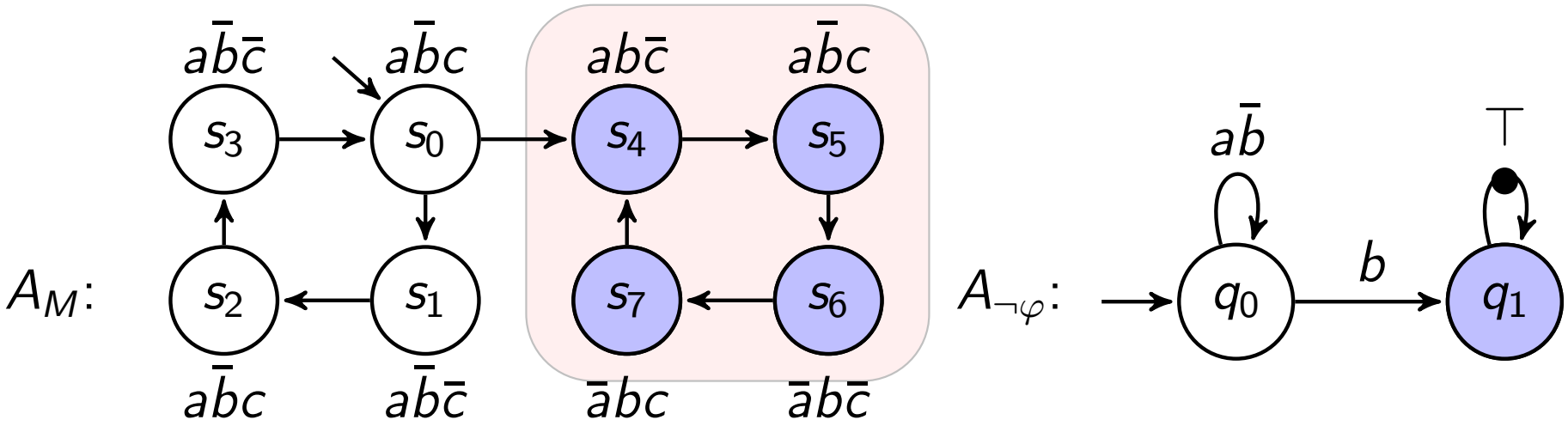
Symbolic Observation Product (SOP)



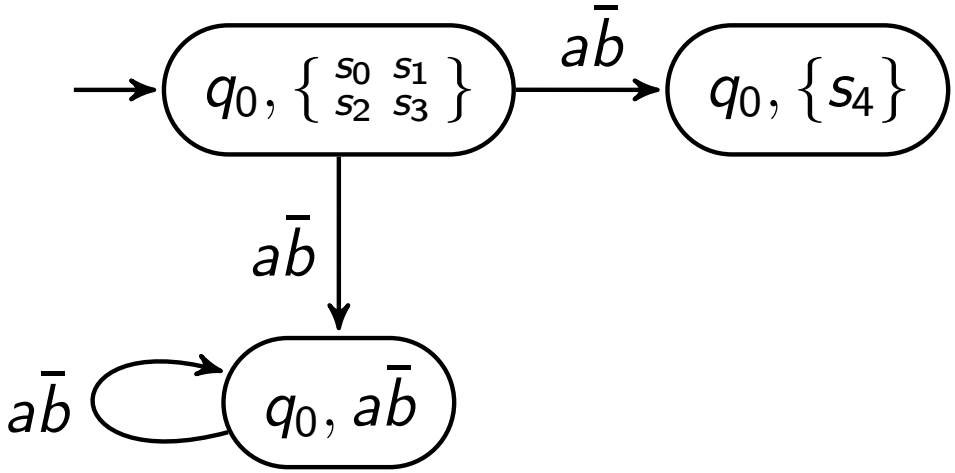
Symbolic Observation **Product**:



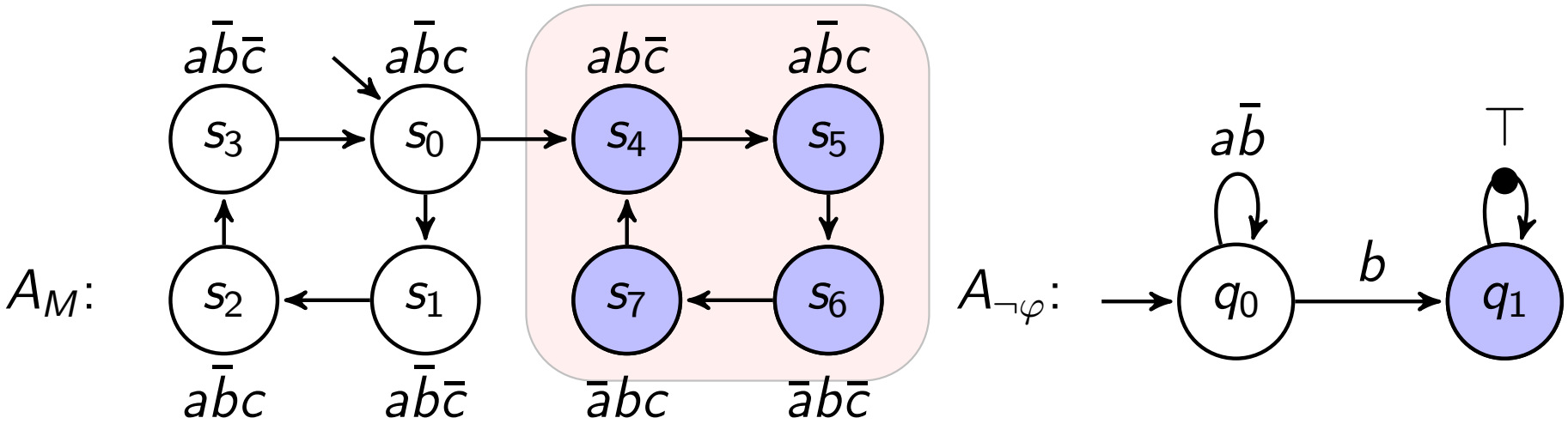
Symbolic Observation Product (SOP)



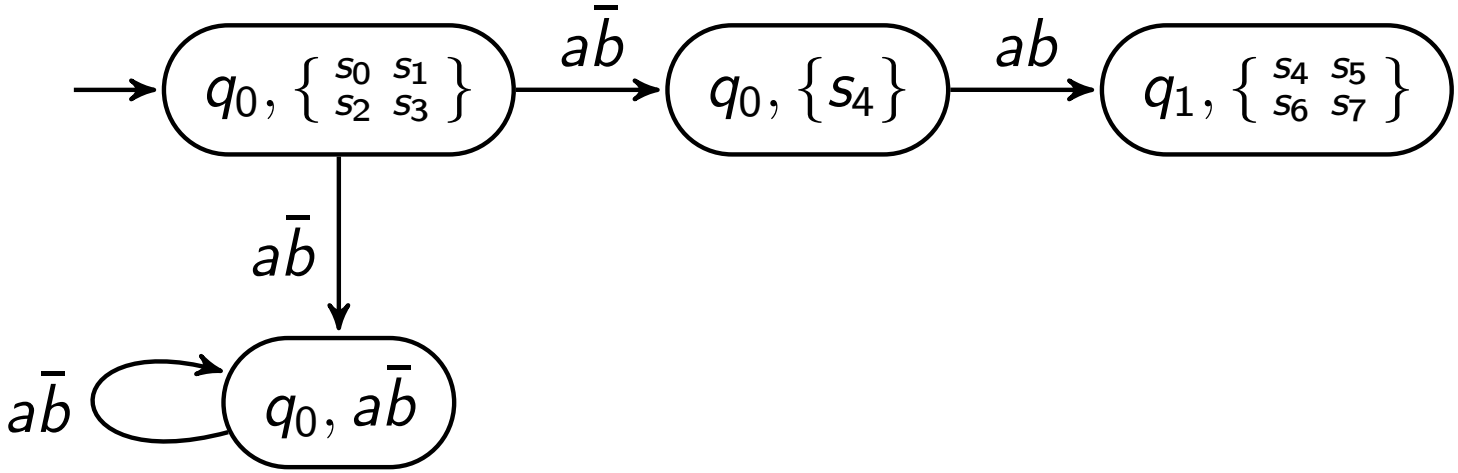
Symbolic Observation **Product**:



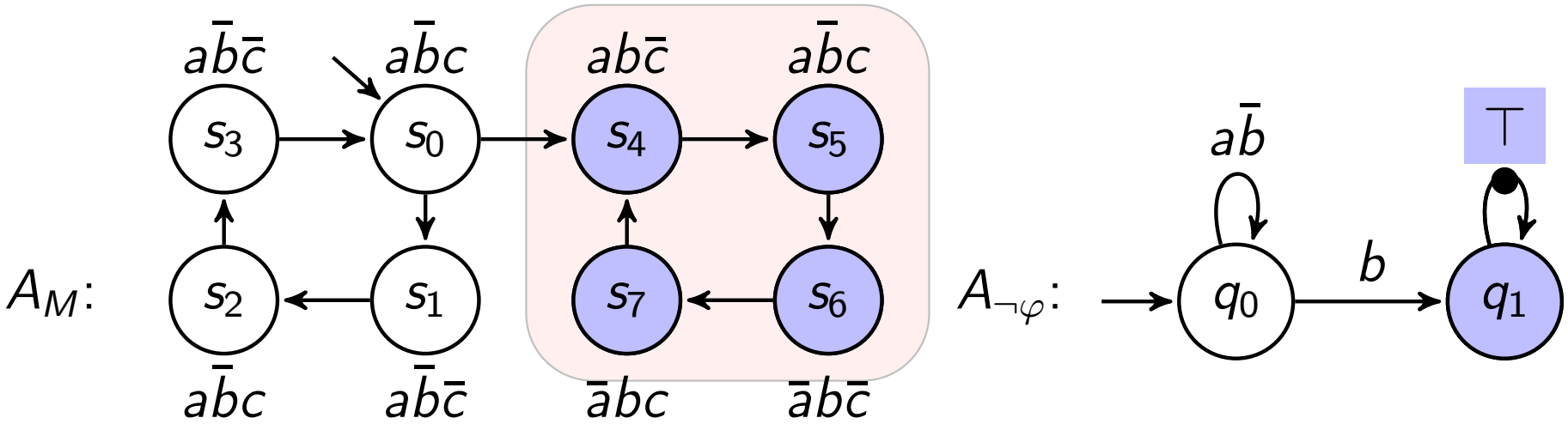
Symbolic Observation Product (SOP)



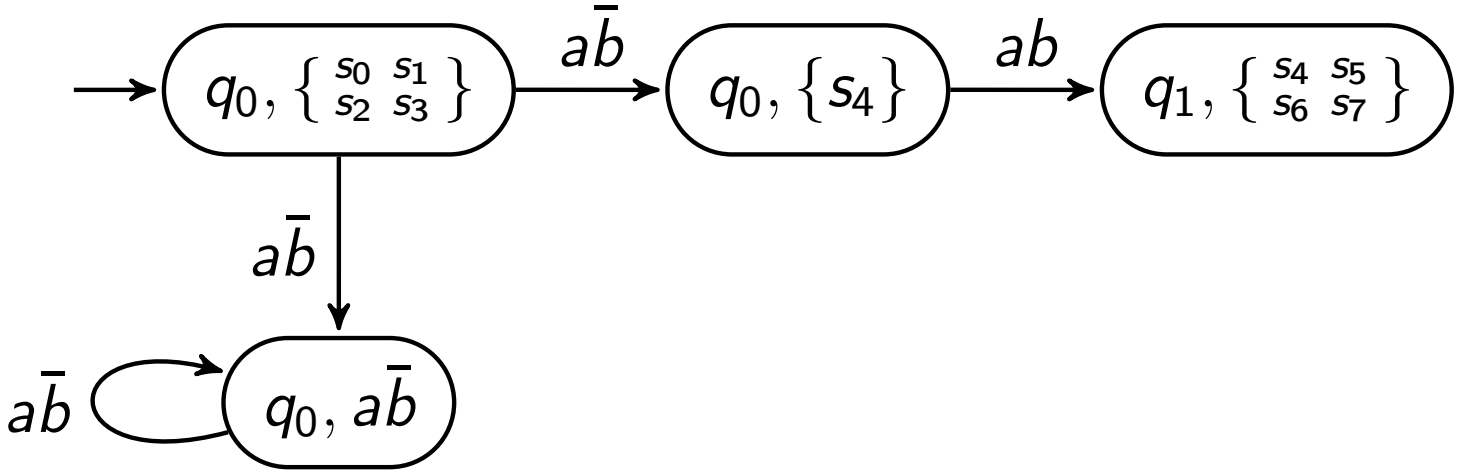
Symbolic Observation **Product**:



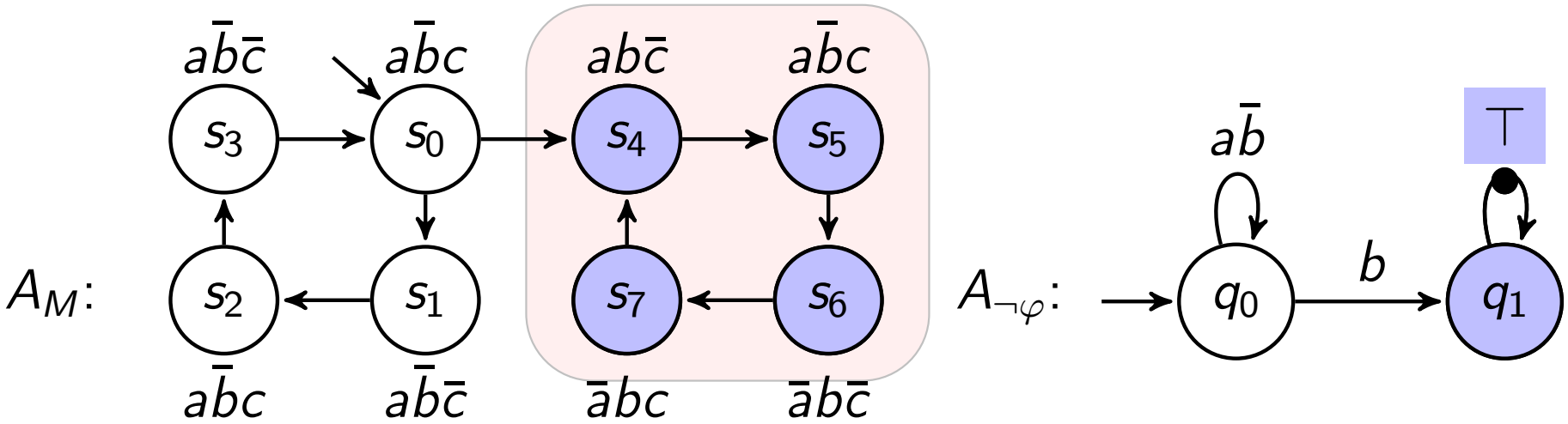
Symbolic Observation Product (SOP)



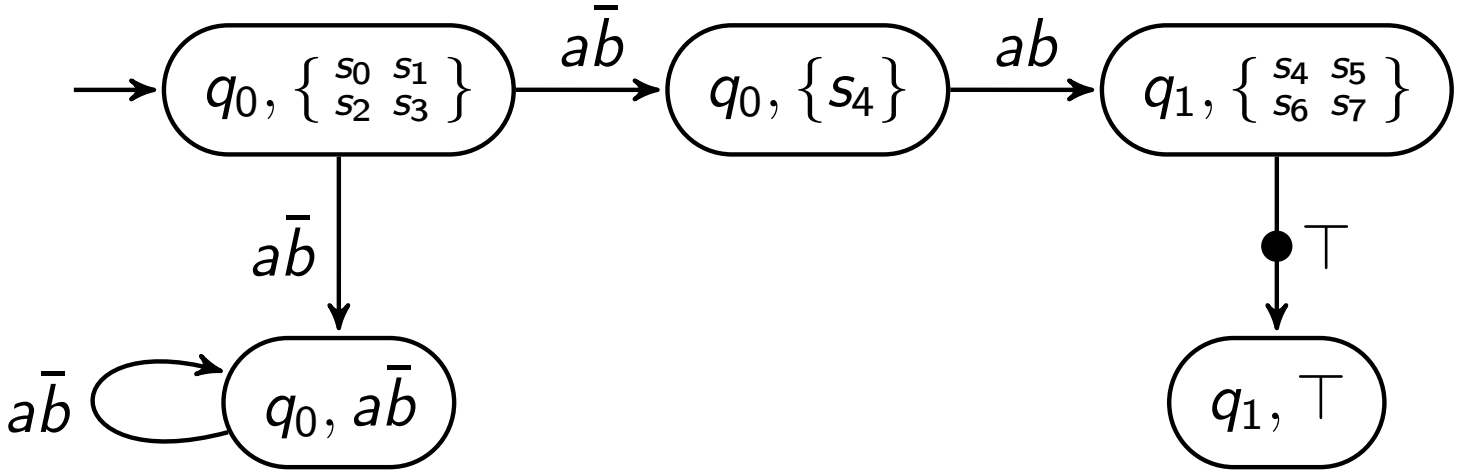
Symbolic Observation **Product**:



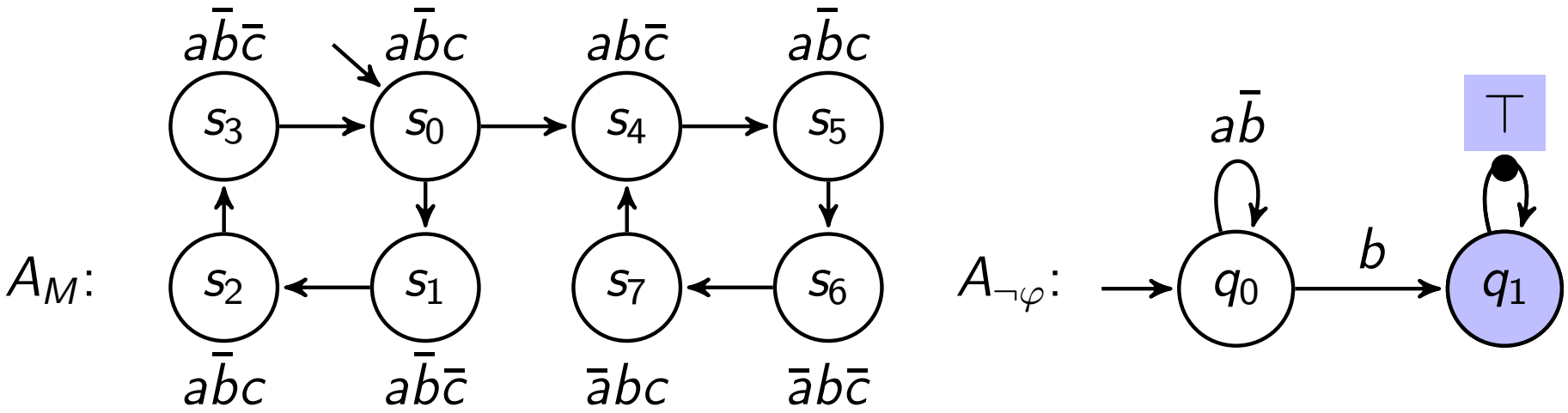
Symbolic Observation Product (SOP)



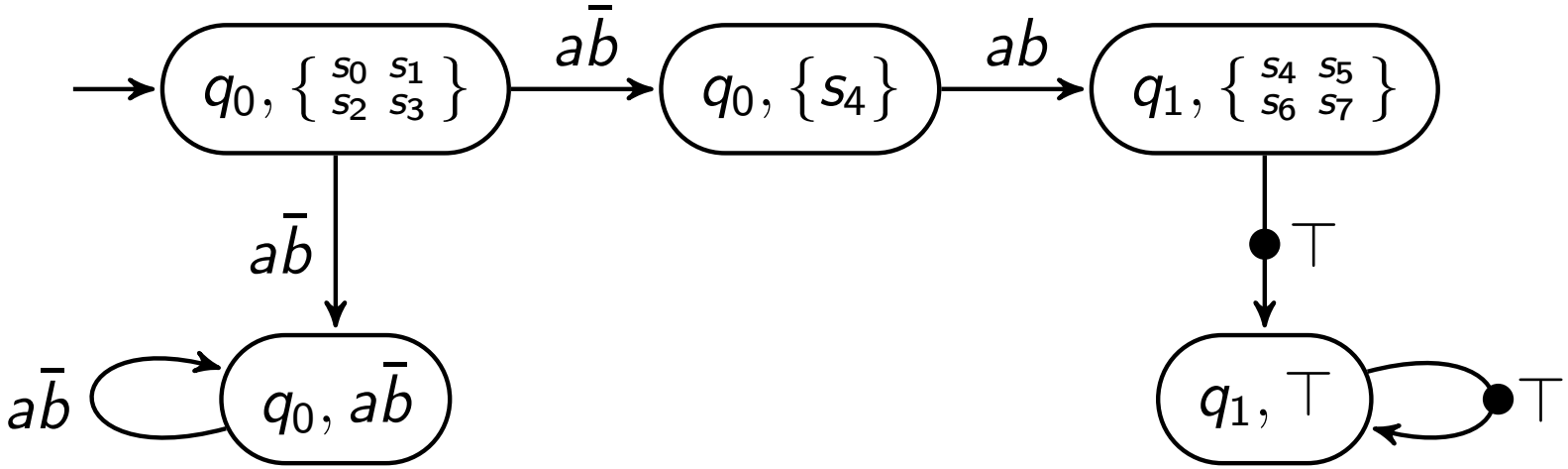
Symbolic Observation **Product**:



Symbolic Observation Product (SOP)

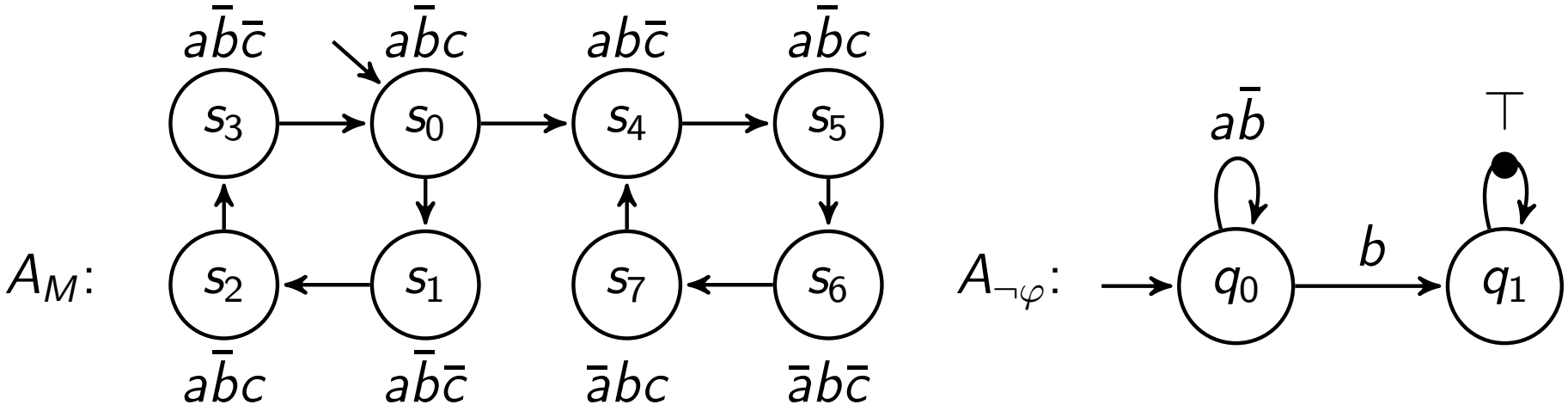


Symbolic Observation **Product**:



Self-Loop Aggregation Product (SLAP)

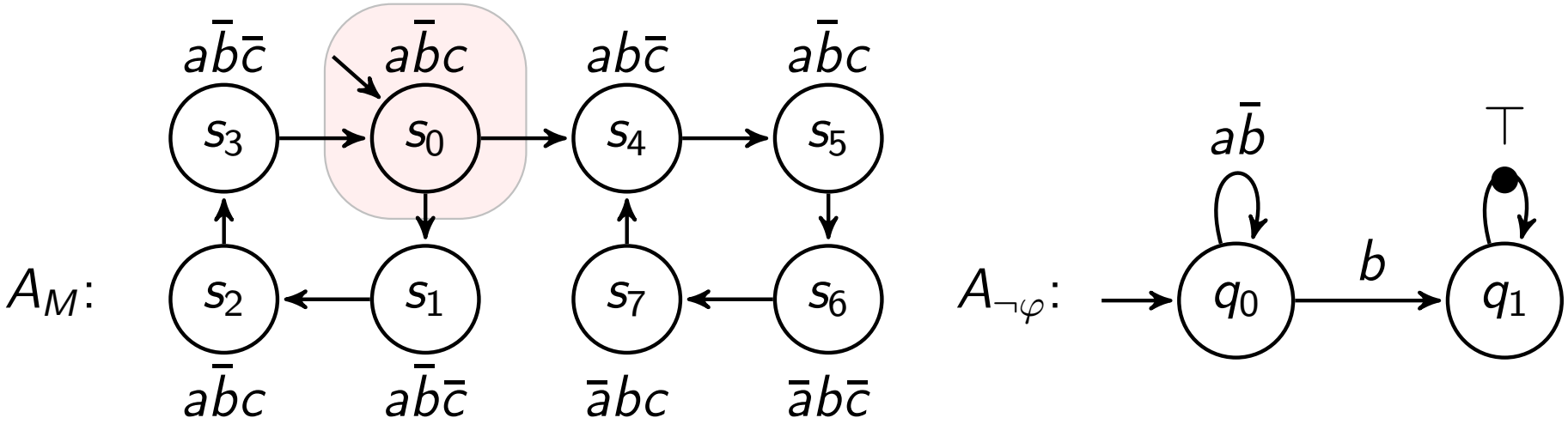
- Self-loop aggregation product - a new hybrid approach to on-the-fly LTL model checking. In ATVA'11



Self-Loop Aggregation **Product**:

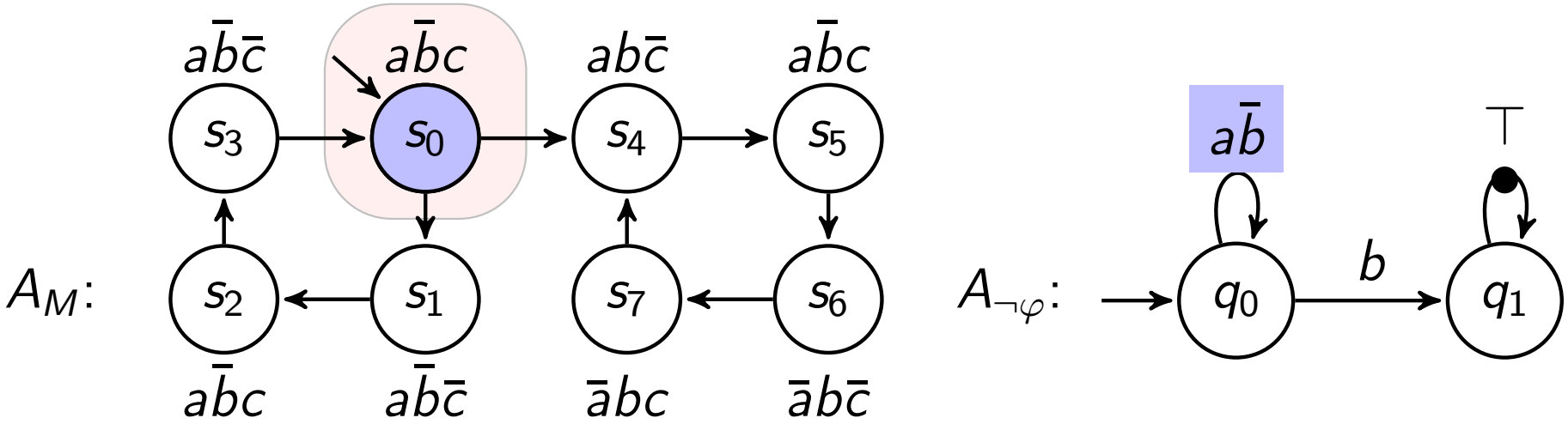
- Specialized synchronized product for full LTL
- Based on the self loops encountered in the property automaton
- Gather all states compatible with the self-loops of the property automaton **at the current point**.

Self-Loop Aggregation Product (SLAP)



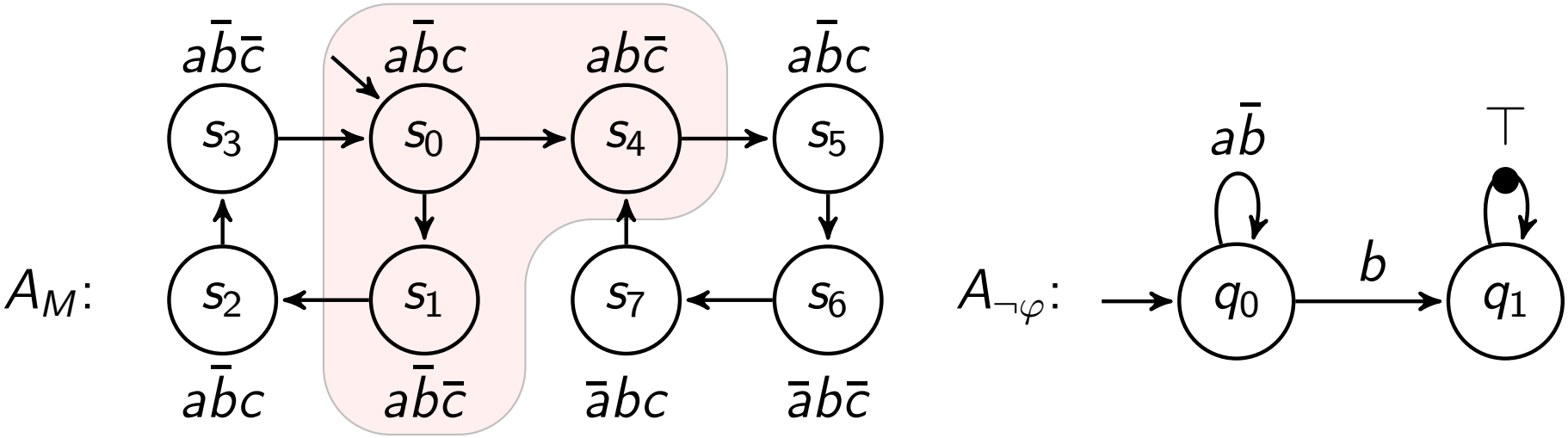
Self-Loop Aggregation **Product**:

Self-Loop Aggregation Product (SLAP)



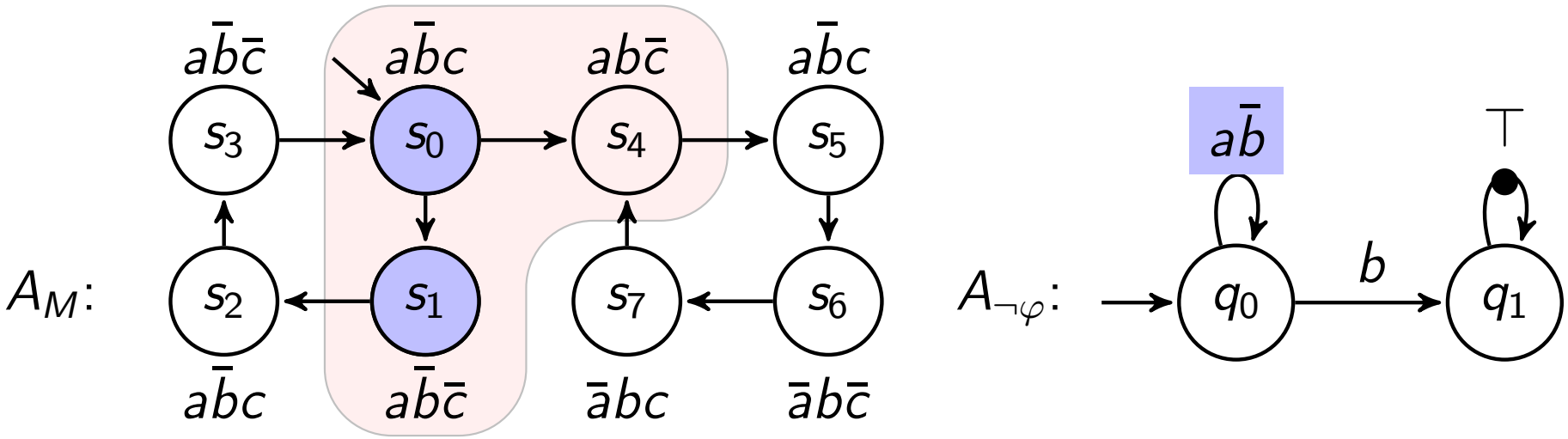
Self-Loop Aggregation **Product**:

Self-Loop Aggregation Product (SLAP)



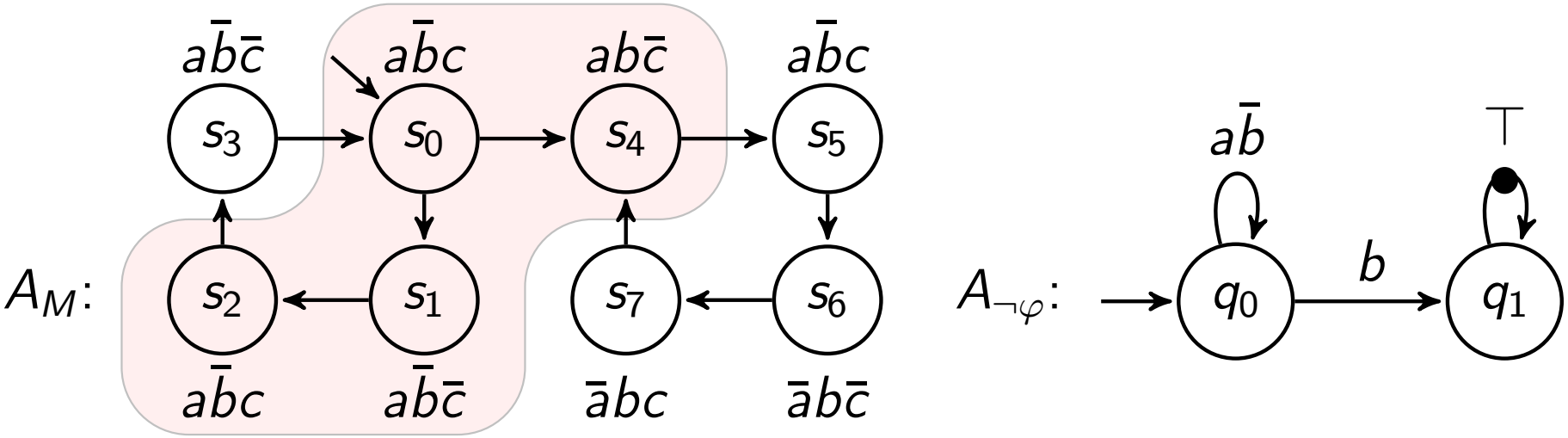
Self-Loop Aggregation **Product**:

Self-Loop Aggregation Product (SLAP)



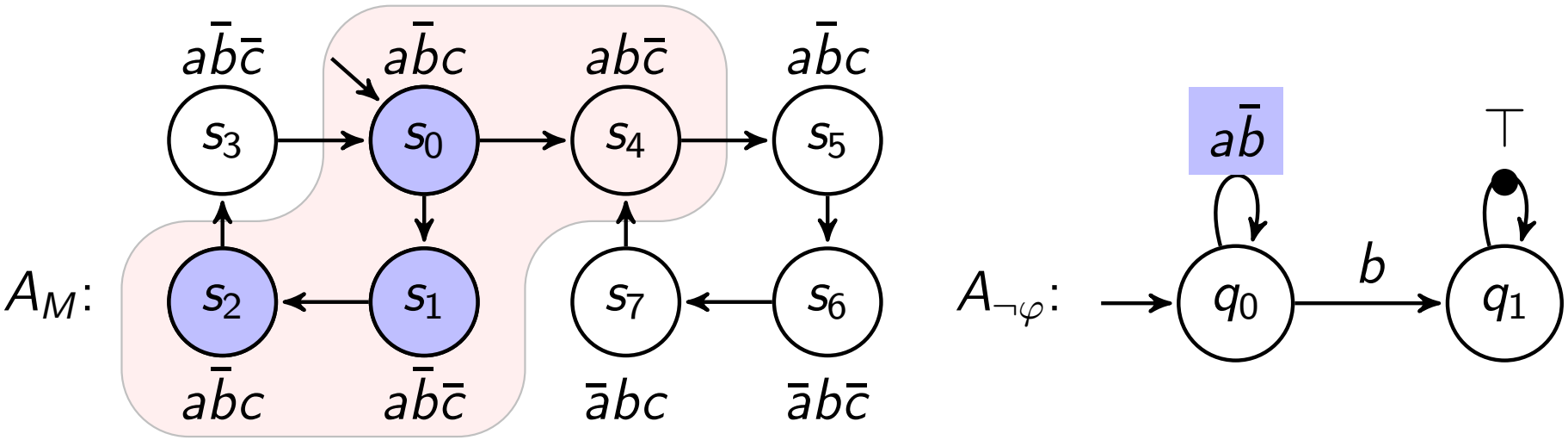
Self-Loop Aggregation **Product**:

Self-Loop Aggregation Product (SLAP)



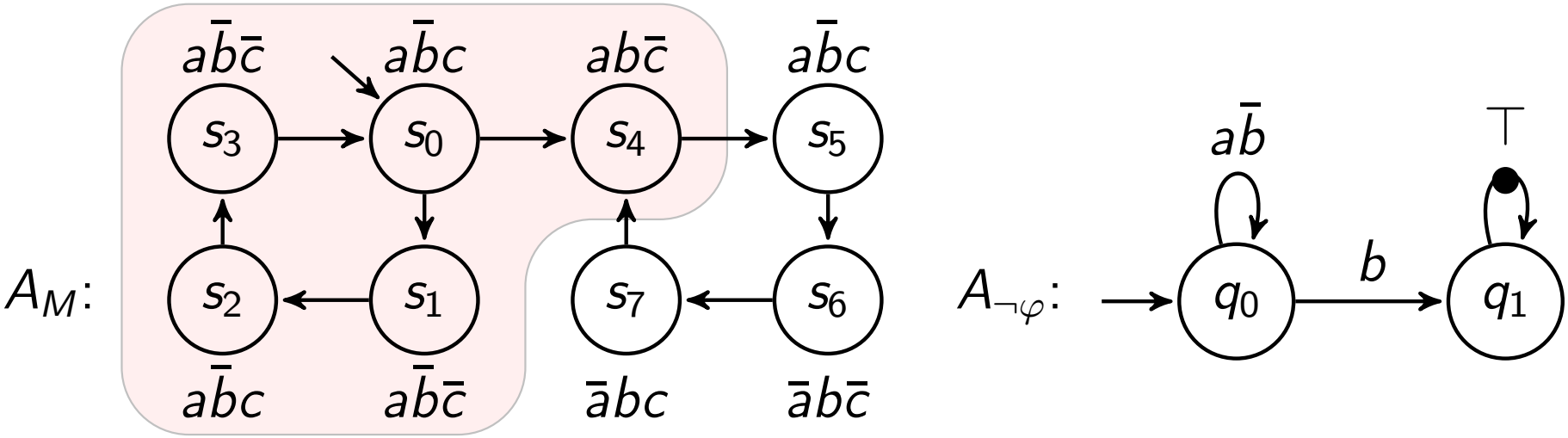
Self-Loop Aggregation **Product**:

Self-Loop Aggregation Product (SLAP)



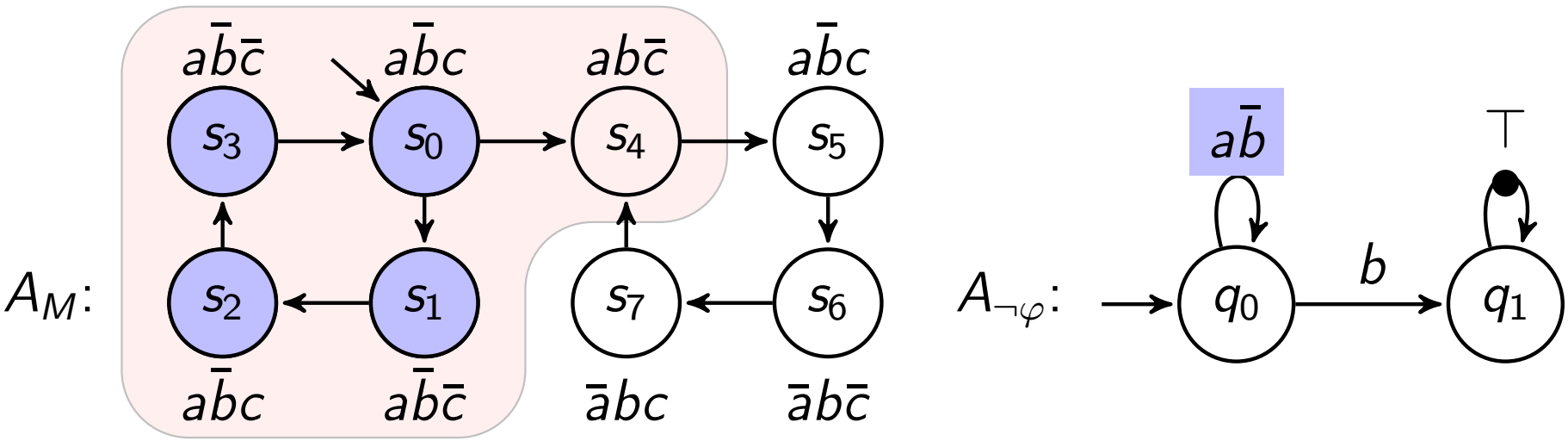
Self-Loop Aggregation **Product**:

Self-Loop Aggregation Product (SLAP)



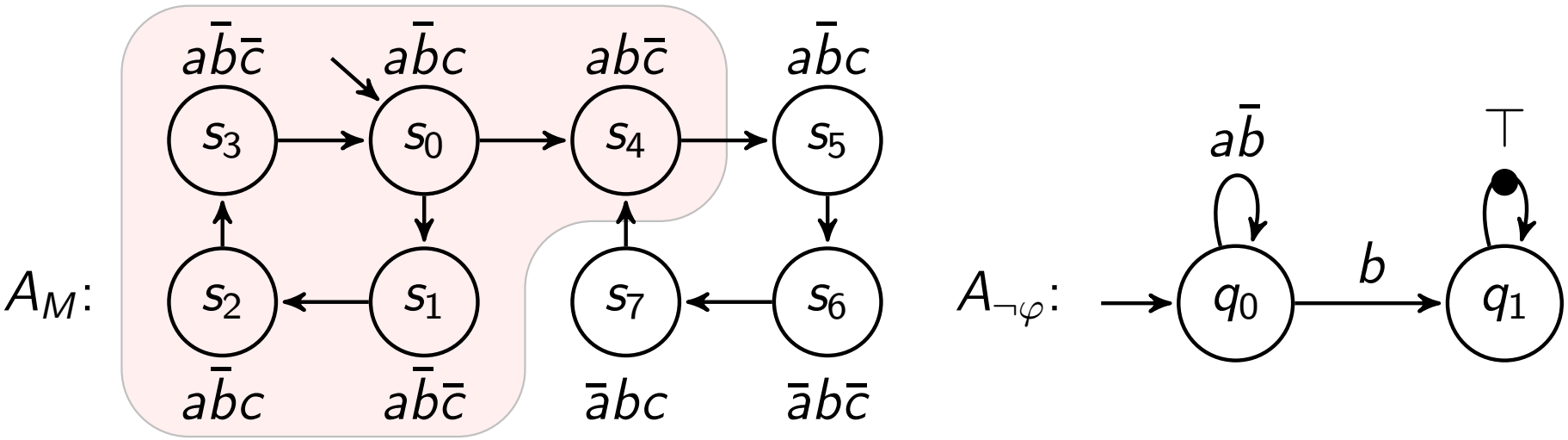
Self-Loop Aggregation **Product**:

Self-Loop Aggregation Product (SLAP)

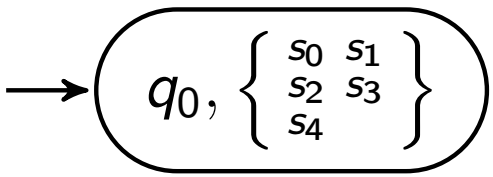


Self-Loop Aggregation **Product**:

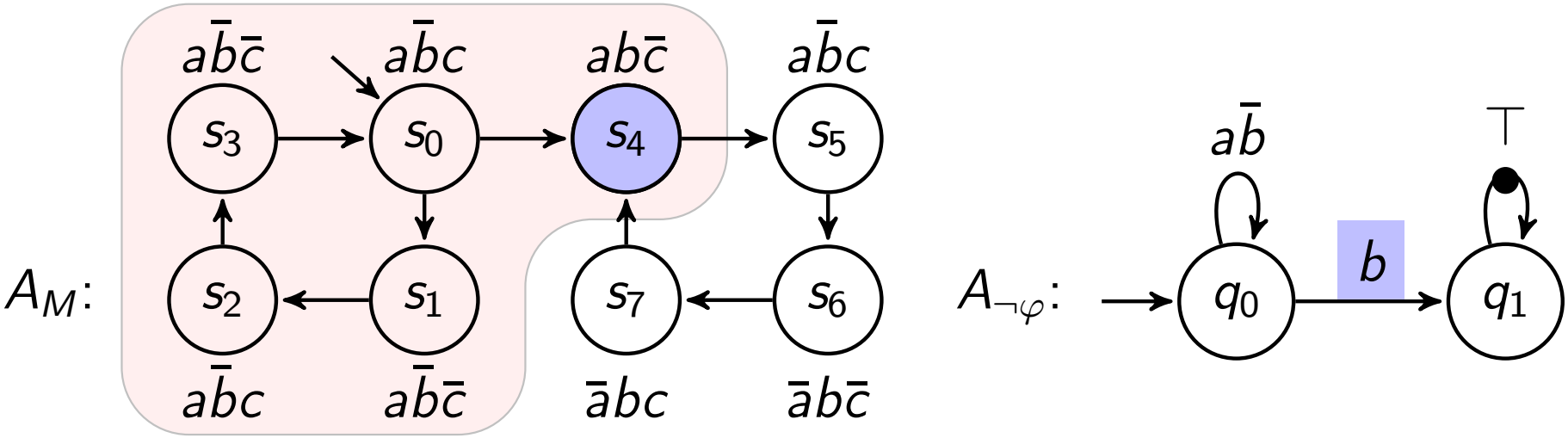
Self-Loop Aggregation Product (SLAP)



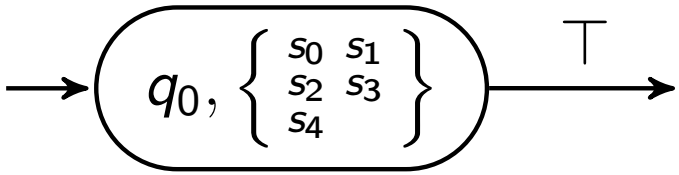
Self-Loop Aggregation **Product**:



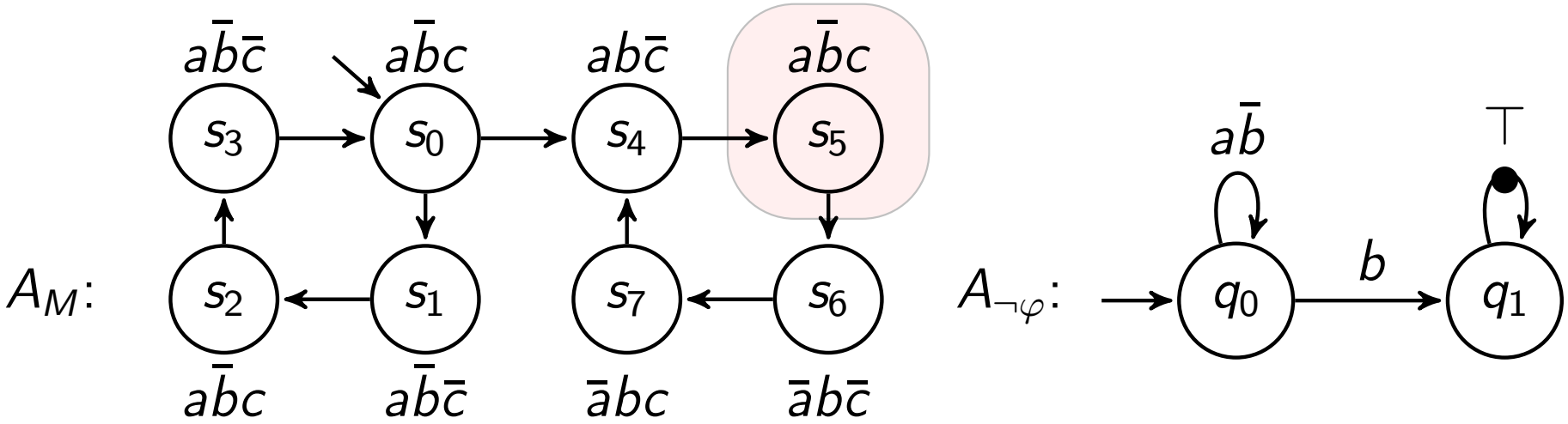
Self-Loop Aggregation Product (SLAP)



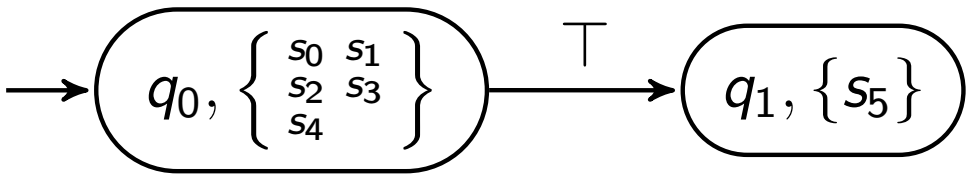
Self-Loop Aggregation **Product**:



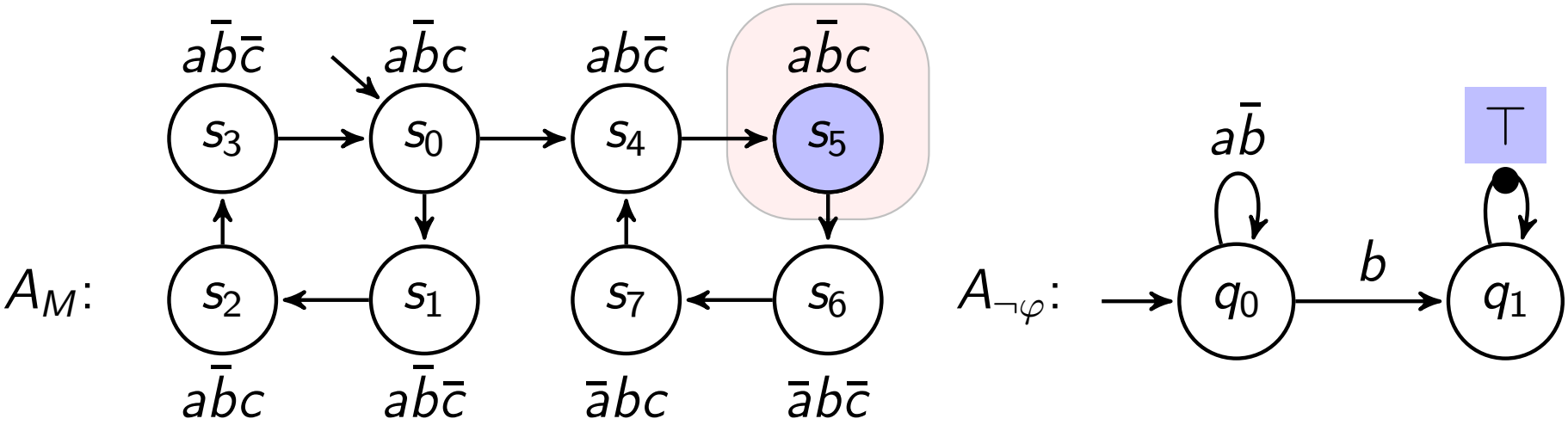
Self-Loop Aggregation Product (SLAP)



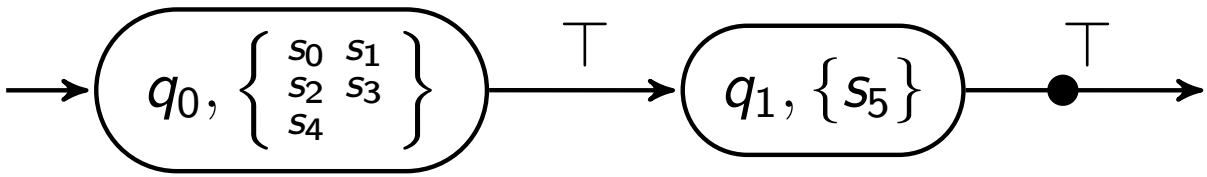
Self-Loop Aggregation **Product**:



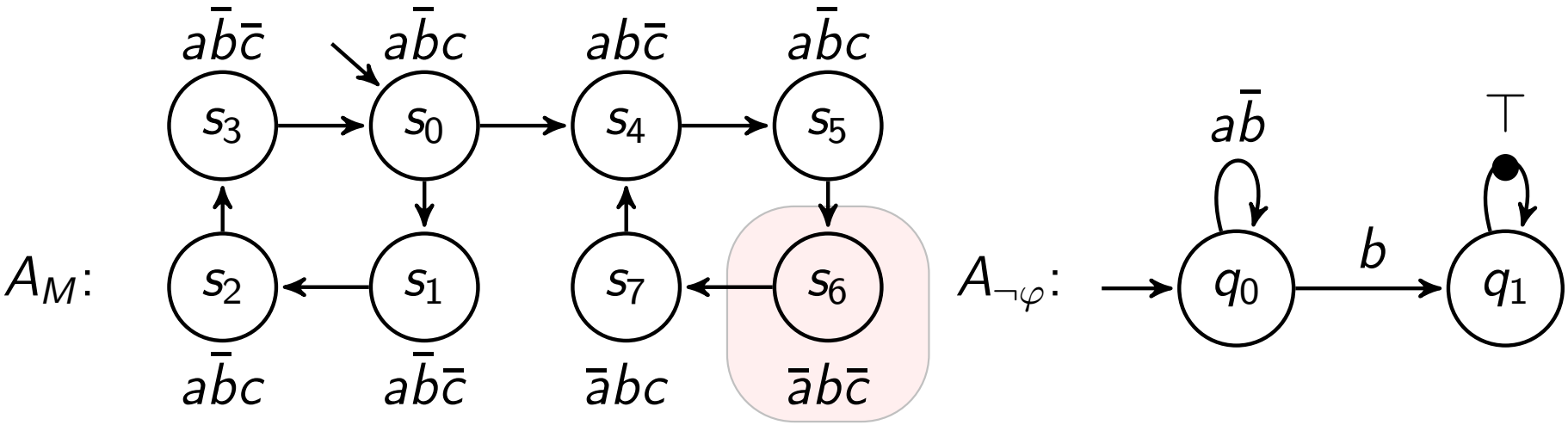
Self-Loop Aggregation Product (SLAP)



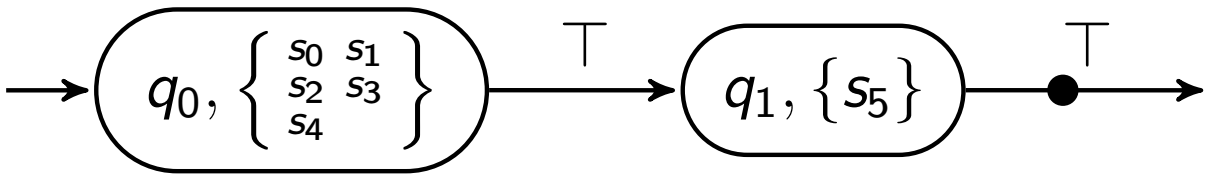
Self-Loop Aggregation **Product**:



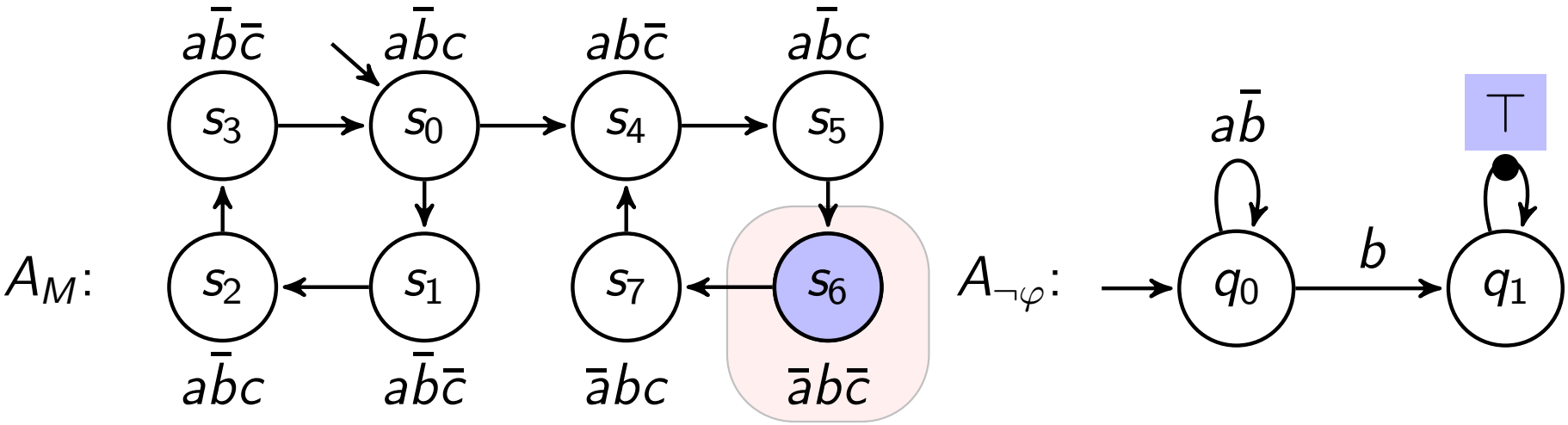
Self-Loop Aggregation Product (SLAP)



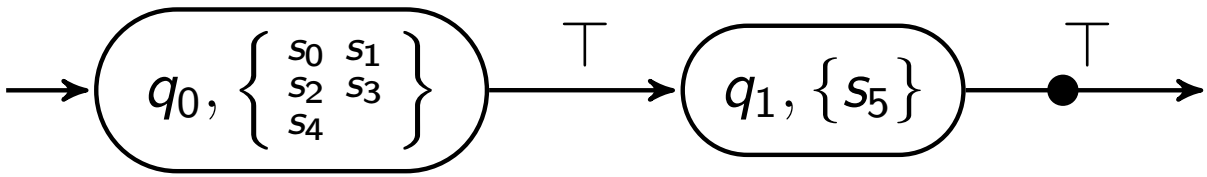
Self-Loop Aggregation **Product**:



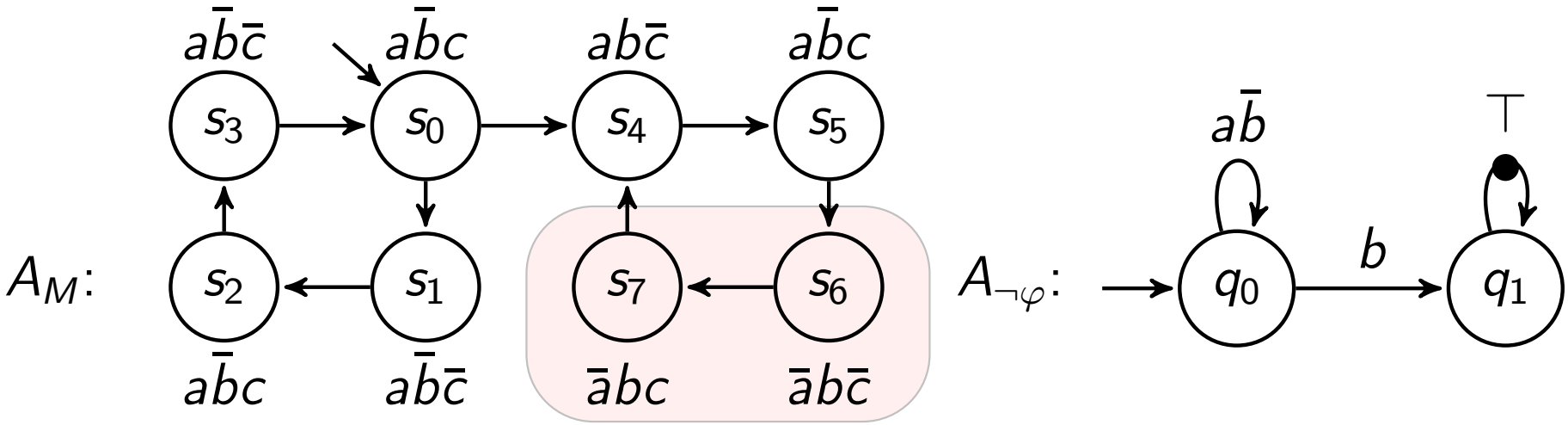
Self-Loop Aggregation Product (SLAP)



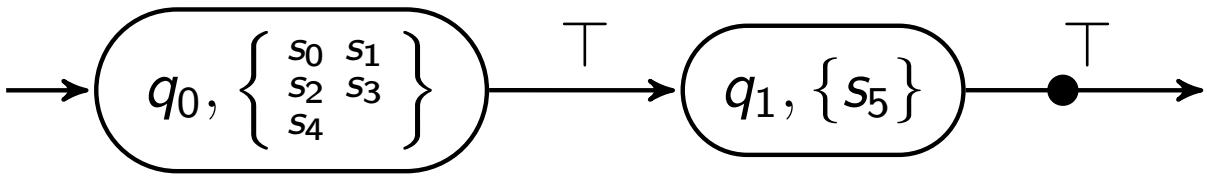
Self-Loop Aggregation **Product**:



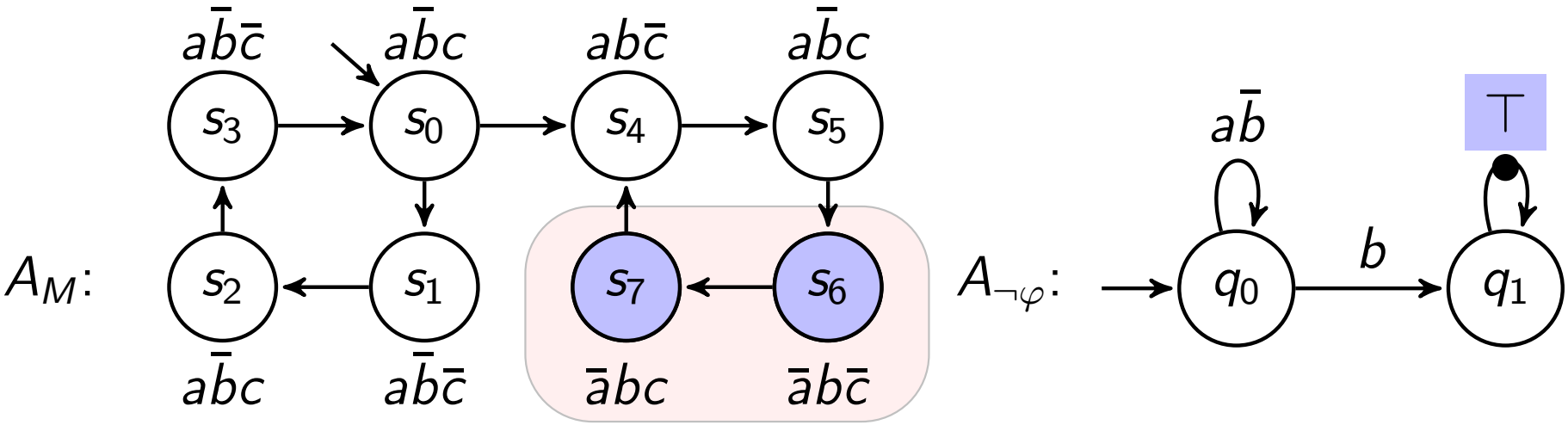
Self-Loop Aggregation Product (SLAP)



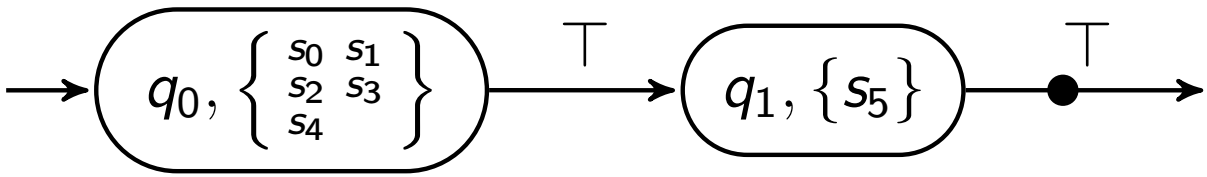
Self-Loop Aggregation **Product**:



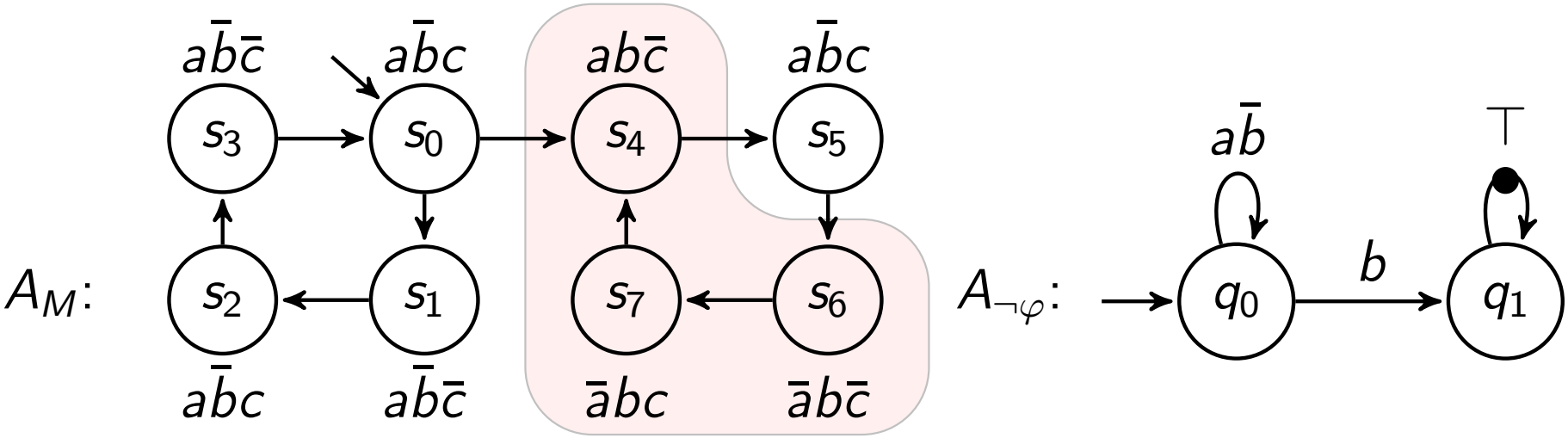
Self-Loop Aggregation Product (SLAP)



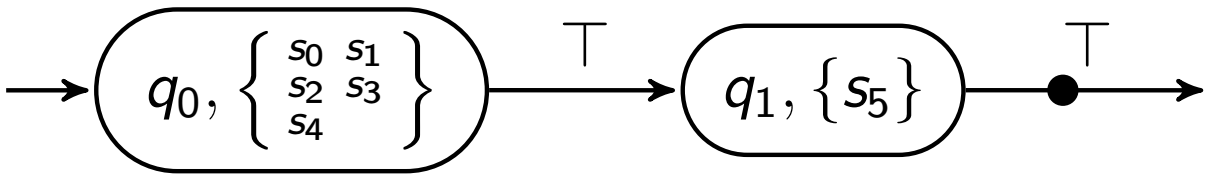
Self-Loop Aggregation **Product**:



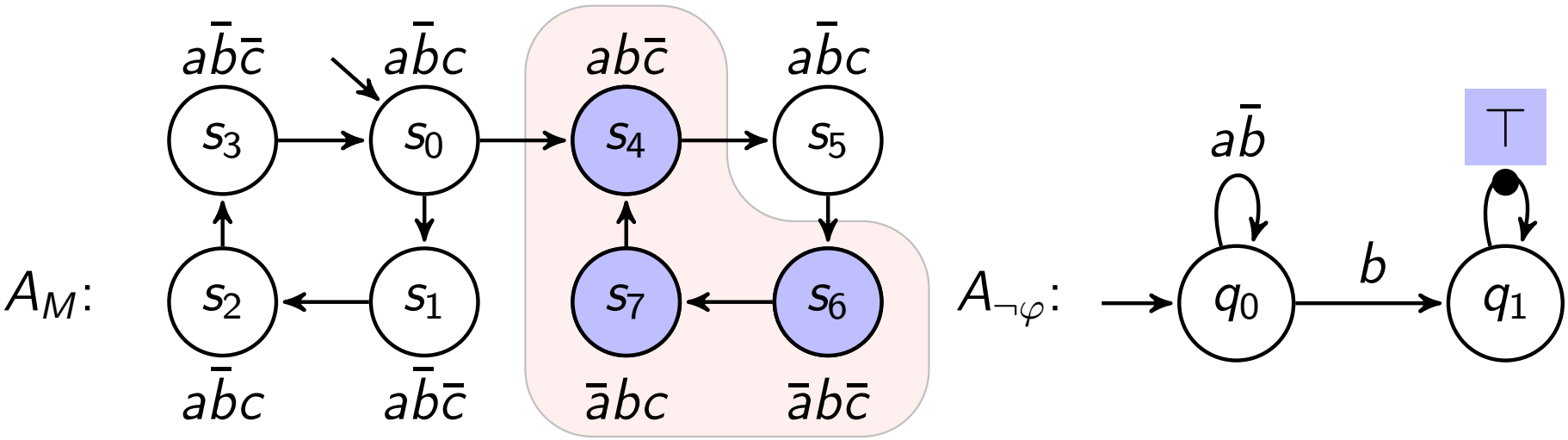
Self-Loop Aggregation Product (SLAP)



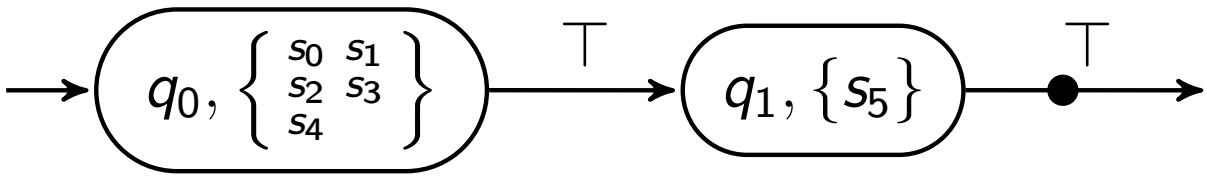
Self-Loop Aggregation **Product**:



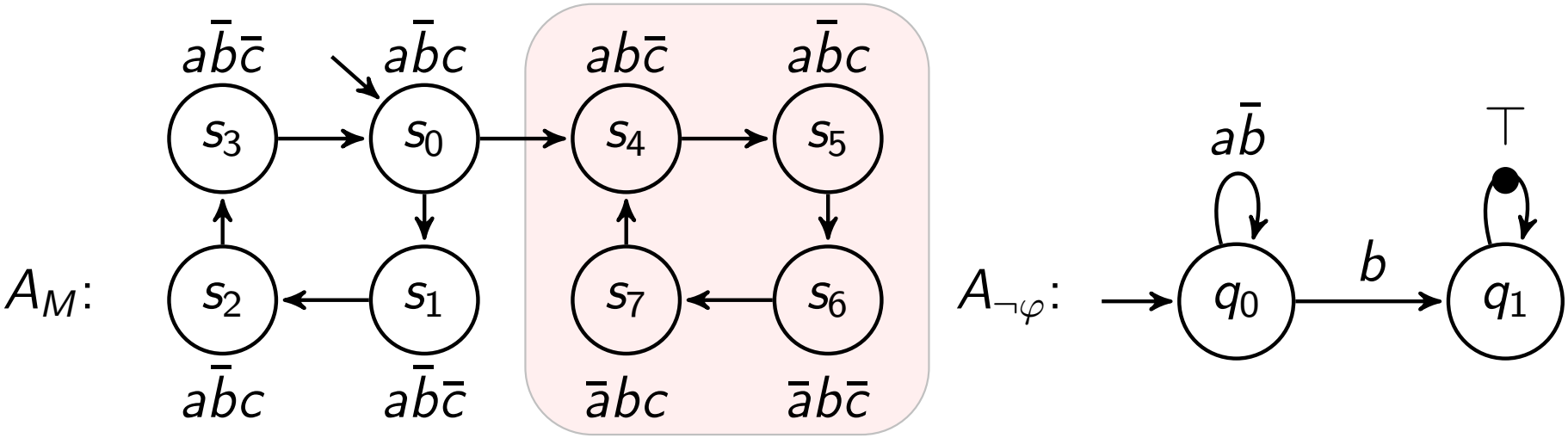
Self-Loop Aggregation Product (SLAP)



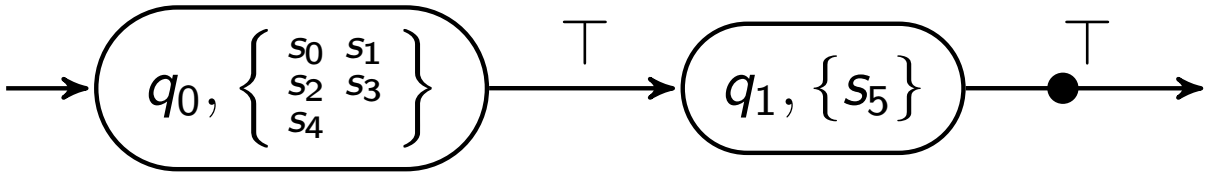
Self-Loop Aggregation **Product**:



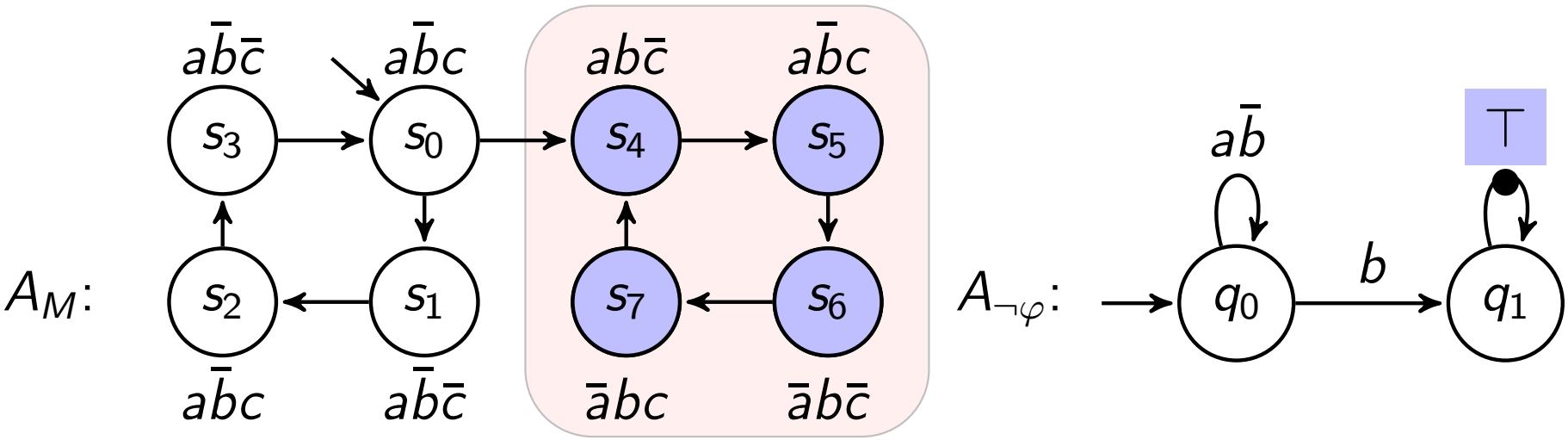
Self-Loop Aggregation Product (SLAP)



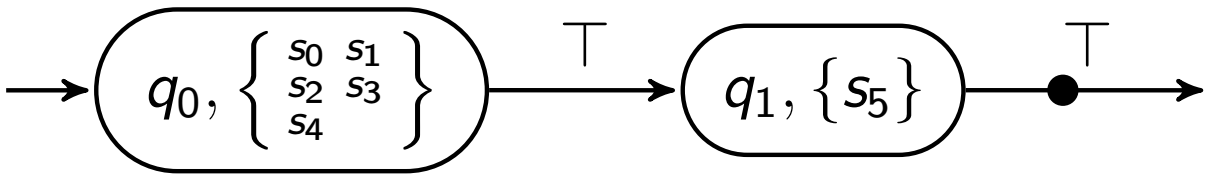
Self-Loop Aggregation **Product**:



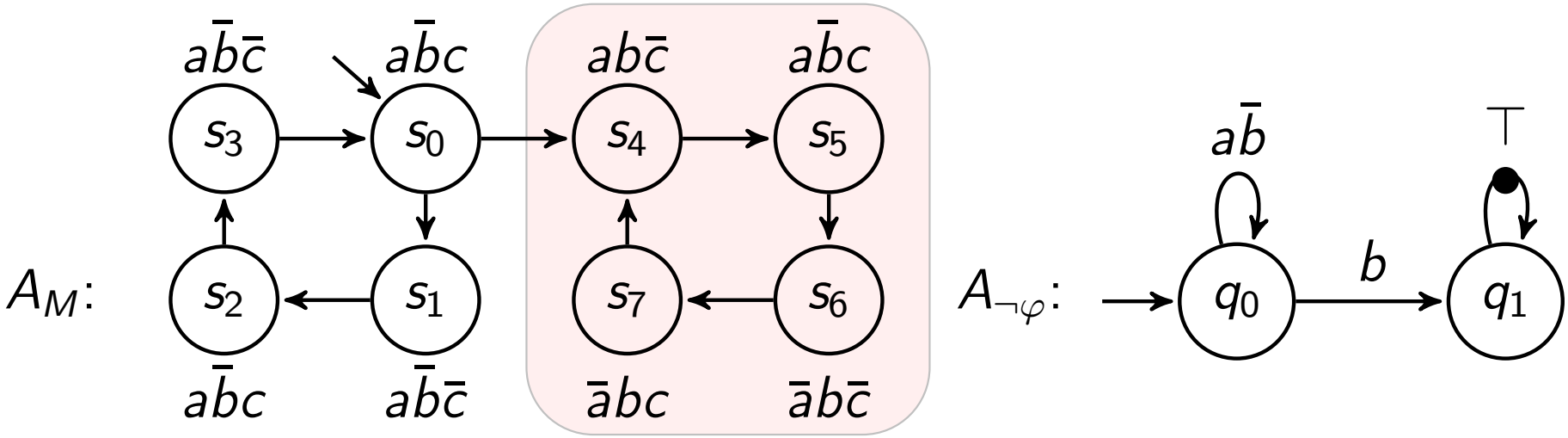
Self-Loop Aggregation Product (SLAP)



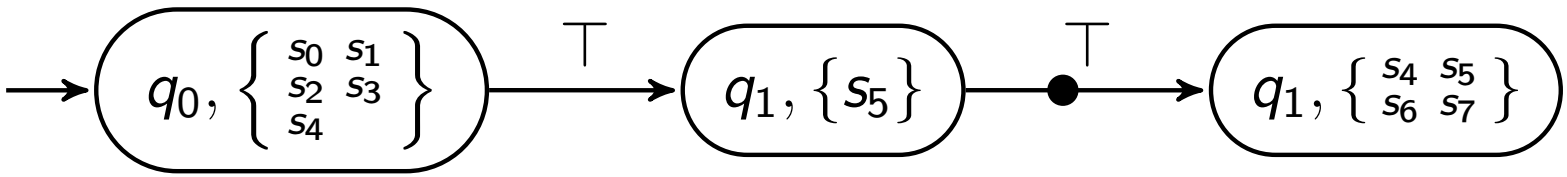
Self-Loop Aggregation **Product**:



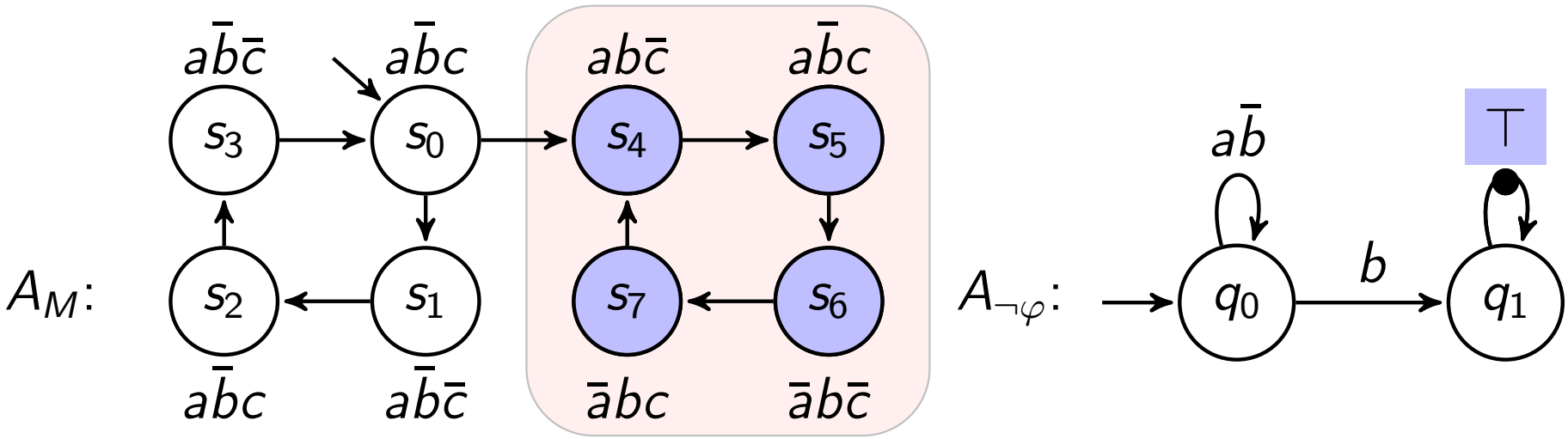
Self-Loop Aggregation Product (SLAP)



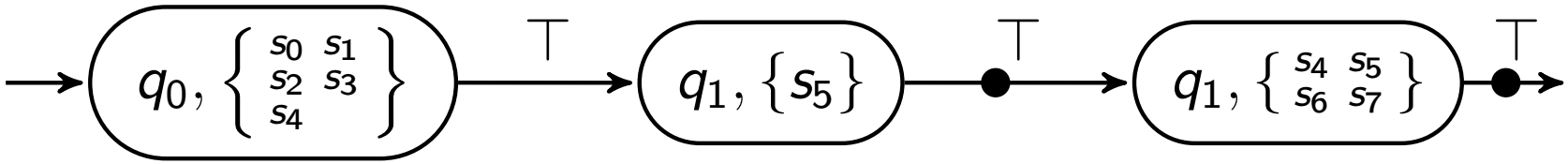
Self-Loop Aggregation **Product**:



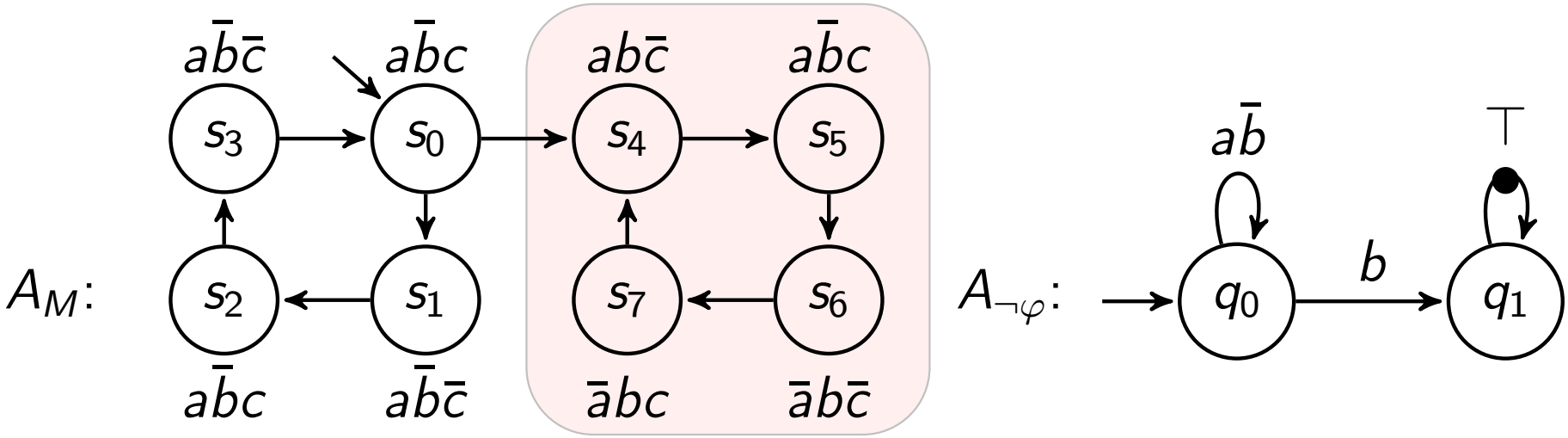
Self-Loop Aggregation Product (SLAP)



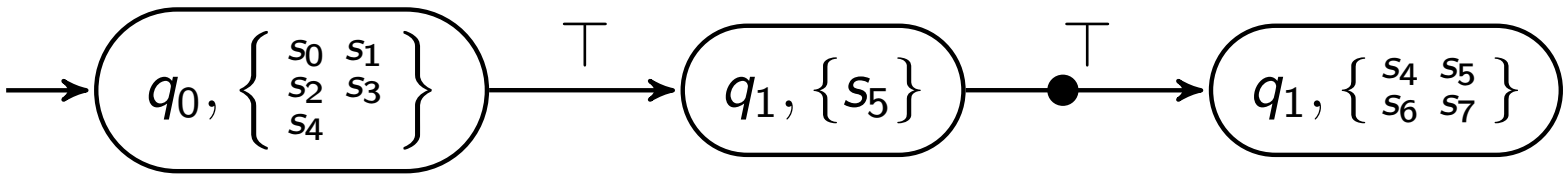
Self-Loop Aggregation **Product**:



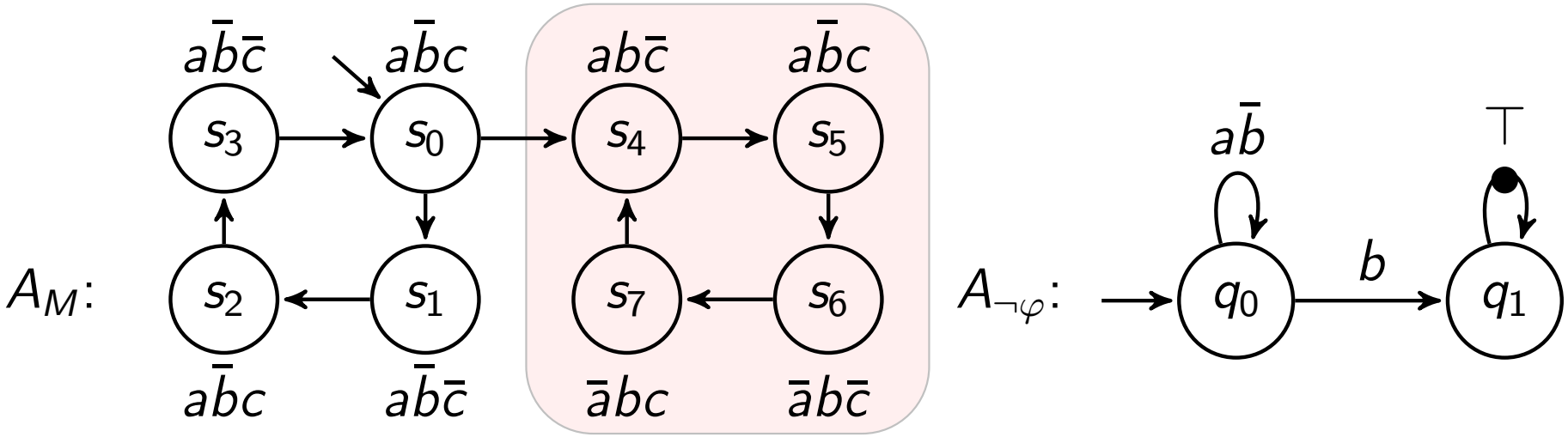
Self-Loop Aggregation Product (SLAP)



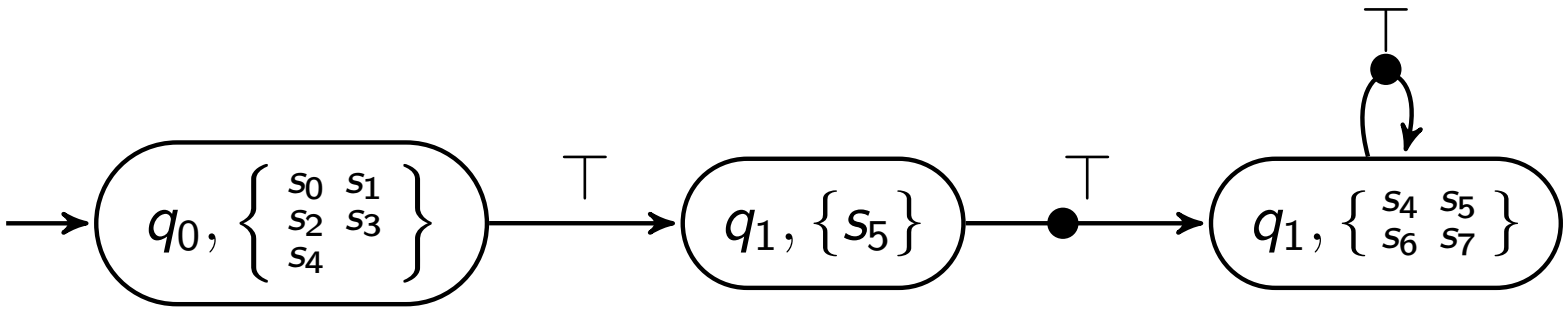
Self-Loop Aggregation **Product**:



Self-Loop Aggregation Product (SLAP)

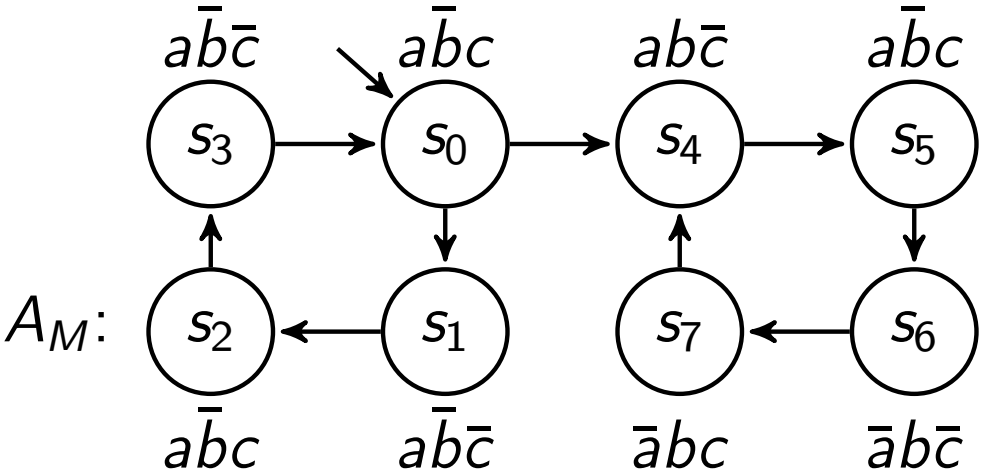


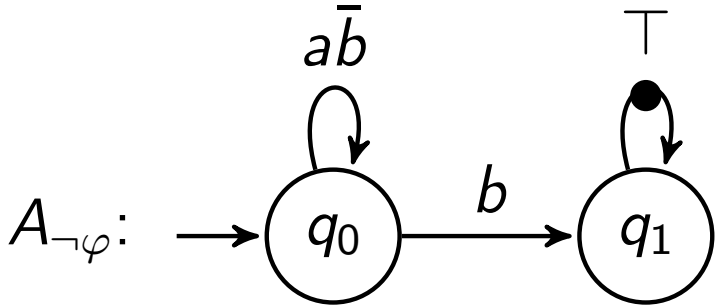
Self-Loop Aggregation **Product**:

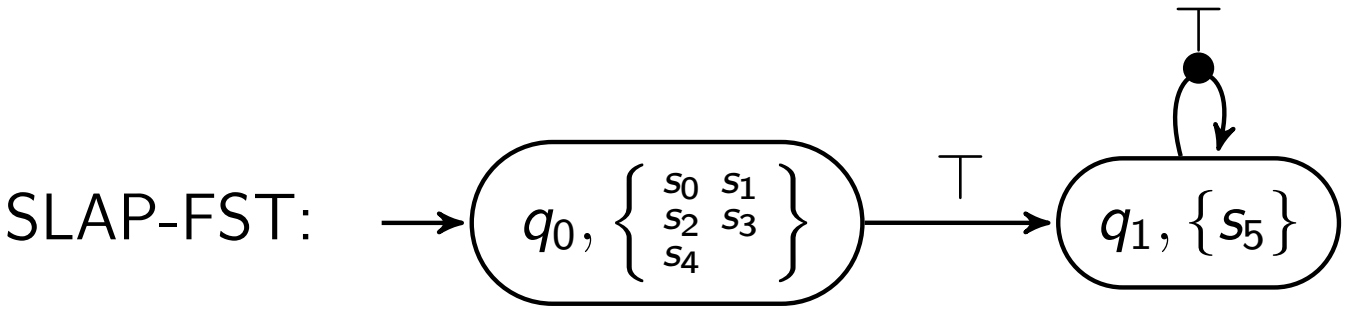


SLAP-FST

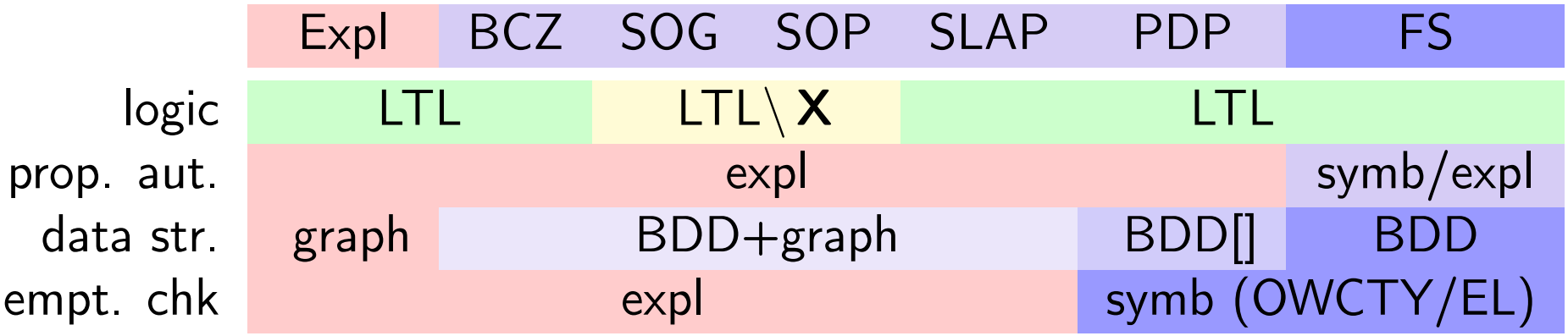
- FST = Fully Symbolic search in Terminal automata states
- When reaching a state of $A_{\neg\varphi}$ which is terminal (and accepting) use a fully symbolic search.







Summary of methods



BCZ: Multiple-State Tableaux, PDP : Property Driven Partitioning

Experimental Framework

Implemented algorithms

BCZ, SOG, SOP, SLAP, SLAP-FST, EL, OWCTY

Tools used

SDD Hierarchical Set Decision Diagrams (`http://ddd.lip6.fr/`)

- Hierarchy (memory gain)
- Automatic saturation (improves fixpoint computations)

Spot Model checking library (`http://spot.lip6.fr/`)

- Good LTL-to-TGBA translation
- Explicit emptiness-check algorithms

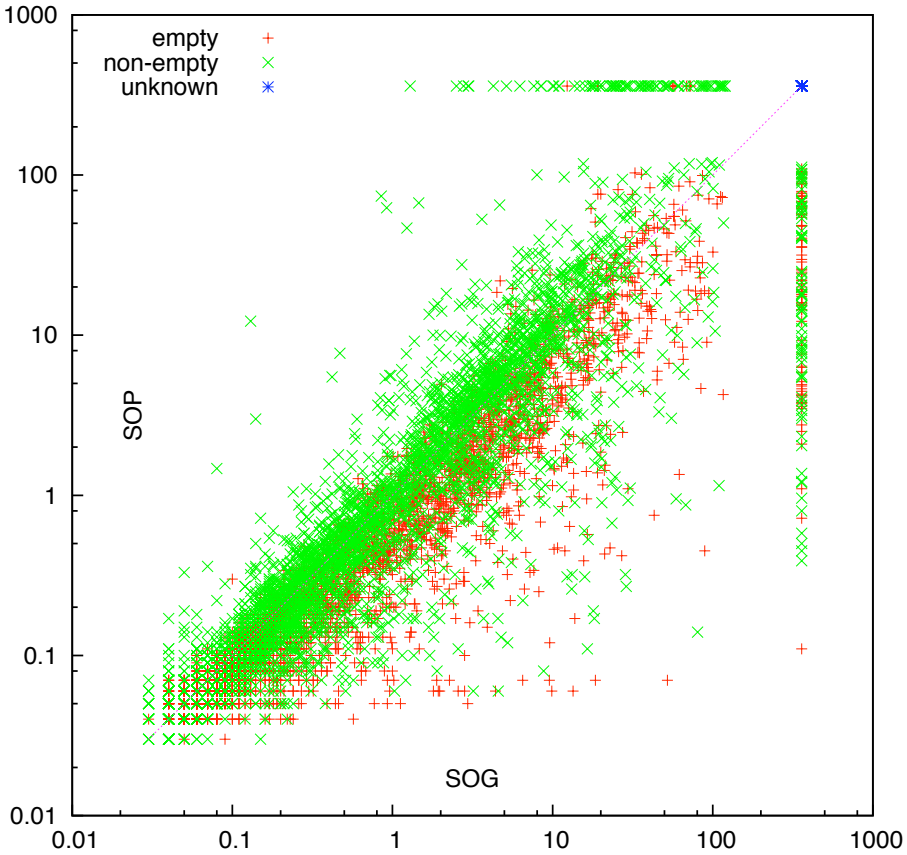
Data

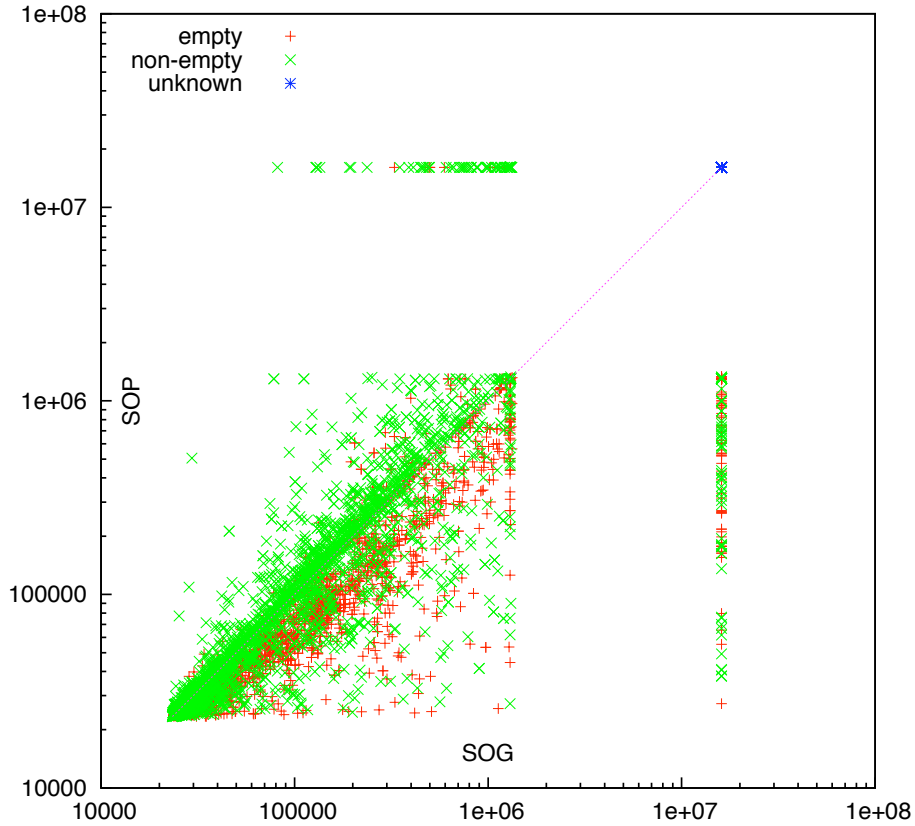
Models Scalable toy models with $10^6 \ldots 10^{66}$ states.

Formulas Random formulas (filtered), plus a set of (random) weak fairness properties.

Scatter Plots: Stuttering invariant algorithms SOG vs SOP

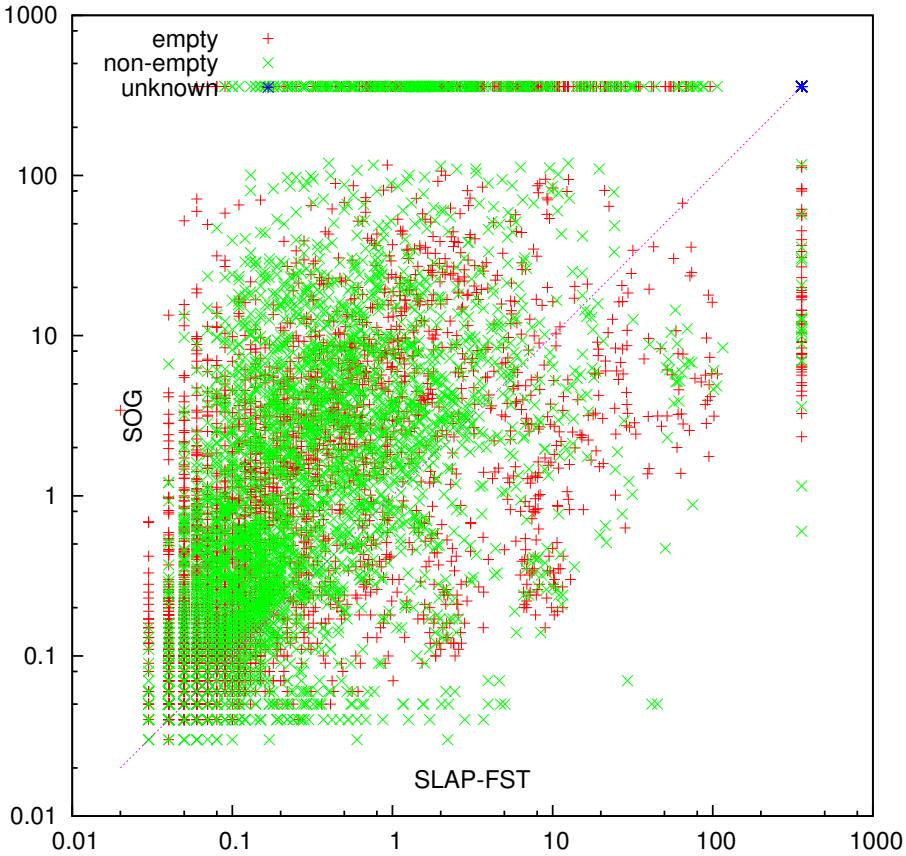
Runtime in seconds. Timeout at 120s. Memory in Kilobytes.
Garbage collection threshold at 1.3 GB

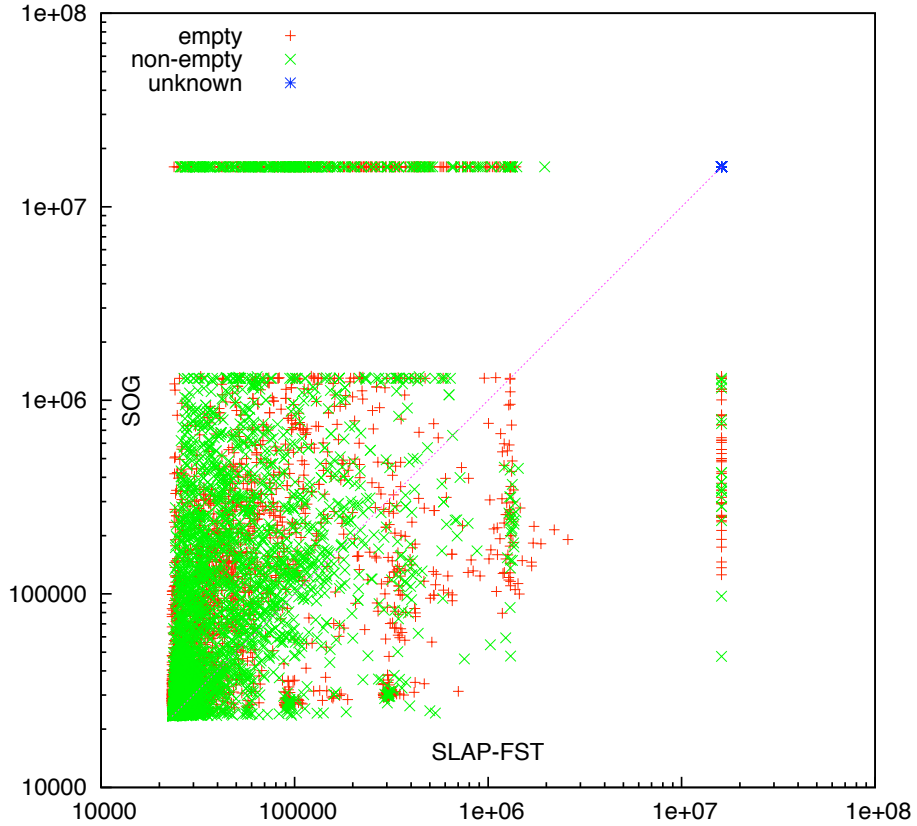




Scatter Plots: SLAP-FST vs SOG

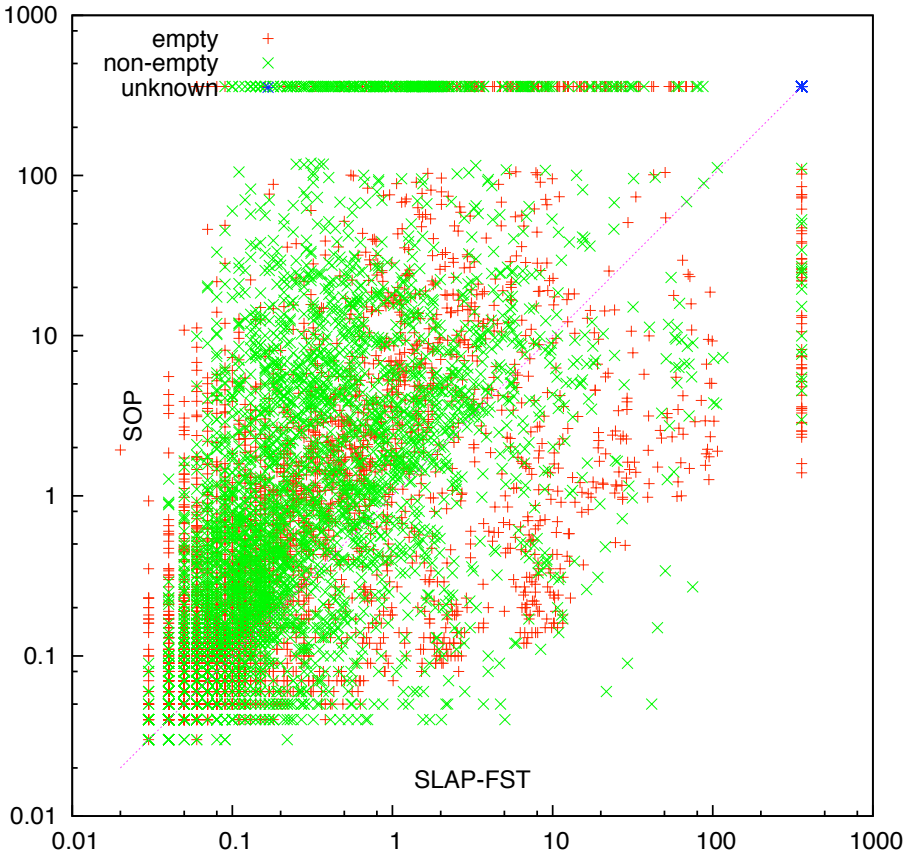
Runtime in seconds. Timeout at 120s. Memory in Kilobytes.
Garbage collection threshold at 1.3 GB

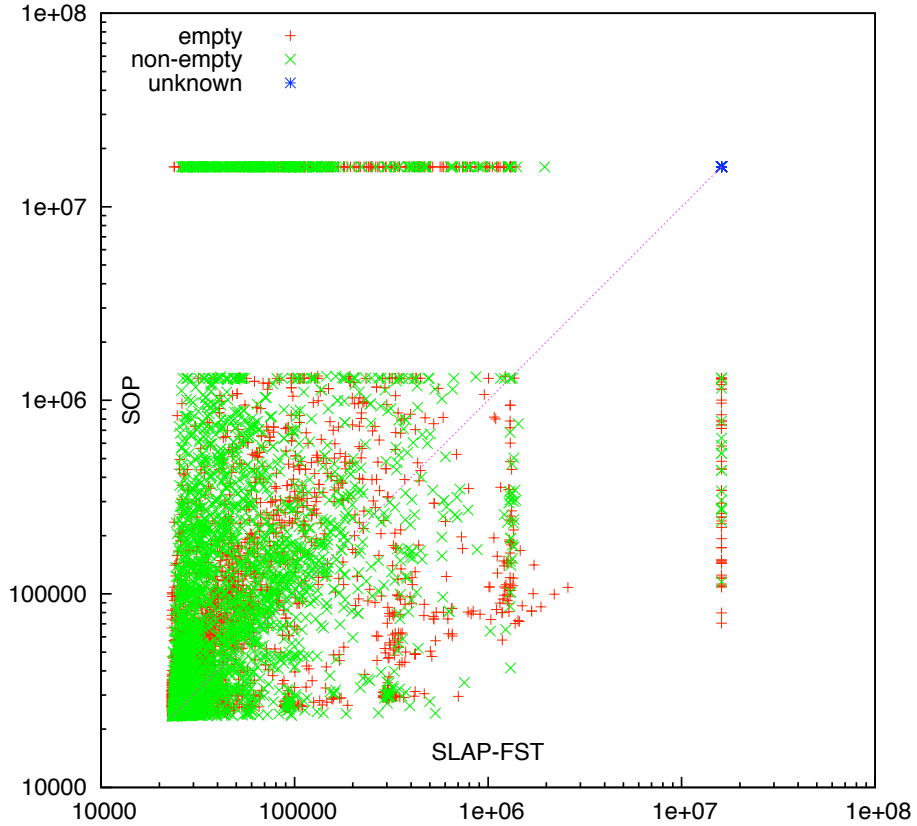




Scatter Plots: SLAP-FST vs SOP

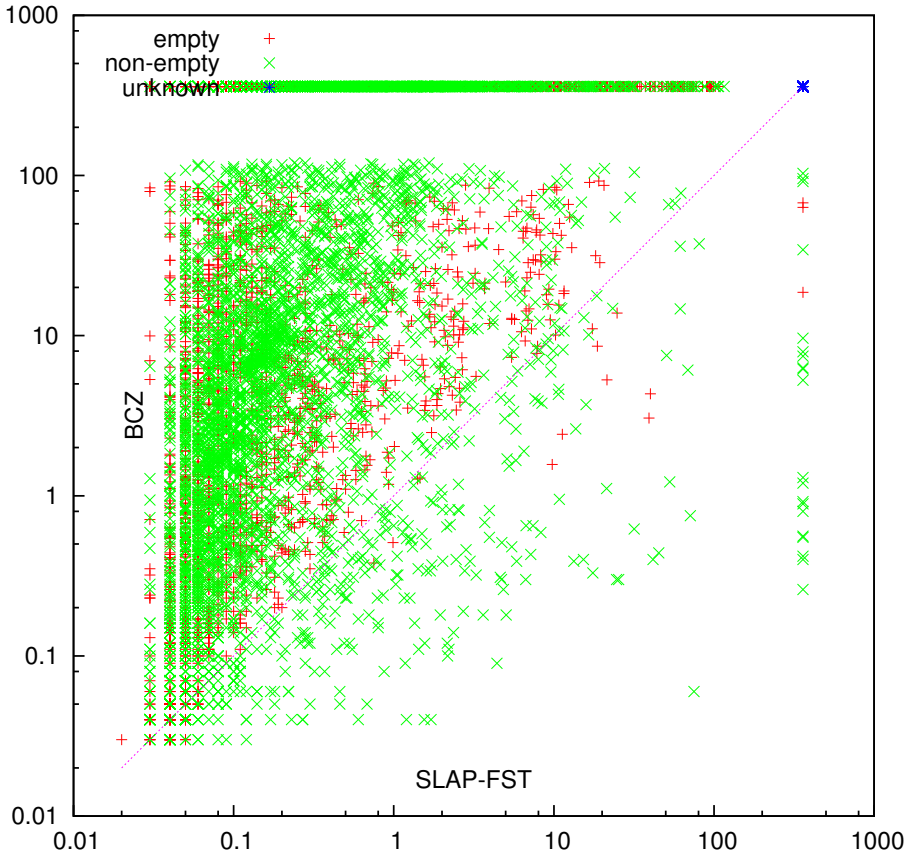
Runtime in seconds. Timeout at 120s. Memory in Kilobytes.
Garbage collection threshold at 1.3 GB

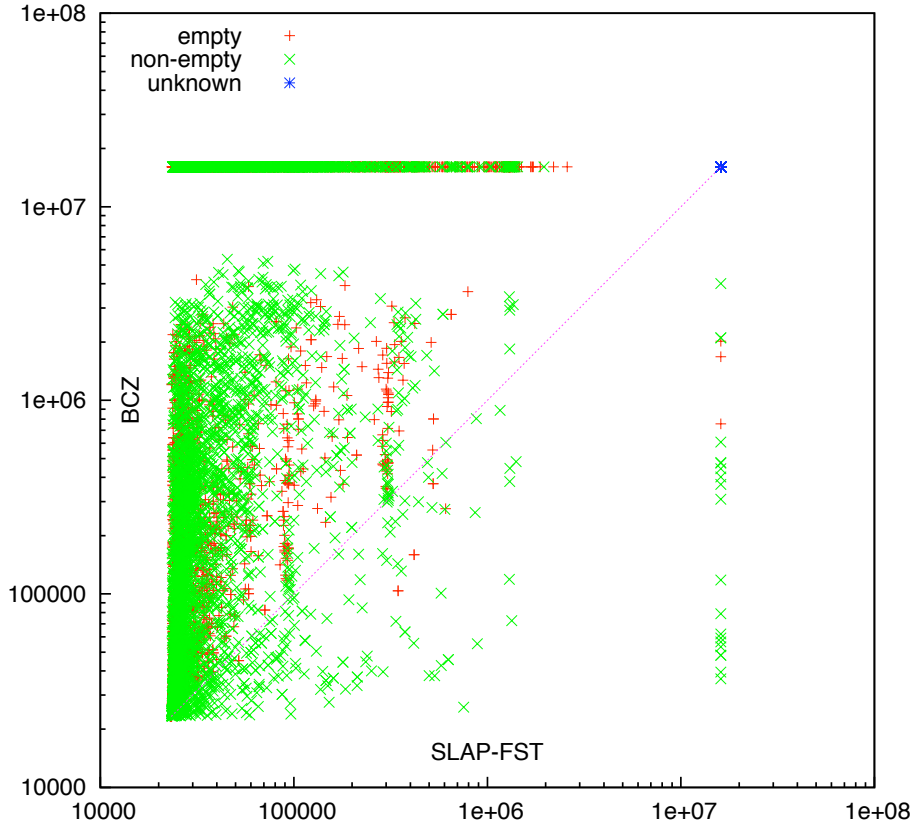




Scatter Plots: SLAP-FST vs. BCZ

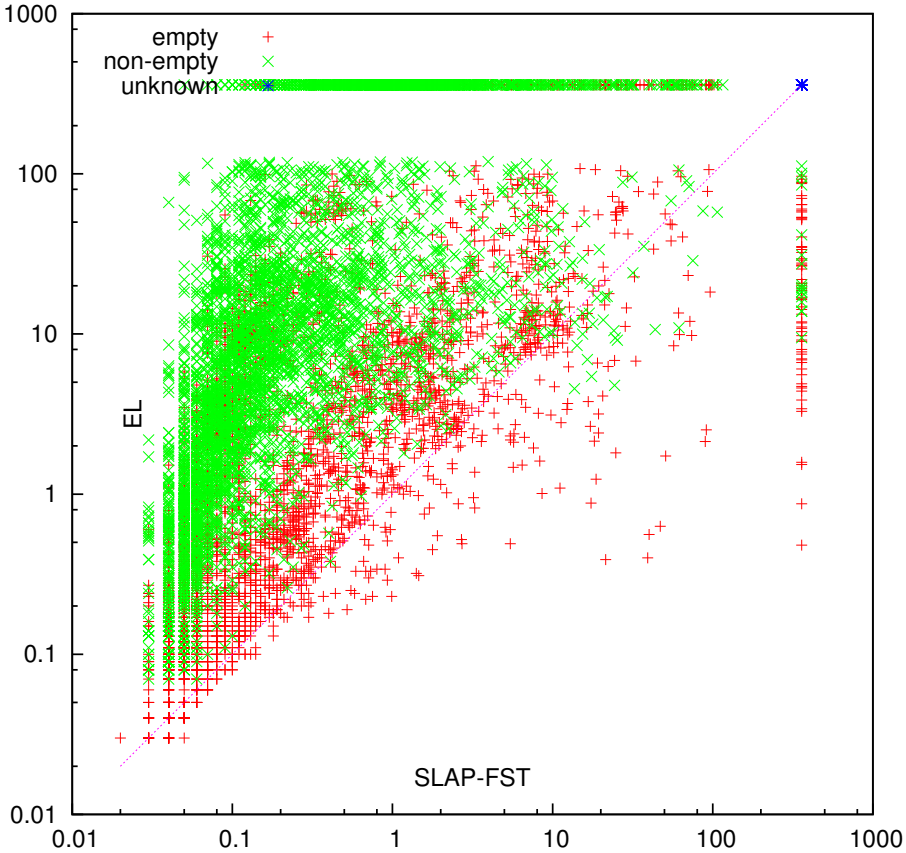
Runtime in seconds. Timeout at 120s. Memory in Kilobytes.
Garbage collection threshold at 1.3 GB

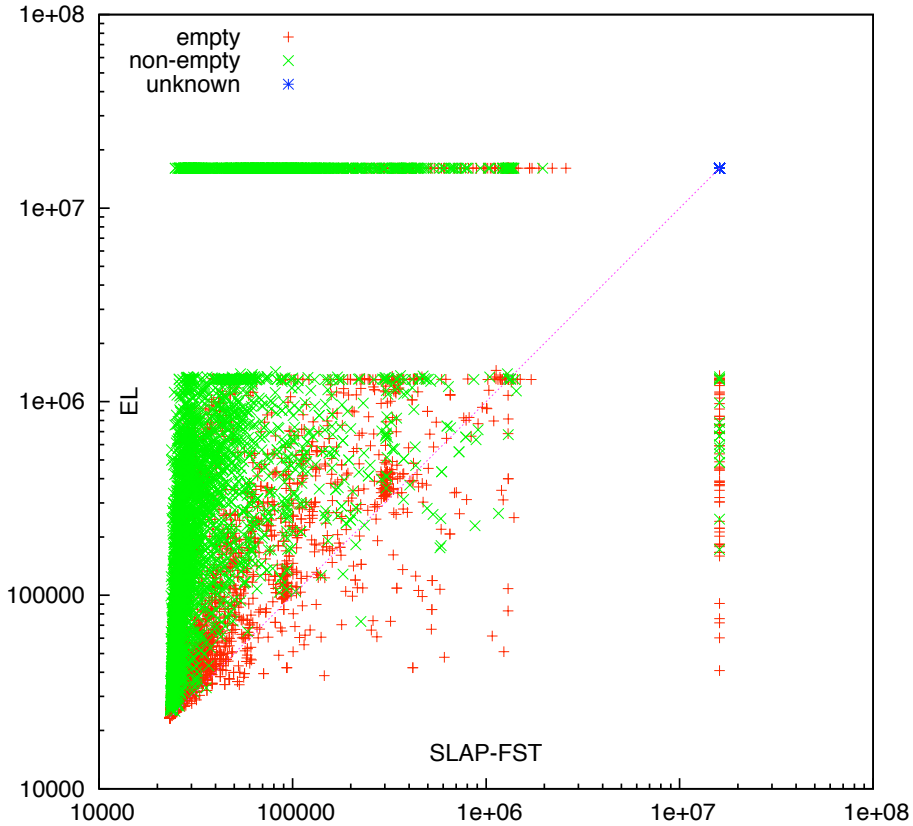




Scatter Plots: SLAP-FST vs. Fully Symbolic

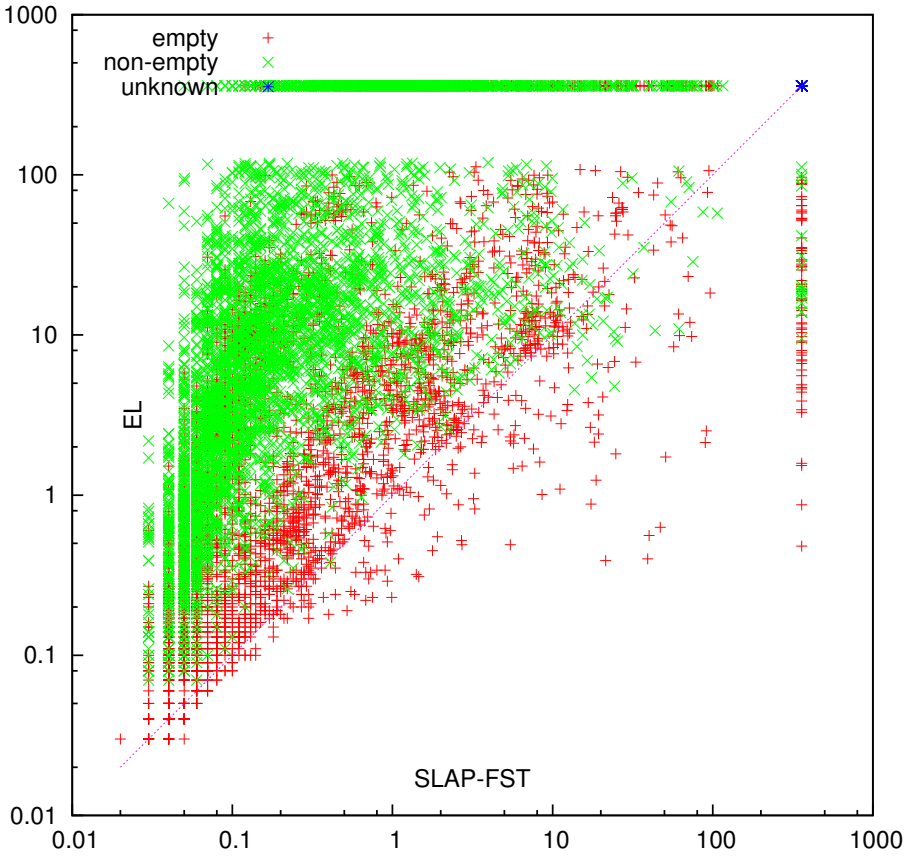
Runtime in seconds. Timeout at 120s. Memory in Kilobytes.
Garbage collection threshold at 1.3 GB

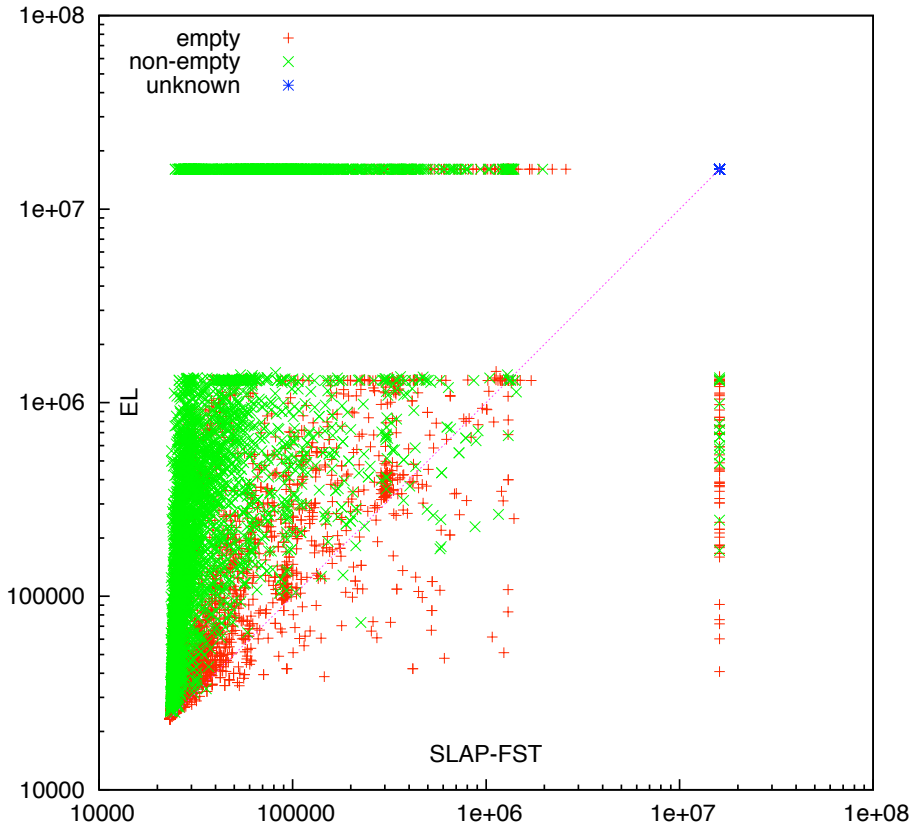




Scatter Plots: SLAP-FST vs. Fully Symbolic

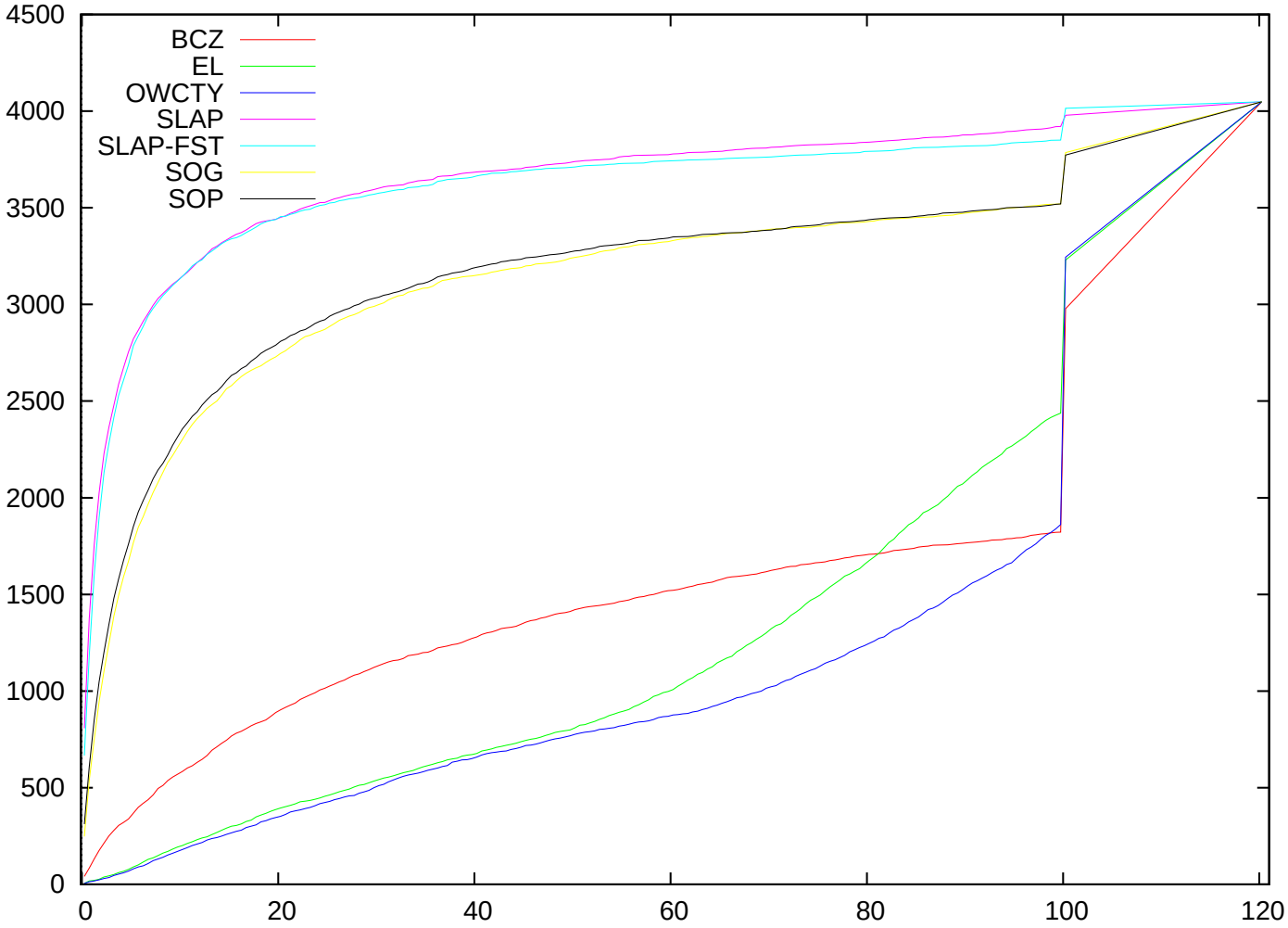
Runtime in seconds. Timeout at 120s. Memory in Kilobytes.
Garbage collection threshold at 1.3 GB



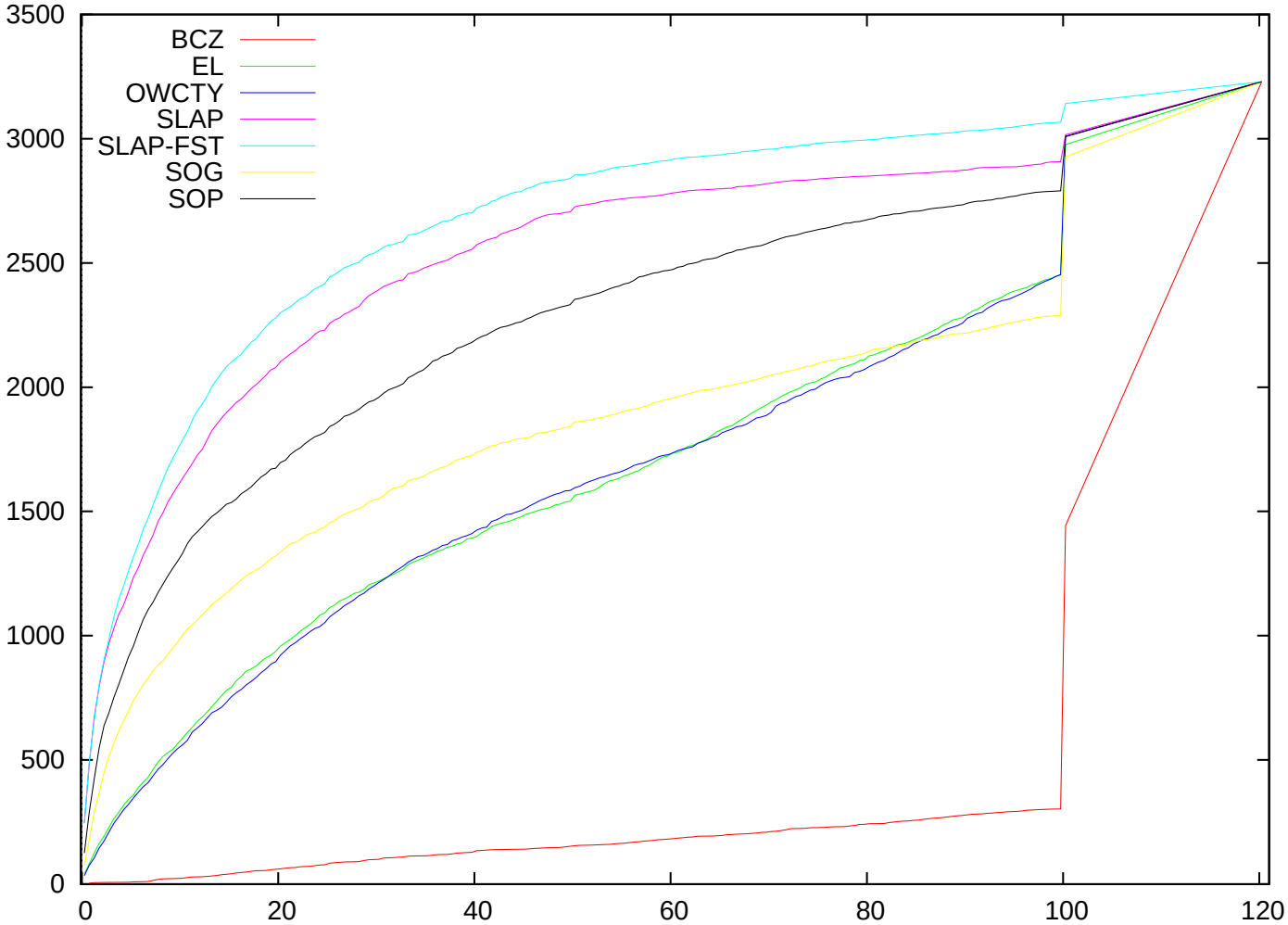


http://move.lip6.fr/software/DDD/ltl_bench.html

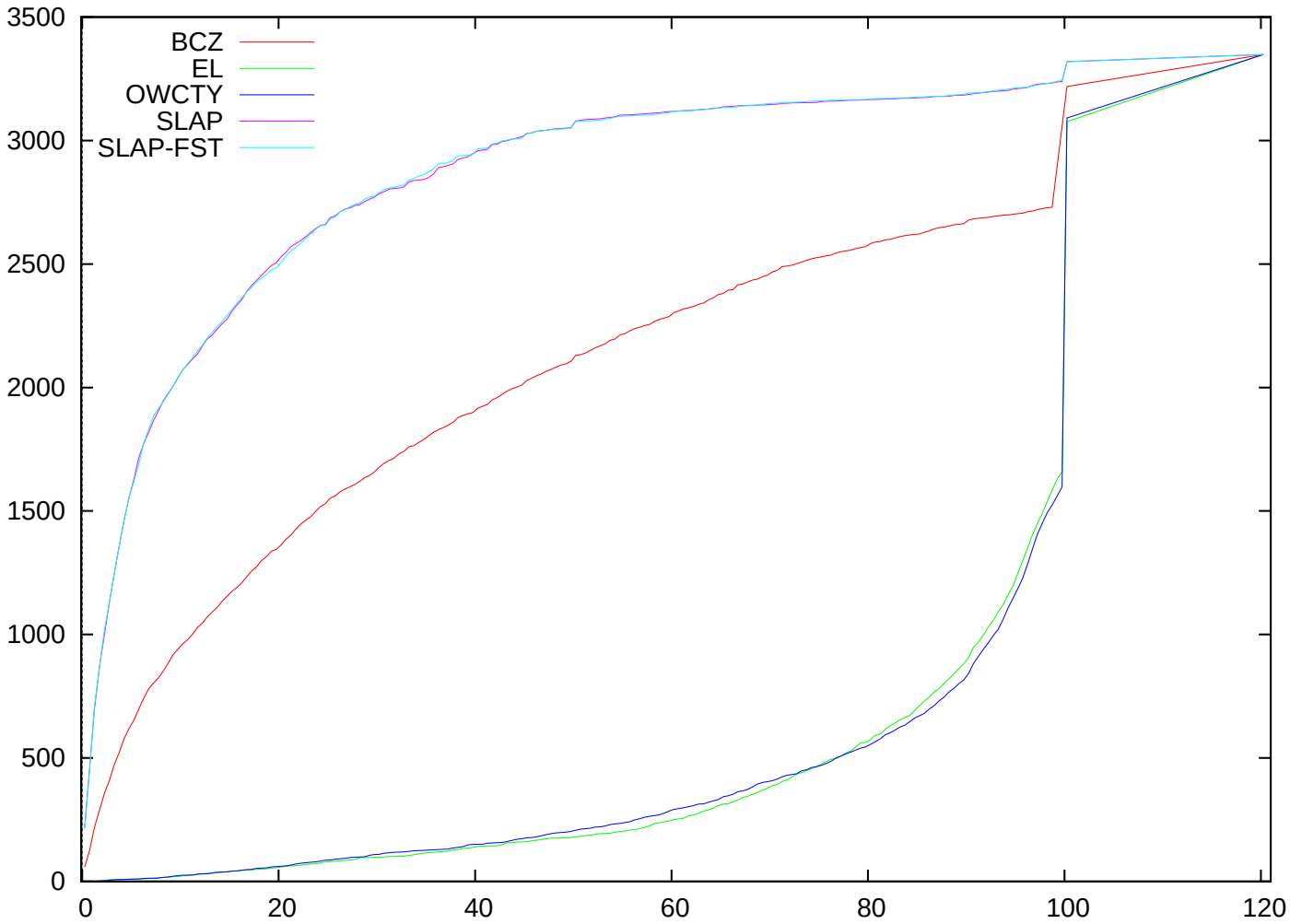
Cumulative Plot: Violated Stuttering Formulas



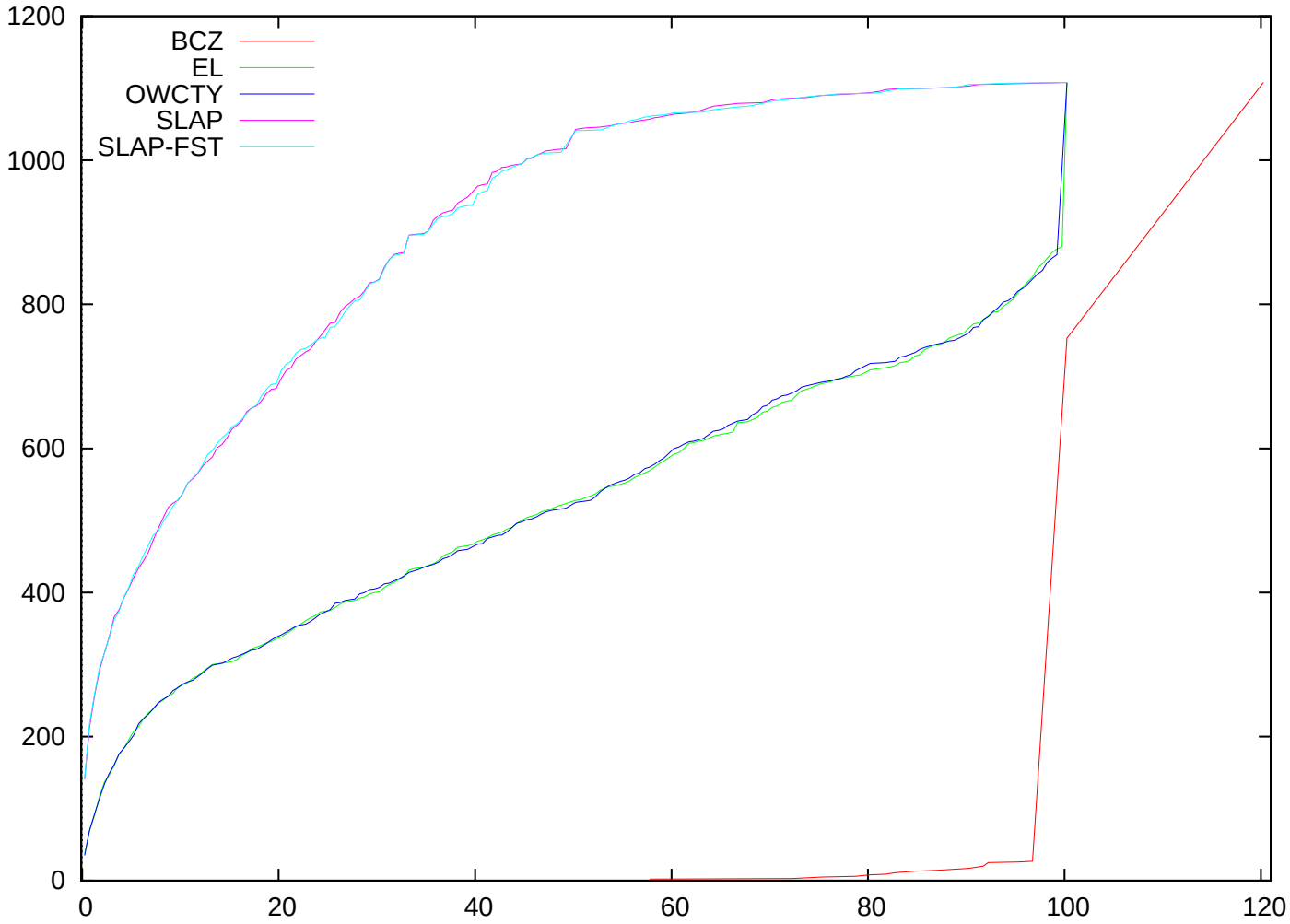
Cumulative Plot: Verified Stuttering Formulas



Cumulative Plot: Violated Non Stuttering Formulas



Cumulative Plot: Verified Non Stuttering Formulas



Conclusion and Perspectives

- Two new hybrid approaches: Aggregates states at the product level, not in the Kripke structure
  - SOP
  - SLAP and SLAP-FST
- SLAP-FST most competitive algorithm of those we compared, **on this benchmark**

Conclusion and Perspectives

- Two new hybrid approaches: Aggregates states at the product level, not in the Kripke structure
  - SOP
  - SLAP and SLAP-FST
- SLAP-FST most competitive algorithm of those we compared, **on this benchmark**

- Compare our hybrid approaches to PDP
- Partial order
- Testing automata
- SOPED SOG and SLAPED SOG