Parameterized verification of round-based shared-memory systems

Nicolas Waldburger [1]
Nathalie Bertrand [1], Nicolas Markey [1], Ocan Sankur [1]

[1]Univ Rennes, Inria, CNRS, IRISA, France

Round-based shared-memory algorithms

The distributed systems considered

- **Parallel**, **identical** processes communicating via **shared memory**
- **Asynchrony**: some processes might be faster than others
- **Non-atomic** read & write combinations, no fault
- **Round-based**: There is a fresh copy of registers at each round
- Processes can be at different rounds; they may read to and write from registers of nearby rounds

[1] James Aspnes, Fast deterministic consensus in a noisy environment, *Journal of Algorithms*, 2002.

Round-based shared-memory algorithms

The distributed systems considered

- **Parallel**, **identical** processes communicating via **shared memory**
- **Asynchrony**: some processes might be faster than others
- **Non-atomic** read & write combinations, no fault
- **Round-based**: There is a fresh copy of registers at each round
- Processes can be at different rounds; they may read to and write from registers of nearby rounds

int $k := 0$, bool $p \in \{0,1\}$, $(\mathrm{rg}_b[r])_{b\in\{0,1\},r\in\mathbb{N}}$ all initialized to no;
while true **do**
  read from $\mathrm{rg}_0[k]$ and $\mathrm{rg}_1[k]$;
  if $\mathrm{rg}_0[k]$ = yes and $\mathrm{rg}_1[k]$ = no **then** $p := 0$;
  else if $\mathrm{rg}_0[k]$ = no and $\mathrm{rg}_1[k]$ = yes **then** $p := 1$;
  write yes to $\mathrm{rg}_p[k]$;
  if $k > 0$ **then**
    read from $\mathrm{rg}_{1-p}[k-1]$;
    if $\mathrm{rg}_{1-p}[k-1]$ = no **then** return p;
  $k := k+1$;

read from registers of rounds k and $k-1$

write to registers of round k

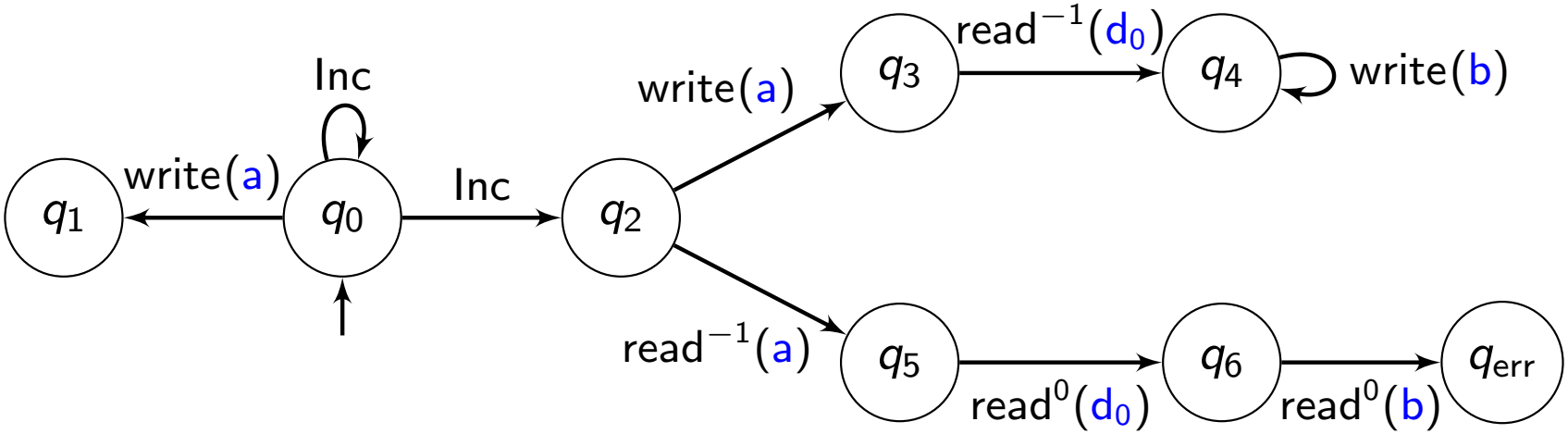
Algorithm 2: Aspnes' consensus algorithm[1].

[1] James Aspnes, Fast deterministic consensus in a noisy environment, *Journal of Algorithms*, 2002.

A model: round-based register protocols

Inspired by models for shared-memory systems without rounds[2][3].

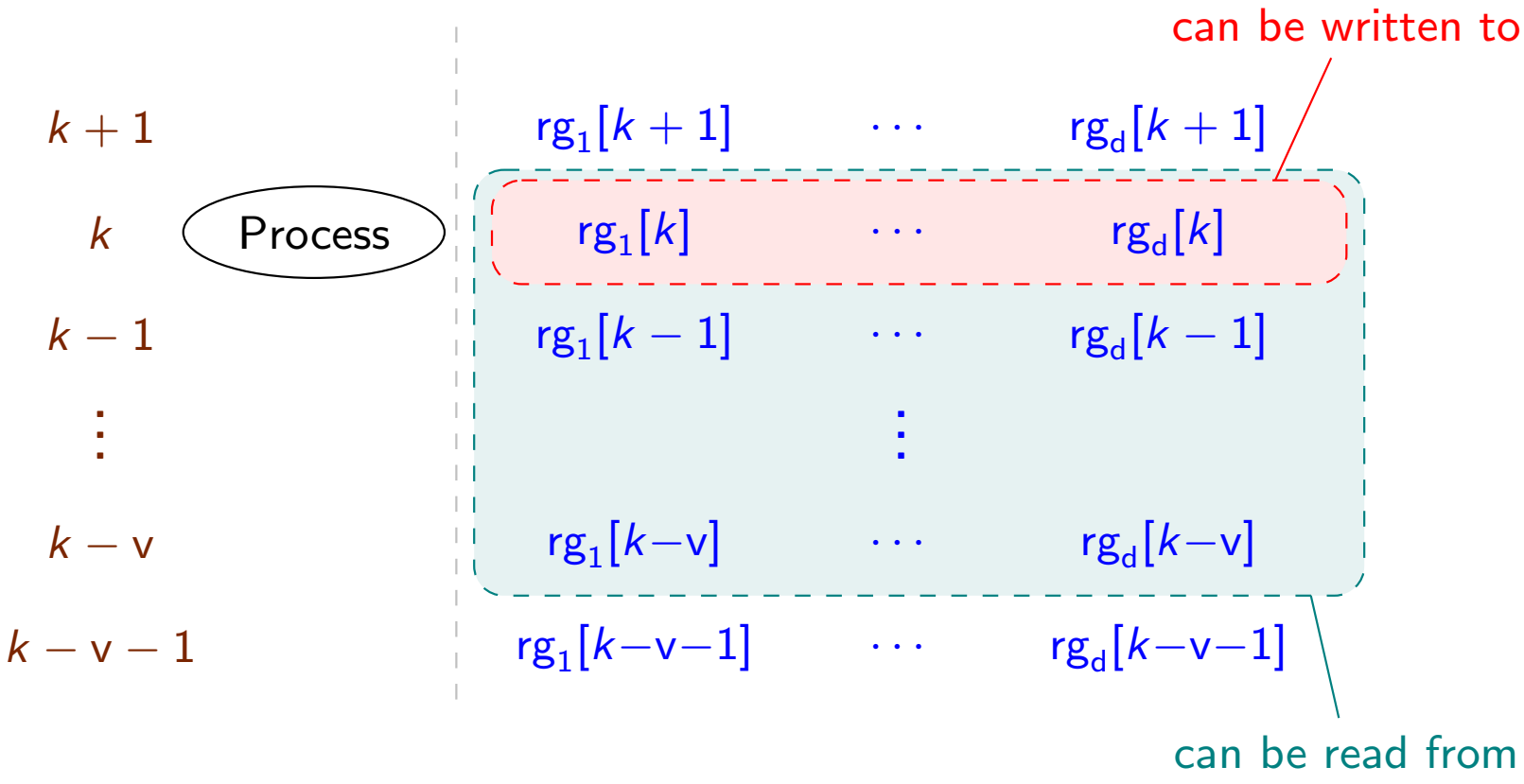
- One model for all processes: a finite automaton
- Transitions are read actions, write actions and round increments
- A fixed number d of registers per round (the total number of registers is hence unbounded)



[2] Javier Esparza, Pierre Ganty, and Rupak Majumdar. Parameterized verification of asynchronous shared-memory systems. *CAV'13*

[3] Patricia Bouyer, Nicolas Markey, Mickael Randour, Arnaud Sangnier, and Daniel Stan. Reachability in networks of register protocols under stochastic schedulers. *ICALP'16*

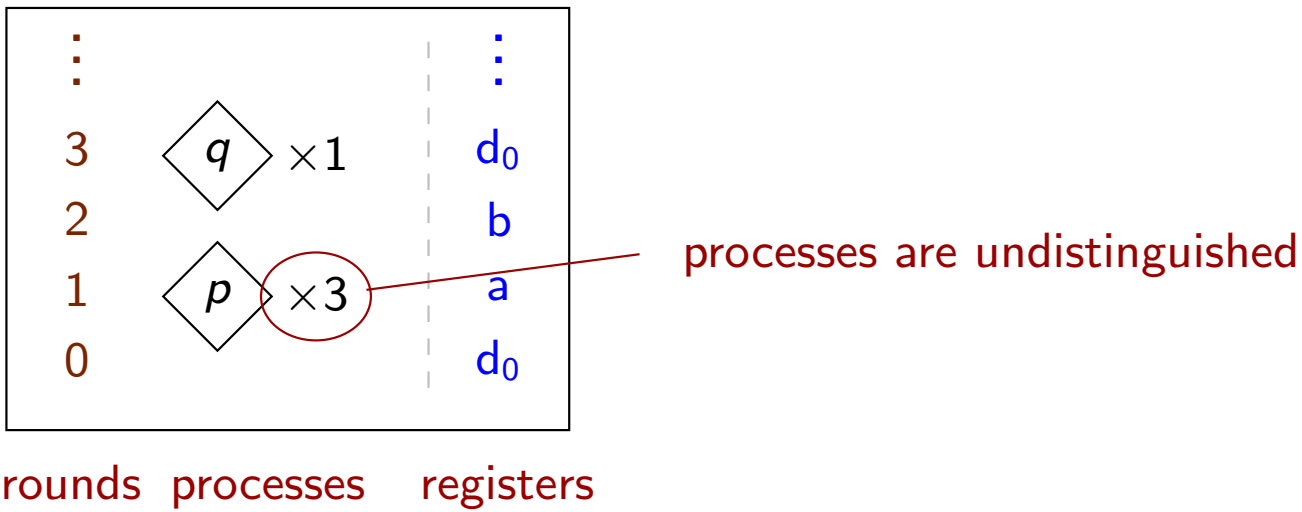
A limited visilibity range



v given in unary (in Aspnes' consensus algorithm, $v = 1$)

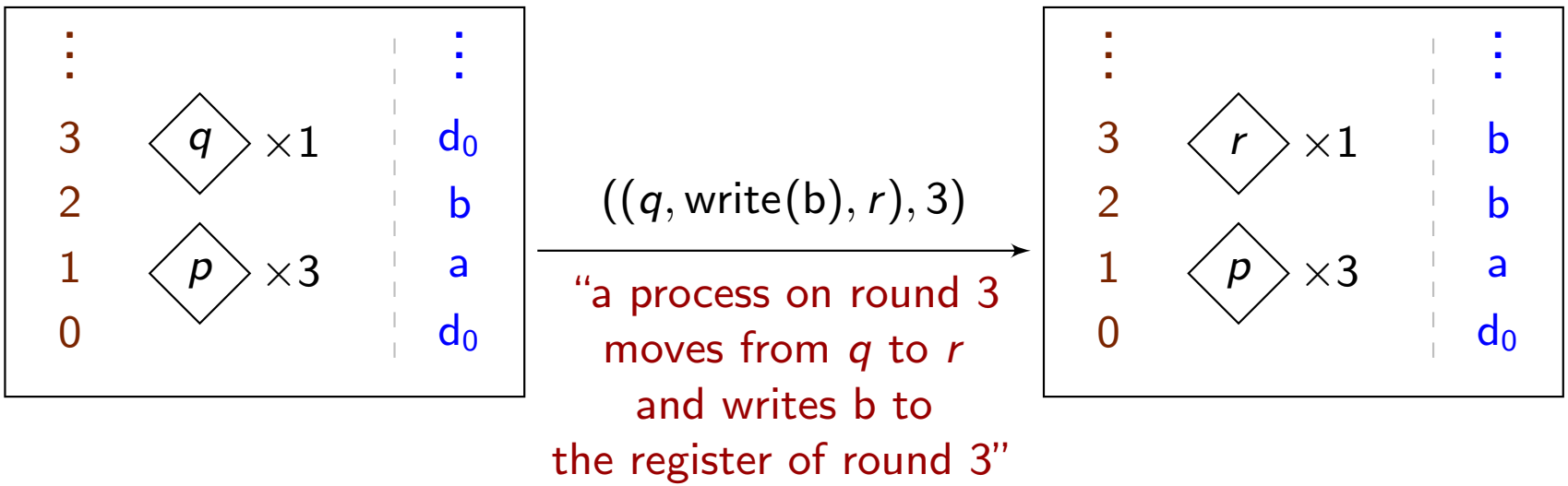
Semantics of the model

From now on, let d = 1: one register per round.



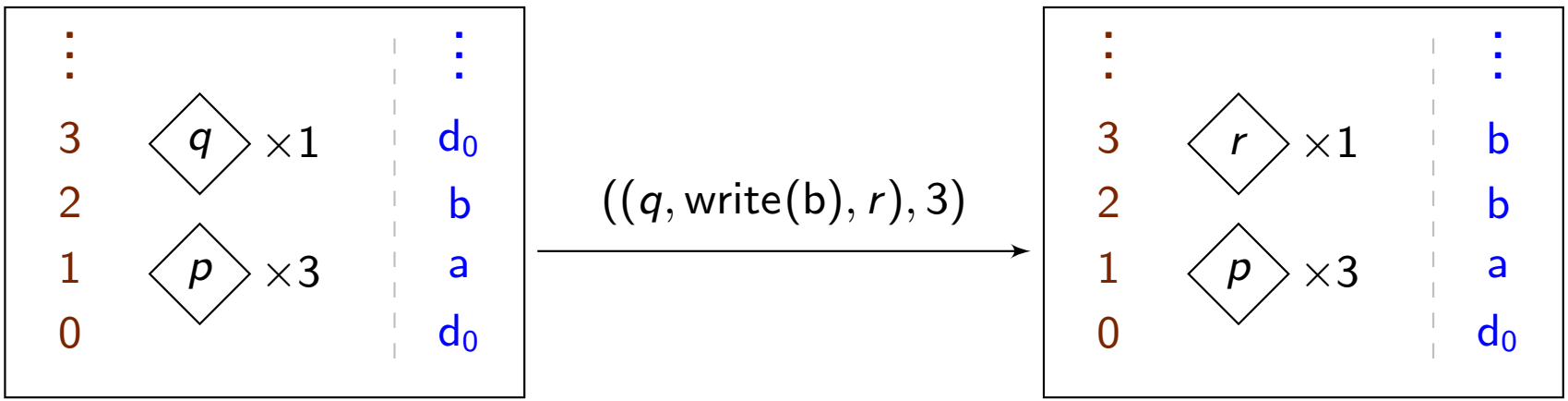
Semantics of the model

From now on, let d = 1: one register per round.

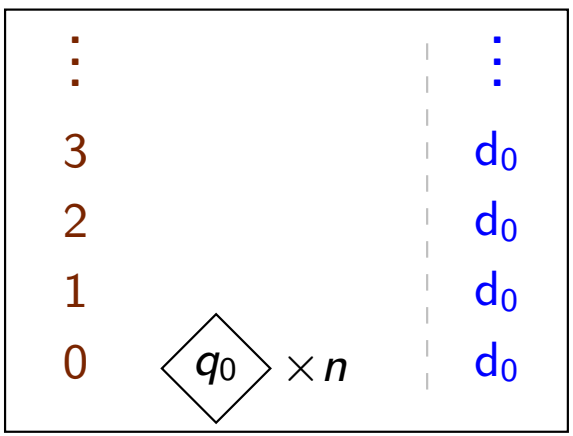


Semantics of the model

From now on, let d = 1: one register per round.



Initial configuration of size n:



The safety problem

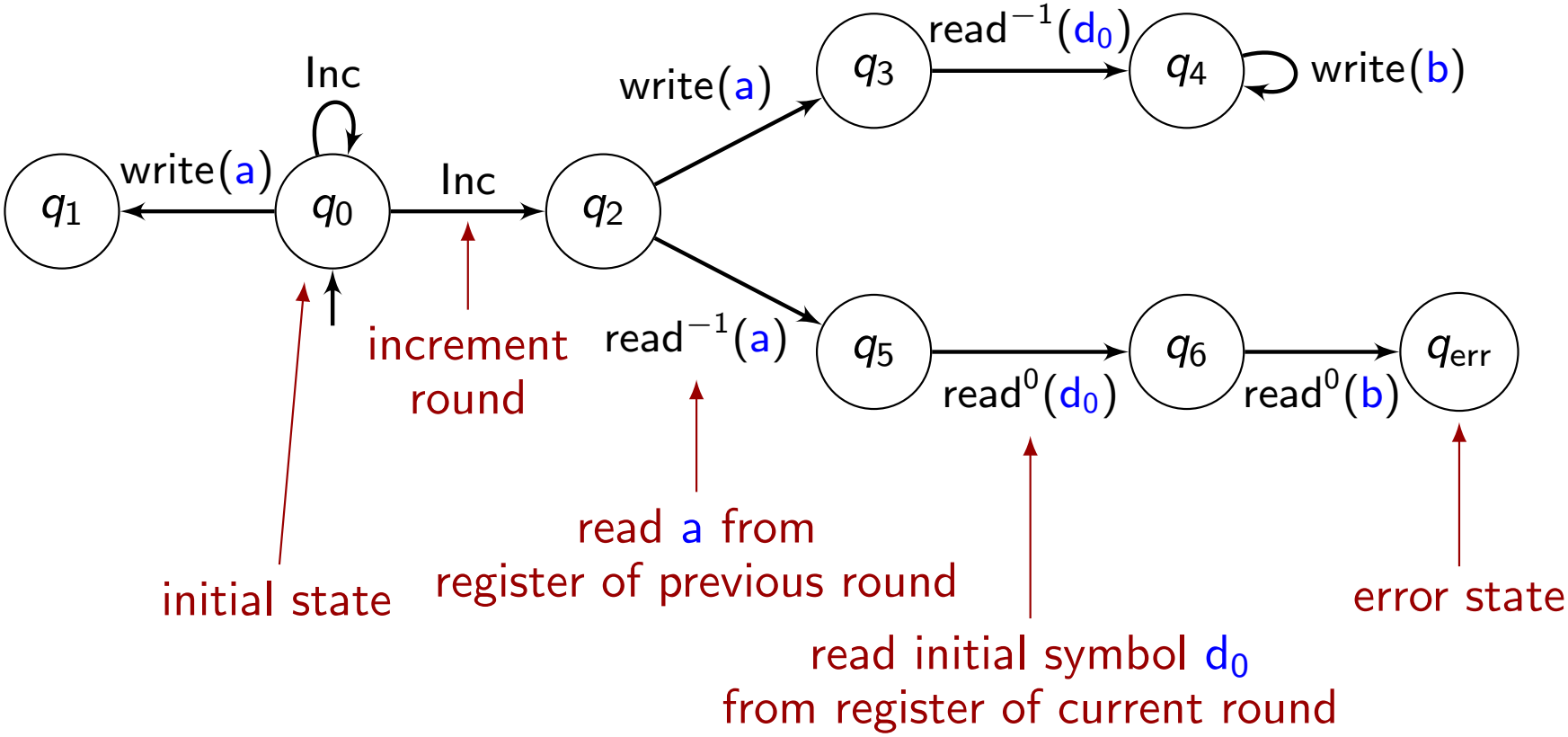
The (parameterized) safety problem

Is it true that, for all numbers of processes n and all executions from the initial configuration of size n, an error state q_{err} is avoided?

If the error state cannot be covered, the system is safe.

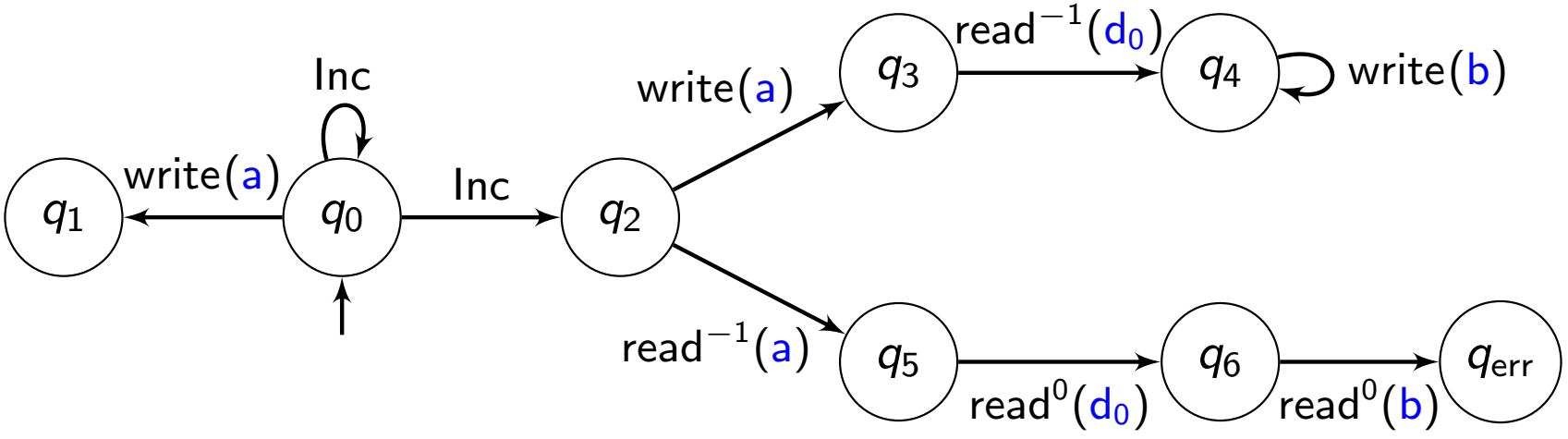
Dual problem: look for an execution *covering* the error.

A small example

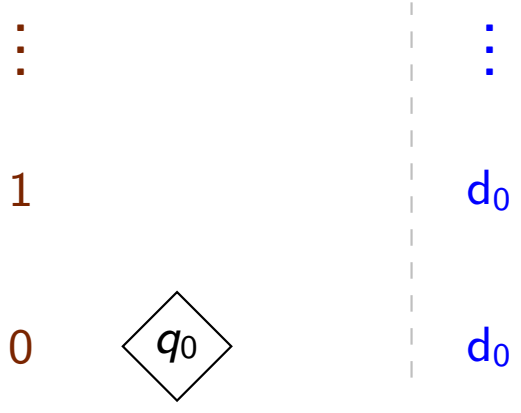


d = 1 (one register per round)
v = 1 (processes can read one round back)

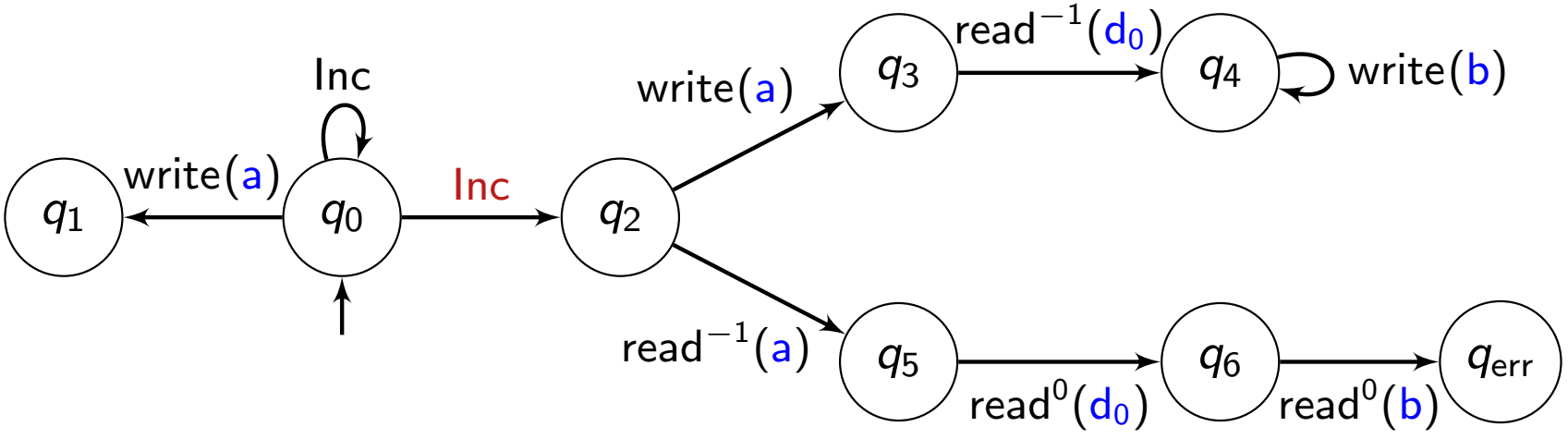
A small example



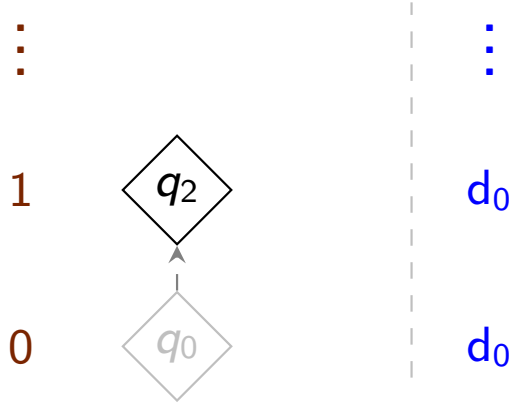
State q_4 can be covered from the initial configuration with one process:



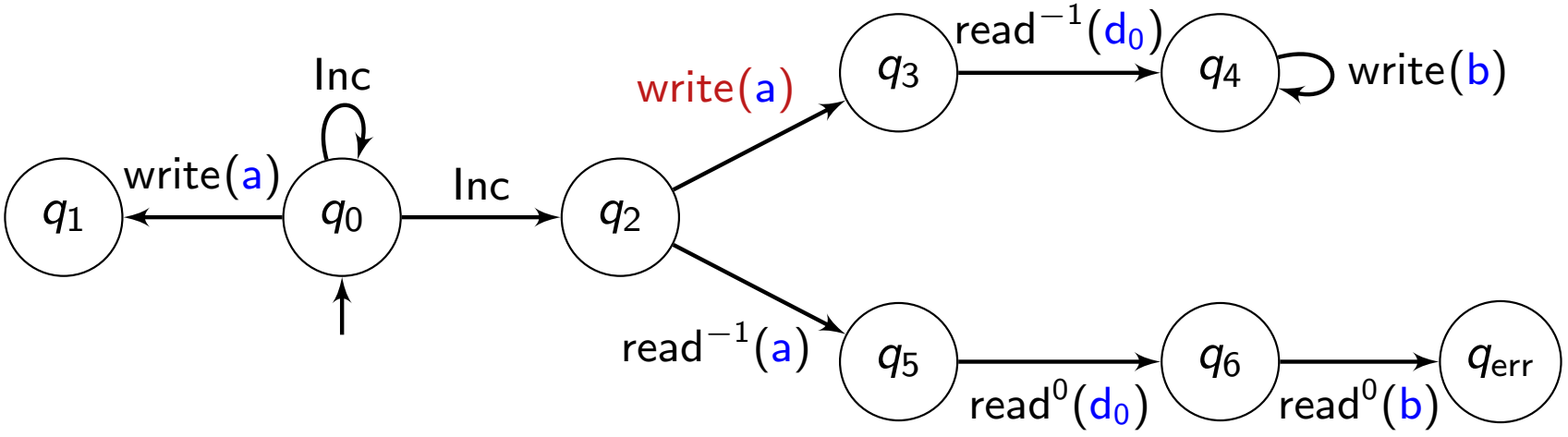
A small example



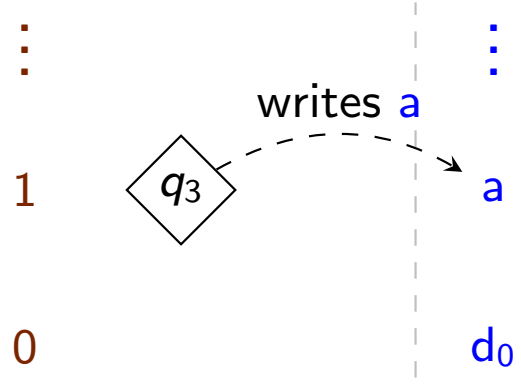
State q_4 can be covered from the initial configuration with one process:



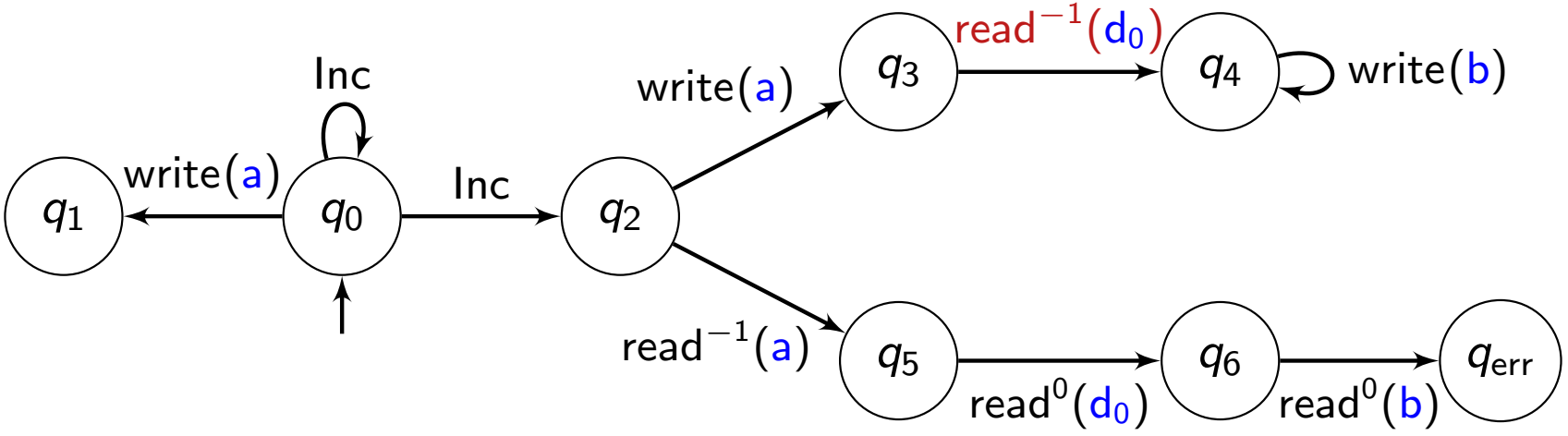
A small example



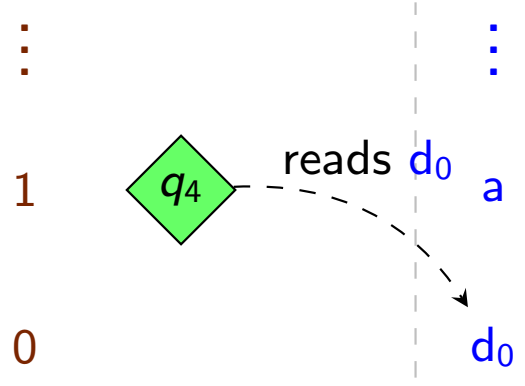
State q_4 can be covered from the initial configuration with one process:



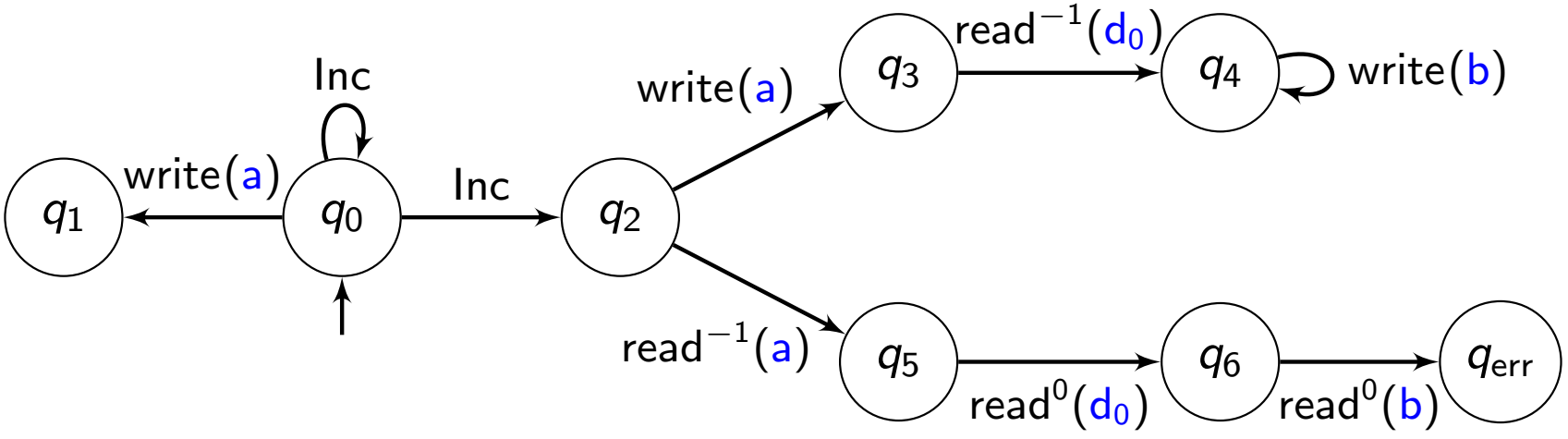
A small example



State q_4 can be covered from the initial configuration with one process:

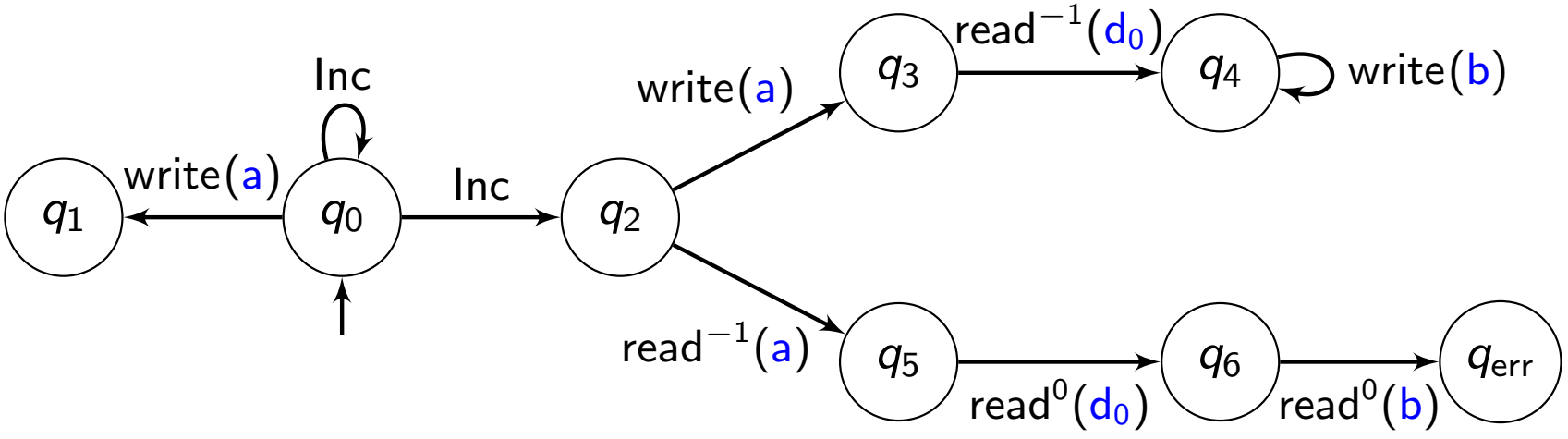


A small example

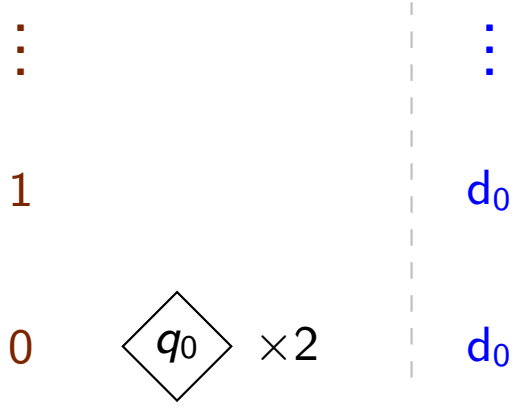


State q_6 can be covered from the initial configuration with two processes:

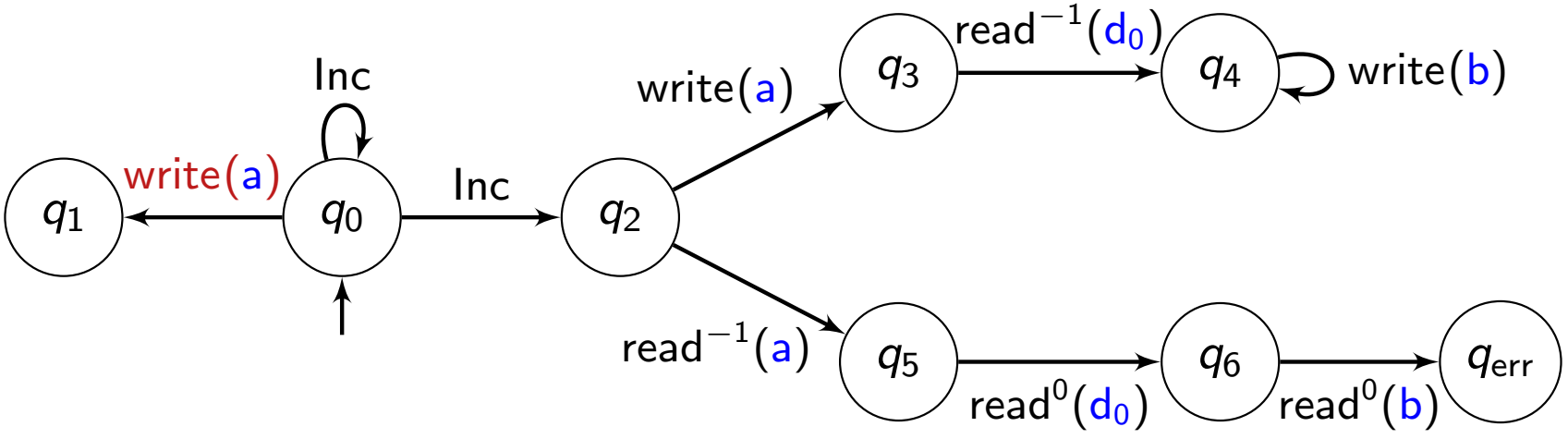
A small example



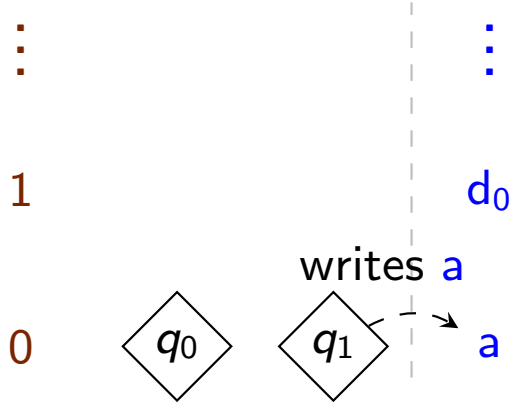
State q_6 can be covered from the initial configuration with two processes:



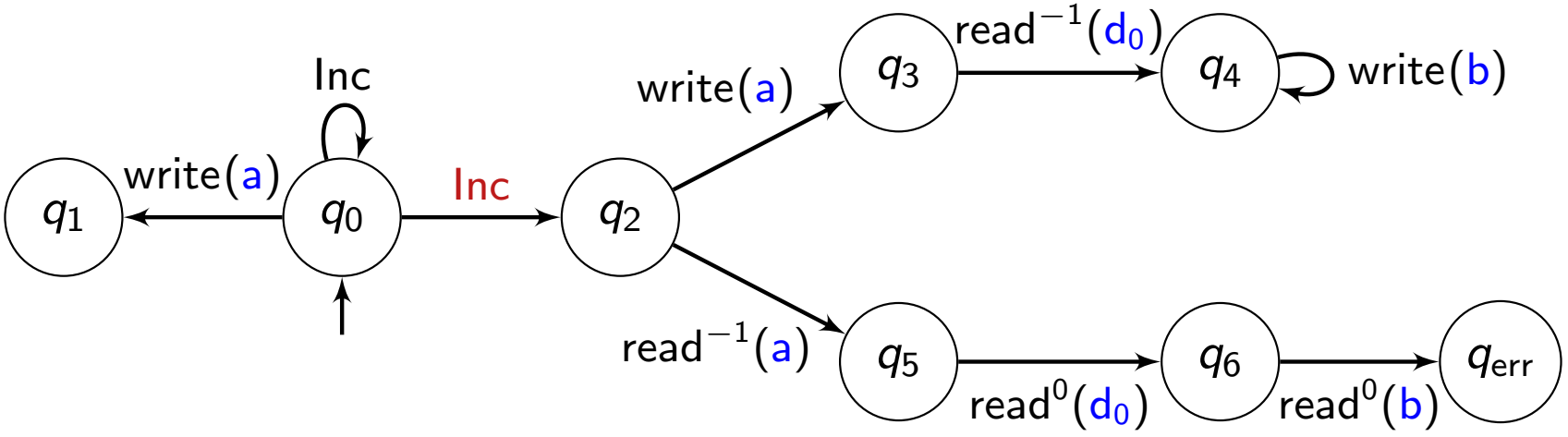
A small example



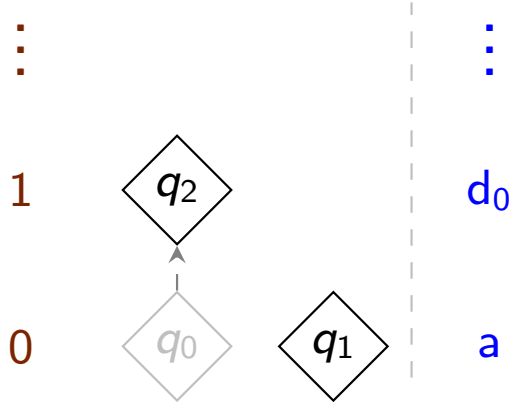
State q_6 can be covered from the initial configuration with two processes:



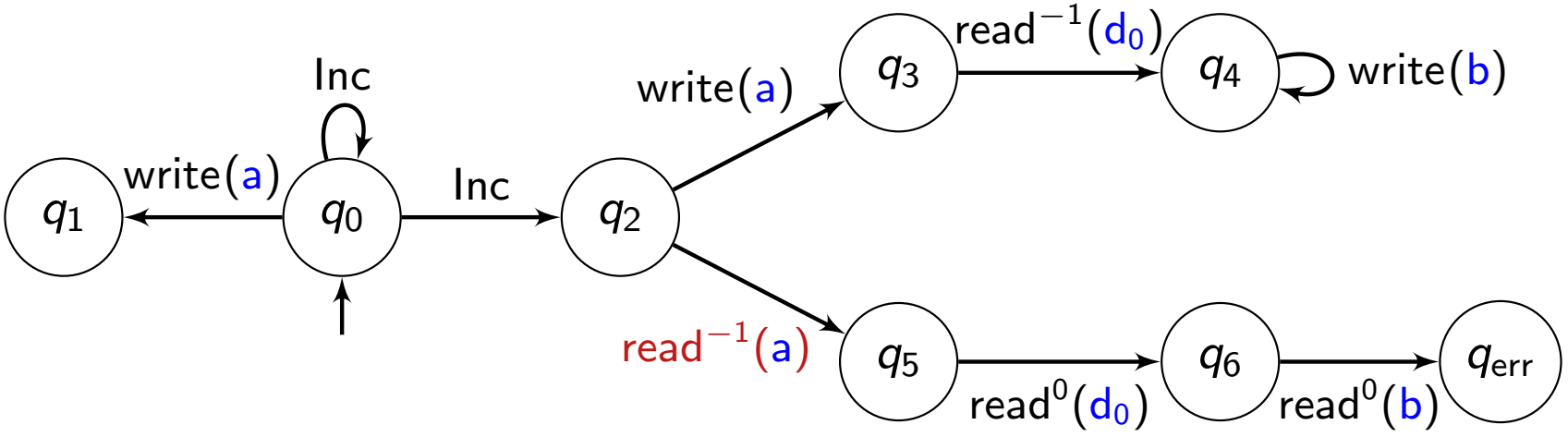
A small example



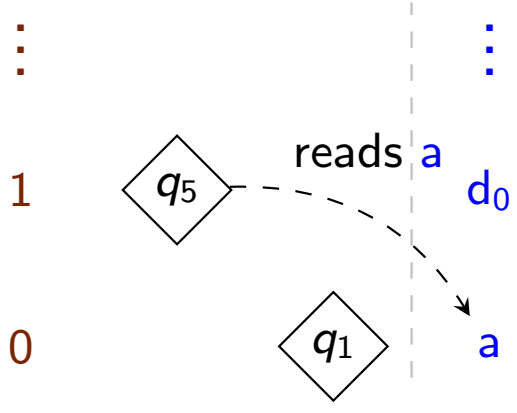
State q_6 can be covered from the initial configuration with two processes:



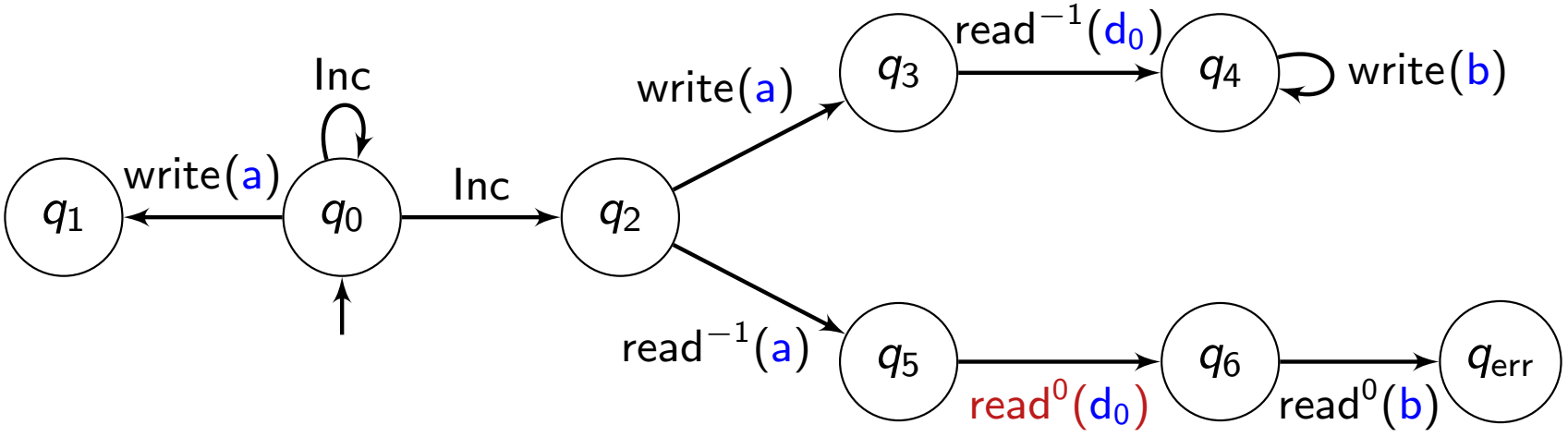
A small example



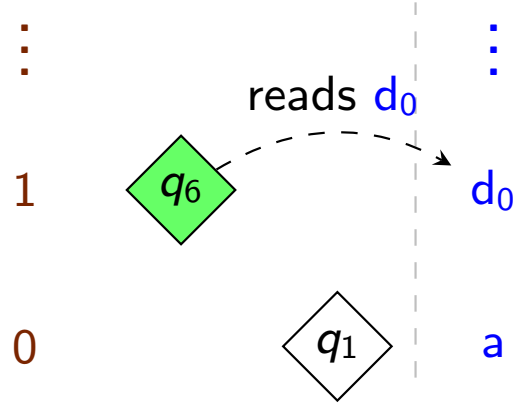
State q_6 can be covered from the initial configuration with two processes:



A small example



State q_6 can be covered from the initial configuration with two processes:



Theorem

Parameterized safety in round-based register protocols is PSPACE-complete.[4]

[4] Nathalie Bertrand, Nicolas Markey, Ocan Sankur, Nicolas Waldburger. Parameterized safety verification of round-based shared-memory systems. *ICALP'22*

Theorem

Parameterized safety in round-based register protocols is PSPACE-complete.[4]

Ingredients of the polynomial-space algorithm

- Copycat property (thanks to non-atomicity)
- Thanks to copycat, define an abstraction where one only remembers which pairs (state,round) are populated by at least one process
- Exploit limited visibility range: reads and writes are local with respect to the round
- Rely on a sliding window along the rounds

[4] Nathalie Bertrand, Nicolas Markey, Ocan Sankur, Nicolas Waldburger. Parameterized safety verification of round-based shared-memory systems. *ICALP'22*

Theorem

Parameterized safety in round-based register protocols is PSPACE-complete.[4]

Ingredients of the polynomial-space algorithm

- Copycat property (thanks to non-atomicity)
- Thanks to copycat, define an abstraction where one only remembers which pairs (state,round) are populated by at least one process
- Exploit limited visibility range: reads and writes are local with respect to the round
- Rely on a sliding window along the rounds

This also allows to prove the following (tight) exponential upper bounds:

Exponential upper bounds

There exists an exponential upper bound on the number of processes and on the number of rounds needed to reach the error state.

[4] Nathalie Bertrand, Nicolas Markey, Ocan Sankur, Nicolas Waldburger. Parameterized safety verification of round-based shared-memory systems. *ICALP'22*

Conclusion

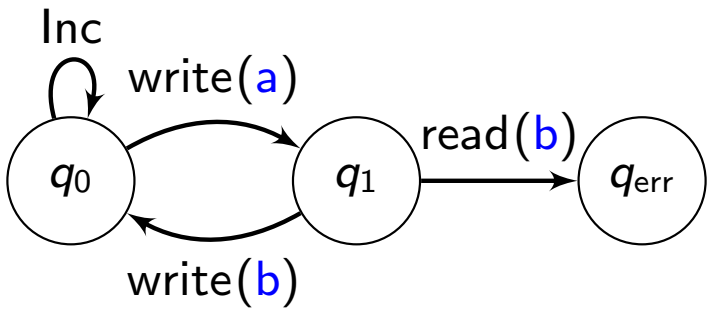
Summary

- Round-based register protocols are a model for round-based shared-memory algorithms such as Aspnes' consensus algorithm
- The verification problem of parameterized safety is PSPACE-complete

Future work

- Other problems on our model: parameterized TARGET, parameterized INEVITABILITY
- Almost-sure reachability in round-based register protocols with stochastic schedulers (termination of Aspnes' algorithm)
- Links with classical notions of fairness

Classical notions of fairness are not satisfactory



q_{err} is reached with probability 1 with a stochastic scheduler with two processes.

Consider the execution with two processes where one process goes to q_1 and back to q_0 on every round, while the other process stays on q_0 forever.

This execution is fair with respect to:

- Fairness on moves: no move is available infinitely often because k increases
- Fairness on transitions: transition from q_1 to q_{err} is never enabled.