On the Verification Problem for Weak Memory Models

M.F. Atig[1] A. Bouajjani[1] S. Burckhardt[2] M. Musuvathi[2]

[1]LIAFA, CNRS & University of Paris 7 (Paris Diderot)

[2]Microsoft Research, Redmond

ACTS II, Chennai, February 3, 2010

[POPL'10]

Concurrent Programs

- Parallel processes with shared memory
- Interleaving (Sequentially Consistent) semantics:
  - Computations of different processes are shuffled
  - Program order is preserved for each process.

Relaxing the program order

$$\begin{array}{lcl} P_1 & : & \mathrm{write}(x,1) \quad ; \quad \mathrm{read}(y,0) \\ P_2 & : & \mathrm{read}(x,0) \end{array}$$

A scheduling for SC semantics: 3 steps

$$\begin{array}{lcl} P_1 & : & \mathrm{write}(x,1)_{(2)} \quad ; \quad \mathrm{read}(y,0)_{(3)} \\ P_2 & : & \mathrm{read}(x,0)_{(1)} \end{array}$$

Relaxing the program order

$$\begin{array}{lcl} P_1 & : & \mathrm{write}(x,1) \quad ; \quad \mathrm{read}(y,0) \\ P_2 & : & \mathrm{read}(x,0) \end{array}$$

A scheduling for SC semantics: 3 steps

$$\begin{array}{lcl} P_1 & : & \mathrm{write}(x,1)_{(2)} \quad ; \quad \mathrm{read}(y,0)_{(3)} \\ P_2 & : & \mathrm{read}(x,0)_{(1)} \end{array}$$

Allowing reordering of actions on different variables: 2 steps !

$$\begin{array}{lcl} P_1 & : & \mathrm{read}(y,0)_{(1)} \quad ; \quad \mathrm{write}(x,1)_{(2)} \\ P_2 & : & \mathrm{read}(x,0)_{(1)} \end{array}$$

$\Rightarrow$ *Weak Memory Models*

Program Order Relaxations: Classification

- $W \rightarrow R$: Write to Read

  write (x) ; read (y) $\mapsto$ read (y) ; write (x)

  $\Rightarrow$ TSO model (Total Store Ordering)

- (+) $W \rightarrow W$: Write to Write

  $\Rightarrow$ PSO model (Partial Store Ordering)

- (+) $R \rightarrow R/W$: Read to Read/Write

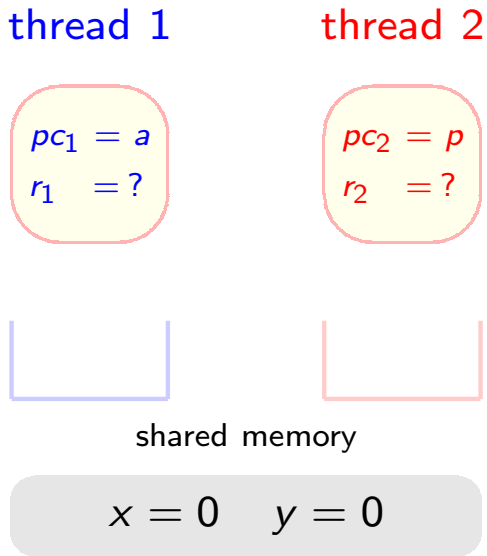
  $\Rightarrow$ $\sim$RMO model (Relaxed Memory Ordering)

Relaxation ⇒ Potential Bad Behaviors

$x = y = 0$

thread 1	thread 2
a: $y = 1$	p: $x = 1$
b: $r_1 = x$	q: $r_2 = y$
c: if($r_1 == 0$) {	s: if($r_2 == 0$) {
d: ...	t: ...
}	}

1- Initial state



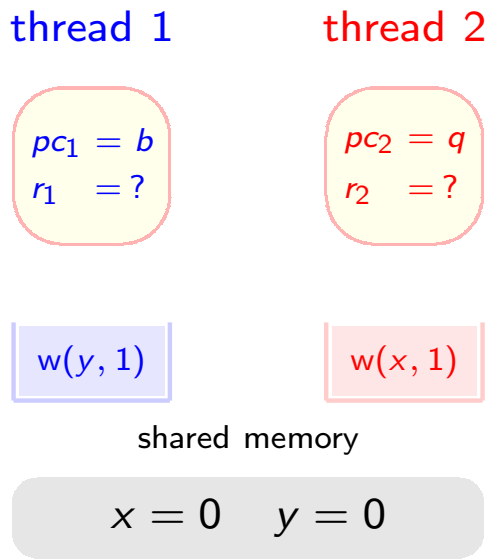
Dekker's mutual exclusion protocol. Fails under Write to Read relaxation.

Relaxation ⇒ Potential Bad Behaviors

$x = y = 0$

thread 1	thread 2
a: $y = 1$	p: $x = 1$
b: $r_1 = x$	q: $r_2 = y$
c: if($r_1 == 0$) {	s: if($r_2 == 0$) {
d: ...	t: ...
}	}

2- Writes are postponed



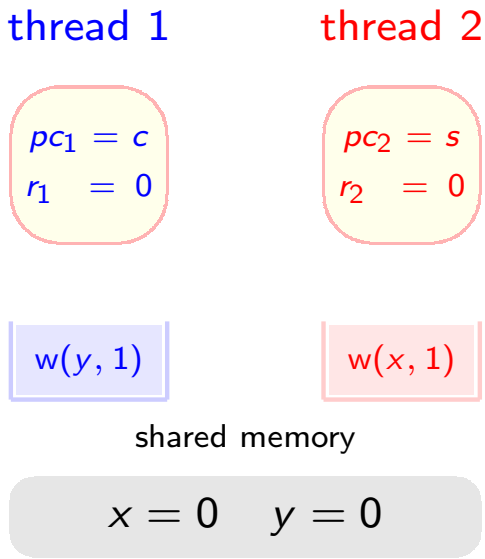
Dekker's mutual exclusion protocol. Fails under Write to Read relaxation.

Relaxation ⇒ Potential Bad Behaviors

$x = y = 0$

thread 1	thread 2
a: $y = 1$	p: $x = 1$
b: $r_1 = x$	q: $r_2 = y$
c: if($r_1 == 0$) {	s: if($r_2 == 0$) {
d: ...	t: ...
}	}

3- Reading from memory



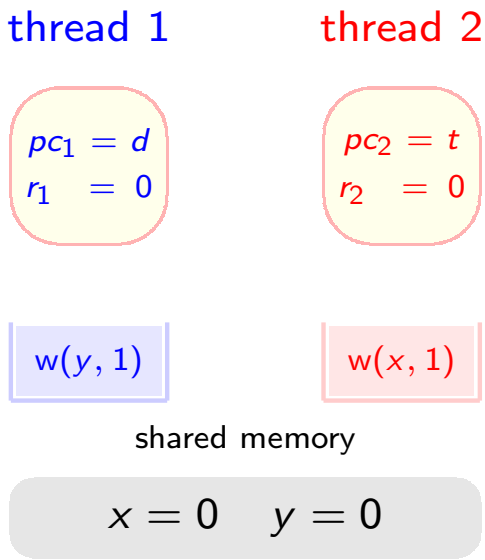
Dekker's mutual exclusion protocol. Fails under Write to Read relaxation.

Relaxation $\Rightarrow$ Potential Bad Behaviors

$x = y = 0$

thread 1	thread 2
a: $y = 1$	p: $x = 1$
b: $r_1 = x$	q: $r_2 = y$
c: if($r_1 == 0$) {	s: if($r_2 == 0$) {
d: ...	t: ...
}	}

4- Accessing critical sections



Dekker's mutual exclusion protocol. Fails under Write to Read relaxation.

Verification Problems

For a memory model μ, a program P, and a (control + memory) state s

- State Reachability Problem (Safety)

  s is reachable in P ?

- Repeated State Reachability Problem (Liveness)

  s is reachable infinitely often in a computation of P ?

Verification Problems

For a memory model μ, a program P, and a (control + memory) state s

- State Reachability Problem (Safety)

  s is reachable in P ?

- Repeated State Reachability Problem (Liveness)

  s is reachable infinitely often in a computation of P ?

Decidability / Complexity ?

Each process is finite-state

- For the SC memory model both problems are PSPACE-complete

Verification Problems

For a memory model μ, a program P, and a (control + memory) state s

- State Reachability Problem (Safety)

  s is reachable in P ?

- Repeated State Reachability Problem (Liveness)

  s is reachable infinitely often in a computation of P ?

Decidability / Complexity ?

Each process is finite-state

- For the SC memory model both problems are PSPACE-complete
- Nontrivial for weak memory models:

  $Paths_{\mu}(P) = Closure_{\mu}(Paths_{SC}(P))$ *is nonregular*

Results

- The state reachability problem is decidable for

  $$W \rightarrow R \text{ (TSO) } \textit{and } W \rightarrow R/W \text{ (PSO)}$$

Results

- The state reachability problem is decidable for
  $$W \rightarrow R \text{ (TSO) } \textit{and } W \rightarrow R/W \text{ (PSO)}$$
- ... but highly complex: Nonprimitive recursive

Results

- The state reachability problem is decidable for
  $$W \rightarrow R \text{ (TSO) } \textit{and } W \rightarrow R/W \text{ (PSO)}$$
- ... but highly complex: Nonprimitive recursive
- The state reachability problem is undecidable for
  $$W \rightarrow R(/W) + R \rightarrow R/W$$

Results

- The state reachability problem is decidable for
  $$W \rightarrow R \text{ (TSO) } \textit{and } W \rightarrow R/W \text{ (PSO)}$$
- ... but highly complex: Nonprimitive recursive
- The state reachability problem is undecidable for
  $$W \rightarrow R(/W) + R \rightarrow R/W$$
- ... but decidable if the number of overtaken reads is always bounded

Results

- The state reachability problem is decidable for
  $$W \rightarrow R \text{ (TSO) } \textit{and } W \rightarrow R/W \text{ (PSO)}$$
- ... but highly complex: Nonprimitive recursive
- The state reachability problem is undecidable for
  $$W \rightarrow R(/W) + R \rightarrow R/W$$
- ... but decidable if the number of overtaken reads is always bounded
- The repeated state reachability problem is undecidable for
  $$W \rightarrow R(/W), \textit{ and } W \rightarrow R/W + R \rightarrow R/W$$

The rest of the talk

- Operational model for TSO

  FSM + unbounded FIFO buffers

The rest of the talk

- Operational model for TSO

  FSM + unbounded FIFO buffers

- Decidability/Complexity:

  Simulations by/of Lossy Channel Systems

The rest of the talk

- Operational model for TSO

  FSM + unbounded FIFO buffers

- Decidability/Complexity:

  Simulations by/of Lossy Channel Systems

- Undecidability for W $\rightarrow$ R/W + R $\rightarrow$ R/W:

  Reduction of the Post Correspondence Problem.

An operational model for TSO

- Finite number of shared variables $\{x, y, x_1 ...\}$
- Finite data domain $\{d, d_1, d_2, ...\}$
- Finite number of finite-control processes $P_1, \ldots, P_n$ with operations:

$$Nop, Write(x, d), Read(x, d), AtomicRW(x, d_1, d_2)$$

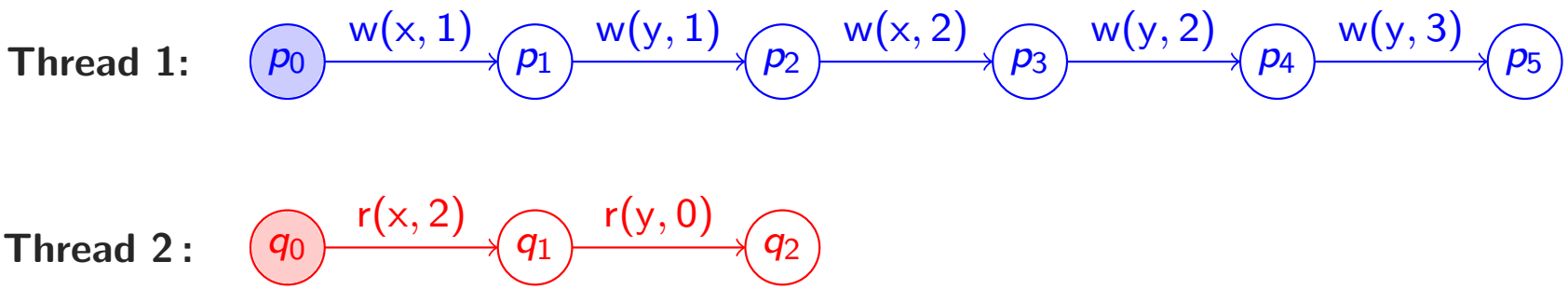
An operational model for TSO

- Finite number of shared variables $\{x, y, x_1 ...\}$
- Finite data domain $\{d, d_1, d_2, ...\}$
- Finite number of finite-control processes $P_1, \ldots, P_n$ with operations:

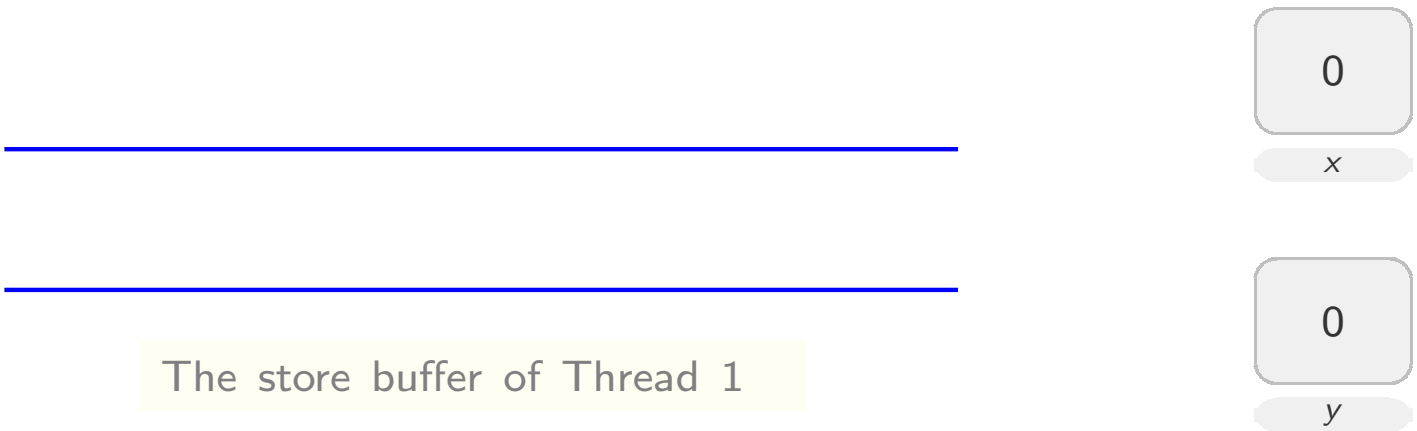
$$Nop, Write(x, d), Read(x, d), AtomicRW(x, d_1, d_2)$$

- Each process has a FIFO buffer
- Configuration = control states + memory state + buffers contents
- Write(x,d) is sent to the buffer
- Memory update = execution of a Write taken from some buffer
- Read(x,d) is executed either if
  - The last Write to x in the buffer is Write(x,d) (Read Own Write)
  - The buffer does not contain a Write to x, and $Memory(x) = d$
- $AtomicRW(x, d_1, d_2)$ requires that the buffer is empty ($\sim$ fence)

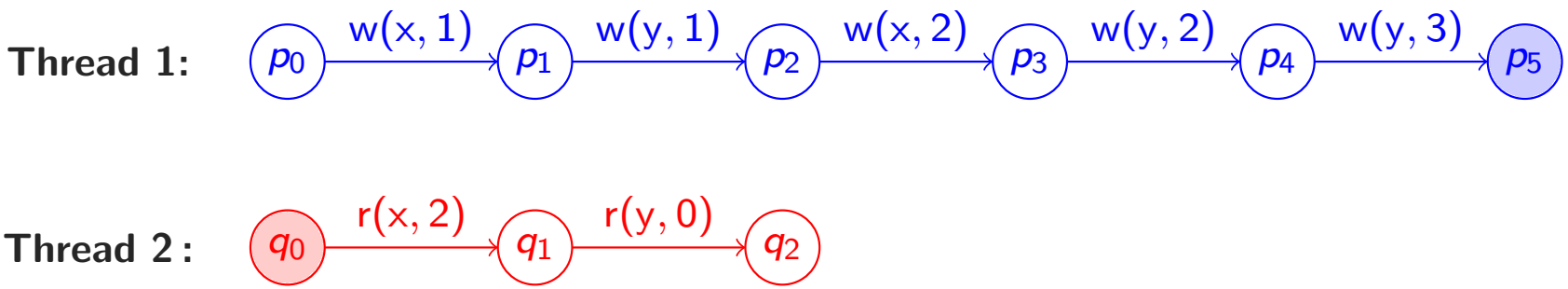
From W $\longrightarrow$ R systems to Lossy Channel Systems



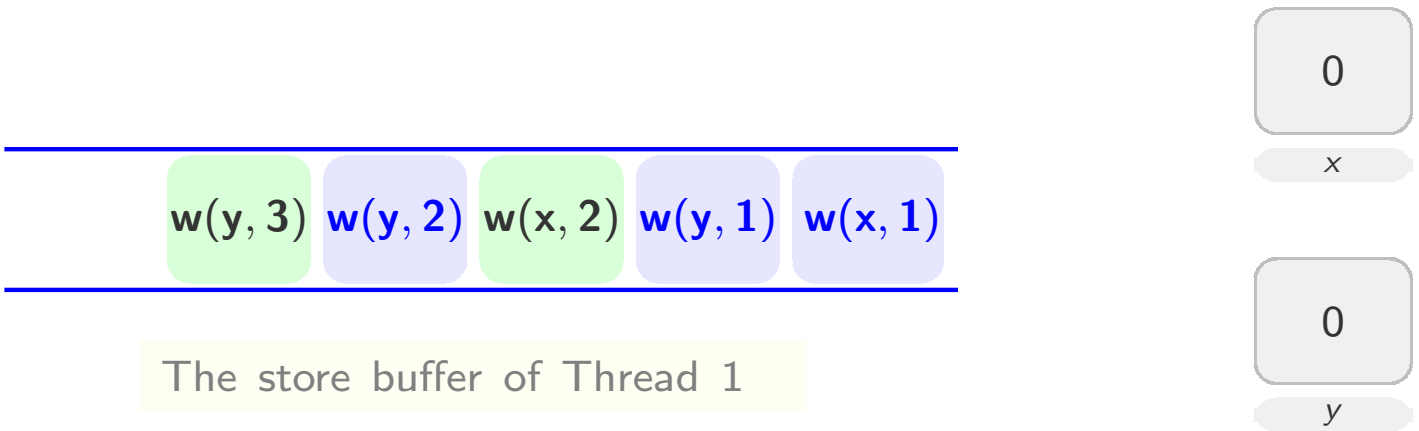
Model: The store buffers are considered as perfect FIFO channels



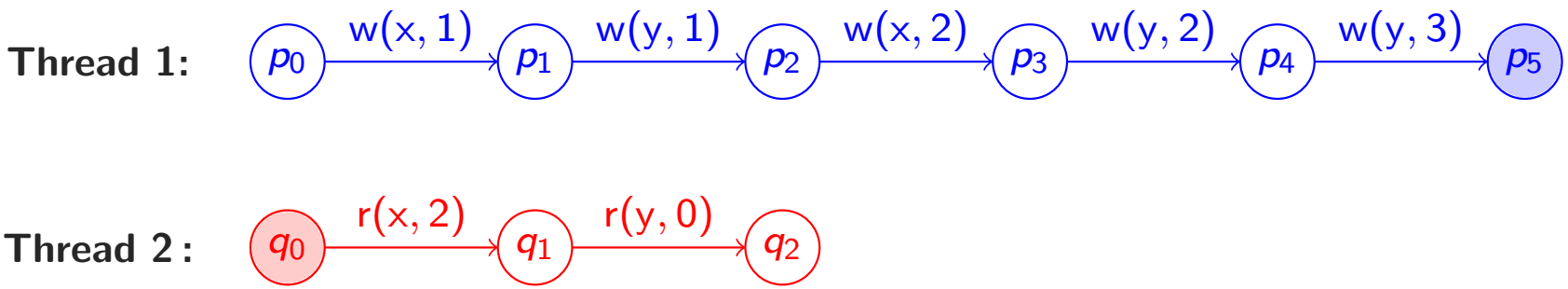
From W $\longrightarrow$ R systems to Lossy Channel Systems



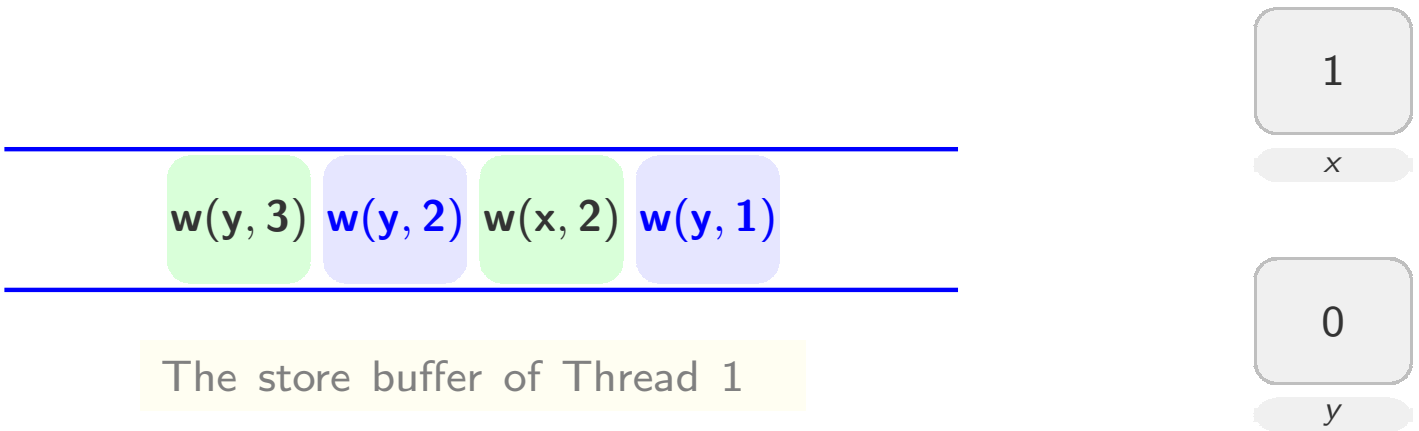
Model: The store buffers are considered as perfect FIFO channels



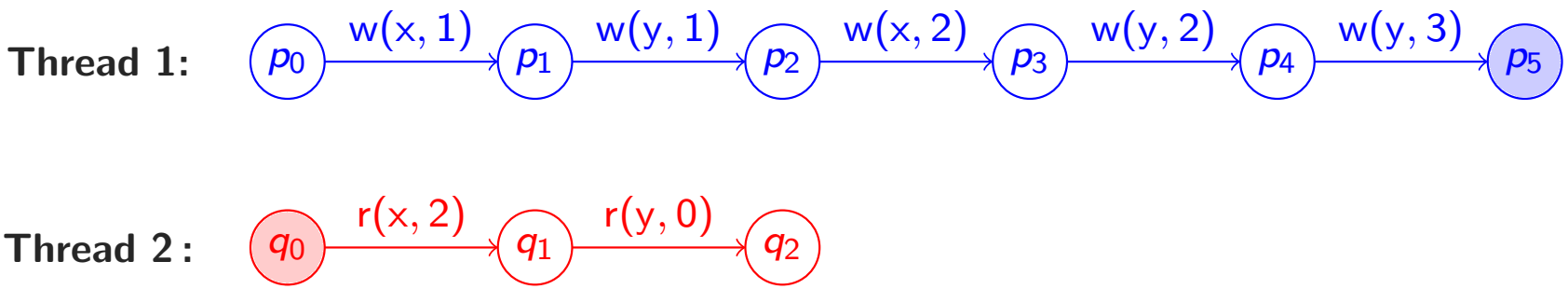
From W $\longrightarrow$ R systems to Lossy Channel Systems



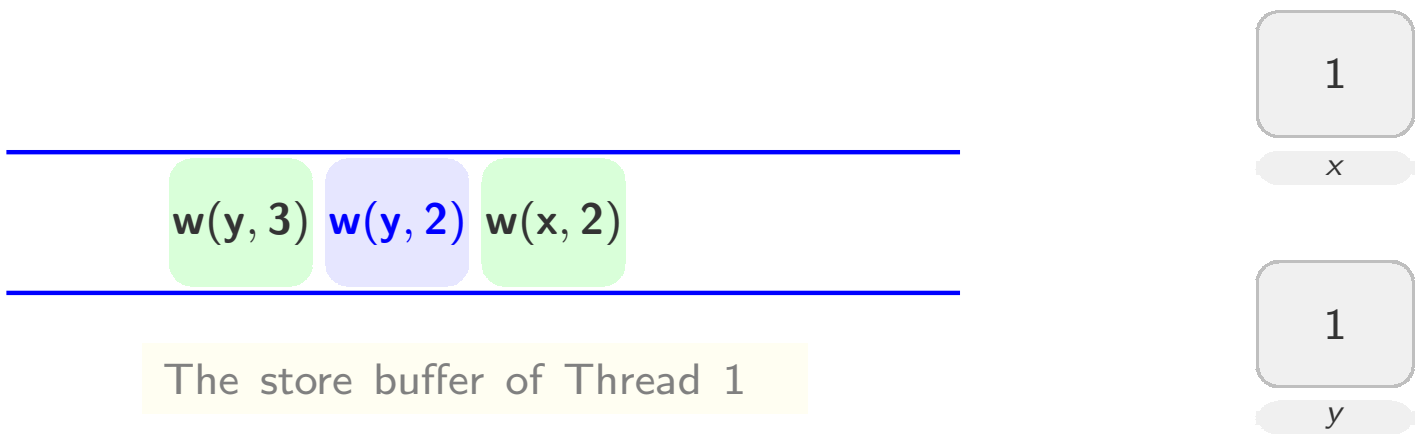
Model: The store buffers are considered as perfect FIFO channels



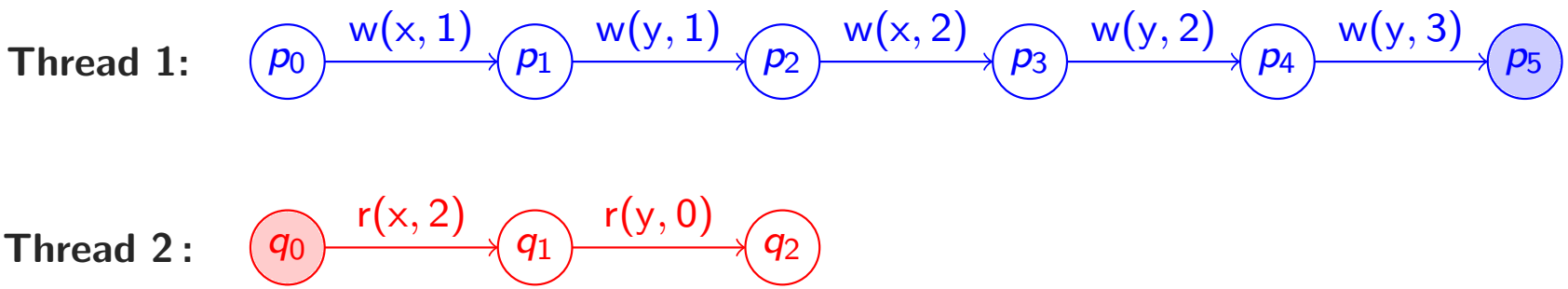
From W $\longrightarrow$ R systems to Lossy Channel Systems



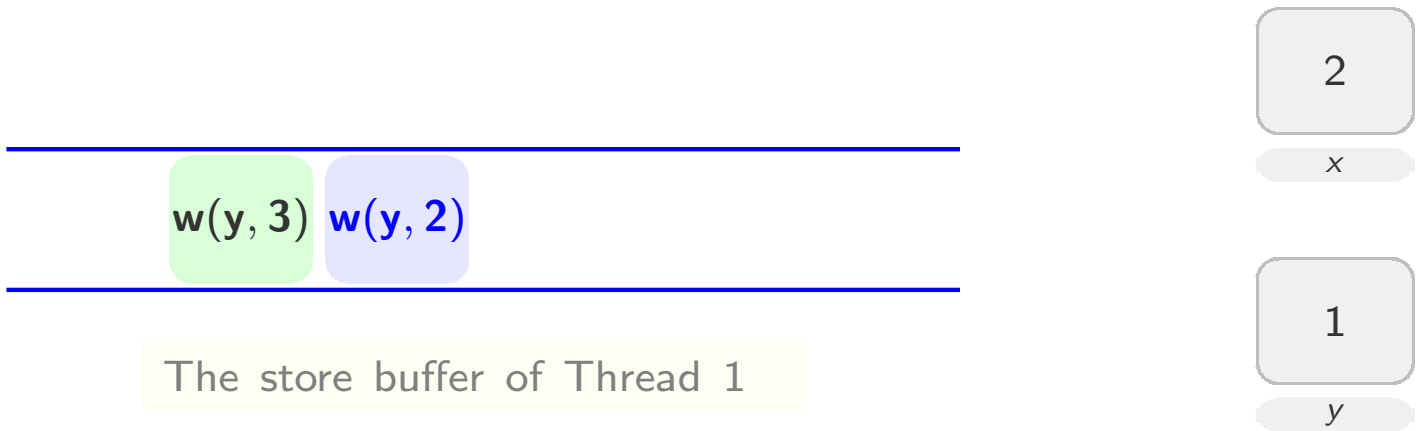
Model: The store buffers are considered as perfect FIFO channels



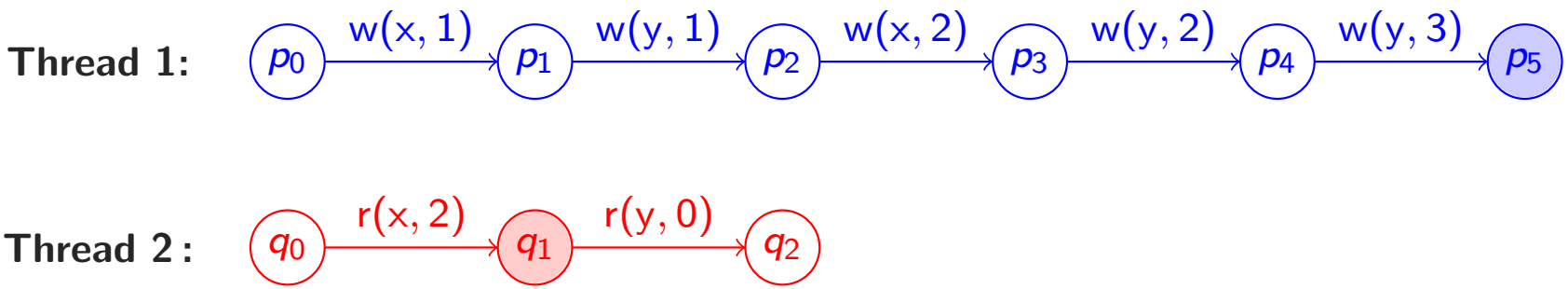
From W $\longrightarrow$ R systems to Lossy Channel Systems



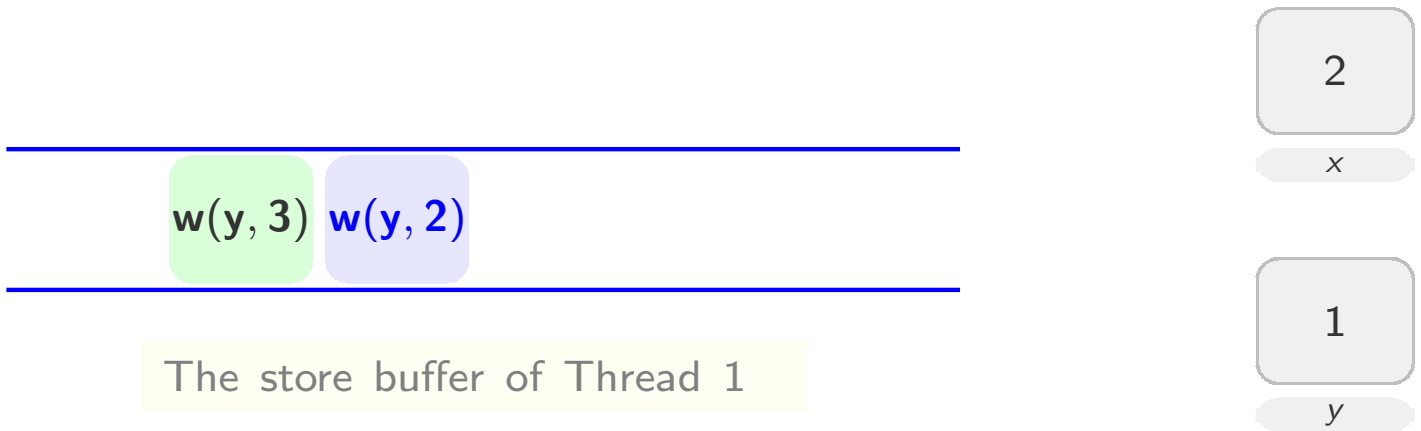
Model: The store buffers are considered as perfect FIFO channels



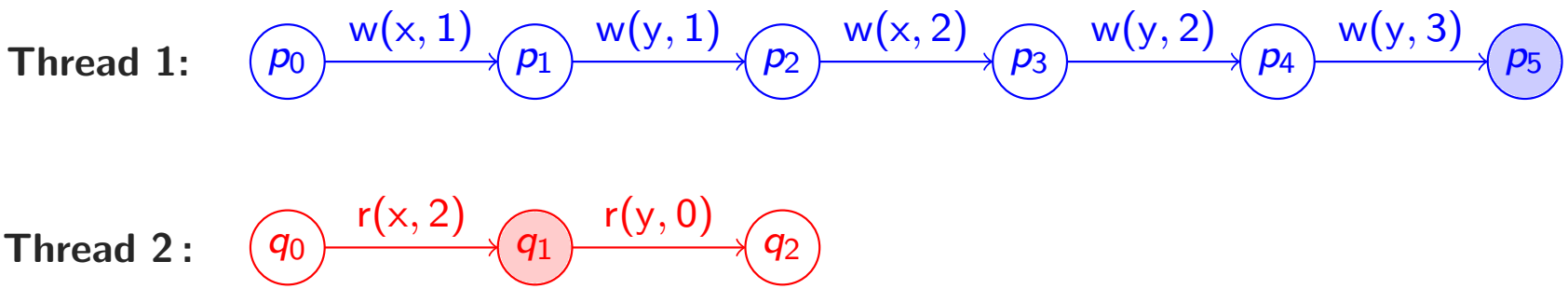
From W → R systems to Lossy Channel Systems



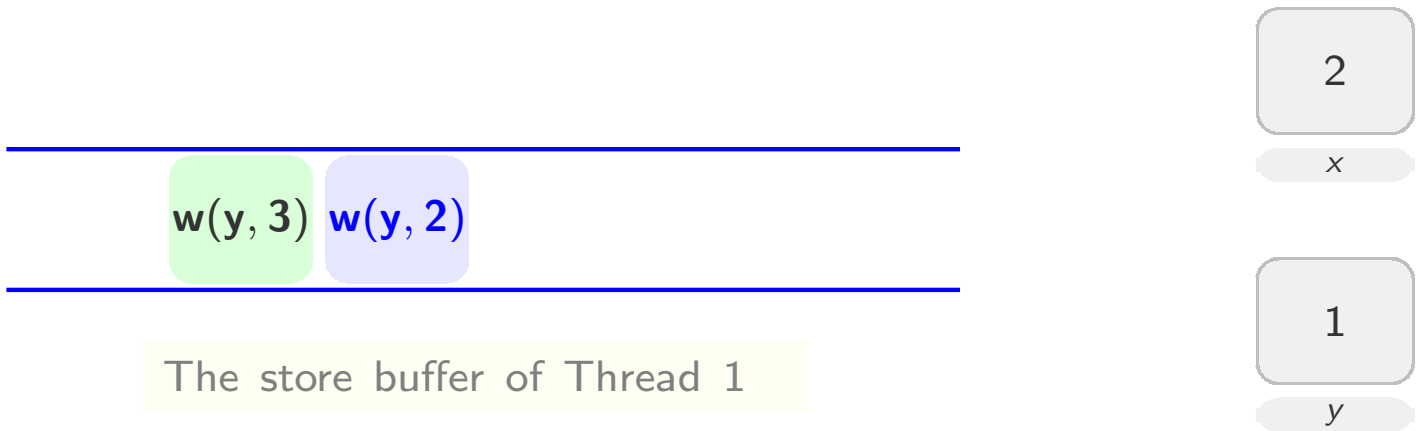
Model: The store buffers are considered as perfect FIFO channels



From W $\longrightarrow$ R systems to Lossy Channel Systems

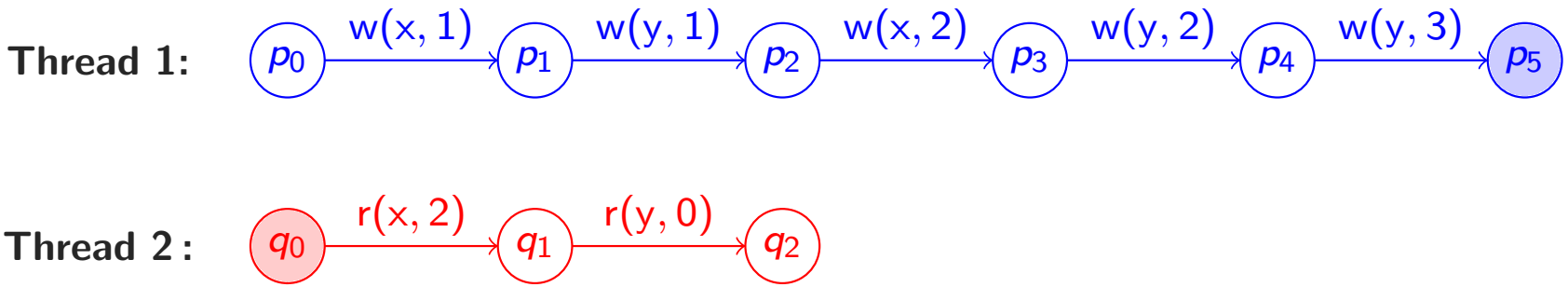


Model: The store buffers are considered as perfect FIFO channels

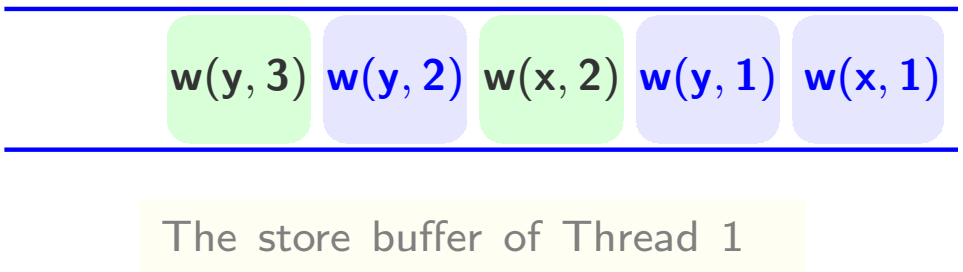


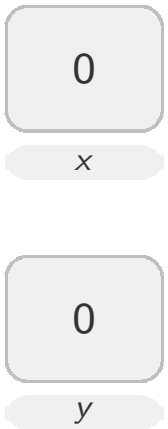
Deadlock

From W $\longrightarrow$ R systems to Lossy Channel Systems

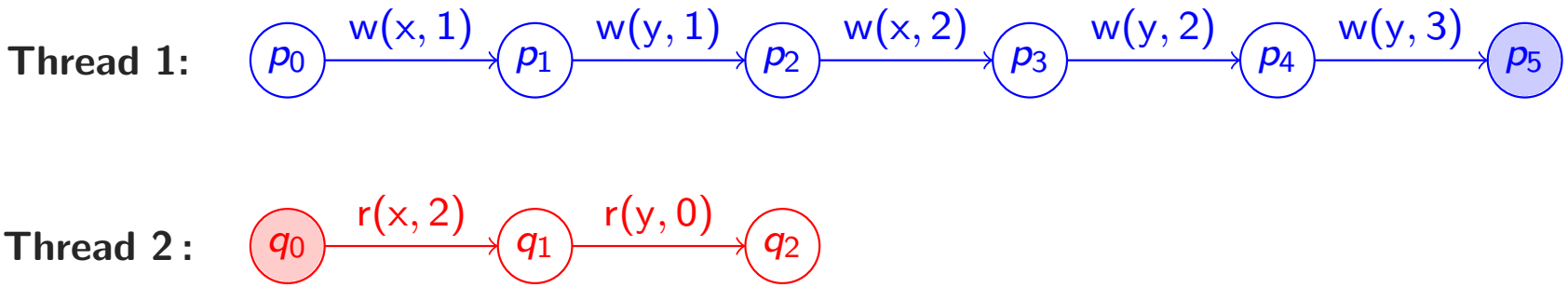


Assume that the store buffers are lossy FIFO channels

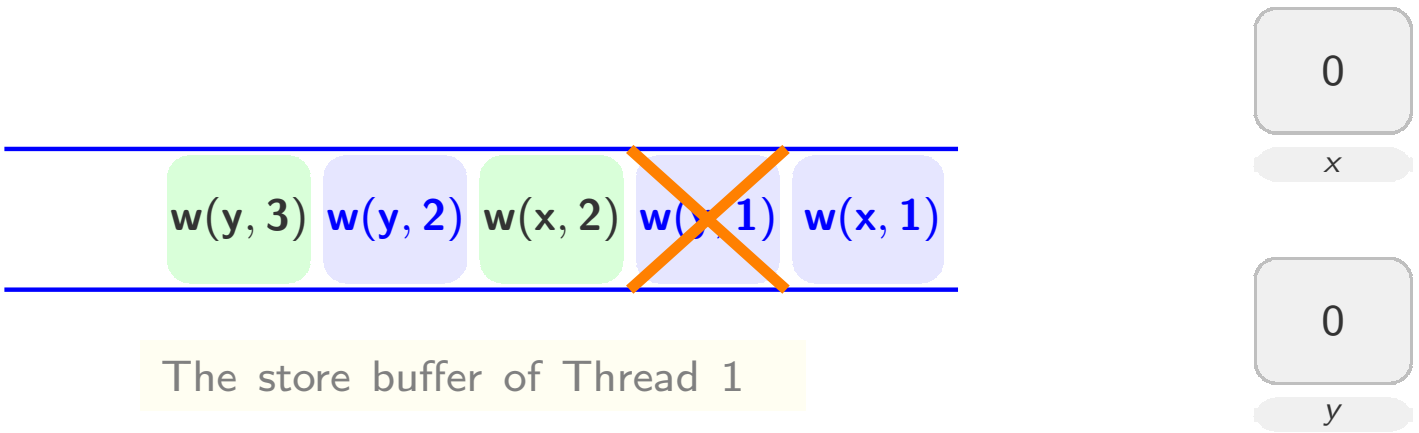




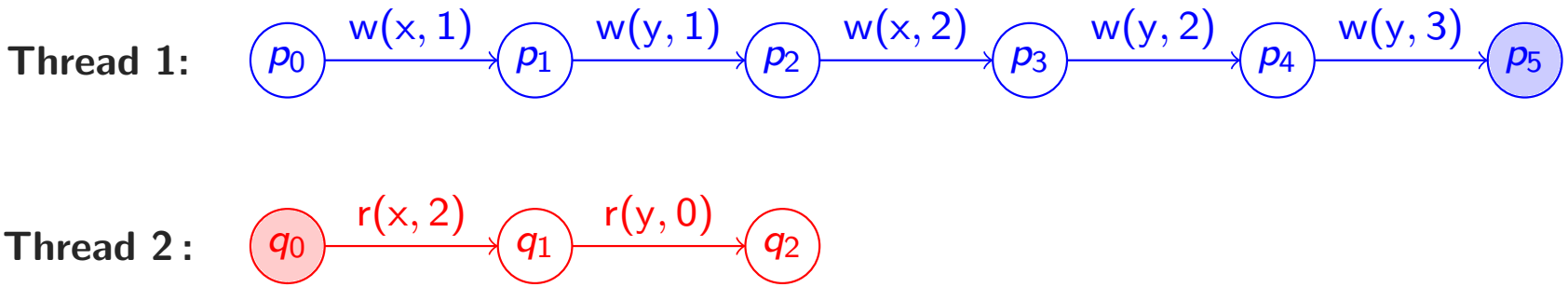
From W $\longrightarrow$ R systems to Lossy Channel Systems



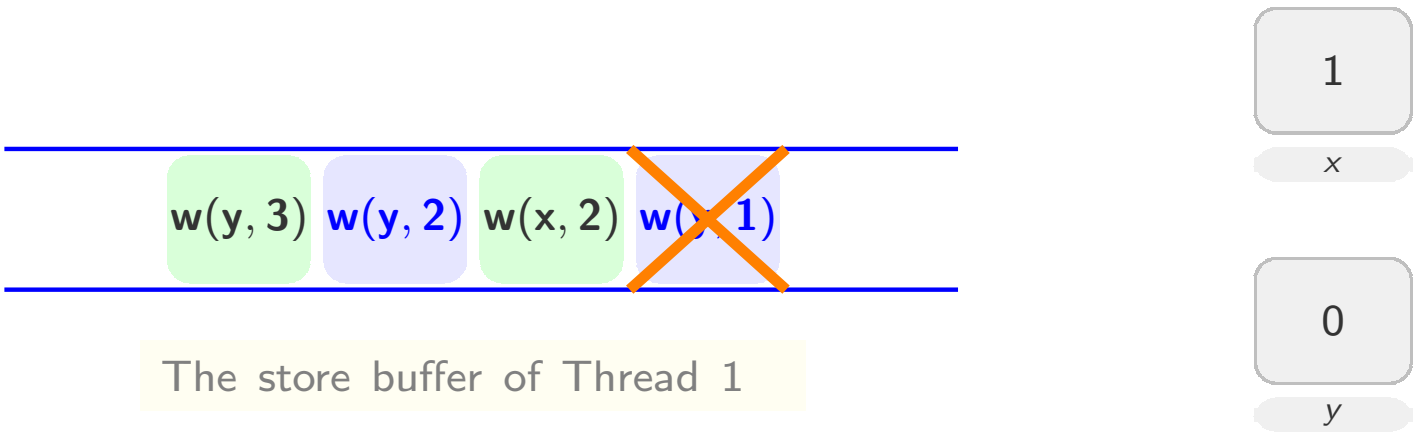
Assume that the store buffers are lossy FIFO channels



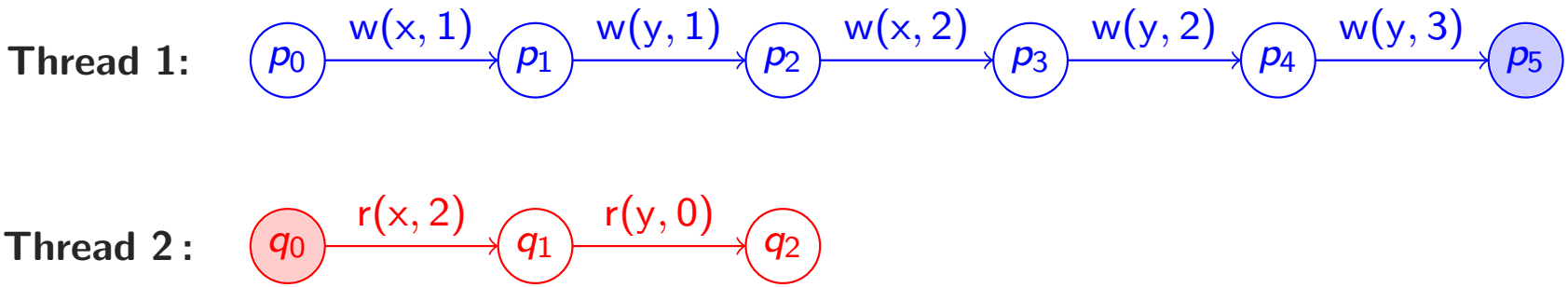
From W $\longrightarrow$ R systems to Lossy Channel Systems



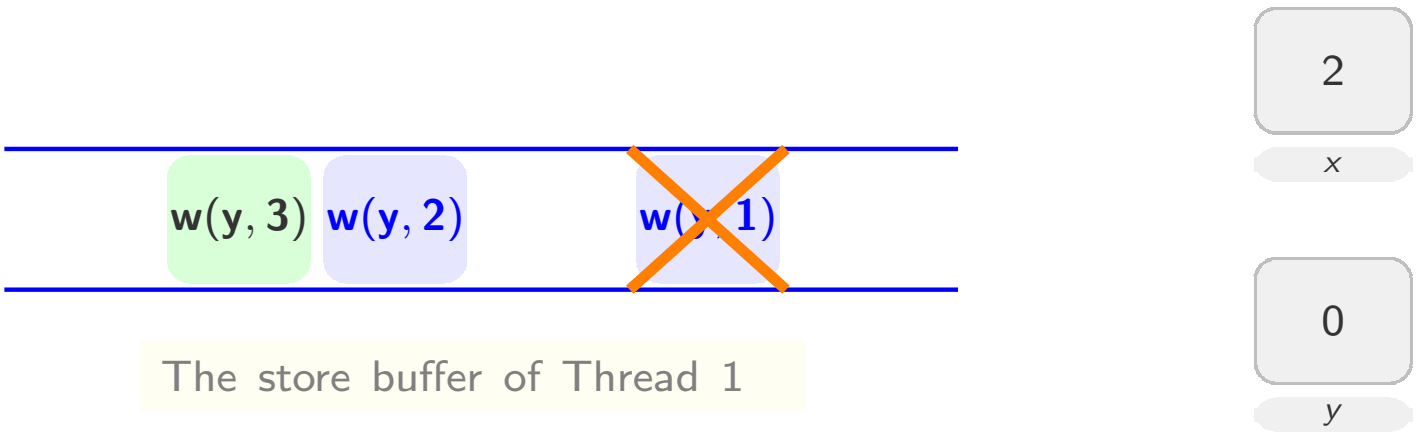
Assume that the store buffers are lossy FIFO channels



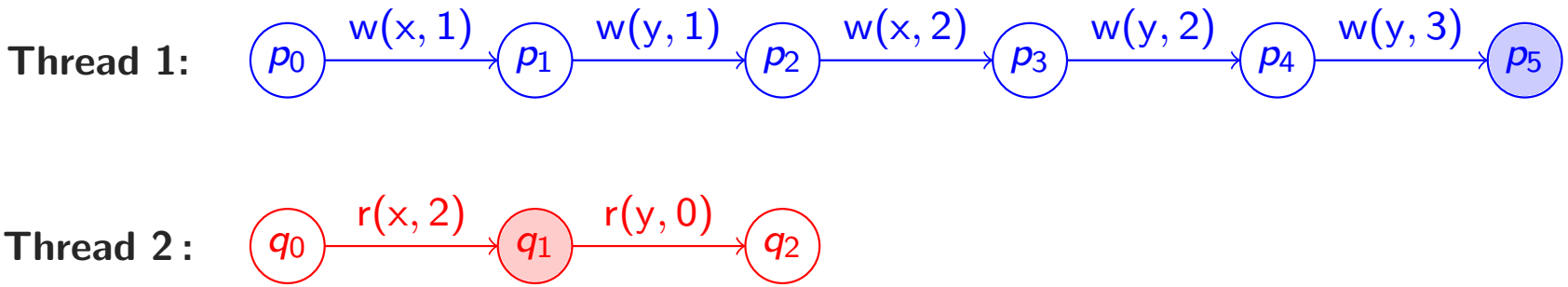
From W $\longrightarrow$ R systems to Lossy Channel Systems



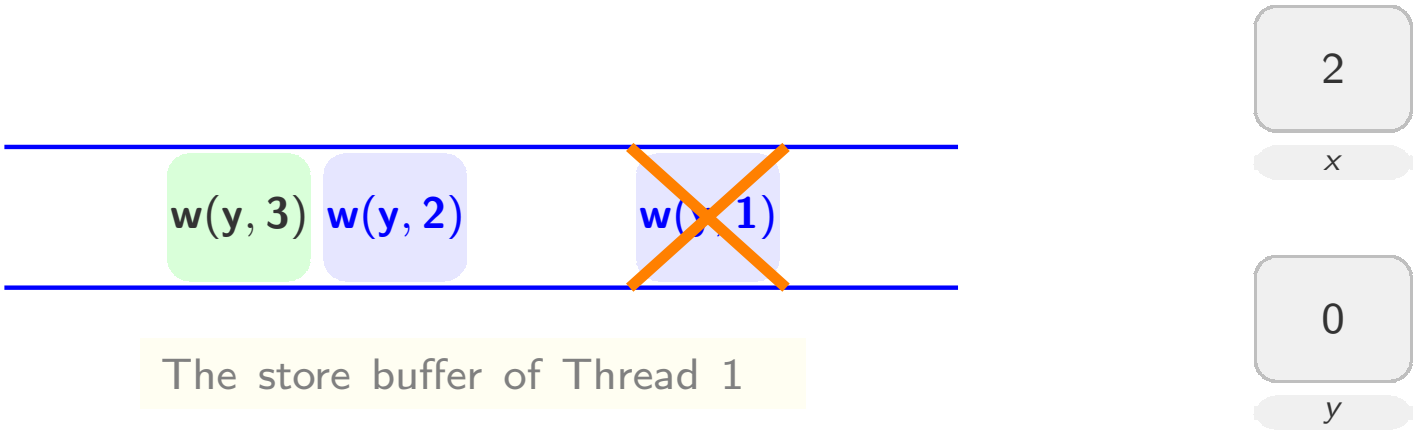
Assume that the store buffers are lossy FIFO channels



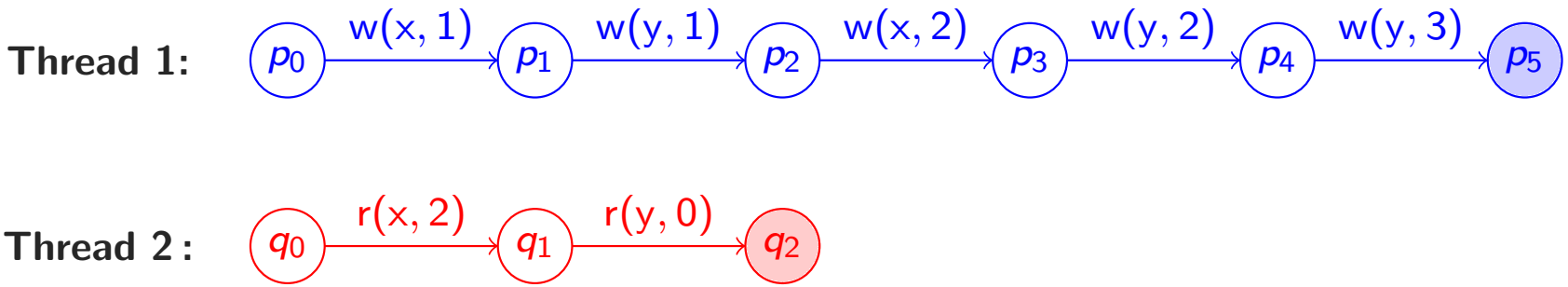
From W $\longrightarrow$ R systems to Lossy Channel Systems



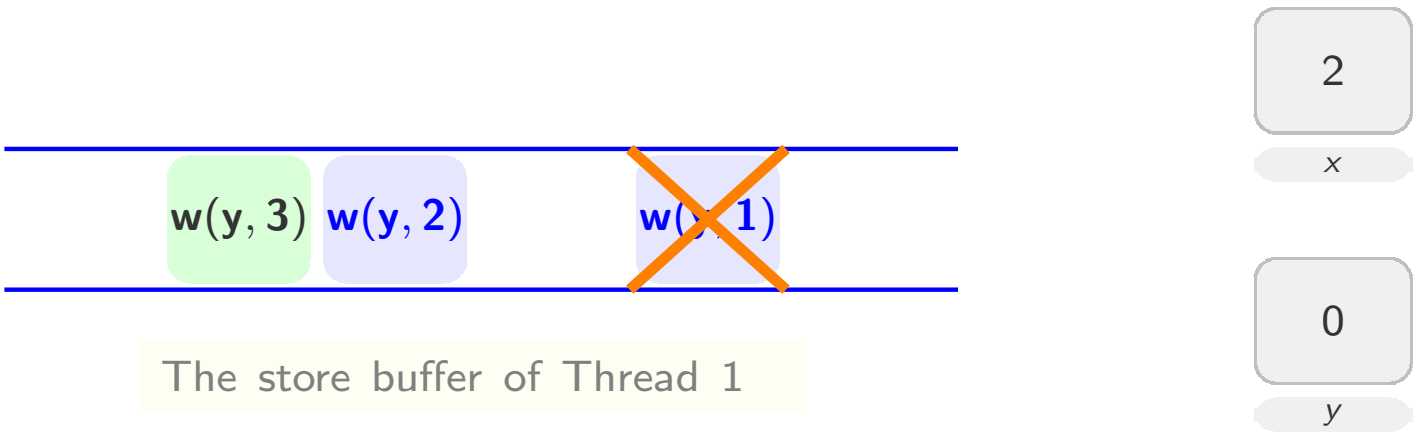
Assume that the store buffers are lossy FIFO channels



From W $\longrightarrow$ R systems to Lossy Channel Systems

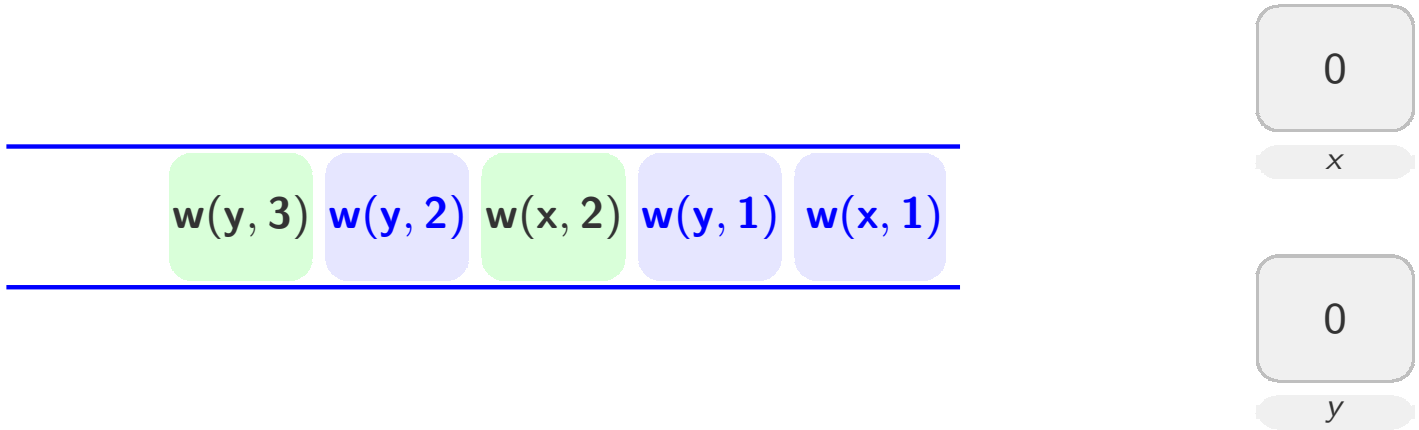


Assume that the store buffers are lossy FIFO channels

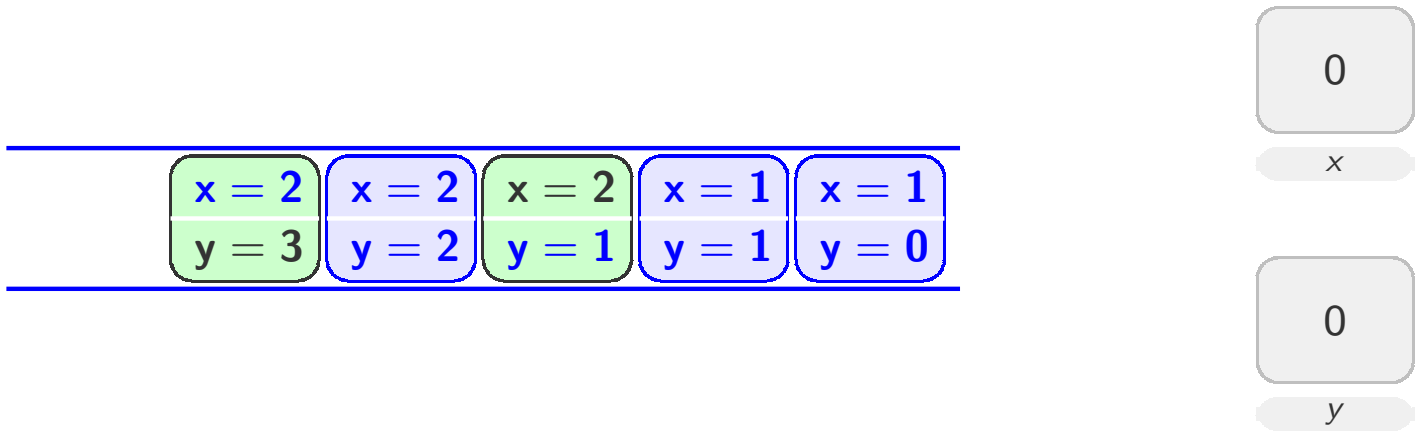


From W $\longrightarrow$ R systems to Lossy Channel Systems

Buffer = perfect FIFO channel

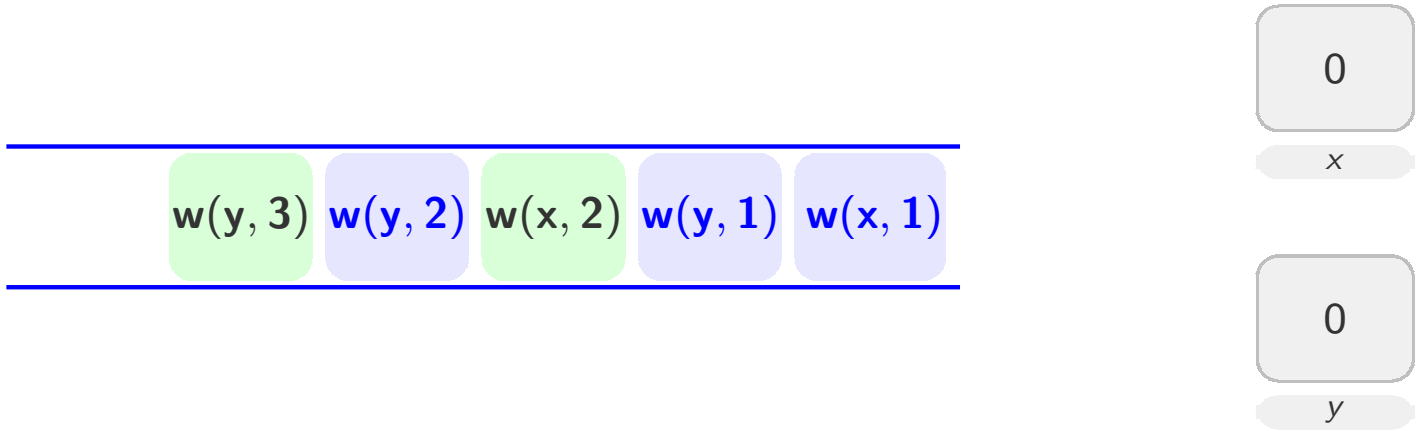


Channel= Sequence of memory states + Lossyness

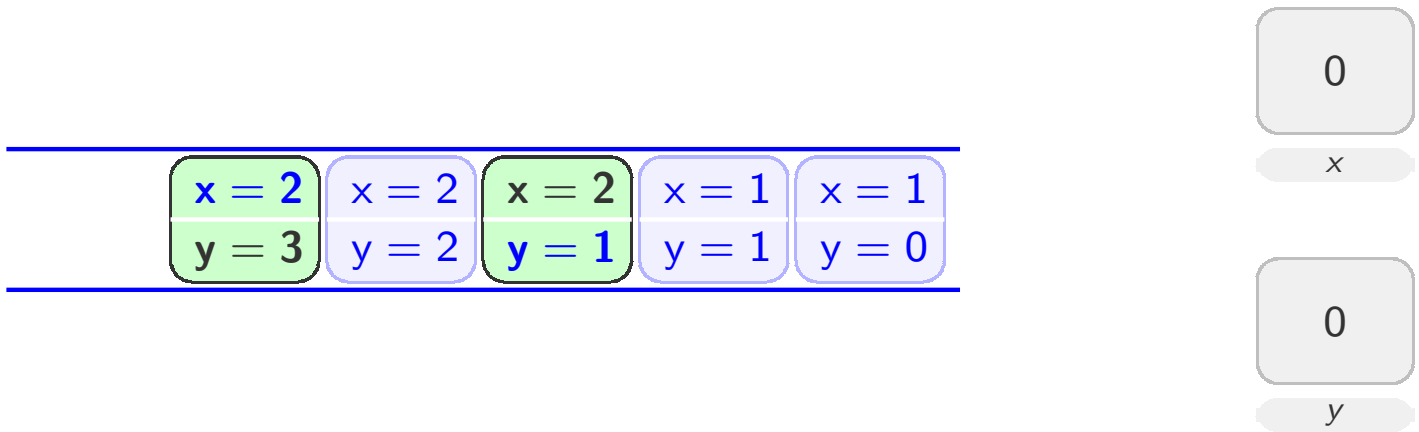


From W $\longrightarrow$ R systems to Lossy Channel Systems

Buffer = perfect FIFO channel



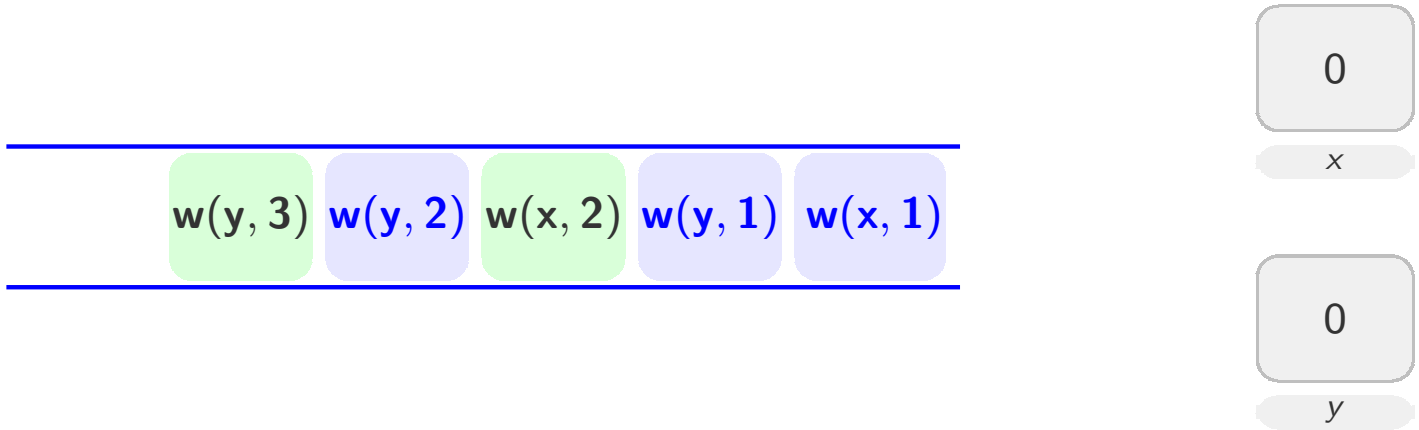
Channel= Sequence of memory states + Lossyness



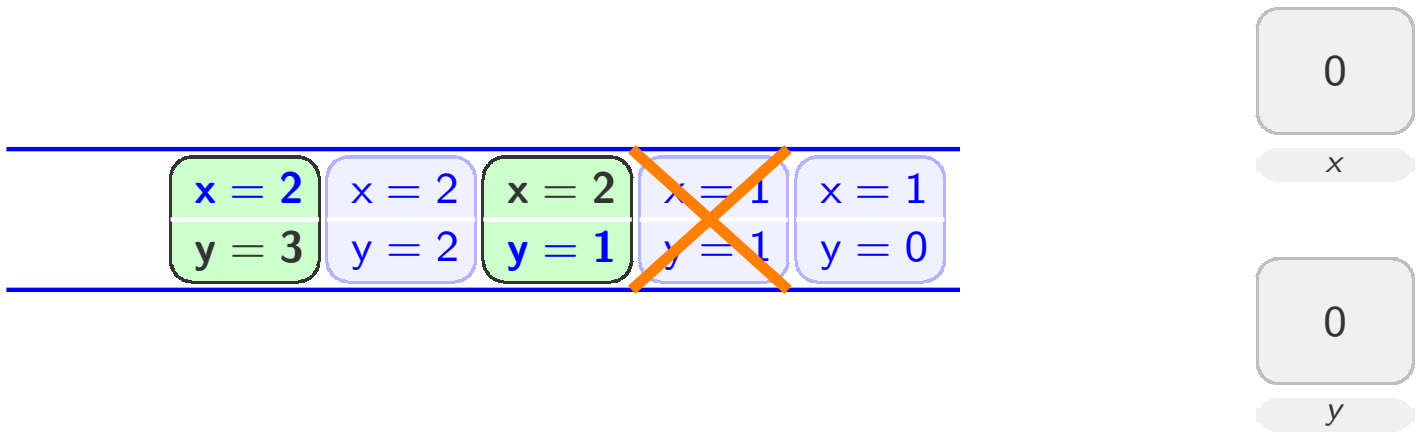
Lossyness= Unobservable memory states

From W $\longrightarrow$ R systems to Lossy Channel Systems

Buffer = perfect FIFO channel



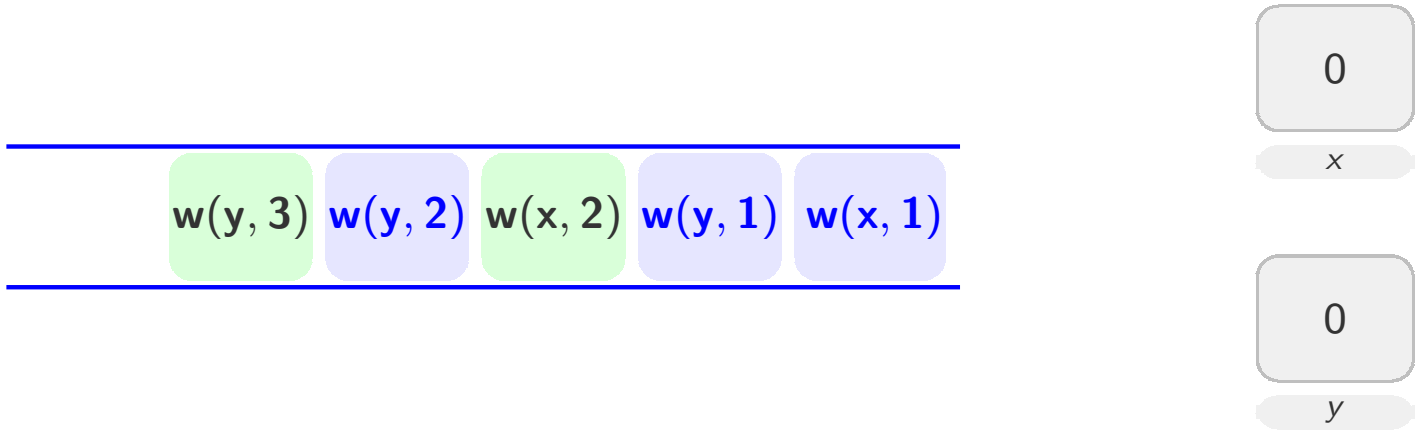
Channel= Sequence of memory states + Lossyness



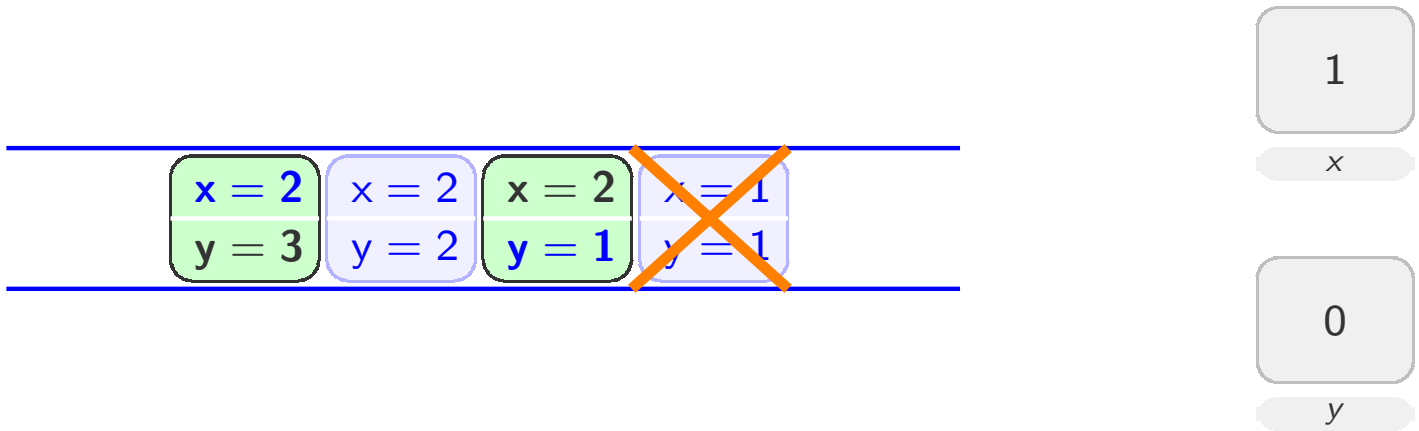
Lossyness= Unobservable memory states

From W $\longrightarrow$ R systems to Lossy Channel Systems

Buffer = perfect FIFO channel



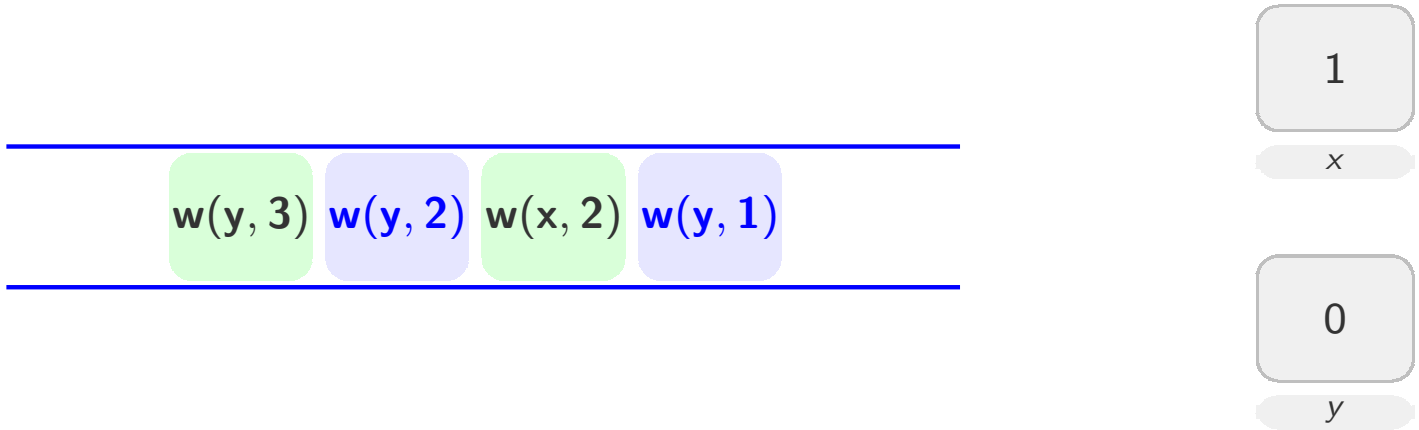
Channel= Sequence of memory states + Lossyness



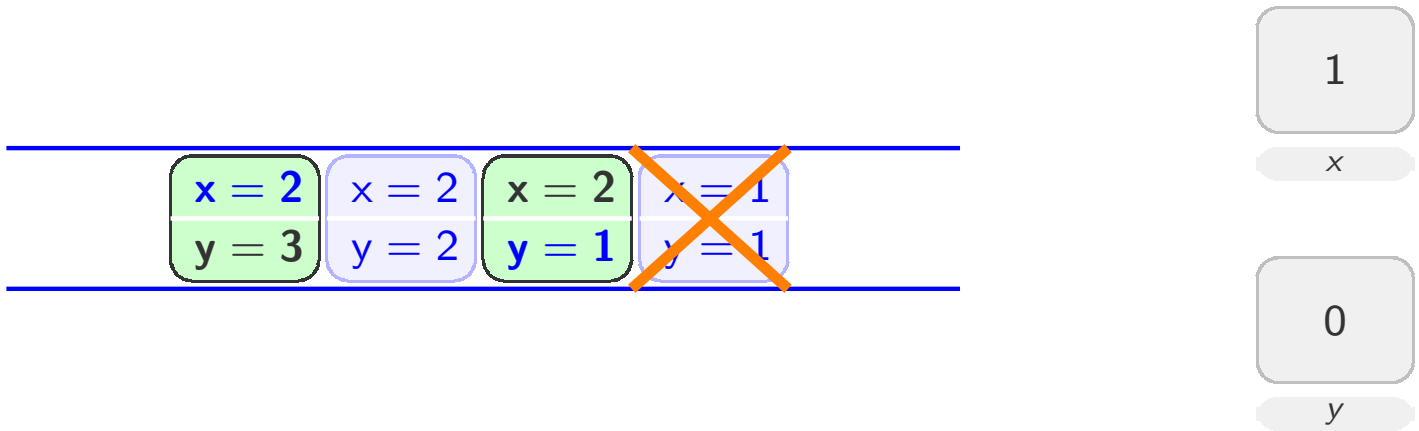
Lossyness= Unobservable memory states

From W $\longrightarrow$ R systems to Lossy Channel Systems

Buffer = perfect FIFO channel



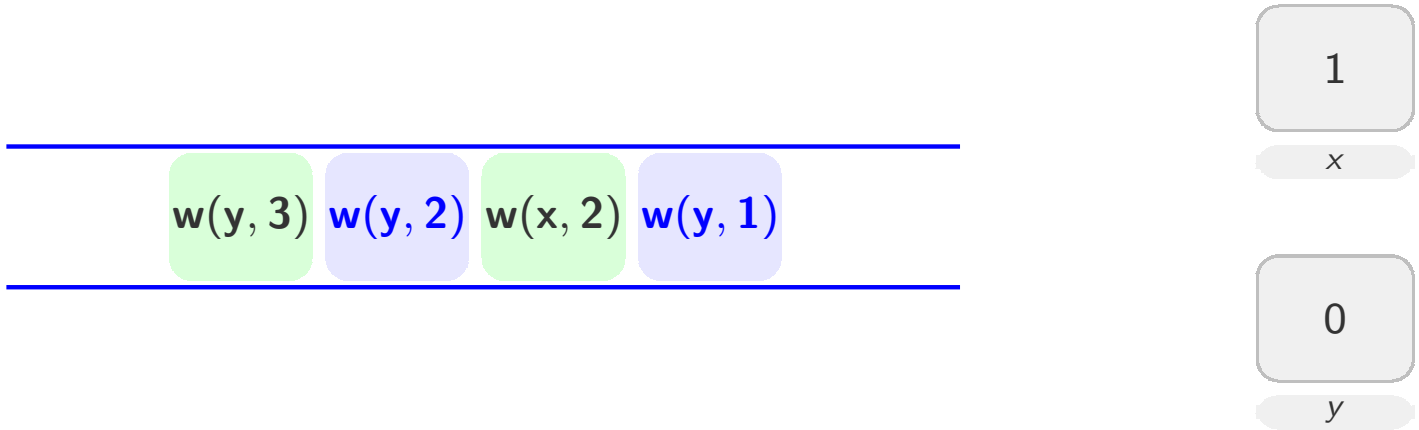
Channel= Sequence of memory states + Lossyness



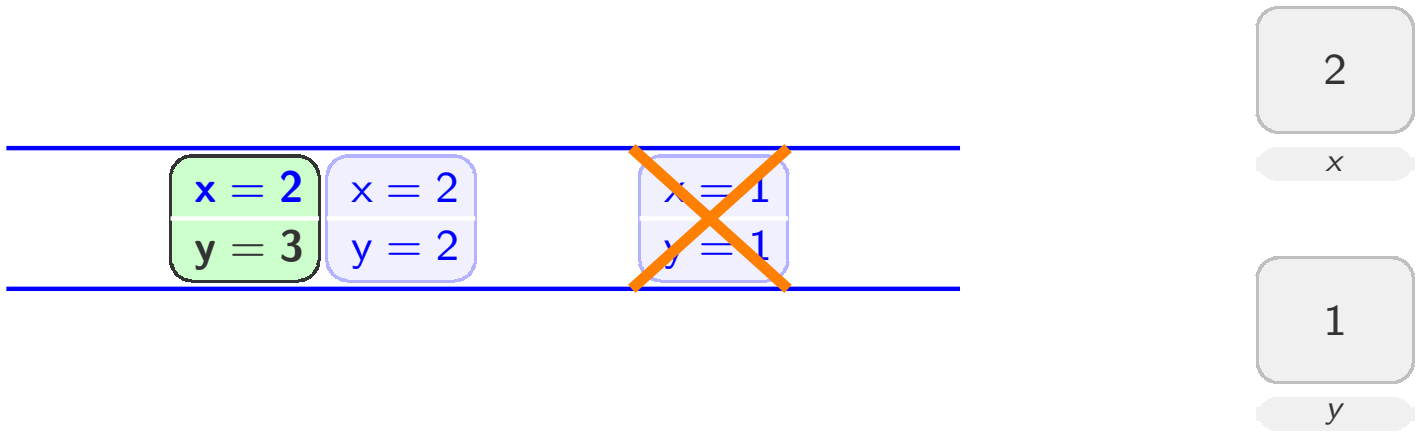
Lossyness= Unobservable memory states

From W $\longrightarrow$ R systems to Lossy Channel Systems

Buffer = perfect FIFO channel



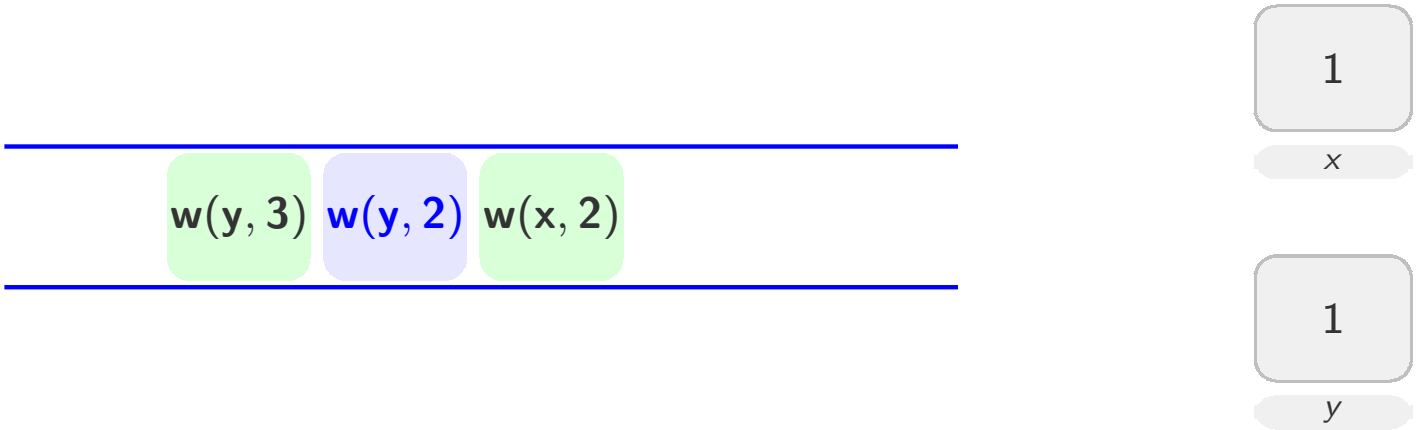
Channel= Sequence of memory states + Lossyness



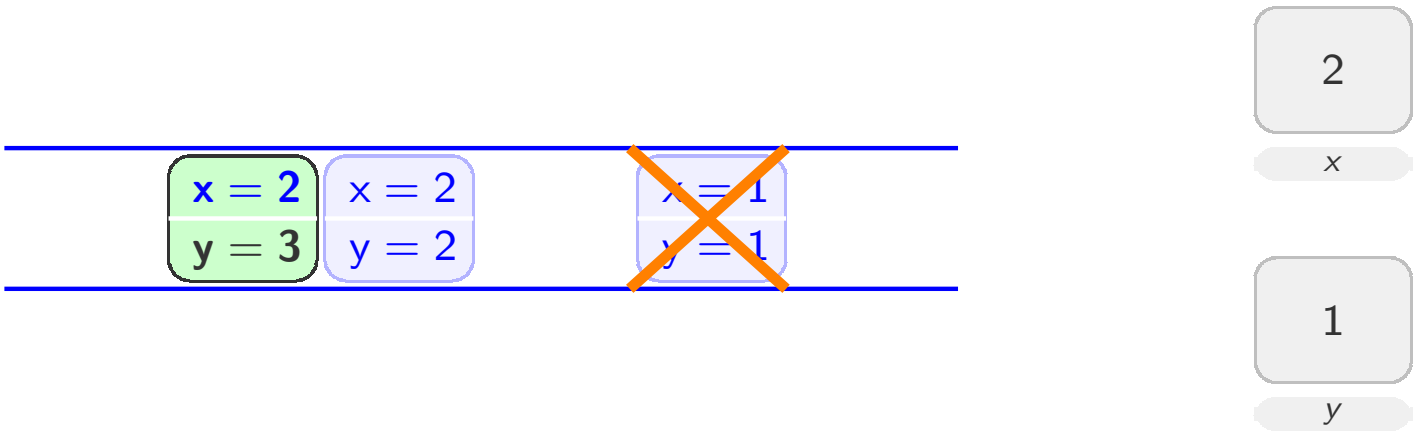
Lossyness= Unobservable memory states

From W → R systems to Lossy Channel Systems

Buffer = perfect FIFO channel



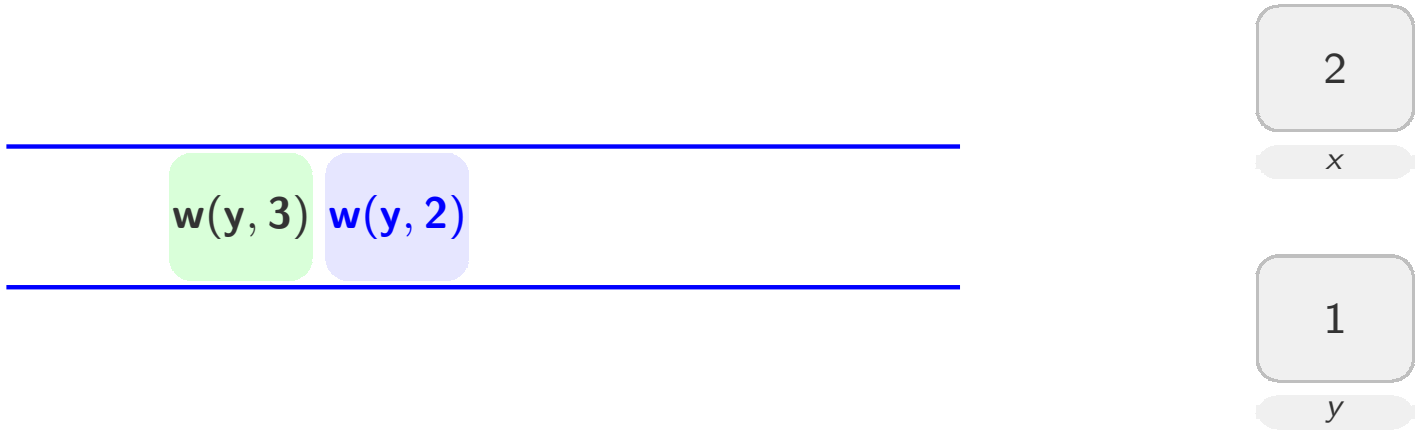
Channel= Sequence of memory states + Lossyness



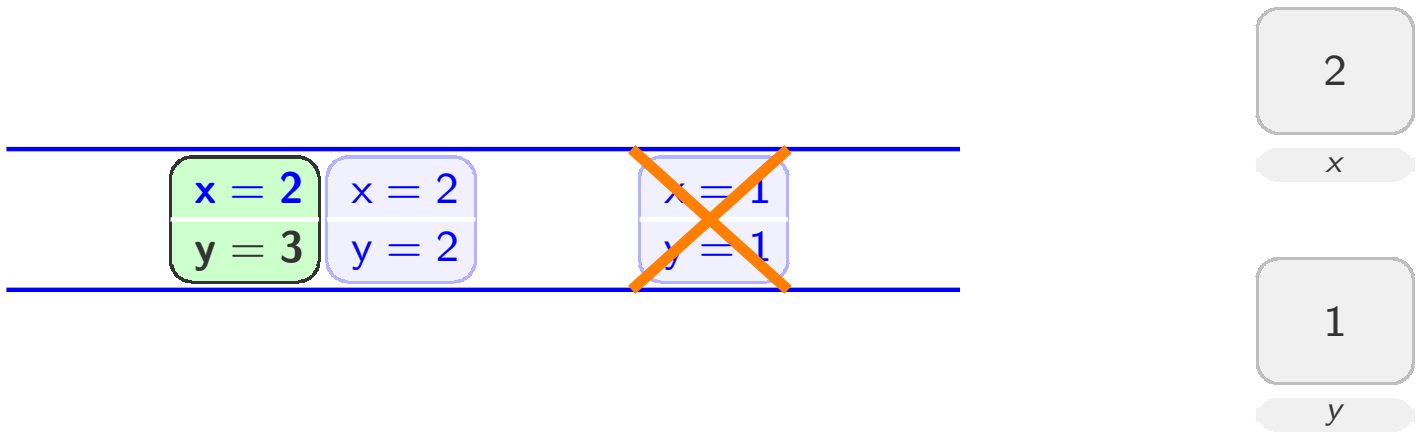
Lossyness= Unobservable memory states

From W → R systems to Lossy Channel Systems

Buffer = perfect FIFO channel

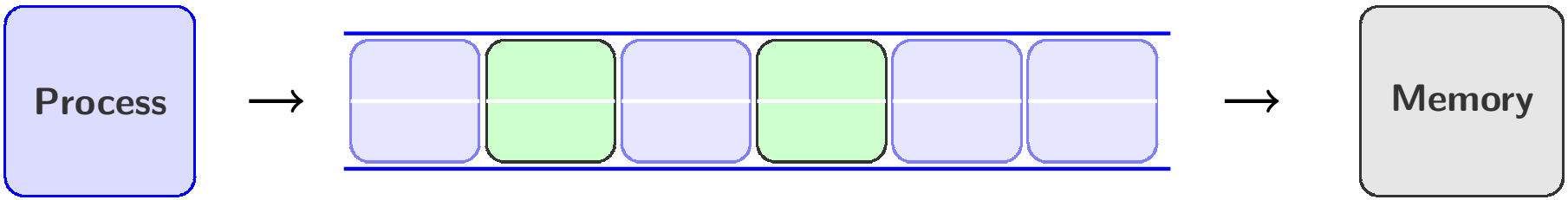


Channel= Sequence of memory states + Lossyness



Lossyness= Unobservable memory states

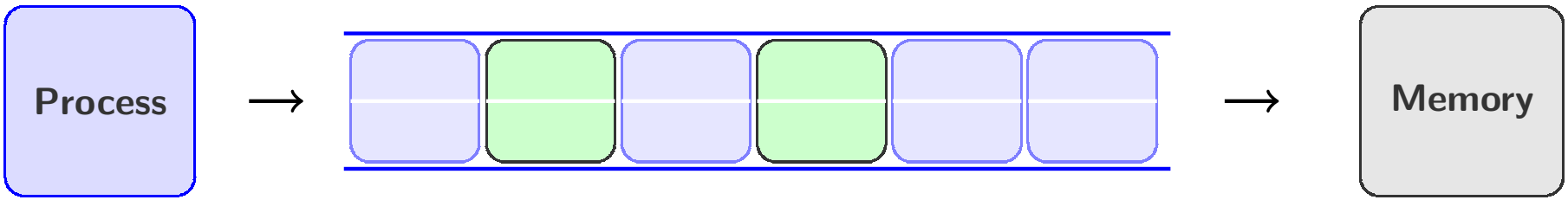
From W $\longrightarrow$ R systems to Lossy Channel Systems



- *Write:* *Compute a new memory state; send it to the channel*
- *Read:* *Check the channel/memory*
- *Memory update:* *Receive a state; copy it to the memory*

From W $\longrightarrow$ R systems to Lossy Channel Systems

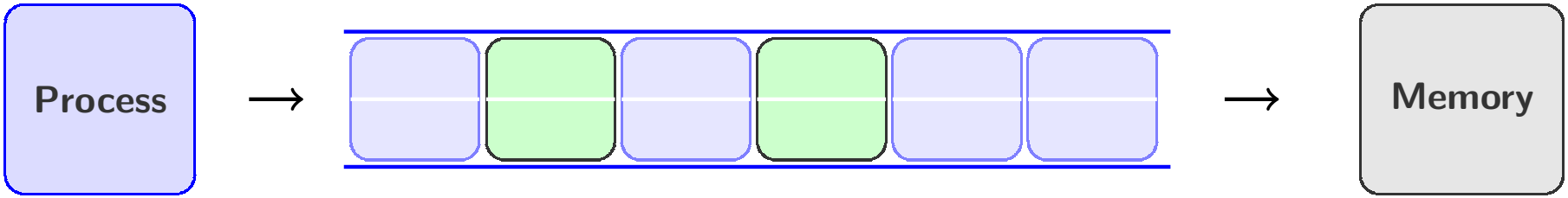
- Problem: Interference between processes ?



- *Write: Compute a new memory state; send it to the channel*
- *Read: Check the channel/memory*
- *Memory update: Receive a state; copy it to the memory*

From W $\longrightarrow$ R systems to Lossy Channel Systems

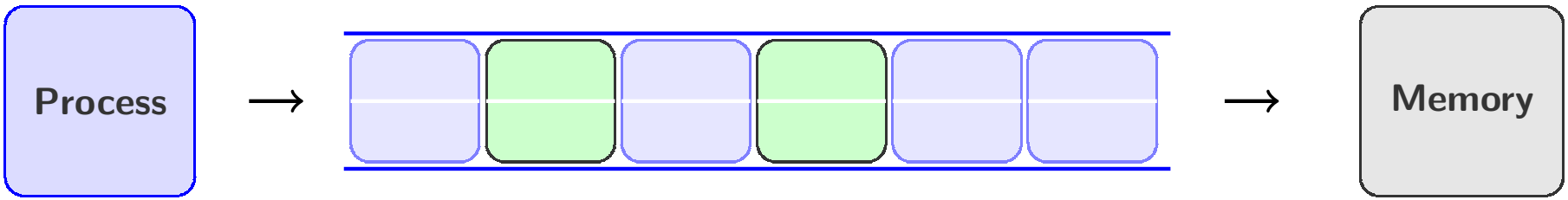
- Problem: Interference between processes ?
- $\Rightarrow$ Each process guesses occurrences of writes by other processes



- *Write: Compute a new memory state; send it to the channel*
- *Read: Check the channel/memory*
- *Memory update: Receive a state; copy it to the memory*
- *Guessed Write: Send the guessed state to the channel*

From W $\longrightarrow$ R systems to Lossy Channel Systems

- Problem: Interference between processes ?
- $\Rightarrow$ Each process guesses occurrences of writes by other processes



  - *Write:* *Compute a new memory state; send it to the channel*
  - *Read:* *Check the channel/memory*
  - *Memory update:* *Receive a state; copy it to the memory*
  - *Guessed Write:* *Send the guessed state to the channel*

- $\Rightarrow$ Check that all process agree on the sequence of states

  Synchronization of the lossy channel machines over send actions

Decidability for the State Reachability Problem

- Thm

  The state reachability problem for TSO (W $\rightarrow$ R) systems is reducible to the control-state reachability problem for LCS. The same holds for PSO (W $\rightarrow$ W/R) systems.

Decidability for the State Reachability Problem

- Thm

  The state reachability problem for TSO (W → R) systems is reducible to the control-state reachability problem for LCS. The same holds for PSO (W → W/R) systems.

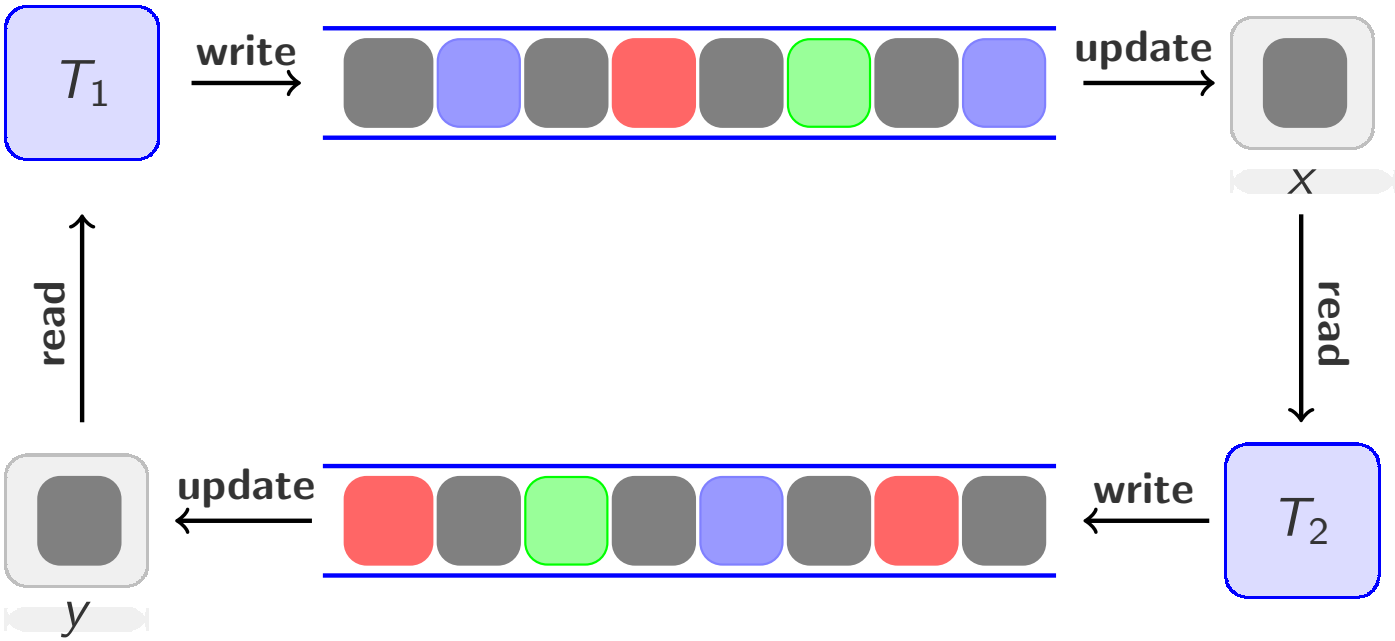
- Thm ([Abdulla, Jonsson, 1993])

  The control-state reachability problem for LCS is decidable

- Corollary

  The state reachability problem for TSO systems is decidable. The same holds for PSO systems.

From Lossy Channel Systems to W $\longrightarrow$ R systems



- T_1 simulates the lossy channel machine:
  - Send operation: Write operation of T_1 to the variable x
  - Read operation: Read operation of T_1 from the variable y
- T_2 transfers the successive values of the variable x to the variable y

Complexity / Undecidability results

- Thm

  Every LCS can be simulated by a TSO (W $\rightarrow$ R) system.
  The same holds for W $\rightarrow$ W/R *systems.*

Complexity / Undecidability results

- Thm

  Every LCS can be simulated by a TSO (W $\rightarrow$ R) system.
  The same holds for W $\rightarrow$ W/R *systems.*

- Thm ([Schnoebelen, 2001])

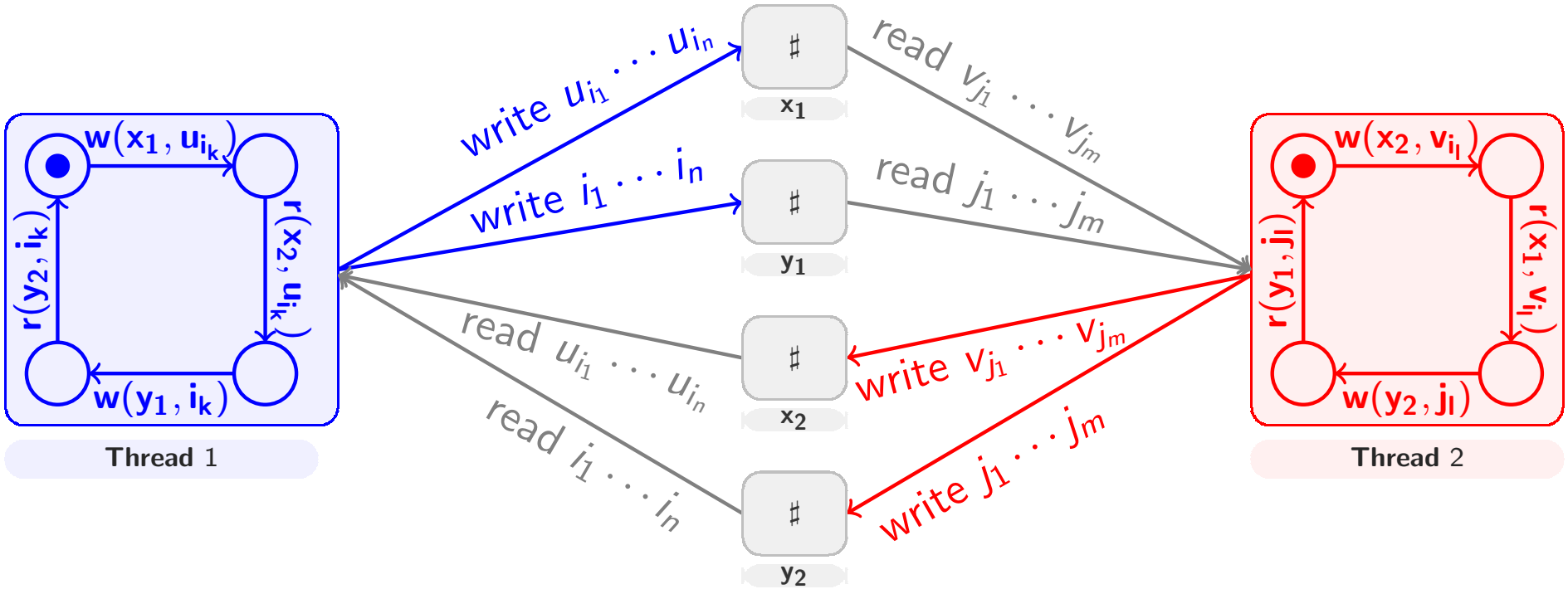
  The control-state reachability problem for LCS is nonprimitive recursive

- Thm ([Abdulla, Jonsson, 1993])

  The repeated control-state reachability problem for LCS is undecidable

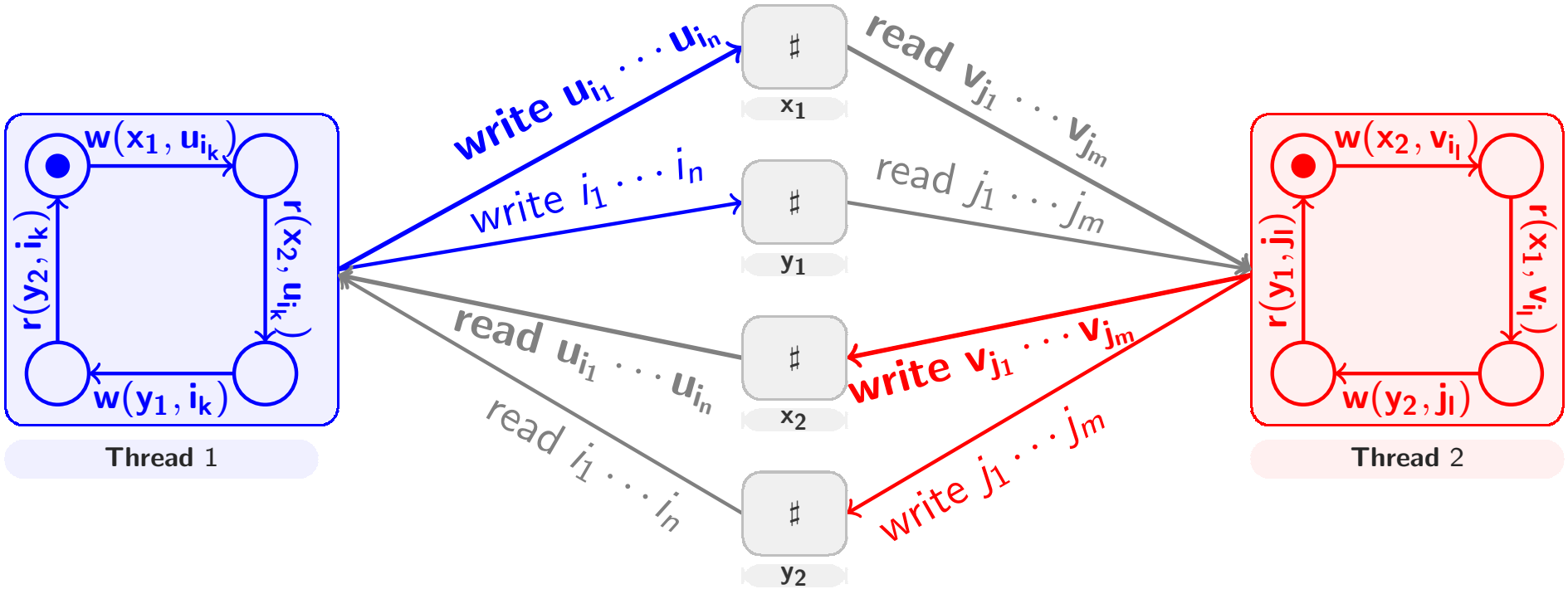
$\Rightarrow$ Lower bound for the state RP, undec. for the repeated state RP.

From PCP to reachability in $WR \rightarrow WR$ systems



A solution for the reachability problem of the $WR \rightarrow WR$ system

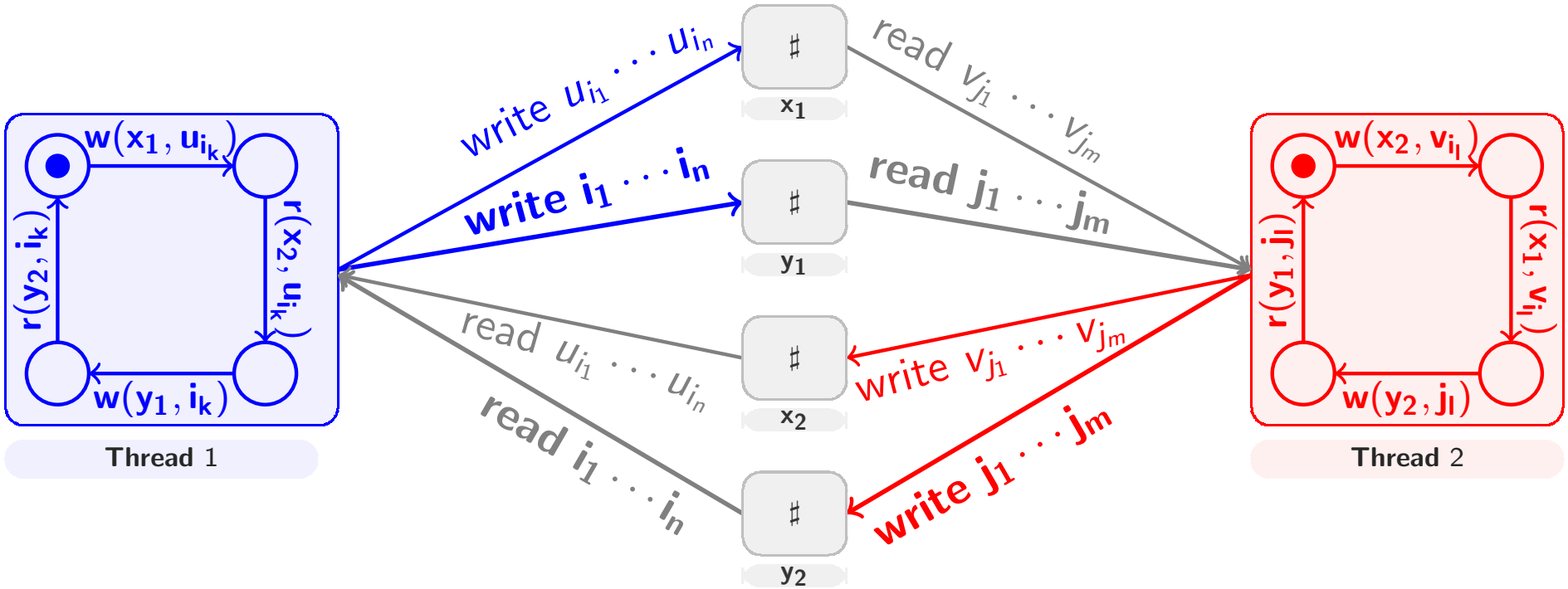
From PCP to reachability in $WR \rightarrow WR$ systems



A solution for the reachability problem of the $WR \rightarrow WR$ system

$$u_{i_1} u_{i_2} \cdots u_{i_n} = v_{j_1} v_{j_2} \cdots v_{j_m}$$

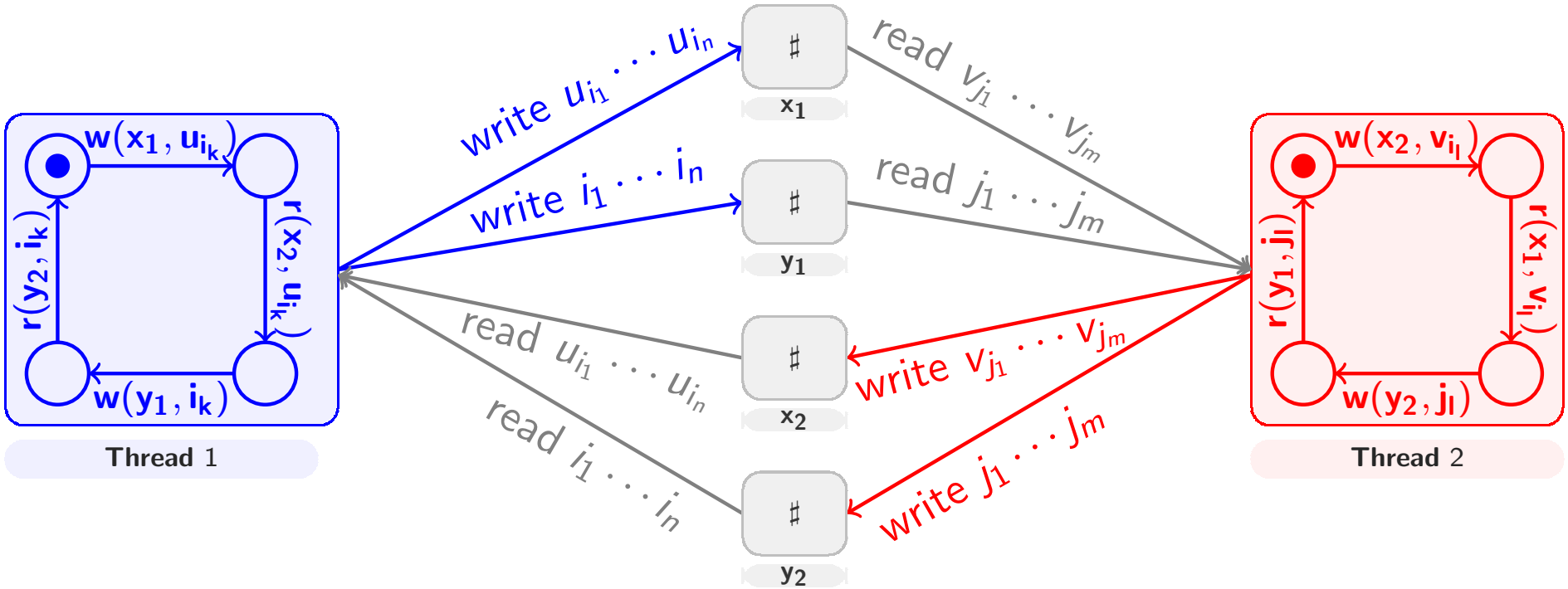
From PCP to reachability in $WR \rightarrow WR$ systems



A solution for the reachability problem of the $WR \rightarrow WR$ system

$$u_{i_1} u_{i_2} \cdots u_{i_n} = v_{j_1} v_{j_2} \cdots v_{j_m} \text{ and } i_1 i_2 \cdots i_n = j_1 j_2 \cdots j_m$$

From PCP to reachability in $WR \rightarrow WR$ systems

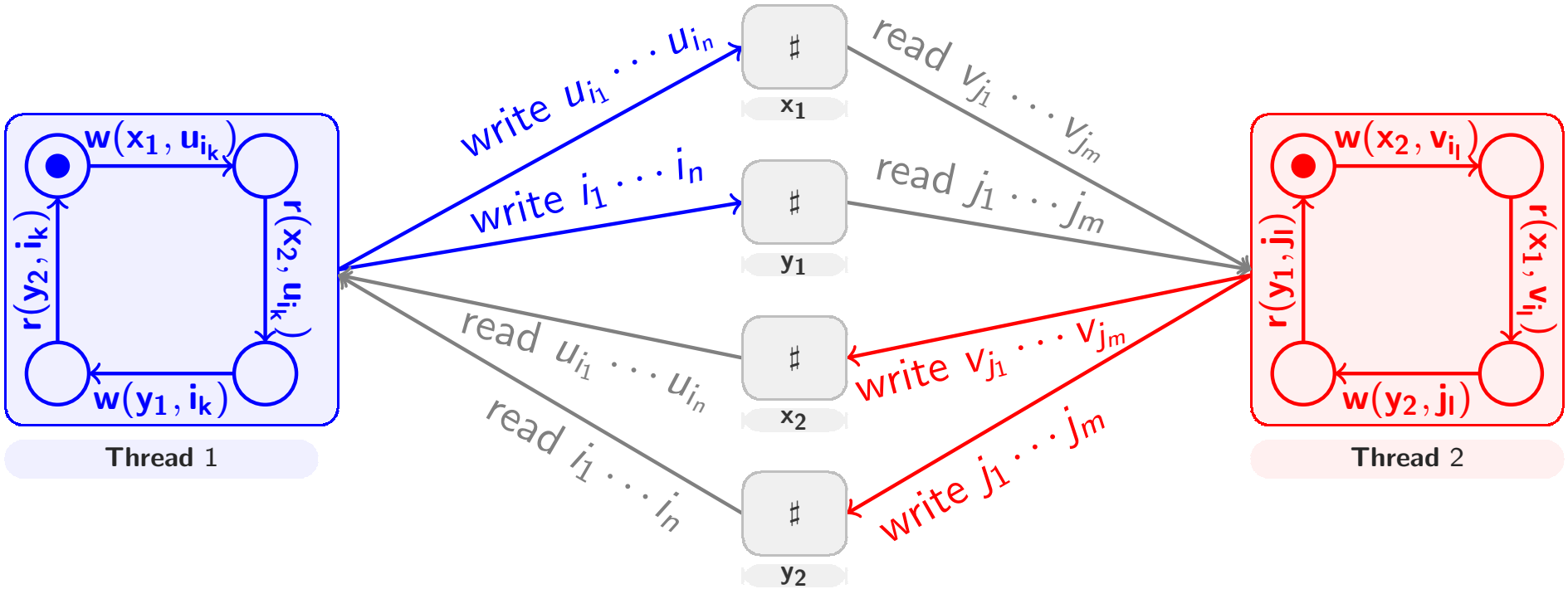


A solution for the reachability problem of the $WR \rightarrow WR$ system

$$u_{i_1} u_{i_2} \cdots u_{i_n} = v_{j_1} v_{j_2} \cdots v_{j_m} \text{ and } i_1 i_2 \cdots i_n = j_1 j_2 \cdots j_m$$

$\Rightarrow$ A solution for the Post Correspondence Problem

From PCP to reachability in $WR \rightarrow WR$ systems

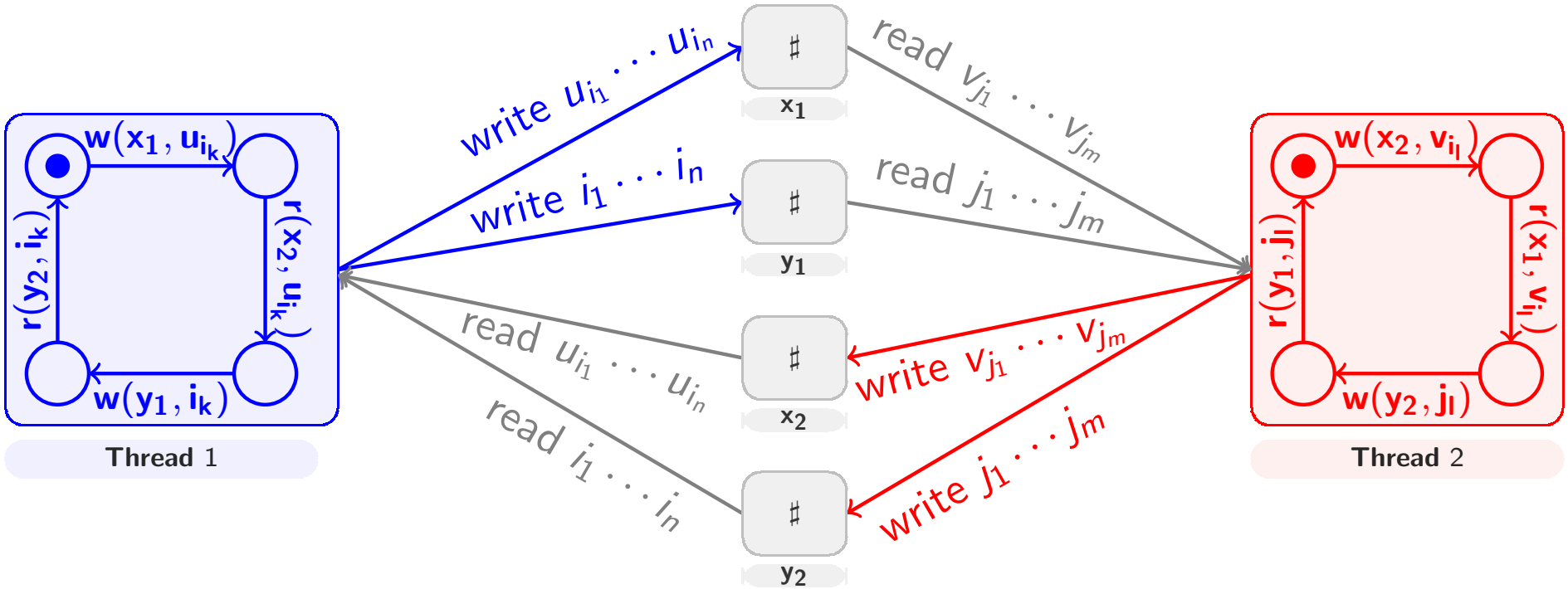


Assume that: $u_{i_1} u_{i_2} \cdots u_{i_n} = v_{j_1} v_{j_2} \cdots v_{j_m}$ and $i_1 i_2 \cdots i_n = j_1 j_2 \cdots j_m$

T_1: $r(y_2, i_n)\ w(y_1, i_n)\ r(x_2, u_{i_n})\ w(x_1, u_{i_n}) \cdots r(y_2, i_1)\ w(y_1, i_1)\ r(x_2, u_{i_1})\ w(x_1, u_{i_1})$

T_2: $r(y_1, j_n)\ w(y_2, j_n)\ r(x_1, v_{j_n})\ w(x_2, v_{j_n}) \cdots r(y_1, j_1)\ w(y_2, j_1)\ r(x_1, v_{j_1})\ w(x_2, v_{j_1})$

From PCP to reachability in $WR \rightarrow WR$ systems

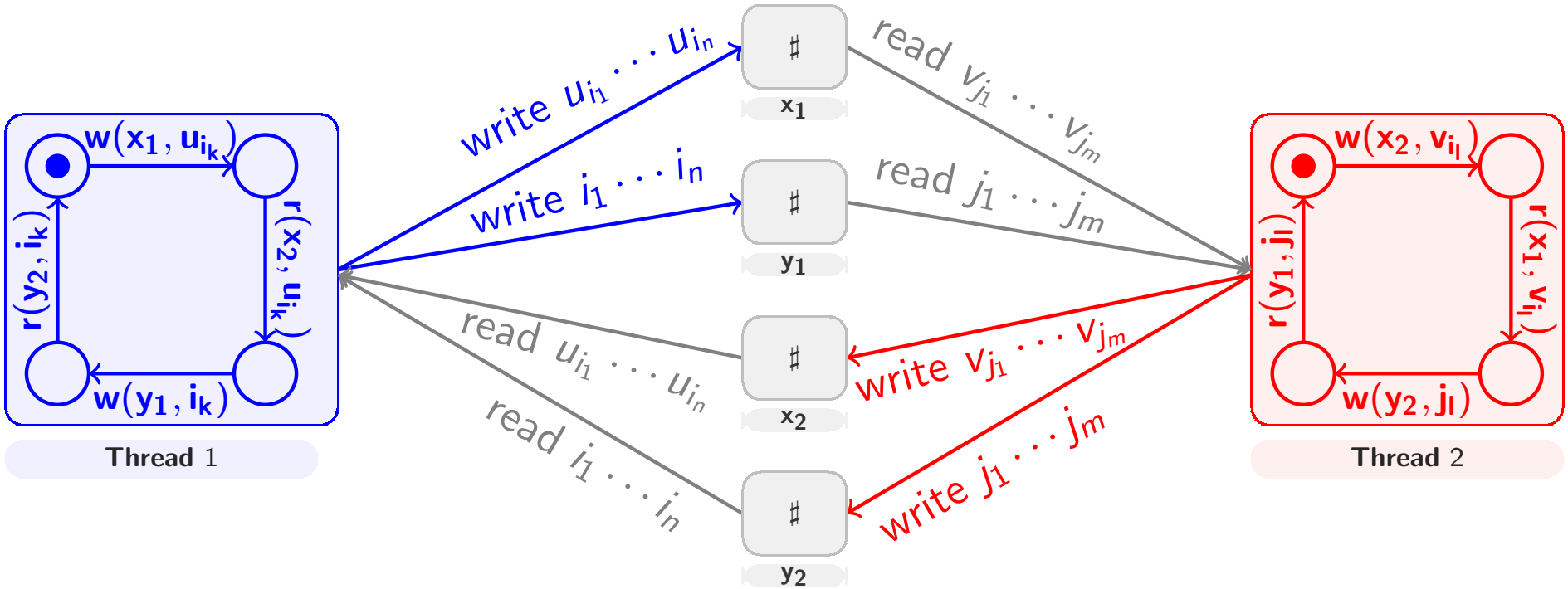


Assume that: $u_{i_1} u_{i_2} \cdots u_{i_n} = v_{j_1} v_{j_2} \cdots v_{j_m}$ and $i_1 i_2 \cdots i_n = j_1 j_2 \cdots j_m$

T_1: $r(y_2, i_n) \cdots r(y_2, i_1)\ r(x_2, u_{i_n}) \cdots r(x_2, u_{i_1})\ w(y_1, i_n) \cdots w(y_1, i_1)\ w(x_1, u_{i_n}) \cdots w(x_1, u_{i_1})$

T_2: $w(y_2, j_n) \cdots w(y_2, j_1)\ w(x_2, v_{j_n}) \cdots w(x_2, v_{j_1})\ r(y_1, j_n) \cdots r(y_1, j_1)\ r(x_1, v_{j_n}) \cdots r(x_1, v_{j_1})$

From PCP to reachability in $WR \rightarrow WR$ systems



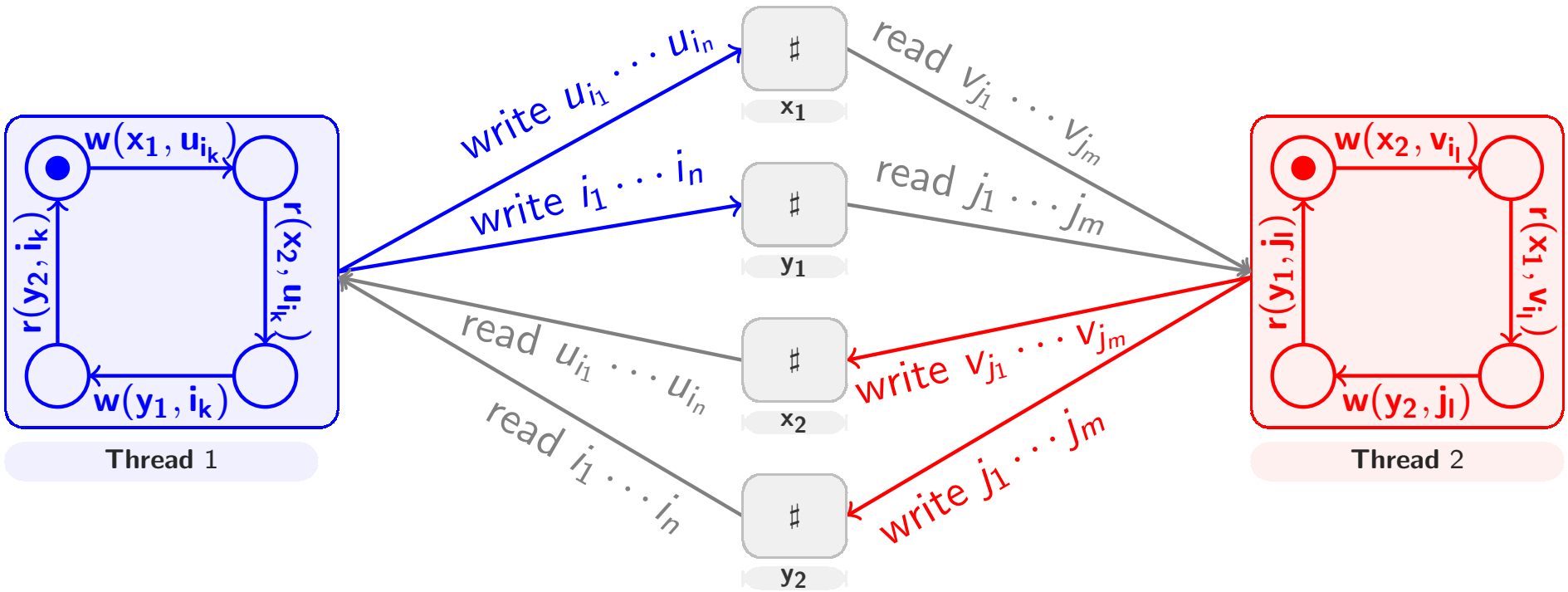
Assume that: $u_{i_1} u_{i_2} \cdots u_{i_n} = v_{j_1} v_{j_2} \cdots v_{j_m}$ and $i_1 i_2 \cdots i_n = j_1 j_2 \cdots j_m$

T_1: $r(y_2, i_n) \cdots r(y_2, i_1)\ r(x_2, u_{i_n}) \cdots r(x_2, u_{i_1})\ w(y_1, i_n) \cdots w(y_1, i_1)\ w(x_1, u_{i_n}) \cdots w(x_1, u_{i_1})$

T_2: $w(y_2, j_n) \cdots w(y_2, j_1)\ w(x_2, v_{j_n}) \cdots w(x_2, v_{j_1})\ r(y_1, j_n) \cdots r(y_1, j_1)\ r(x_1, v_{j_n}) \cdots r(x_1, v_{j_1})$

$\Rightarrow$ A solution for the reachability problem of the $WR \rightarrow WR$ system

From PCP to reachability in $(W \rightarrow R) + (R \rightarrow WR)$ systems

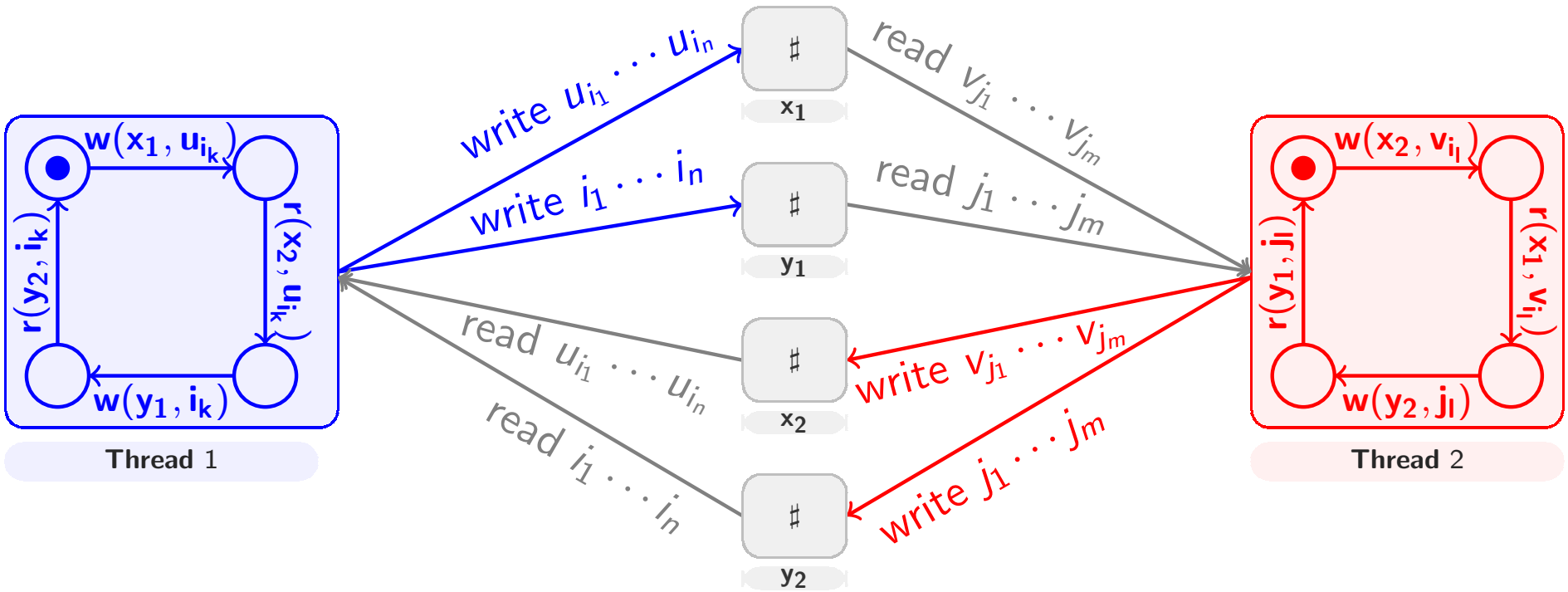


Assume that: $u_{i_1} u_{i_2} \cdots u_{i_n} = v_{j_1} v_{j_2} \cdots v_{j_m}$ and $i_1 i_2 \cdots i_n = j_1 j_2 \cdots j_m$

T_1: $r(y_2, i_n)\ w(y_1, i_n)\ r(x_2, u_{i_n})\ w(x_1, u_{i_n}) \cdots r(y_2, i_1)\ w(y_1, i_1)\ r(x_2, u_{i_1})\ w(x_1, u_{i_1})$

T_2: $r(y_1, j_n)\ w(y_2, j_n)\ r(x_1, v_{j_n})\ w(x_2, v_{j_n}) \cdots r(y_1, j_1)\ w(y_2, j_1)\ r(x_1, v_{j_1})\ w(x_2, v_{j_1})$

From PCP to reachability in $(W \rightarrow R) + (R \rightarrow WR)$ systems

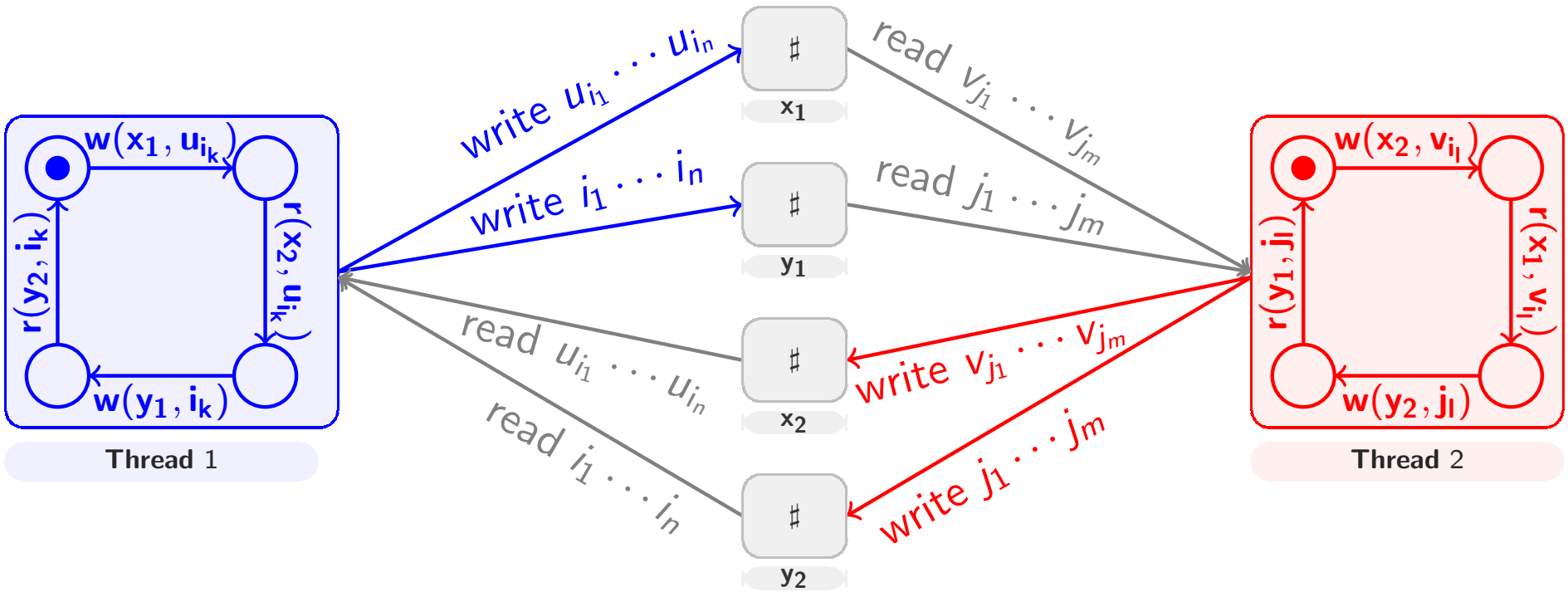


Assume that: $u_{i_1} u_{i_2} \cdots u_{i_n} = v_{j_1} v_{j_2} \cdots v_{j_m}$ and $i_1 i_2 \cdots i_n = j_1 j_2 \cdots j_m$

T_1: $r(y_2, i_n)\ r(x_2, u_{i_n}) \cdots r(y_2, i_1)\ r(x_2, u_{i_1}) \cdots w(y_1, i_n)\ w(x_1, u_{i_n}) \cdots w(y_1, i_1)\ w(x_1, u_{i_1})$

T_2: $w(y_2, j_n)\ w(x_2, v_{j_n}) \cdots w(y_2, j_1)\ w(x_2, v_{j_1}) \cdots r(y_1, j_n)\ r(x_1, v_{j_n}) \cdots r(y_1, j_1)\ r(x_1, v_{j_1})$

From PCP to reachability in $(W \rightarrow R) + (R \rightarrow WR)$ systems



Assume that: $u_{i_1} u_{i_2} \cdots u_{i_n} = v_{j_1} v_{j_2} \cdots v_{j_m}$ and $i_1 i_2 \cdots i_n = j_1 j_2 \cdots j_m$

T_1: $r(y_2, i_n)\ r(x_2, u_{i_n}) \cdots r(y_2, i_1)\ r(x_2, u_{i_1}) \cdots w(y_1, i_n)\ w(x_1, u_{i_n}) \cdots w(y_1, i_1)\ w(x_1, u_{i_1})$

T_2: $w(y_2, j_n)\ w(x_2, v_{j_n}) \cdots w(y_2, j_1)\ w(x_2, v_{j_1}) \cdots r(y_1, j_n)\ r(x_1, v_{j_n}) \cdots r(y_1, j_1)\ r(x_1, v_{j_1})$

$\Rightarrow$ Reachability in the $(W \rightarrow R) + (R \rightarrow RW)$ system

(Un)decidability results for WR $\rightarrow$ WR systems

- Thm

  PCP is reducible to state reachability in W/R $\rightarrow$ W/R *systems. The same holds for* (W $\rightarrow$ R) + (R $\rightarrow$ R/W) *systems.*

(Un)decidability results for WR $\rightarrow$ WR systems

- Thm

  PCP is reducible to state reachability in W/R $\rightarrow$ W/R *systems. The same holds for* (W $\rightarrow$ R) + (R $\rightarrow$ R/W) *systems.*

- $\Rightarrow$ Bounded speculation:

  Reads are stored (guessed) and checked later + Bound on the number of Reads in the buffers.

(Un)decidability results for WR $\rightarrow$ WR systems

- Thm

  PCP is reducible to state reachability in W/R $\rightarrow$ W/R *systems. The same holds for* (W $\rightarrow$ R) + (R $\rightarrow$ R/W) *systems.*

- $\Rightarrow$ Bounded speculation:

  Reads are stored (guessed) and checked later + Bound on the number of Reads in the buffers.

- Thm

  State reachability in bounded-speculation W/R $\rightarrow$ W/R *systems is decidable.*
  The same holds for (W $\rightarrow$ R) + (R $\rightarrow$ R/W) *systems.*

Conclusion and future work

- Decidability of the state reachability problem for W $\rightarrow$ R (/W)

Conclusion and future work

- Decidability of the state reachability problem for W $\longrightarrow$ R (/W)
- High complexity (in theory)

  $$\textit{PSPACE-completeness for SC} \rightsquigarrow \textit{Nonprimitive recursiveness}$$

  $\Rightarrow$ Use efficient (sound and complete) analysis based on iterative computations of upper/lower approximations [Geeraerts et al.]

Conclusion and future work

- Decidability of the state reachability problem for W $\longrightarrow$ R (/W)
- High complexity (in theory)

  $$\textit{PSPACE-completeness for SC} \rightsquigarrow \textit{Nonprimitive recursiveness}$$

  $\Rightarrow$ Use efficient (sound and complete) analysis based on iterative computations of upper/lower approximations [Geeraerts et al.]
- Undecidability for the repeated state reachability problem (liveness)

  $\Rightarrow$ Conditions ensuring decidability ?

Conclusion and future work

- Decidability of the state reachability problem for W $\longrightarrow$ R (/W)
- High complexity (in theory)

  $$\textit{PSPACE-completeness for SC} \rightsquigarrow \textit{Nonprimitive recursiveness}$$

  $\Rightarrow$ Use efficient (sound and complete) analysis based on iterative computations of upper/lower approximations [Geeraerts et al.]
- Undecidability for the repeated state reachability problem (liveness)

  $\Rightarrow$ Conditions ensuring decidability ?
- Unbounded R $\longrightarrow$ R/W relaxation (speculation) is dangerous

Conclusion and future work

- Decidability of the state reachability problem for W $\longrightarrow$ R (/W)
- High complexity (in theory)

  $$\textit{PSPACE-completeness for SC} \rightsquigarrow \textit{Nonprimitive recursiveness}$$

  $\Rightarrow$ Use efficient (sound and complete) analysis based on iterative computations of upper/lower approximations [Geeraerts et al.]
- Undecidability for the repeated state reachability problem (liveness)

  $\Rightarrow$ Conditions ensuring decidability ?
- Unbounded R $\longrightarrow$ R/W relaxation (speculation) is dangerous
- Other classes of weak memory models ?

  $\Rightarrow$ Accurate formal models, boundaries of decidability/complexity, etc.

Conclusion and future work

- Decidability of the state reachability problem for W $\rightarrow$ R (/W)
- High complexity (in theory)

  $$\textit{PSPACE-completeness for SC} \rightsquigarrow \textit{Nonprimitive recursiveness}$$

  $\Rightarrow$ Use efficient (sound and complete) analysis based on iterative computations of upper/lower approximations [Geeraerts et al.]
- Undecidability for the repeated state reachability problem (liveness)

  $\Rightarrow$ Conditions ensuring decidability ?
- Unbounded R $\rightarrow$ R/W relaxation (speculation) is dangerous
- Other classes of weak memory models ?

  $\Rightarrow$ Accurate formal models, boundaries of decidability/complexity, etc.
- Fence insertion (make a program robust wrt SC)

Some work on WMM

- Adve and Gharachorloo,
  Shared Memory Consistency Models: A Tutorial, 1995
- S. Park, D. Dill,
  An executable specification, analyzer and verifier for RMO, 1995
- P. Chatterjee, G. Gopalakrishnan,
  A Specification and Verification Framework for Developing Weak Shared Memory Consistency Protocols, 2002
- S. Burckhardt, R. Alur, M. K. Martin,
  CheckFence: checking consistency of concurrent data types on relaxed memory models, 2007
- S. Burckhardt, M. Musuvathi,
  Effective Program Verification for Relaxed Memory Models, 2008
- S. Sarkar, P. Sewell, F. Zappa Nardelli, S. Owens, T. Ridge, T. Braibant, M. O. Myreen, J. Alglave,
  The semantics of x86-CC multiprocessor machine code, 2009
- G. Boudol, G. Petri,
  Relaxed memory models: an operational approach, 2009