Universality Issues in Reversible Computing Systems and Cellular Automata

Kenichi Morita
Hiroshima University

Contents

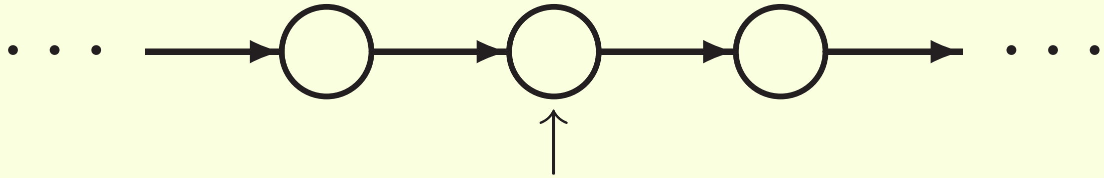
1. Introduction
2. Reversible Turing machines (RTMs)
3. Reversible logic elements and circuits
4. Reversible cellular automata (RCAs)

Even very simple reversible systems have universal computing ability!

1. Introduction

Reversible Computing

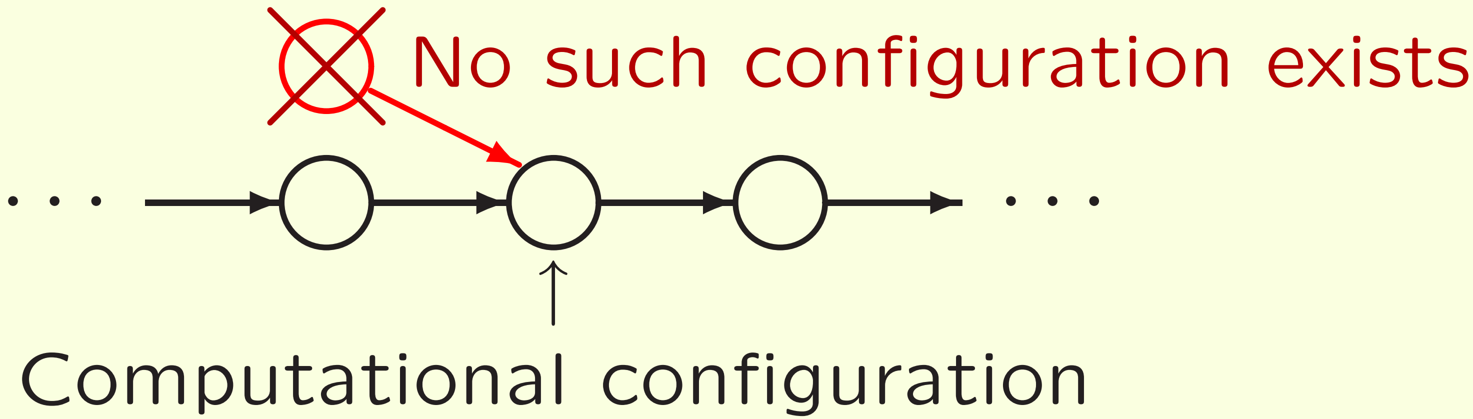
- Roughly speaking, it is a "backward deterministic" computing; i.e., every computational configuration has at most one predecessor.



- Though its definition is rather simple, it reflects physical reversibility well.

Reversible Computing

- Roughly speaking, it is a “backward deterministic” computing; i.e., every computational configuration has at most one predecessor.



- Though its definition is rather simple, it reflects physical reversibility well.

Several Models of Reversible Computing

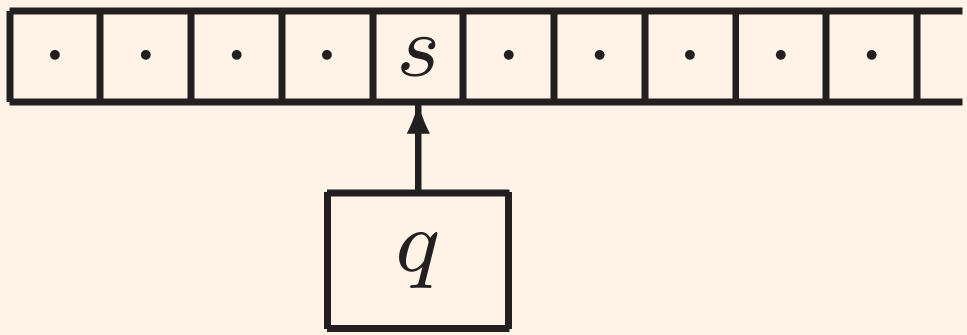
- Reversible Turing machines (RTMs)
- Reversible logic elements and circuits
- Reversible cellular automata (RCAs)
- Reversible counter machines (RCMs)
- Others

- These models are closely related each other.
- Reversible computers work in a very different fashion from classical computers!

2. Reversible Turing Machines

Reversible Turing Machines (RTMs)

A "backward deterministic" TM.



Definition of a TM

$$T = (Q, S, q_0, q_f, s_0, \delta)$$

Q: a finite set of states.
S: a finite set of tape symbols.
q_0: an initial state $q_0 \in Q$.
q_f: a final state $q_f \in Q$.
s_0: a blank symbol $s_0 \in S$.
δ: a move relation given by a set of quintuples $[p, s, s', d, q] \in Q \times S \times S \times \{-, 0, +\} \times Q$.

Definition of an RTM

A TM $T = (Q, S, q_0, q_f, s_0, \delta)$ is called *reversible* iff the following condition holds for any pair of distinct quintuples $[p_1, s_1, s'_1, d_1, q_1]$ and $[p_2, s_2, s'_2, d_2, q_2]$.

If $q_1 = q_2$, then $s'_1 \neq s'_2 \wedge d_1 = d_2$

(If the next states are the same, then the written symbols must be different and the shift directions must be the same.)

Universality of RTMs

Theorem [Bennett, 1973]
For any one-tape (irreversible) TM T, there is a garbage-less 3-tape reversible TM which simulates the former.

A Small Universal RTM (URTM)

A URTM is an RTM that can compute *any* recursive function.

Theorem The following URTMs exist:
17-state 5-symbol URTM [Morita and Yamaguchi, 2007]
15-state 6-symbol URTM [Morita, 2008]

A Small Universal RTM (URTM)

A URTM is an RTM that can compute *any* recursive function.

Theorem The following URTMs exist:
17-state 5-symbol URTM [Morita and Yamaguchi, 2007]
15-state 6-symbol URTM [Morita, 2008]

These URTMs can simulate any cyclic tag system [Cook, 2004], which is proved to be universal.

Cyclic Tag System (CTAG) [Cook, 2004]

$$C = (k, \{Y, N\}, (\mathsf{halt}, p_1, \cdots, p_{k-1}))$$

- k: the length of a cycle (positive integer).
- $\{Y, N\}$: the alphabet used in a CTAG.
- $(p_1, \cdots, p_{k-1}) \in (\{Y, N\}^*)^{k-1}$: production rules.

An *instantaneous description* (ID) is a pair (v, i), where $v \in \{Y, N\}^*$ and $i \in \{0, \cdots, k-1\}$.

For any $(v, i), (w, j) \in \{Y, N\}^* \times \{0, \cdots, k-1\}$,

$$\begin{aligned}
(Yv, i) \Rightarrow (w, j) \text{ iff } & [m \neq 0] \wedge [j = i + 1 \bmod k] \\
& \wedge [w = vp_i], \\
(Nv, i) \Rightarrow (w, j) \text{ iff } & [j = i + 1 \bmod k] \wedge [w = v].
\end{aligned}$$

A Simple Example of a CTAG System

$$C_1 = (3, \{Y, N\}, (\text{halt}, \; YN, \; YY))$$

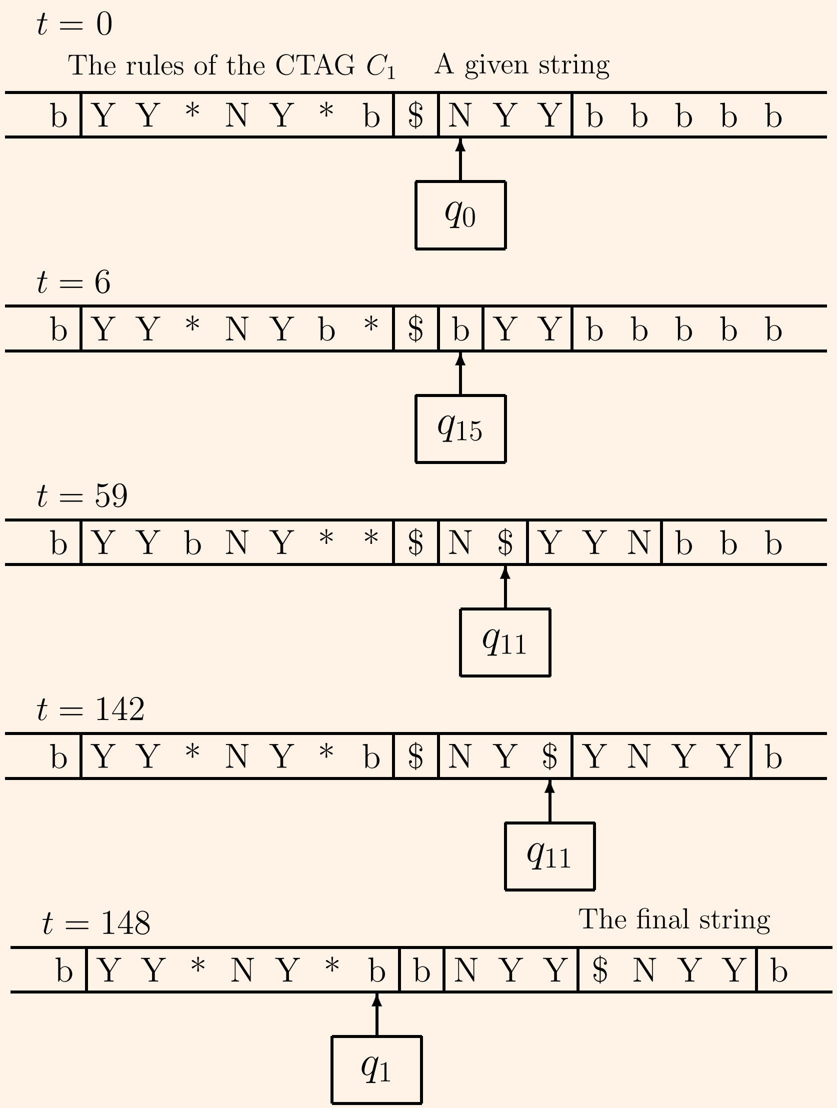
If an initial word NYY is given, the computing on C_1 proceeds as follows:

$$
\begin{aligned}
& (\; N\,Y\,Y \;, \quad 0\;) \\
\Rightarrow & (\; Y\,Y \;, \quad 1\;) \\
\Rightarrow & (\; Y\,Y\,N \;, \quad 2\;) \\
\Rightarrow & (\; Y\,N\,Y\,Y \;, \quad 0\;)
\end{aligned}
$$

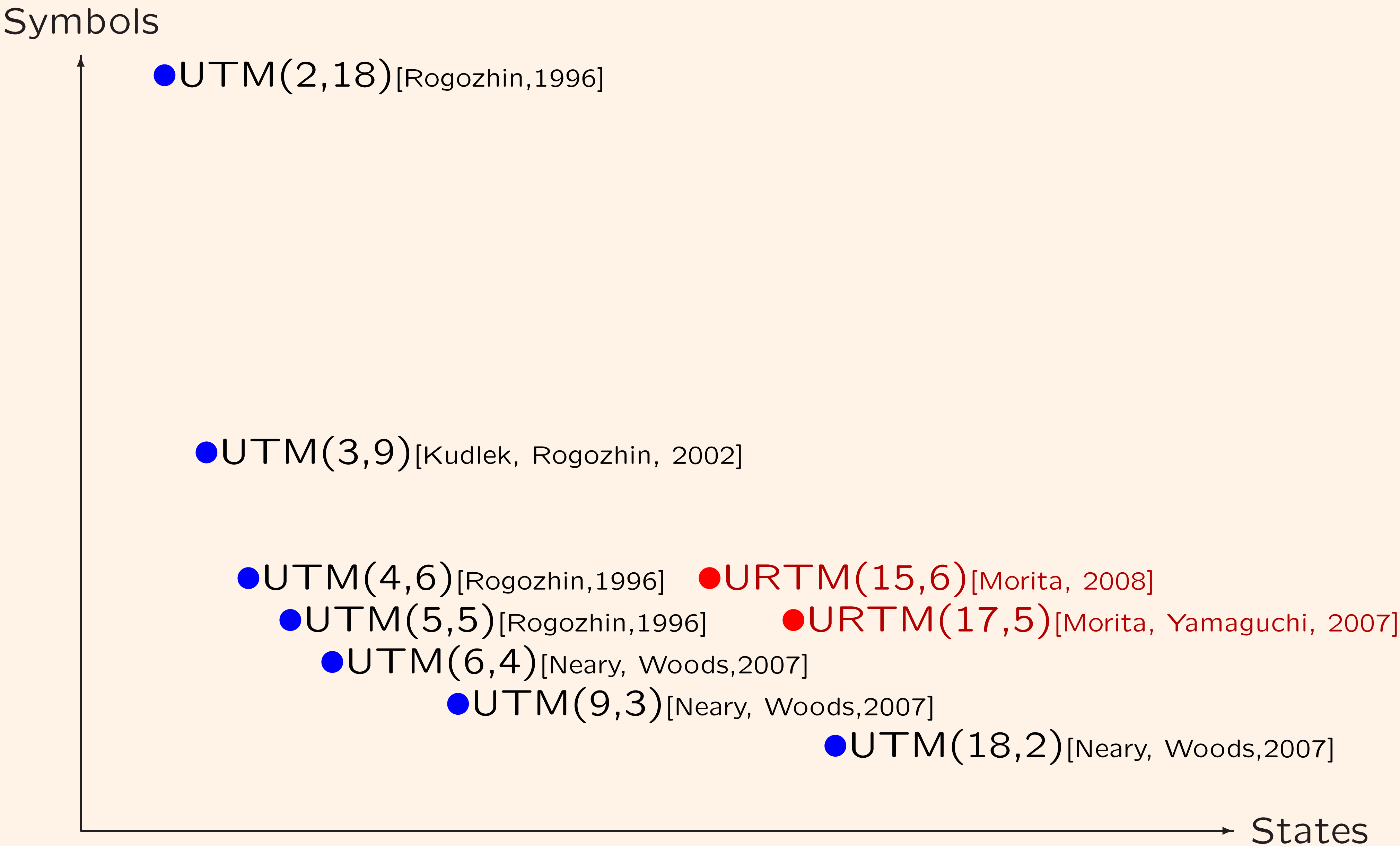
The quintuple set of the URTM(17,5)

	b	Y	N	$*$	$
q_0	$\$ - q_2$	$\$ - q_1$	$b - q_{13}$		
q_1	halt	$Y - q_1$	$N - q_1$	$* + q_0$	$b - q_1$
q_2	$* - q_3$	$Y - q_2$	$N - q_2$	$* - q_2$	null
q_3	$b + q_{12}$	$b + q_4$	$b + q_7$	$b + q_{10}$	
q_4	$Y + q_5$	$Y + q_4$	$N + q_4$	$* + q_4$	$\$ + q_4$
q_5	$b - q_6$				
q_6	$Y - q_3$	$Y - q_6$	$N - q_6$	$* - q_6$	$\$ - q_6$
q_7	$N + q_8$	$Y + q_7$	$N + q_7$	$* + q_7$	$\$ + q_7$
q_8	$b - q_9$				
q_9	$N - q_3$	$Y - q_9$	$N - q_9$	$* - q_9$	$\$ - q_9$
q_{10}		$Y + q_{10}$	$N + q_{10}$	$* + q_{10}$	$\$ + q_{11}$
q_{11}		$Y + q_{11}$	$N + q_{11}$	$* + q_{11}$	$Y + q_0$
q_{12}		$Y + q_{12}$	$N + q_{12}$	$* + q_{12}$	$\$ - q_3$
q_{13}	$* - q_{14}$	$Y - q_{13}$	$N - q_{13}$	$* - q_{13}$	$\$ - q_{13}$
q_{14}	$b + q_{16}$	$Y - q_{14}$	$N - q_{14}$	$b + q_{15}$	
q_{15}	$N + q_0$	$Y + q_{15}$	$N + q_{15}$	$* + q_{15}$	$\$ + q_{15}$
q_{16}		$Y + q_{16}$	$N + q_{16}$	$* + q_{16}$	$\$ - q_{14}$

Simulating the CTAG C_1 by the URTM(17,5)



Small UTMs and URTMs



3. Reversible Logic Elements

Reversible Logic Element

A logic element whose function is described by a one-to-one mapping.

(1) Reversible logic elements without memory
(i.e., reversible logic gates):

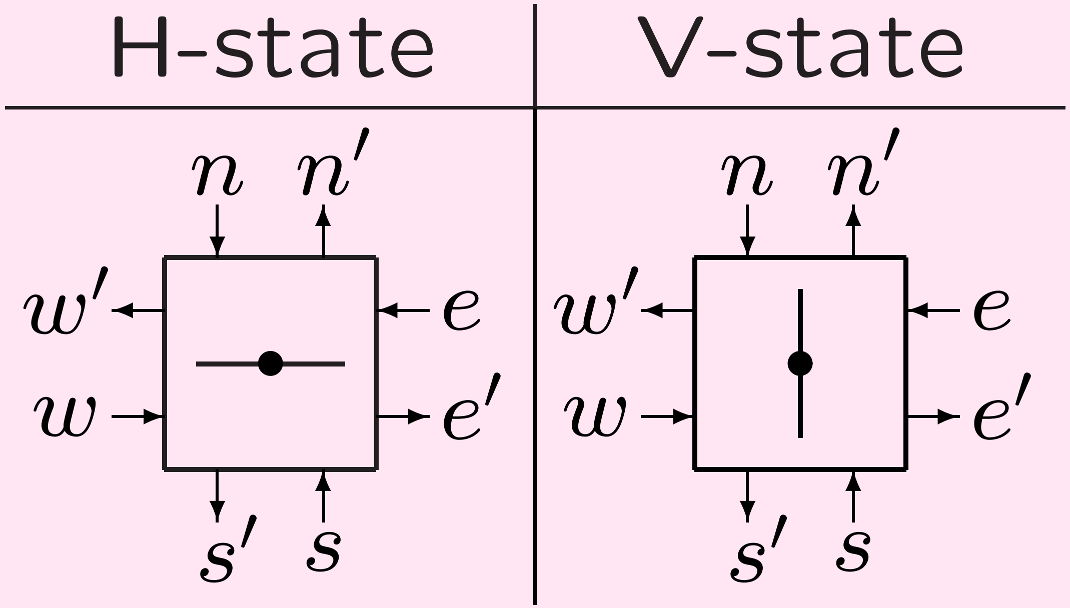
- Toffoli gate [Toffoli, 1980]
- Fredkin gate [Fredkin and Toffoli, 1982]
- etc.

(2) Reversible logic elements with memory:

- Rotary element (RE) [Morita, 2001]
- etc.

Rotary element (RE)

A 2-state 4-input-line 4-output-line element.



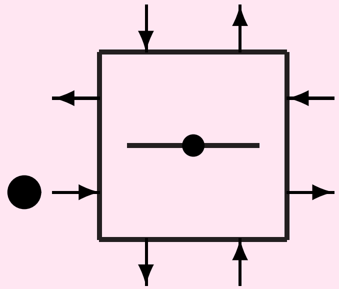
(Remark)

We assume signal "1" is given at most one input line.

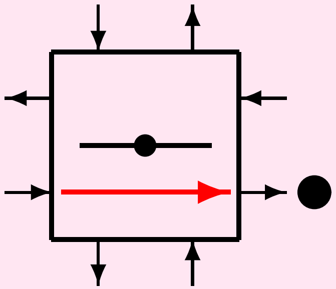
Operations of an RE

- Parallel case:

$t = 0$

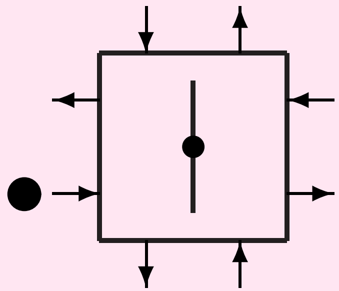


$t = 1$

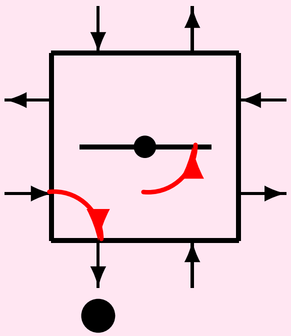


- Orthogonal case:

$t = 0$

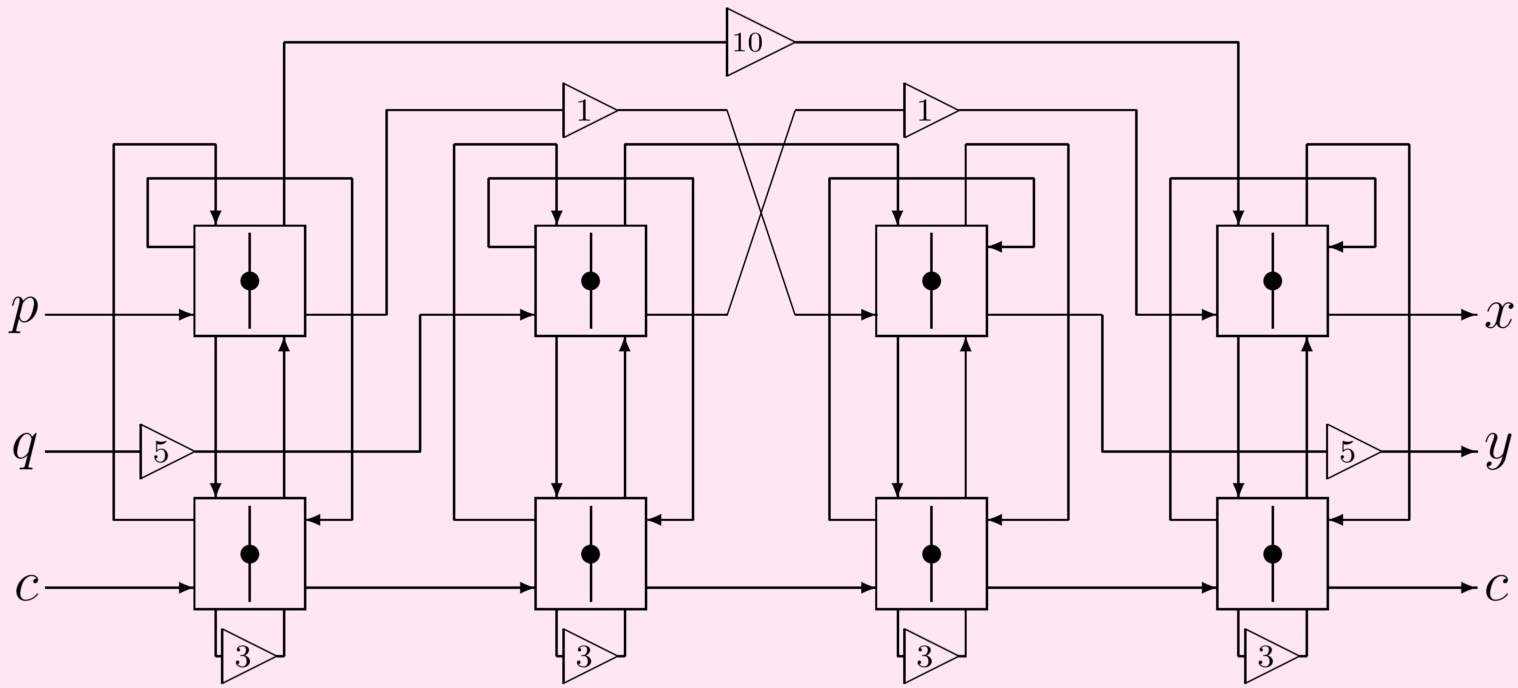


$t = 1$



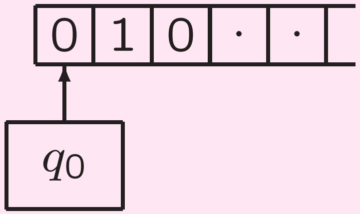
Logical Universality of a Rotary Element

A Fredkin gate can be composed of REs and delay elements.

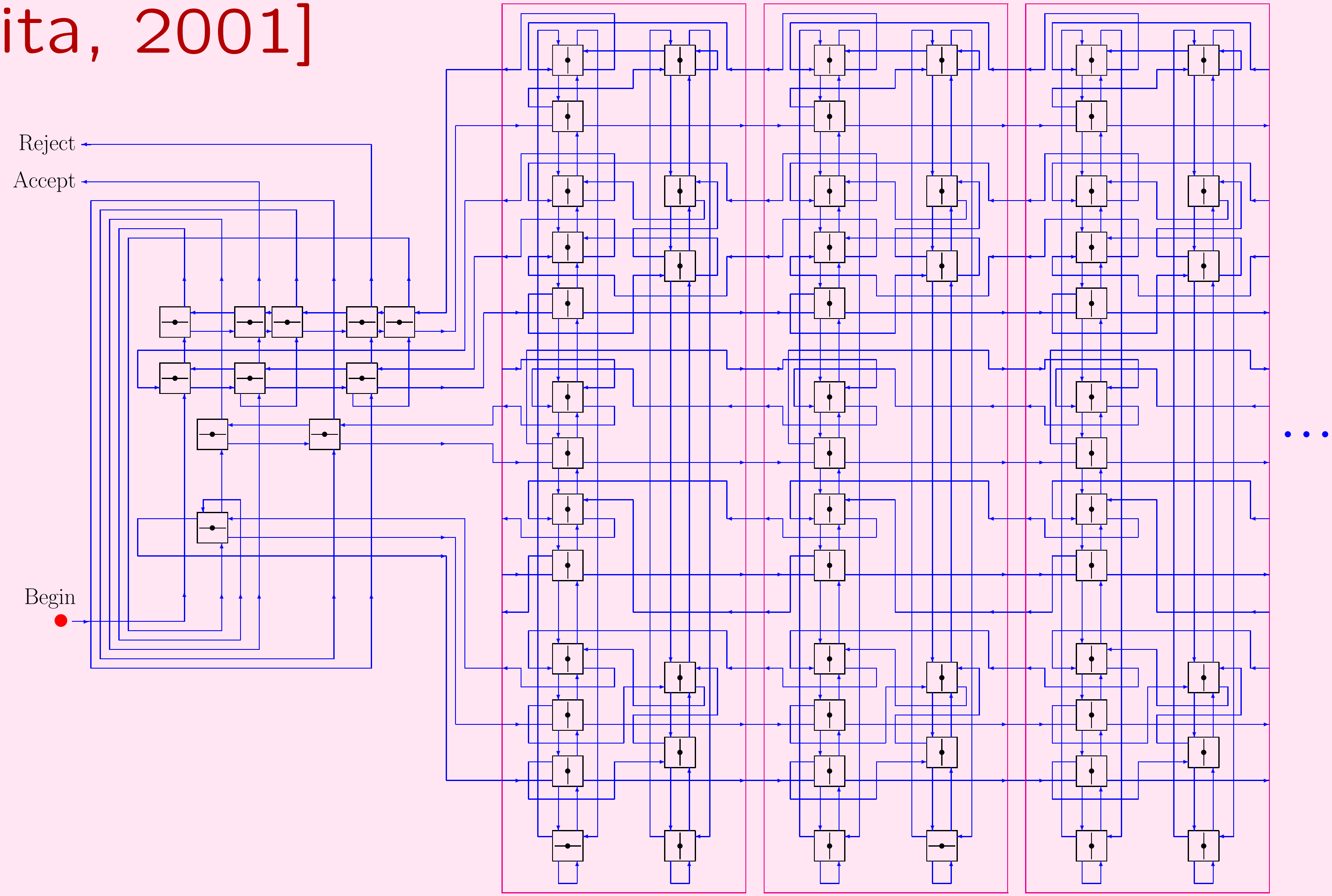


(Remark) But, this is not a good method to use REs.

Any Reversible Turing Machine Can Be Composed Only of REs



[Morita, 2001]



A Simple Example of an RTM T_{parity}

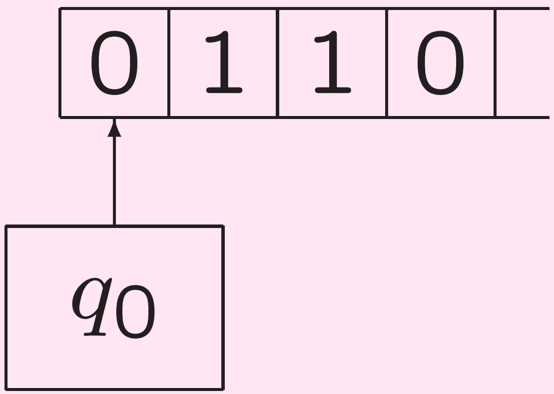
$$T_{\mathsf{parity}} = (Q, \{0, 1\}, q_0, q_{\mathsf{acc}}, 0, \delta)$$

$$Q = \{q_0, q_1, q_2, q_{\mathsf{acc}}, q_{\mathsf{rej}}\}$$

$$\begin{aligned}\delta = \{ & [\, q_0, 0, 1, R, q_1 \,], \\ & [\, q_1, 0, 1, N, q_{\mathsf{acc}} \,], \\ & [\, q_1, 1, 0, R, q_2 \,], \\ & [\, q_2, 0, 1, N, q_{\mathsf{rej}} \,], \\ & [\, q_2, 1, 0, R, q_1 \,] \ \}.\end{aligned}$$

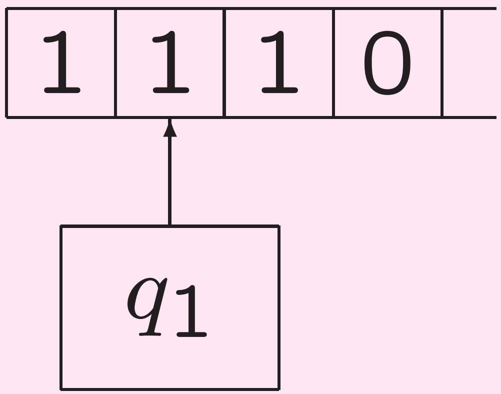
A Simple Example of an RTM T_{parity}

$$t = 0$$



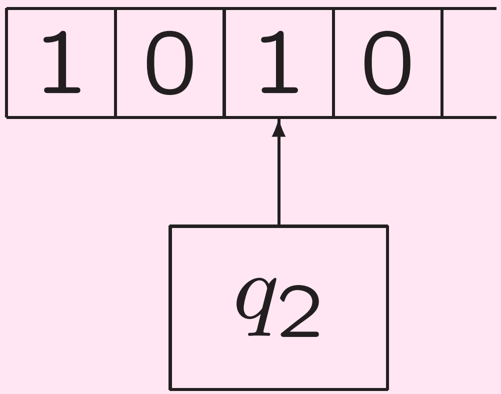
A Simple Example of an RTM T_{parity}

$t = 1$



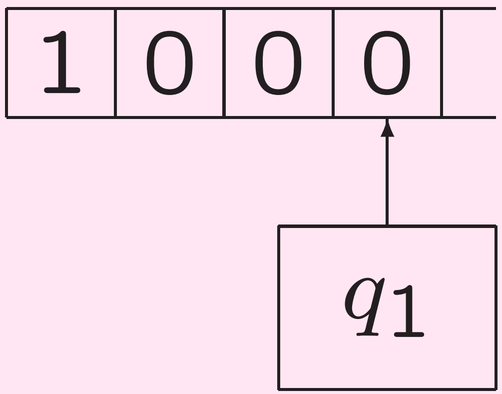
A Simple Example of an RTM T_{parity}

$$t = 2$$



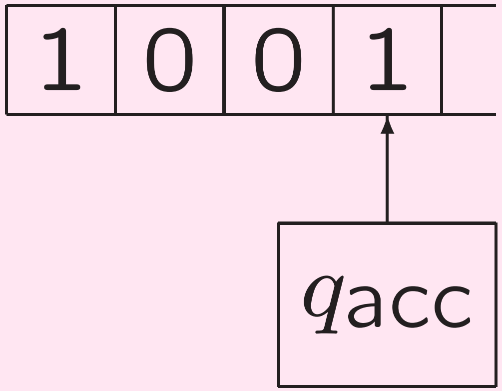
A Simple Example of an RTM T_{parity}

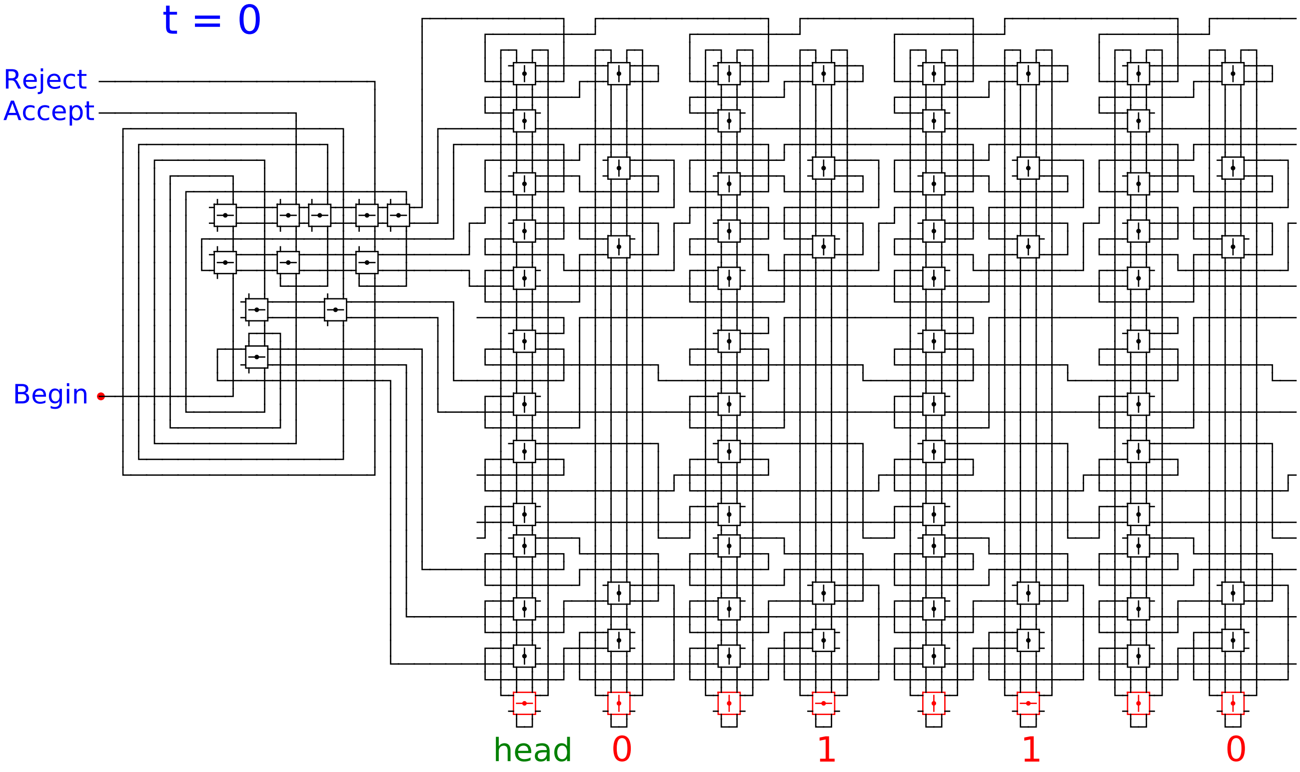
$t = 3$

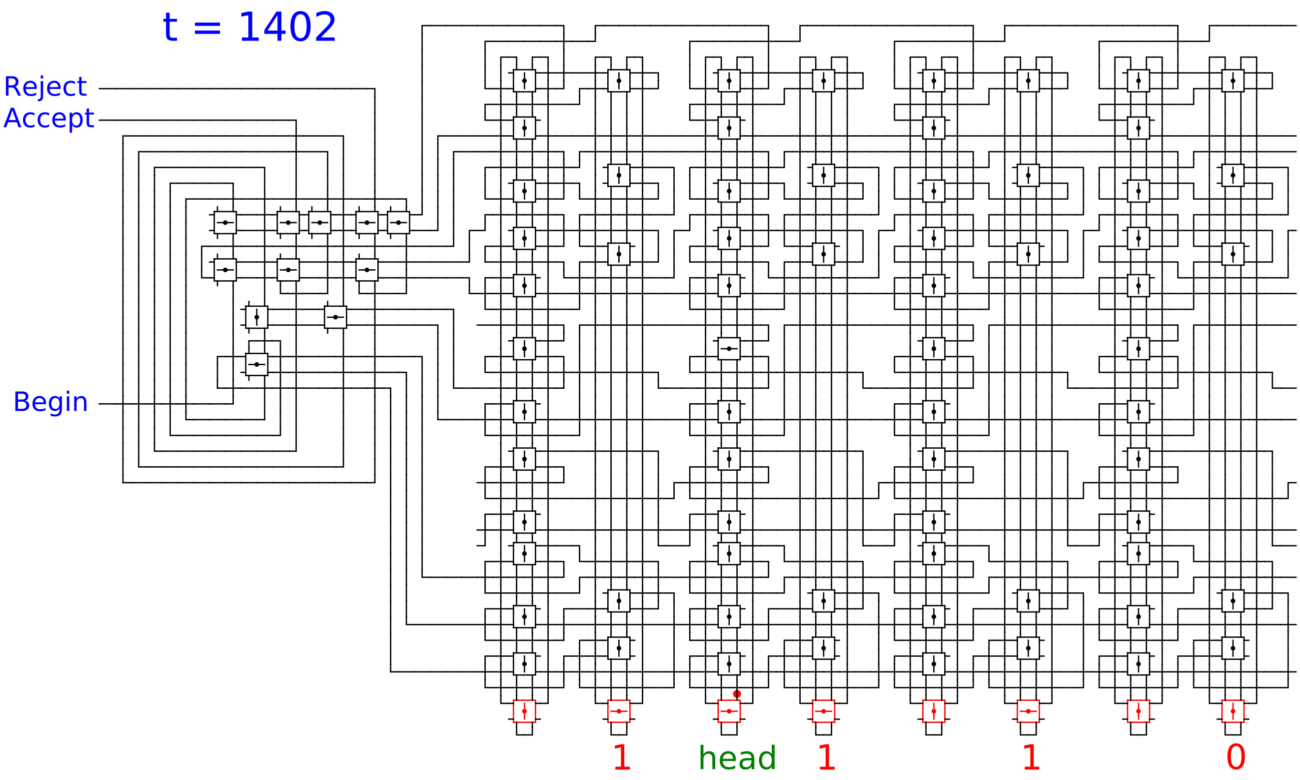


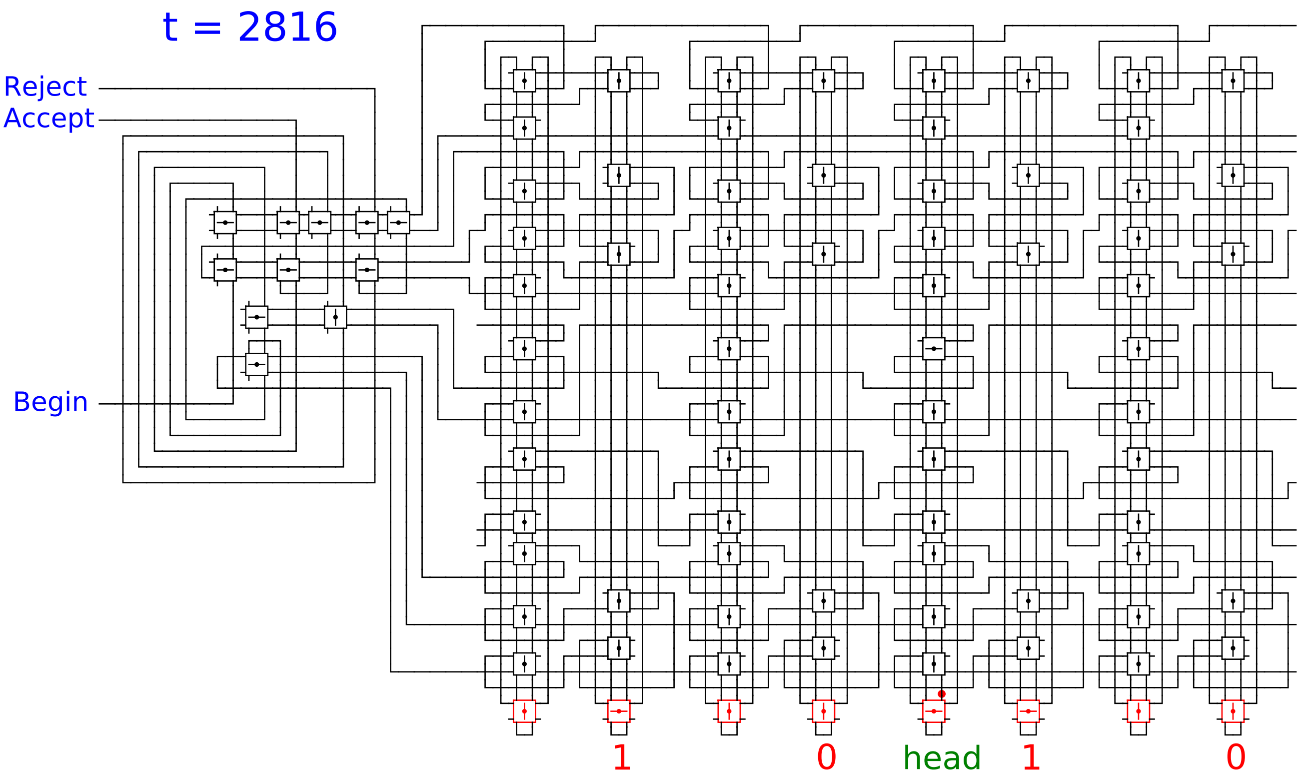
A Simple Example of an RTM T_{parity}

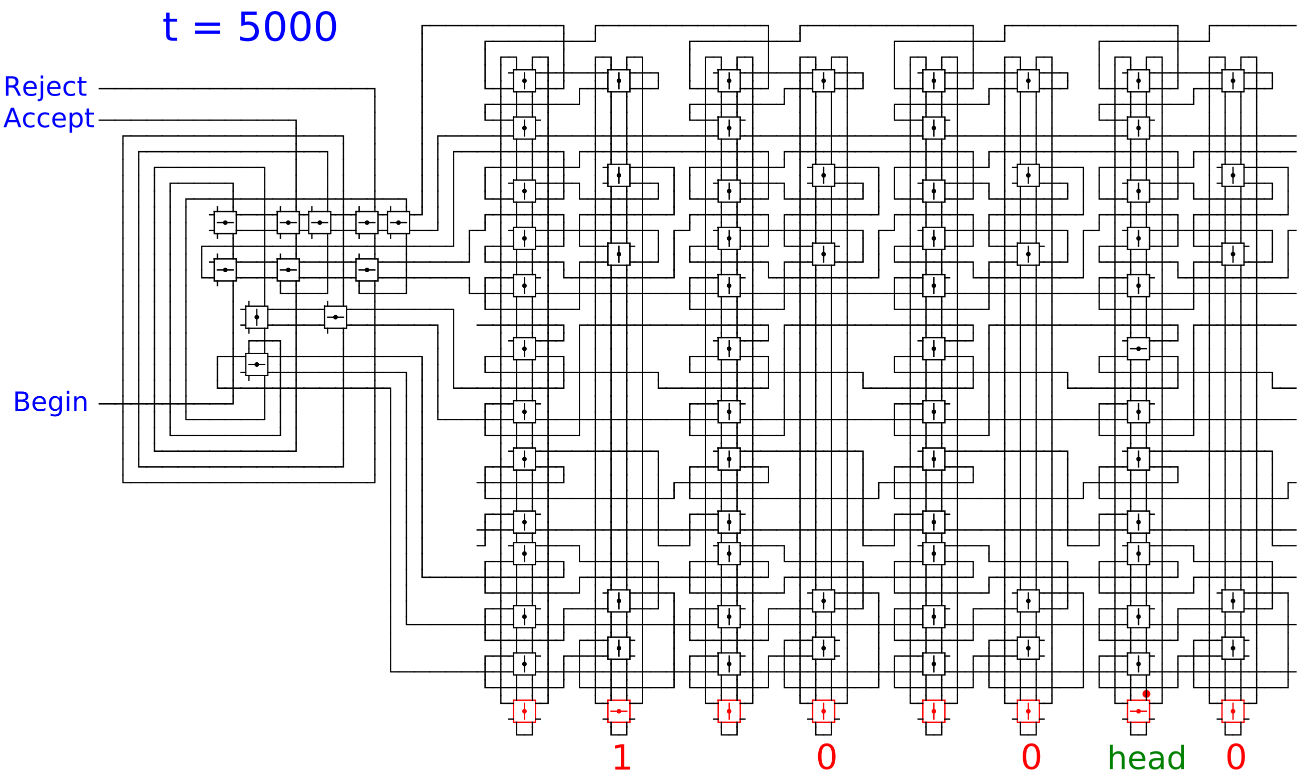
$$t = 3$$



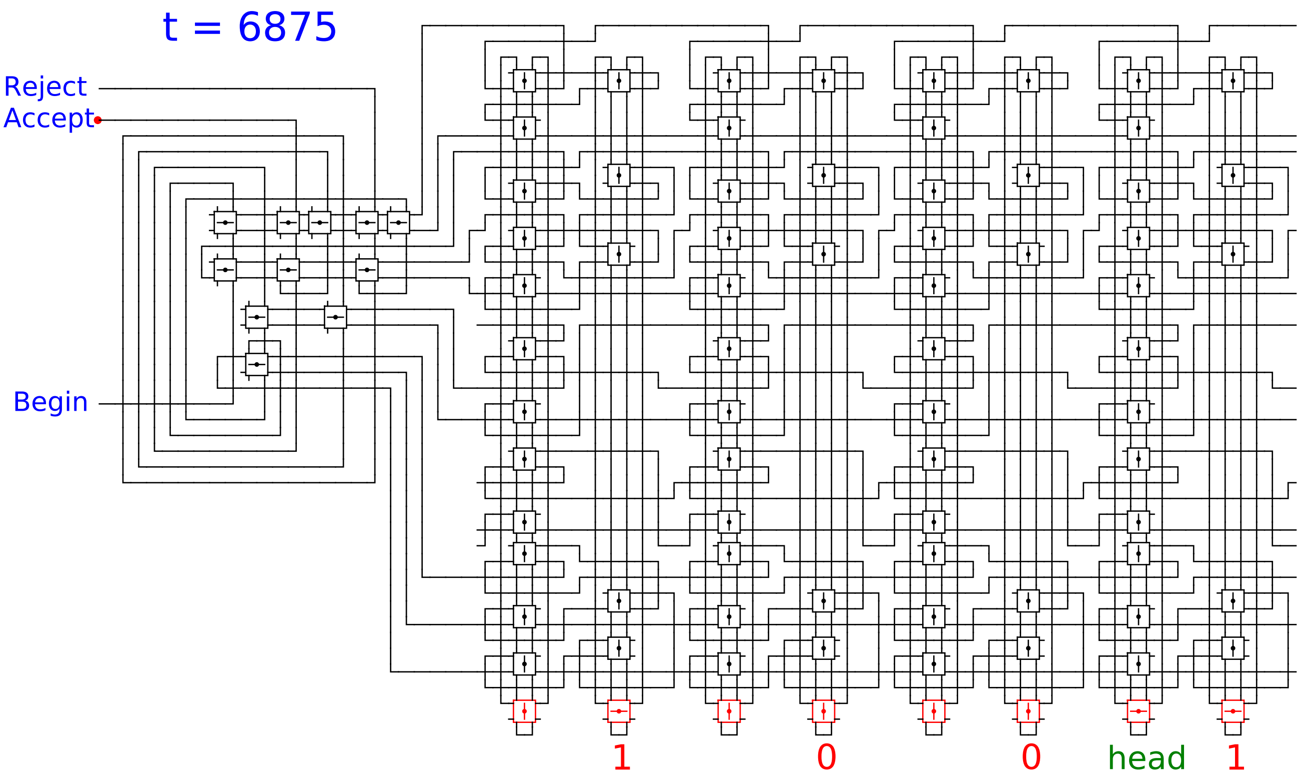

t = 0
Reject
Accept
Begin
head
0
1
1
0


t = 1402
Reject
Accept
Begin
1
head
1
1
0


t = 2816
Reject
Accept
Begin
1
0
head
1
0

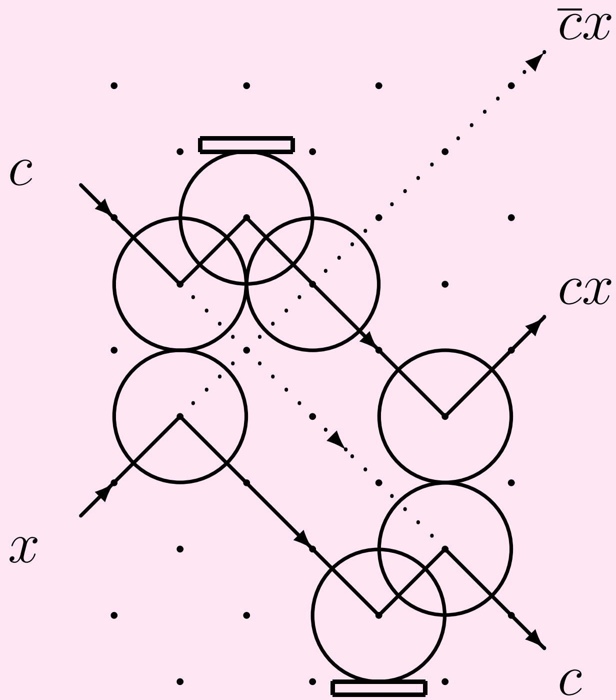


t = 5000
Reject
Accept
Begin
1
0
0
head
0

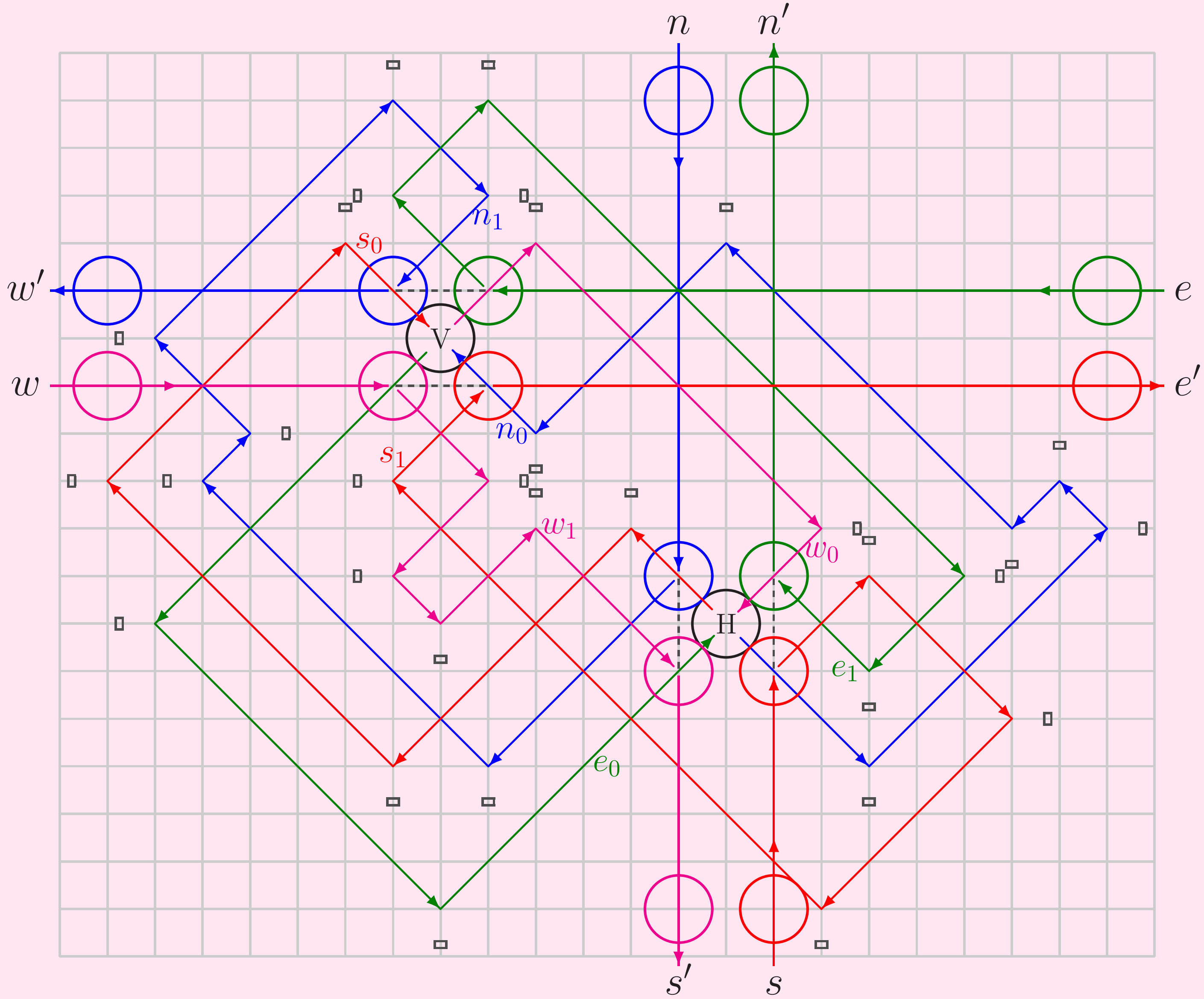

t = 6875
Reject
Accept
Begin
1
0
0
head
1

Billiard Ball Model (BBM)
– A reversible physical model of computing –

[Fredkin and Toffoli, 1982]

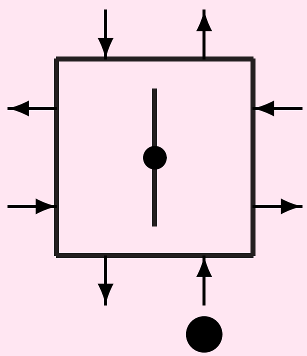


Realization of an RE by BBM [Morita, 2008]

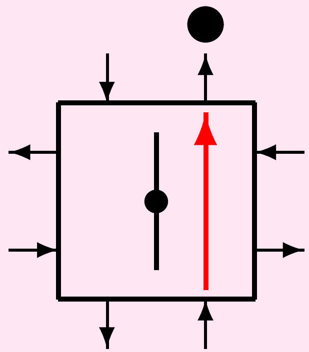


Parallel Case

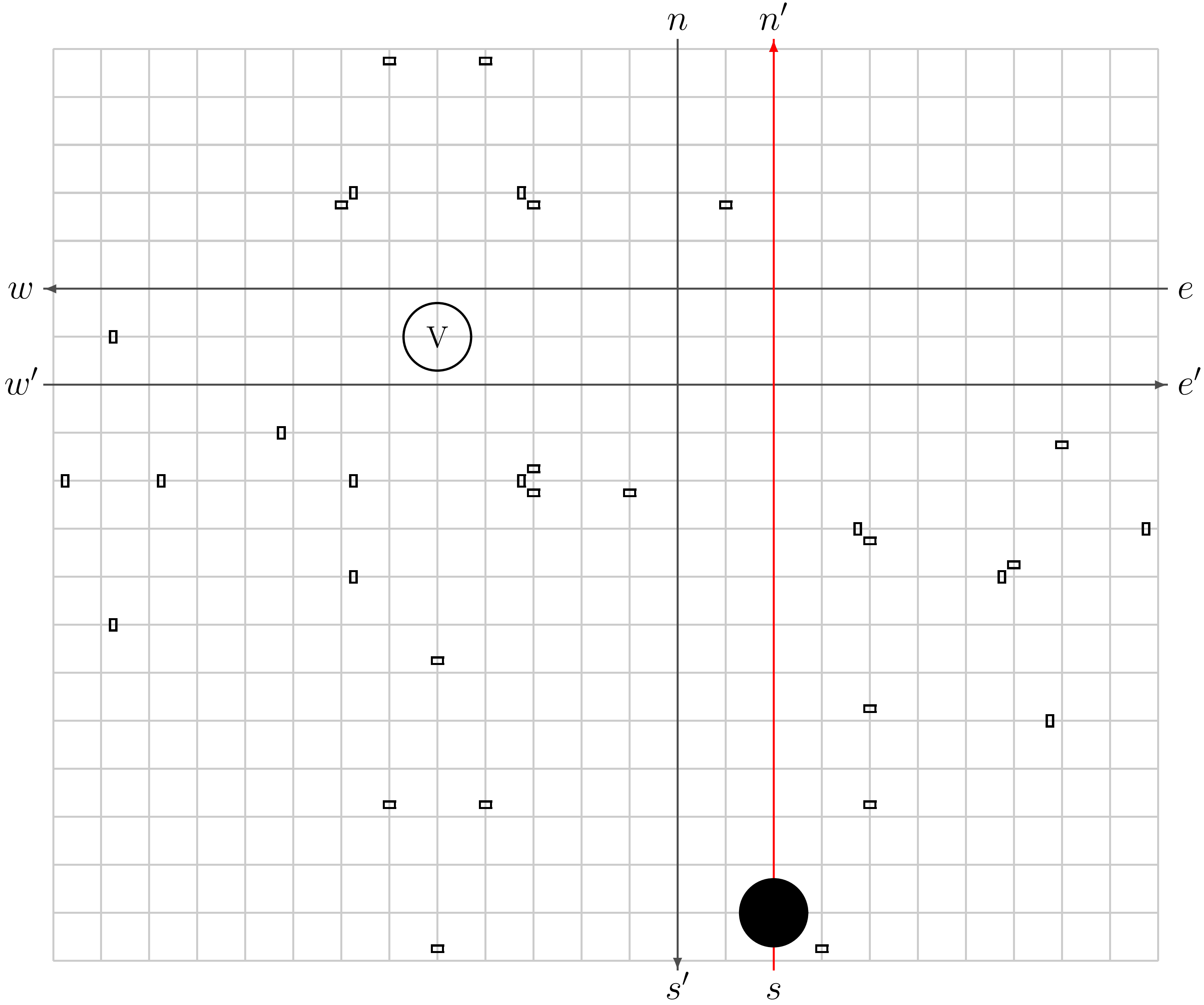
$t = 0$



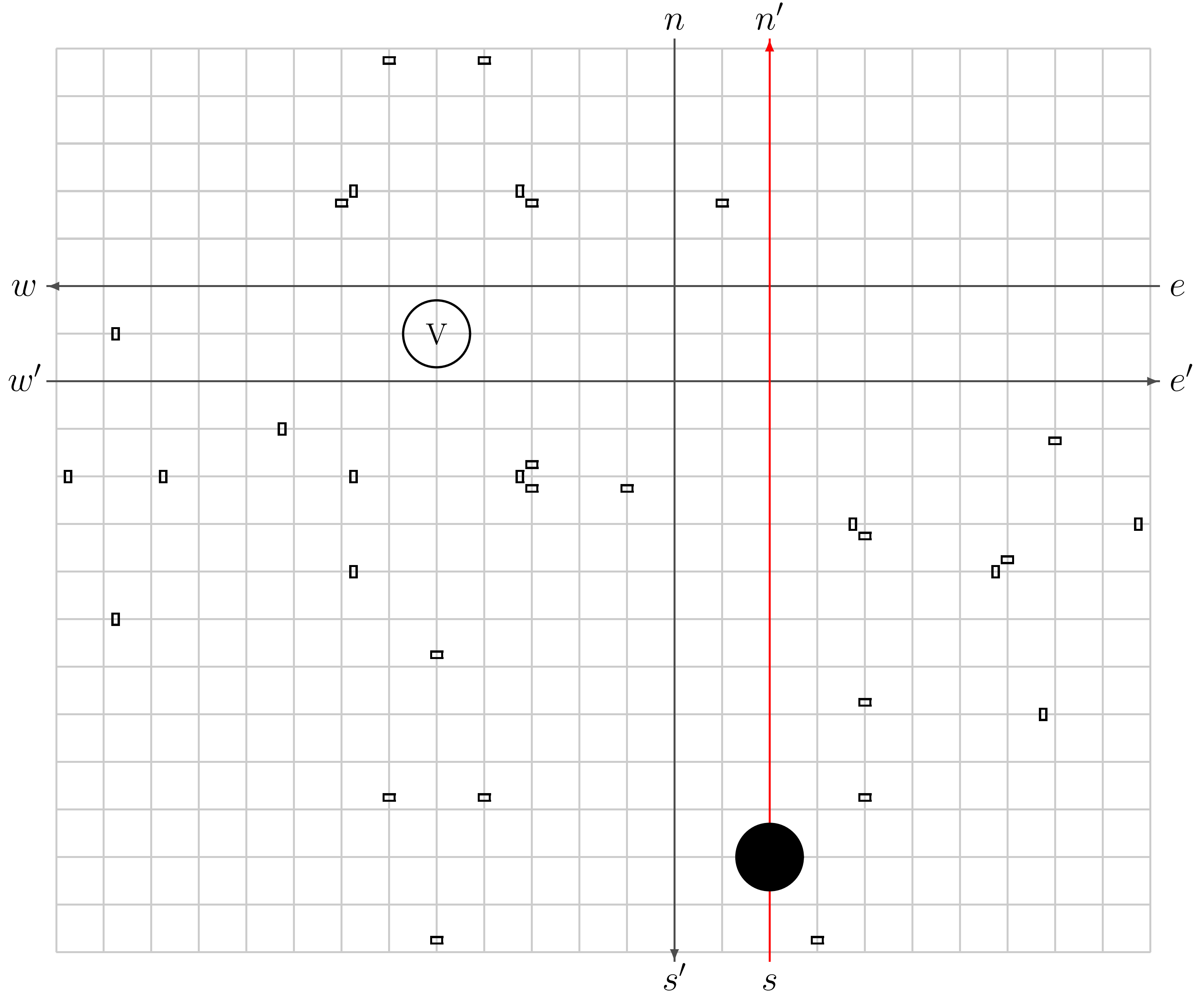
$t = 1$



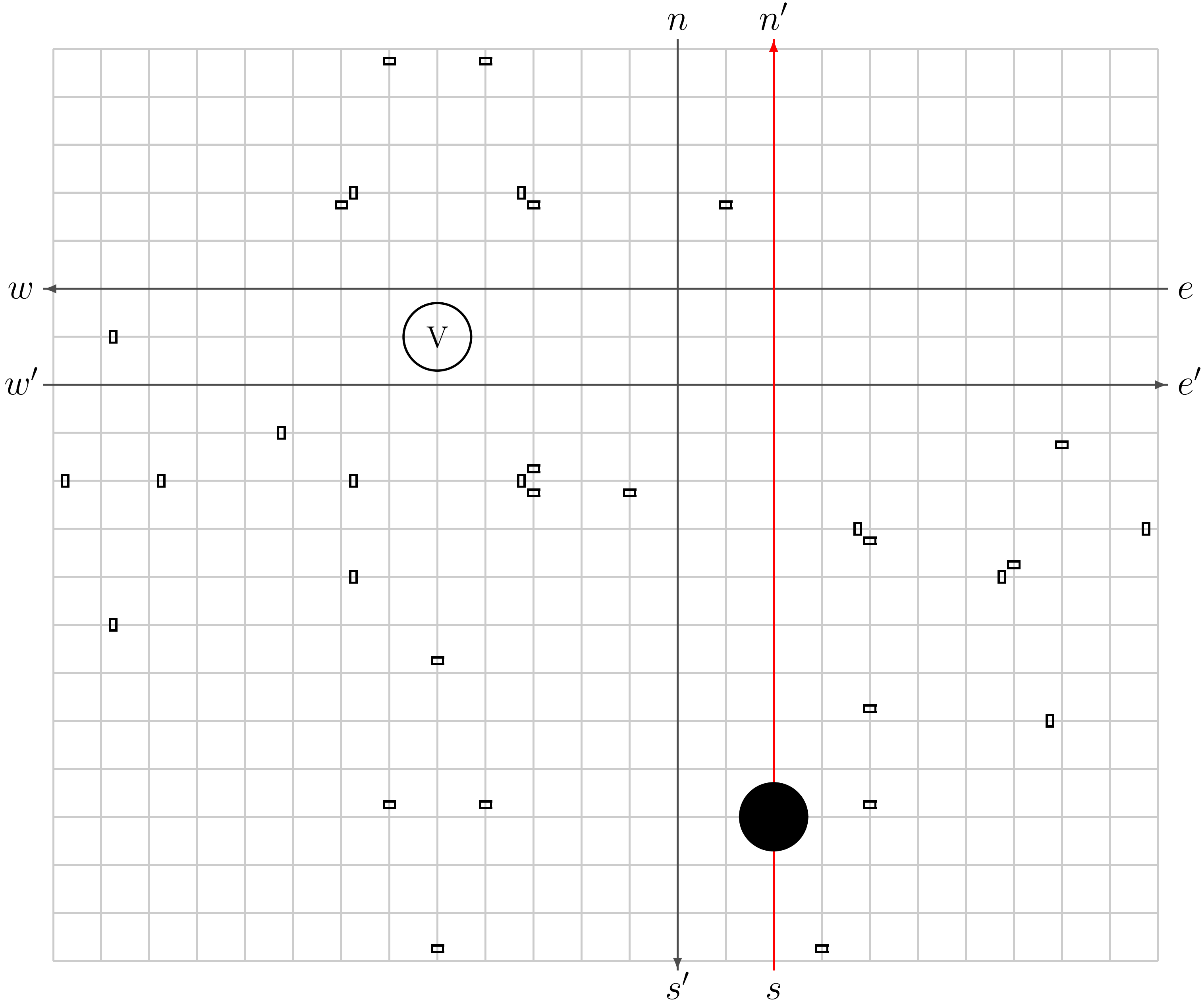
Movements of Balls (State: V, Input: s)



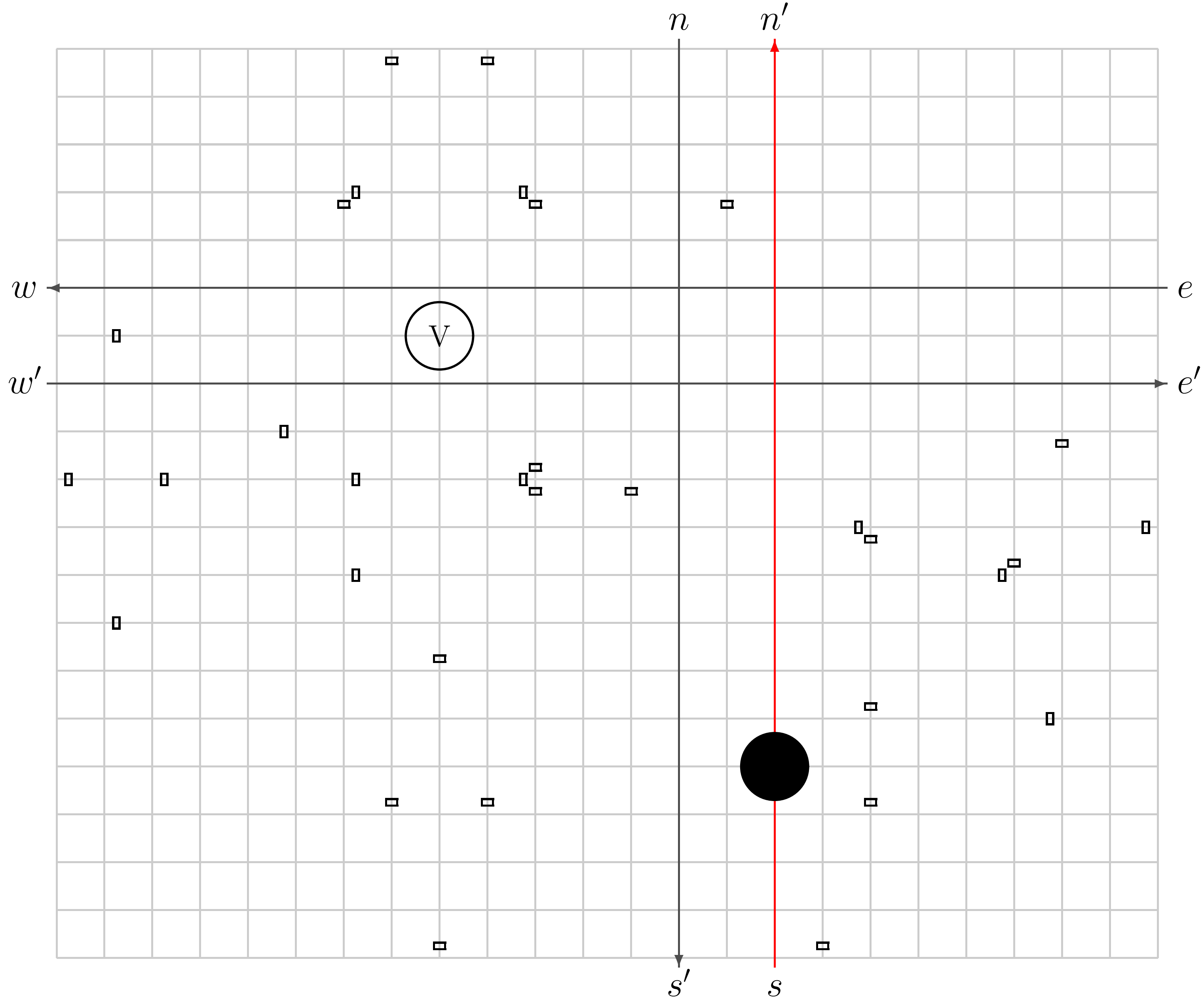
Movements of Balls (State: V, Input: s)



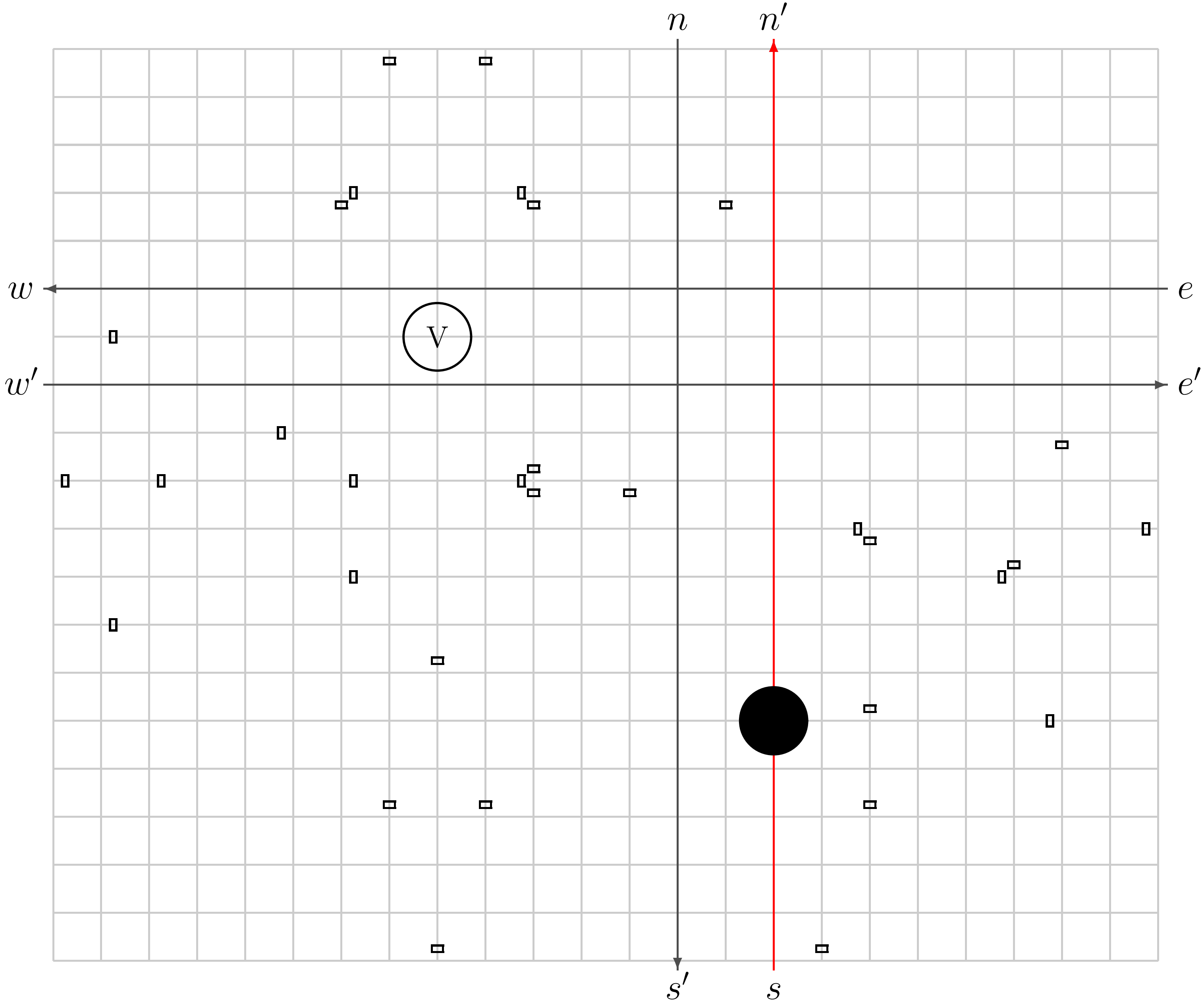
Movements of Balls (State: V, Input: s)



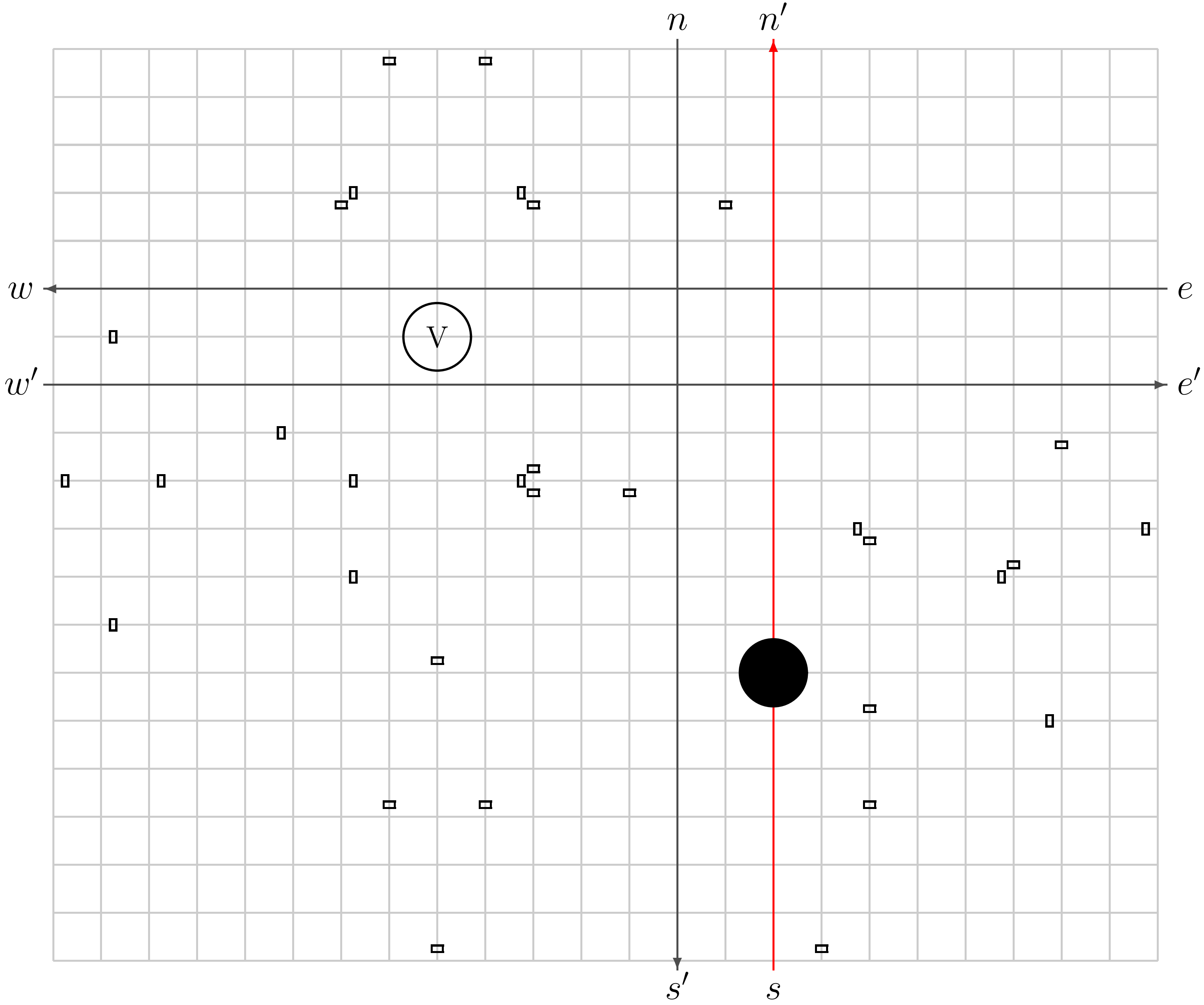
Movements of Balls (State: V, Input: s)



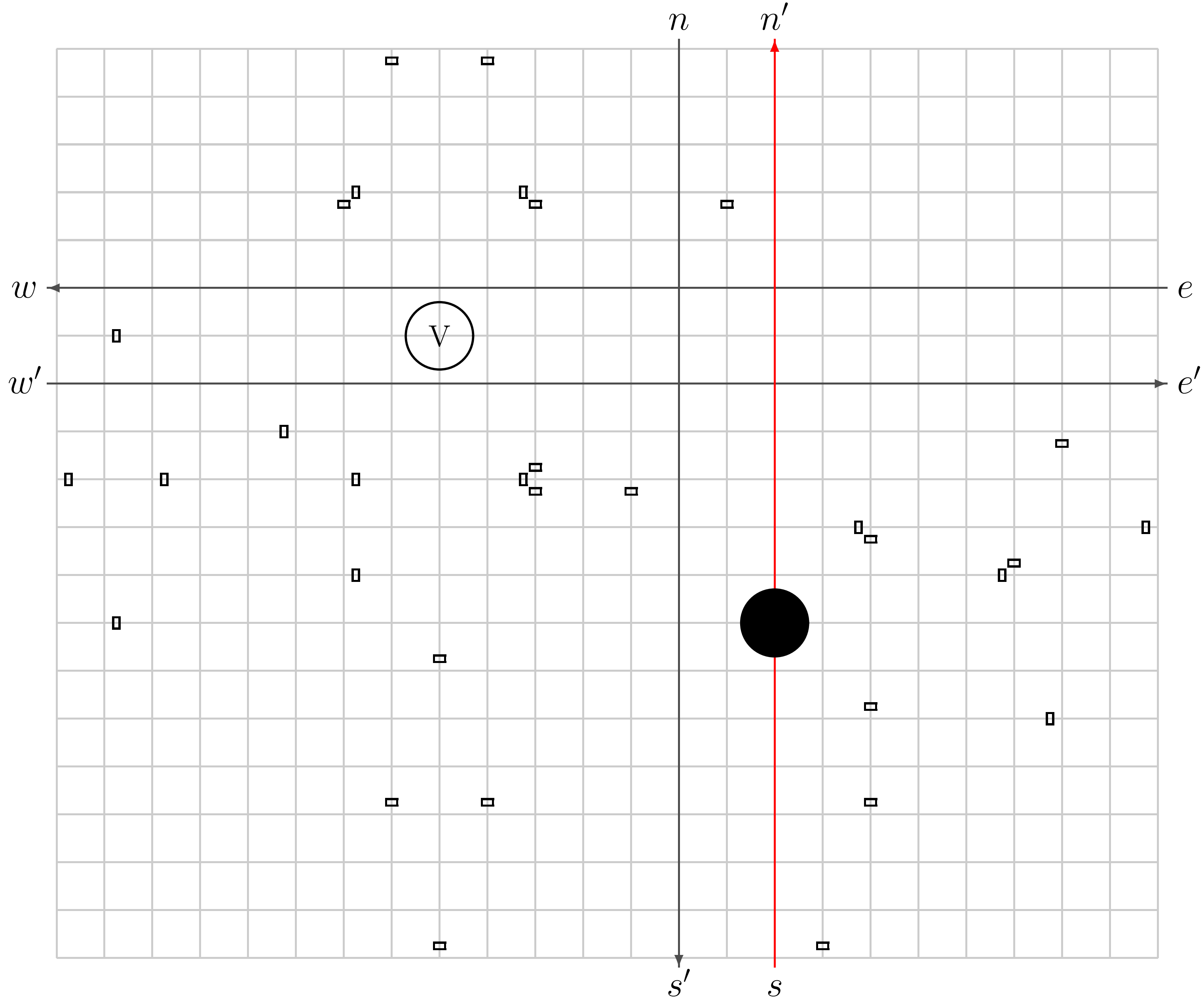
Movements of Balls (State: V, Input: s)



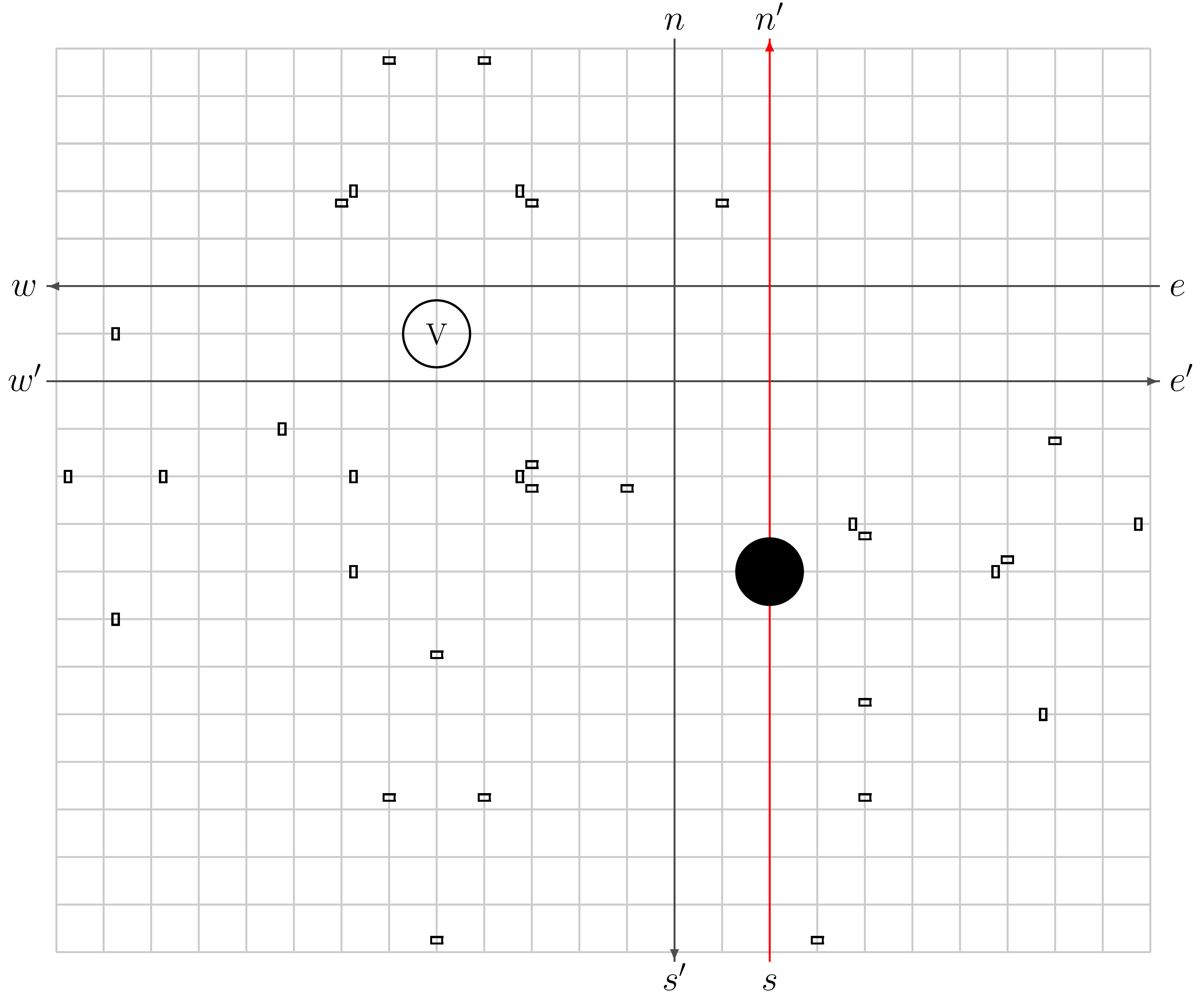
Movements of Balls (State: V, Input: s)



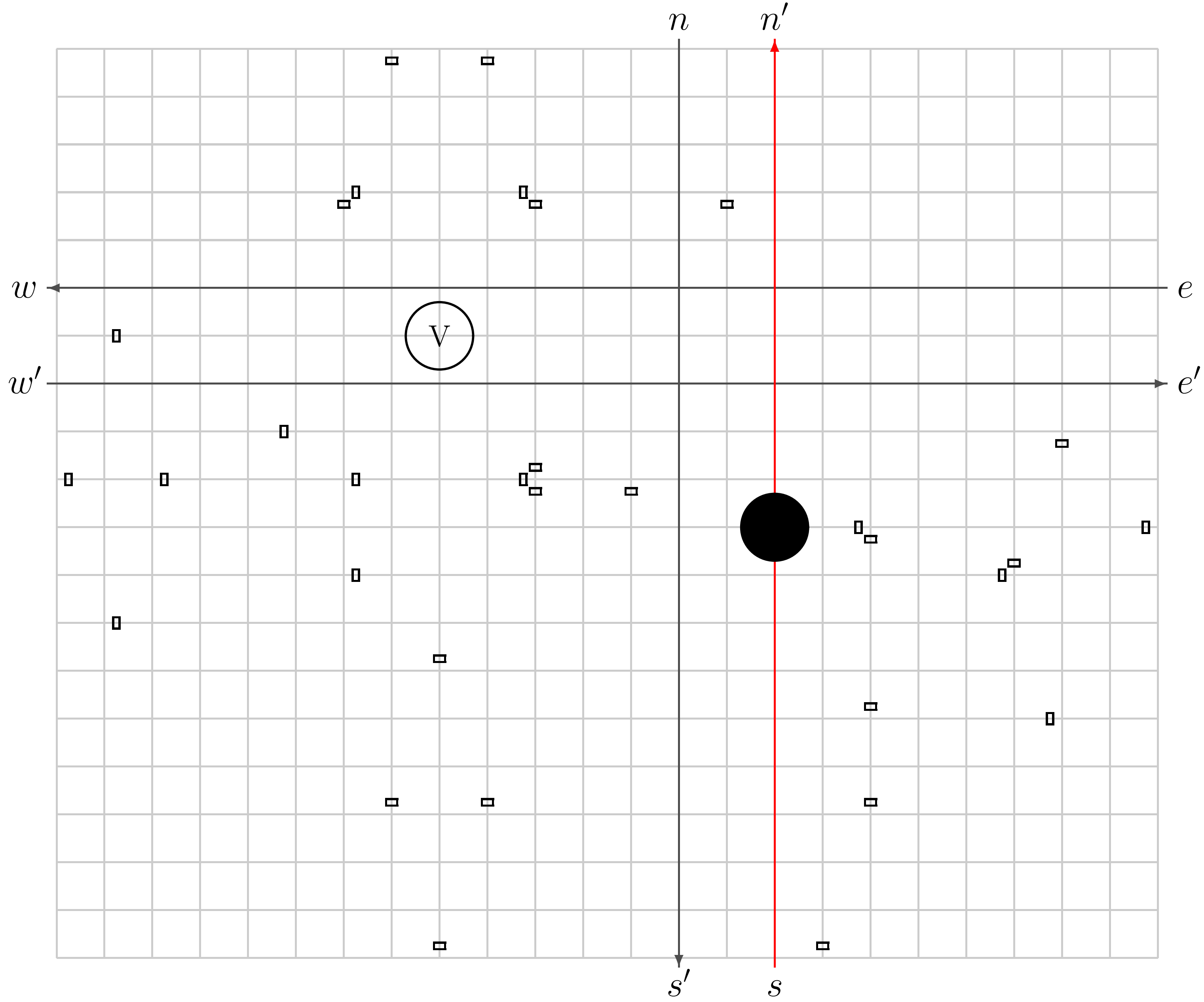
Movements of Balls (State: V, Input: s)



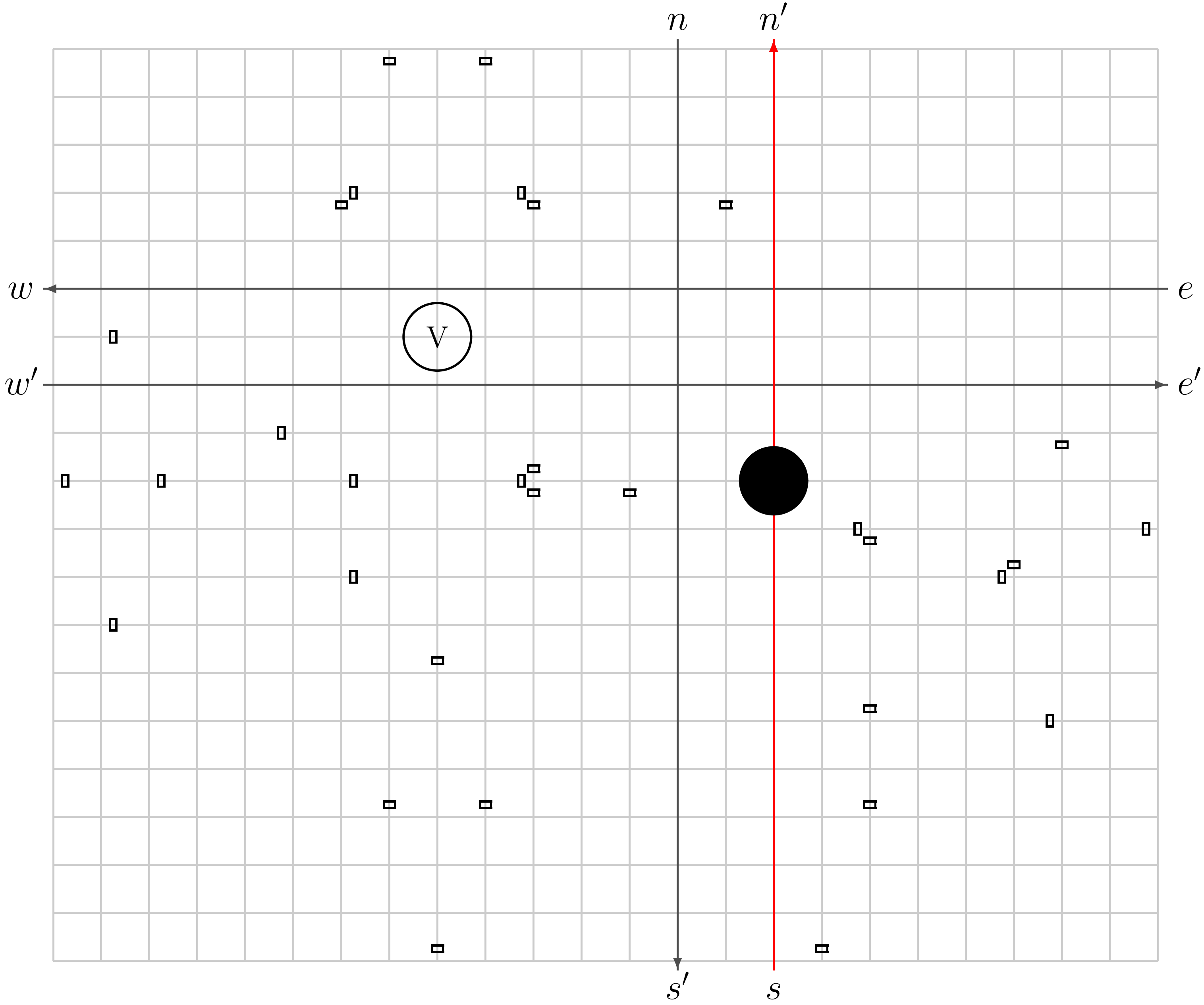
Movements of Balls (State: V, Input: s)



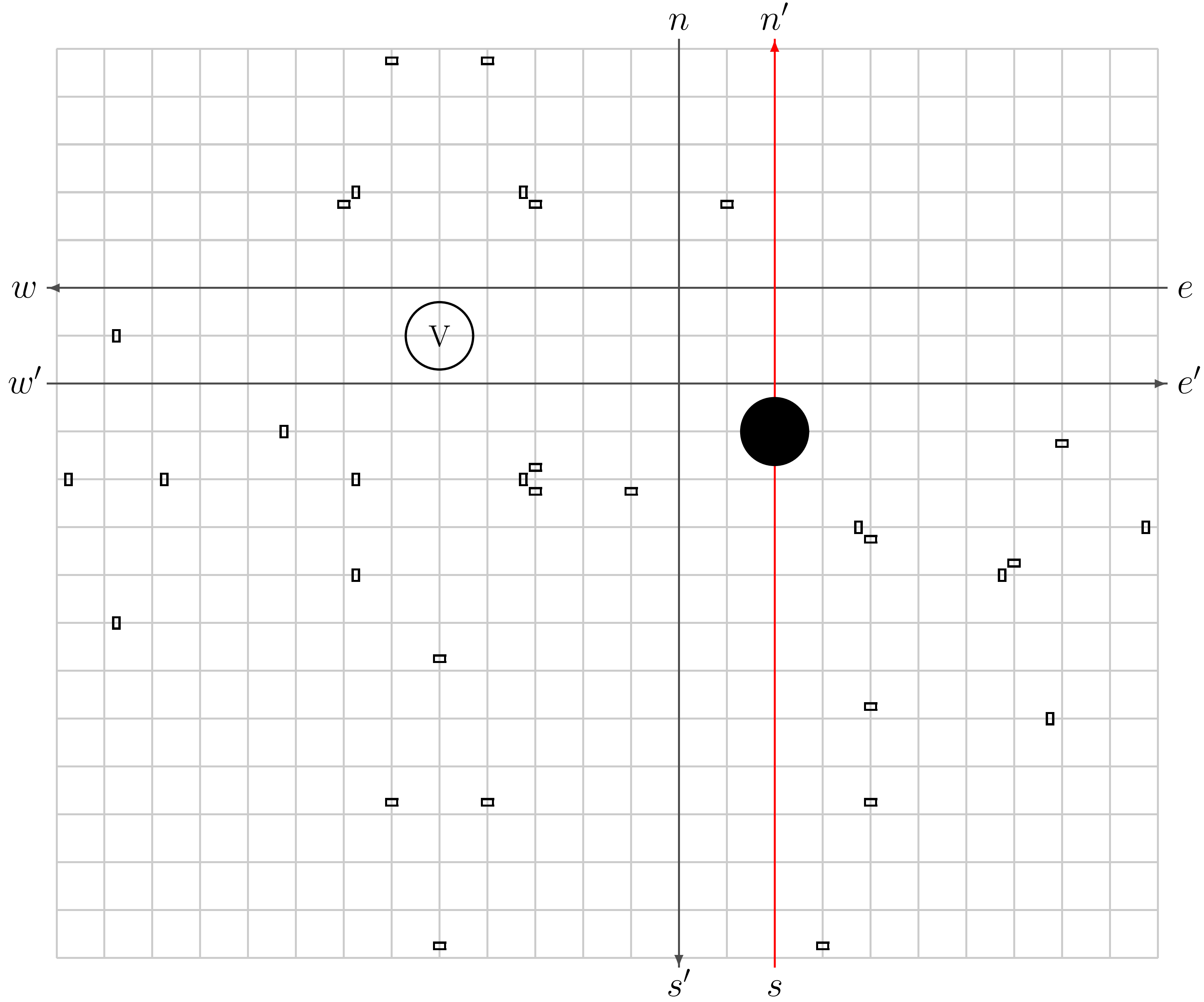
Movements of Balls (State: V, Input: s)



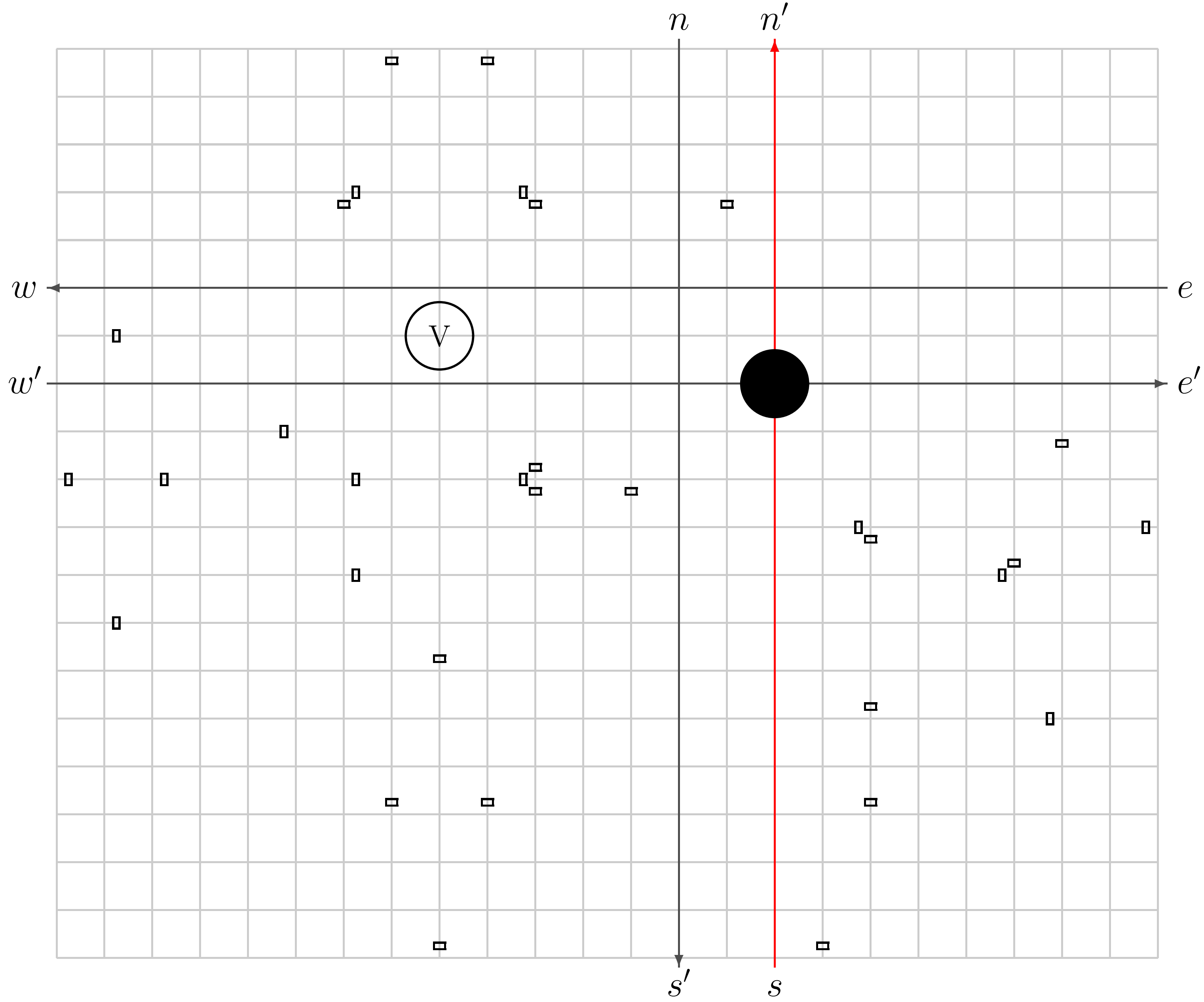
Movements of Balls (State: V, Input: s)



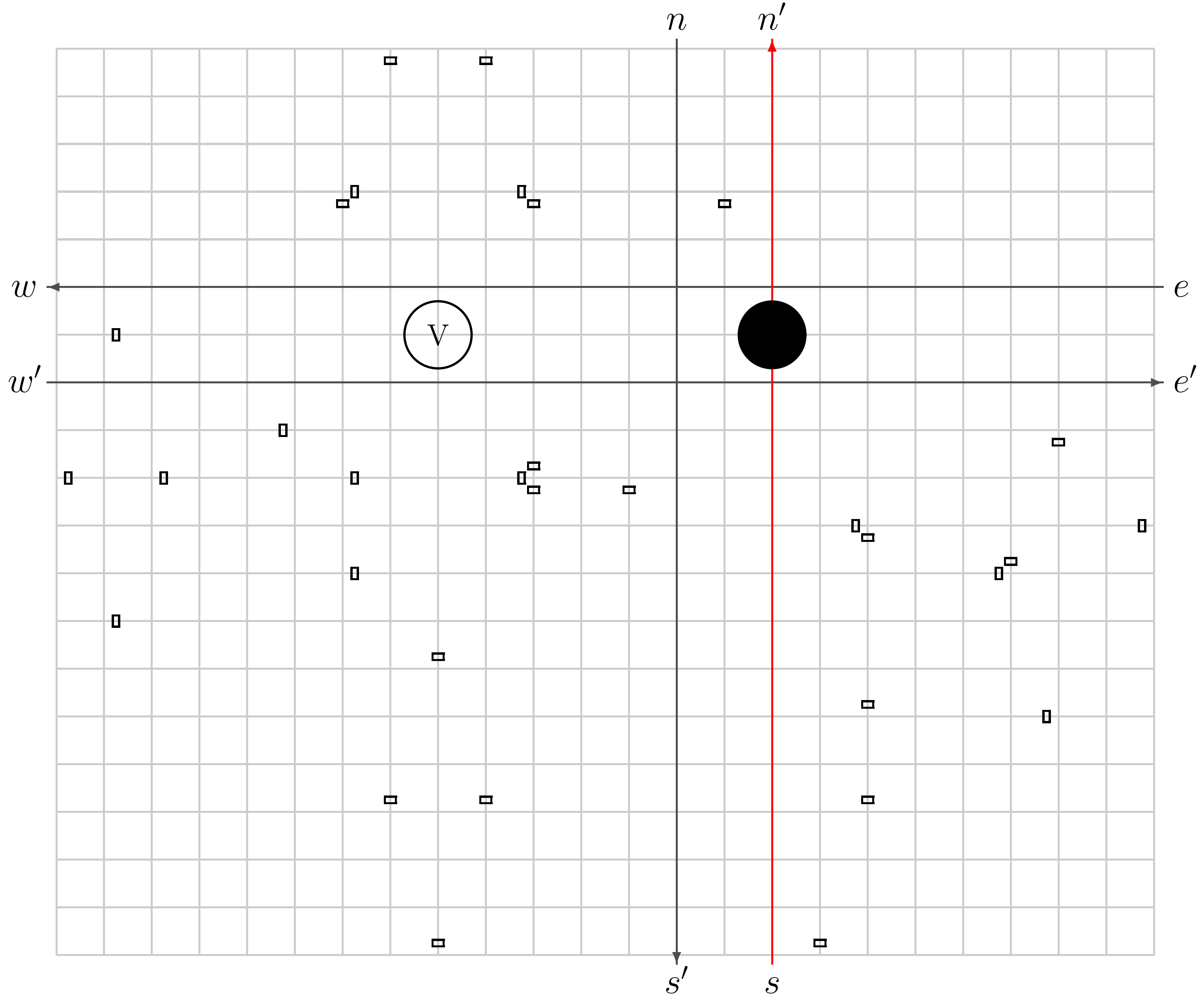
Movements of Balls (State: V, Input: s)



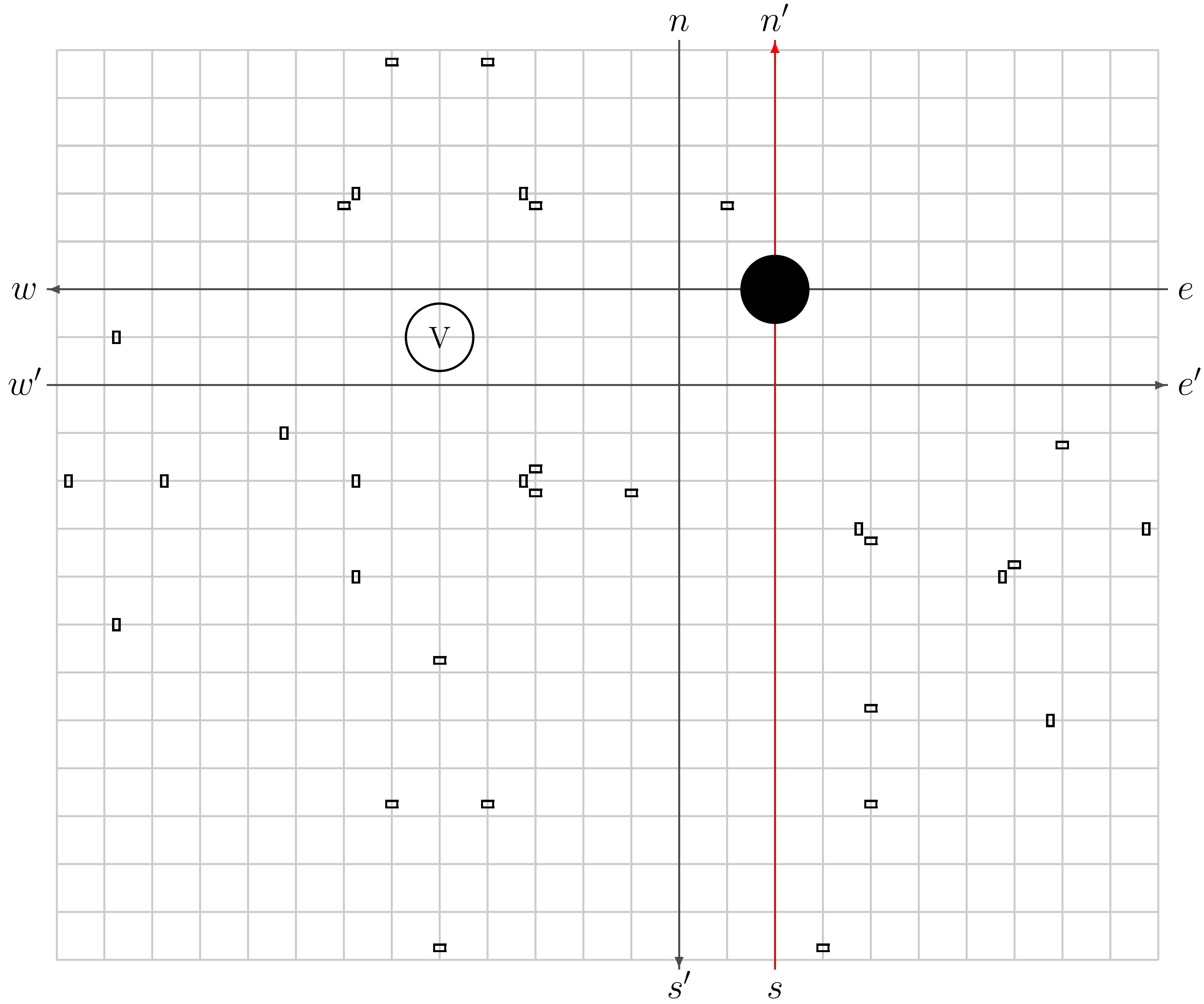
Movements of Balls (State: V, Input: s)



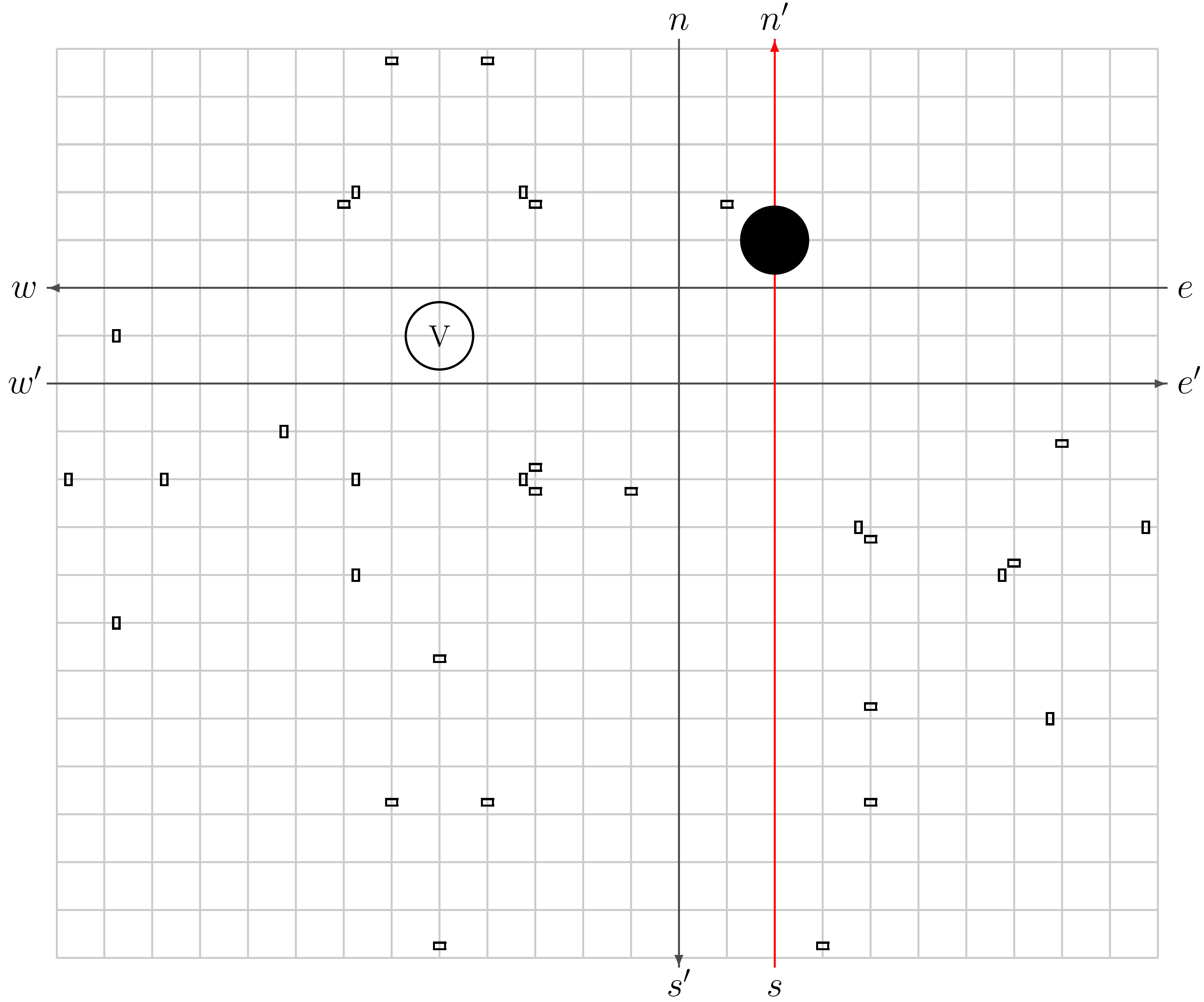
Movements of Balls (State: V, Input: s)



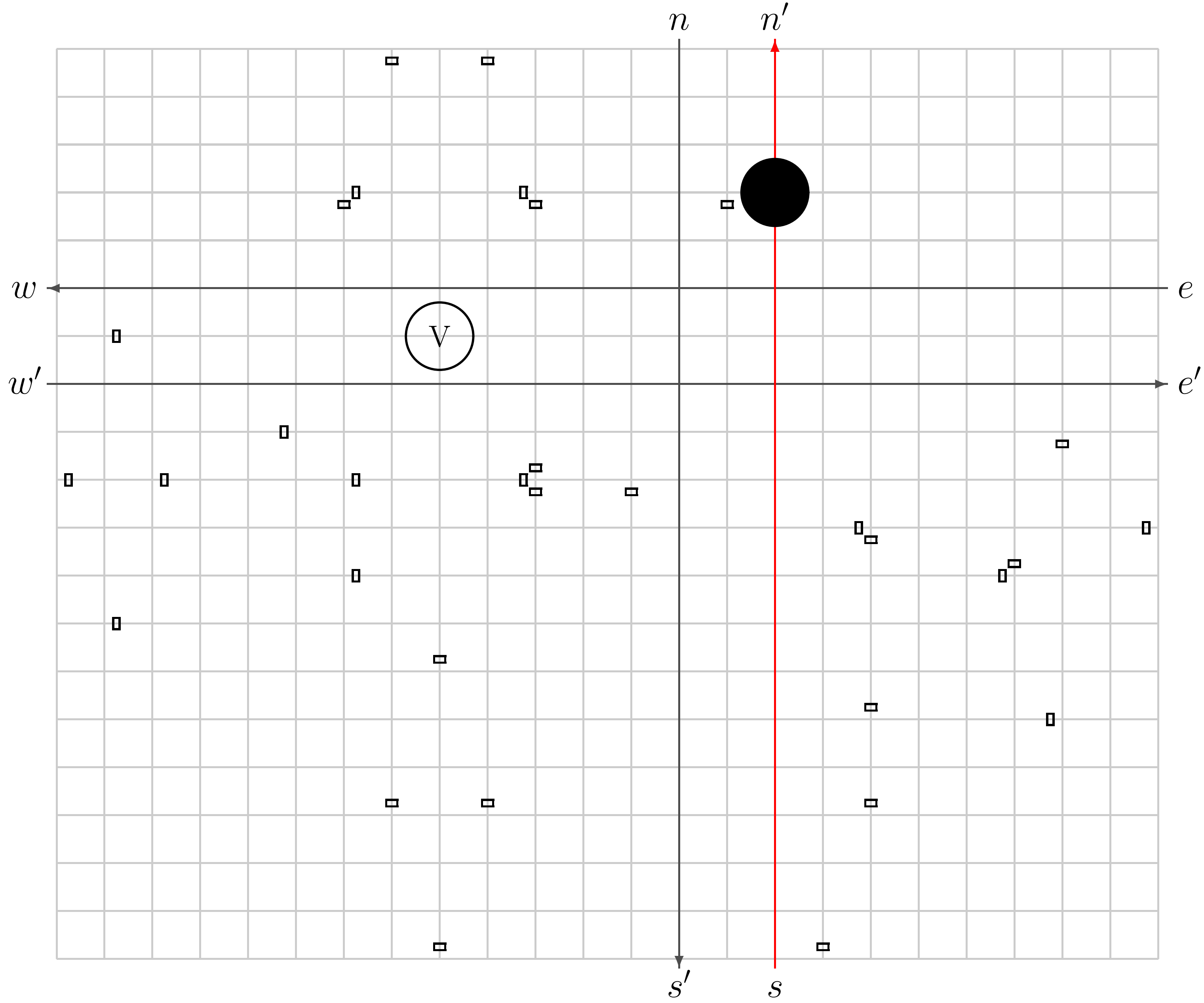
Movements of Balls (State: V, Input: s)



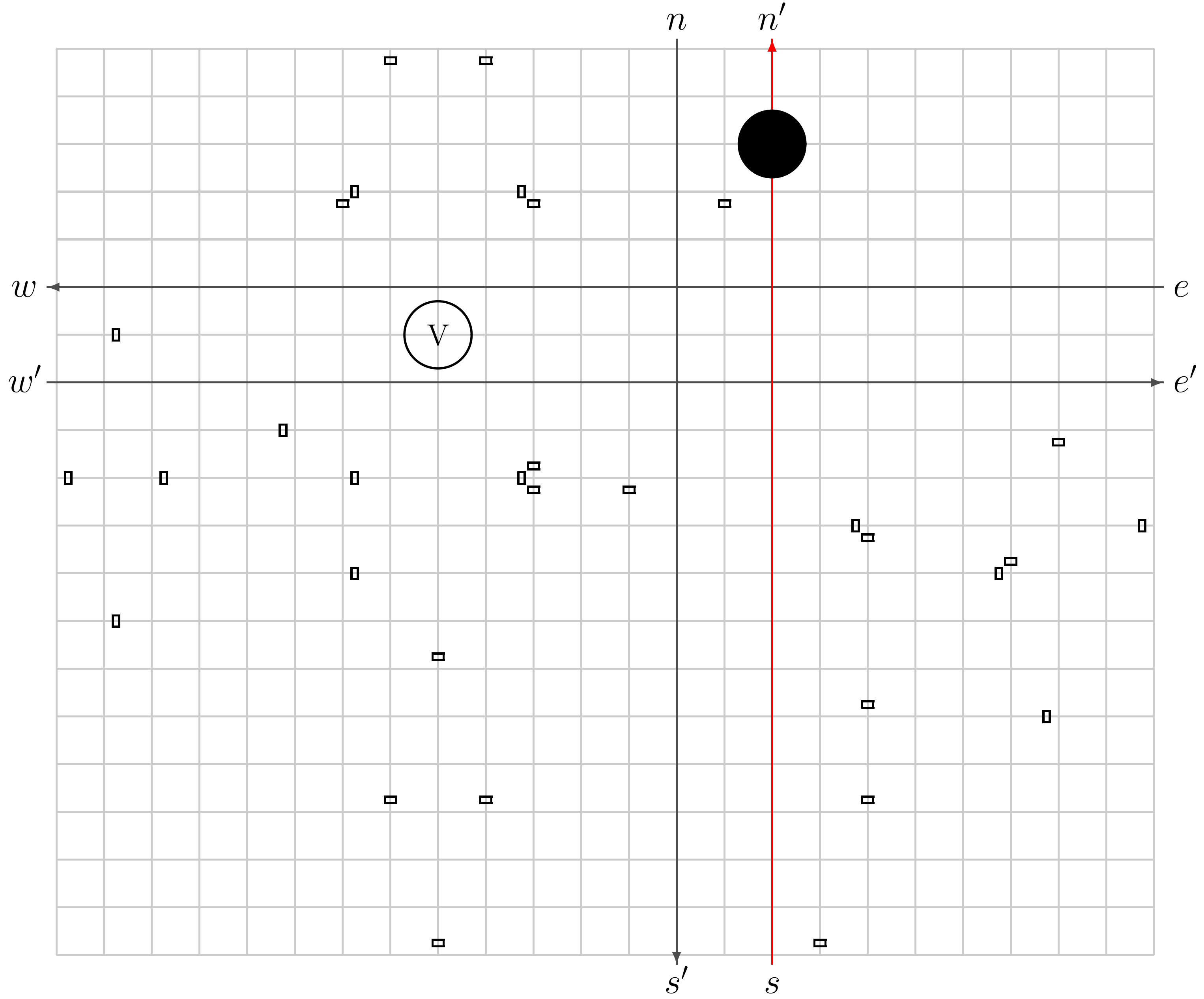
Movements of Balls (State: V, Input: s)



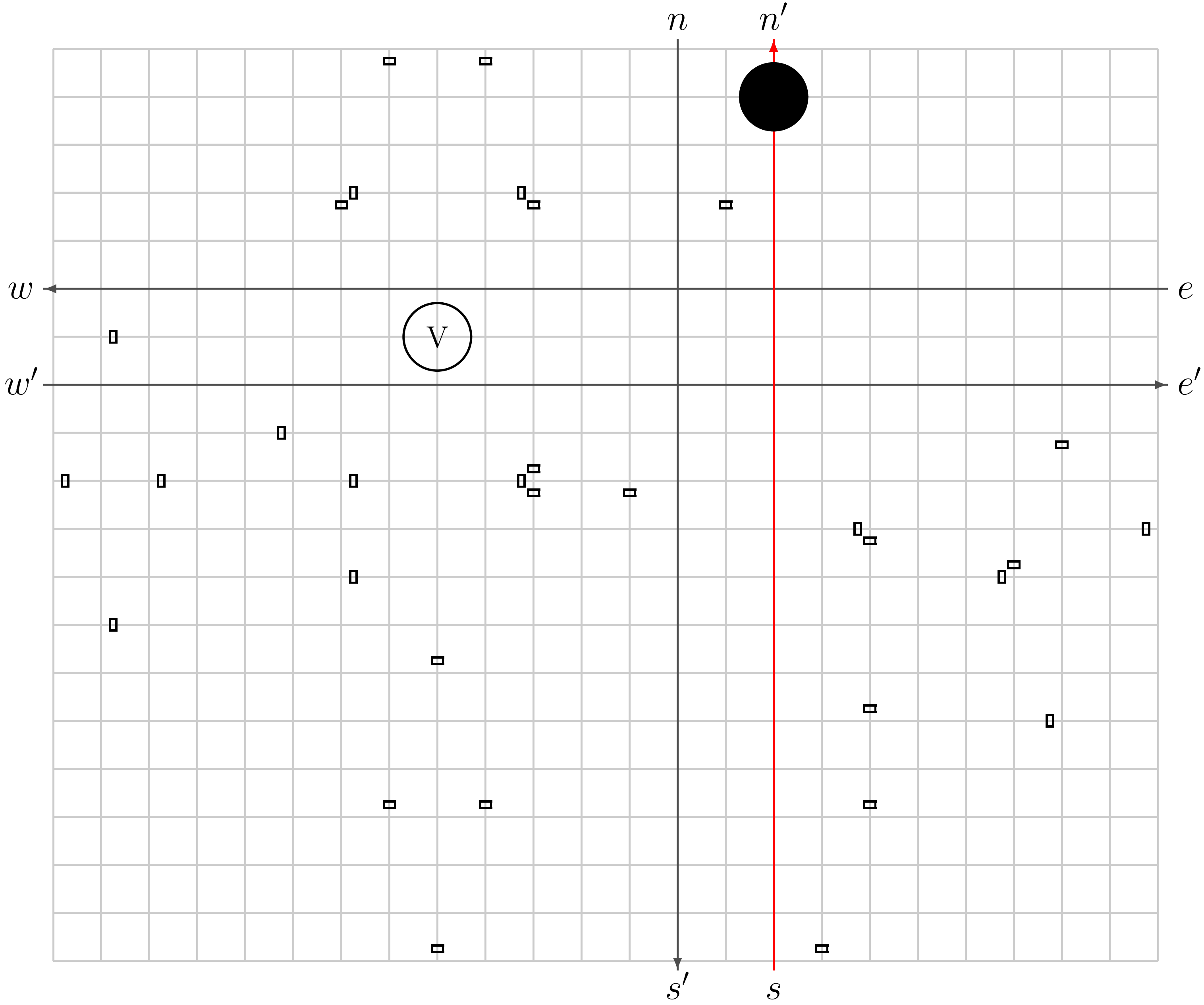
Movements of Balls (State: V, Input: s)



Movements of Balls (State: V, Input: s)

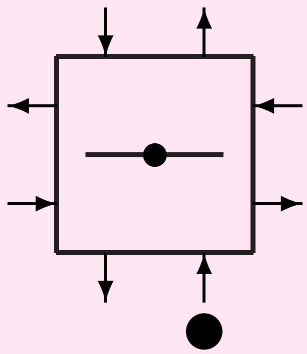


Movements of Balls (State: V, Input: s)

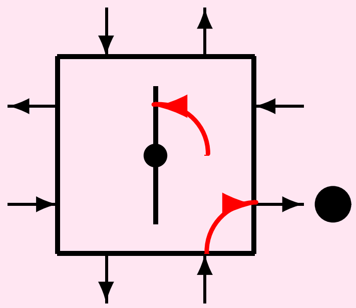


Orthogonal Case

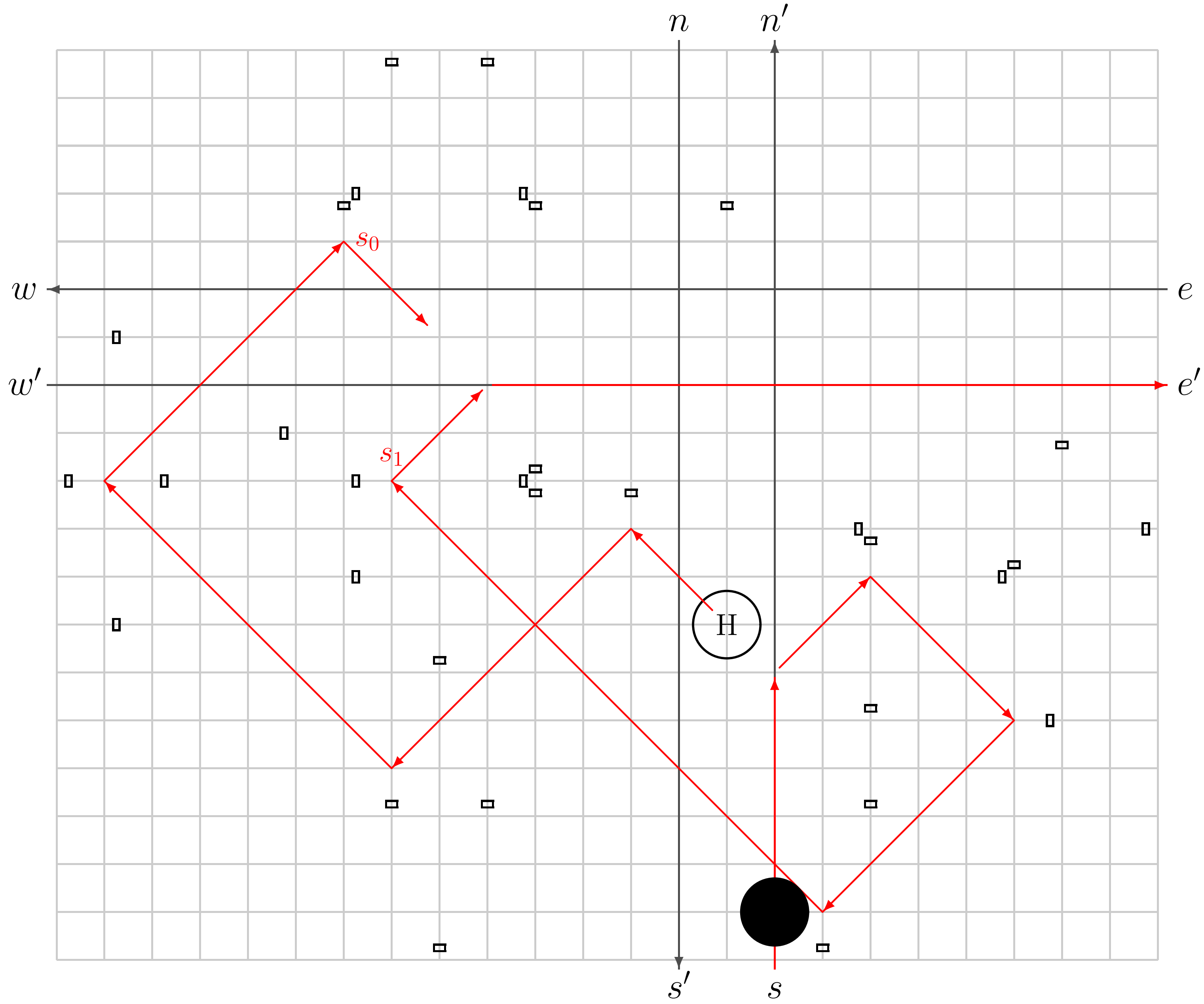
$t = 0$



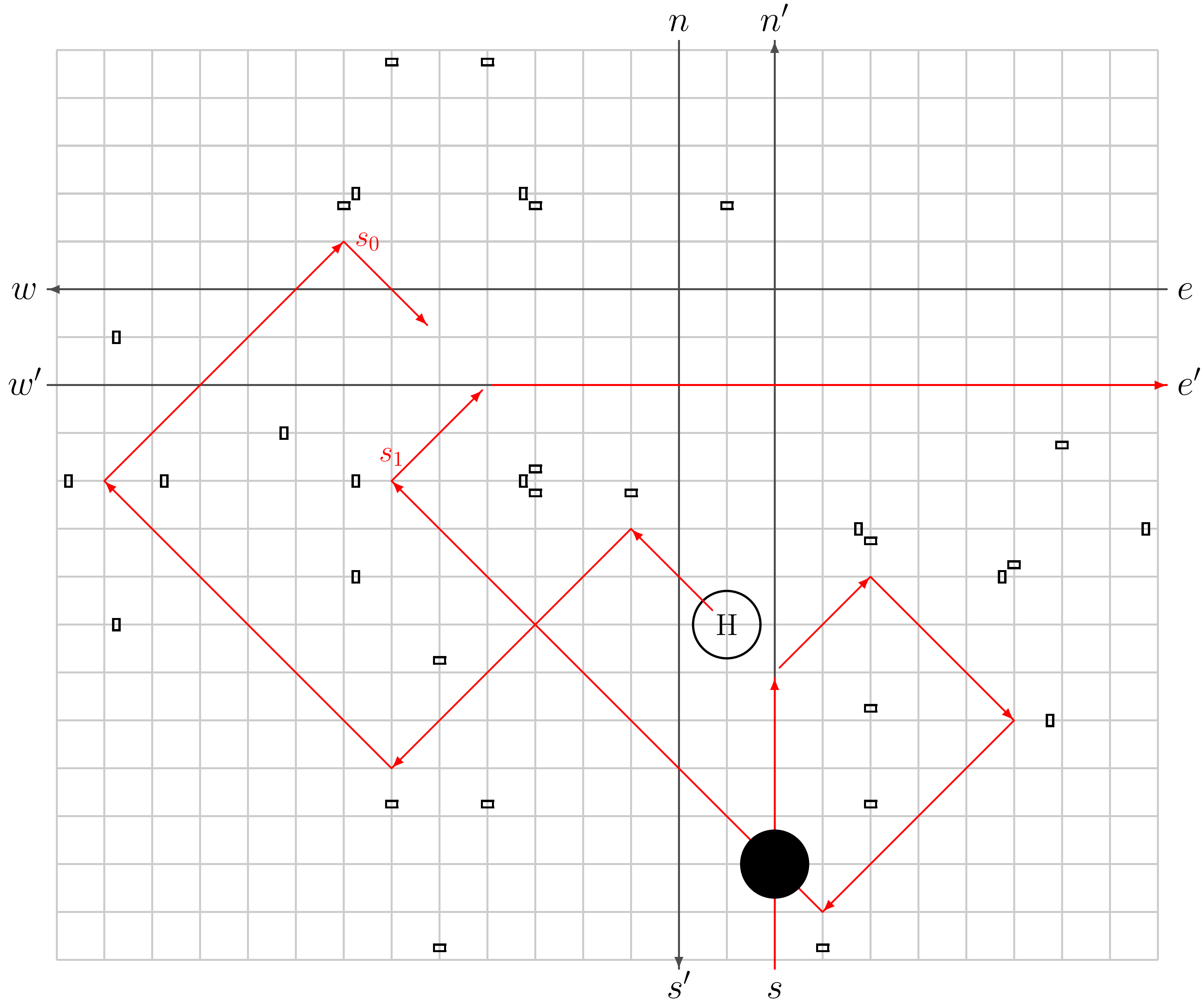
$t = 1$



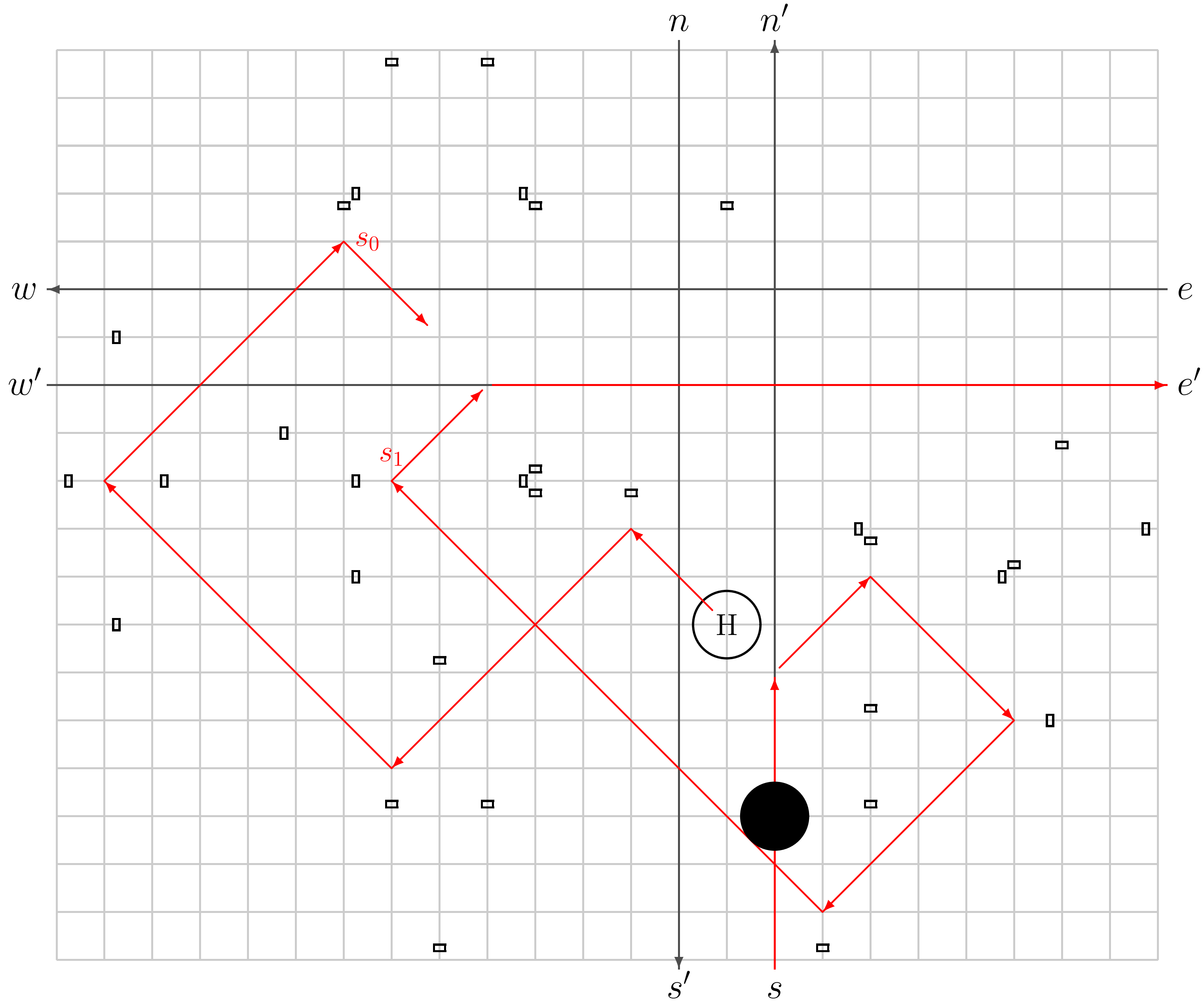
Movements of Balls (State: H, Input: s)



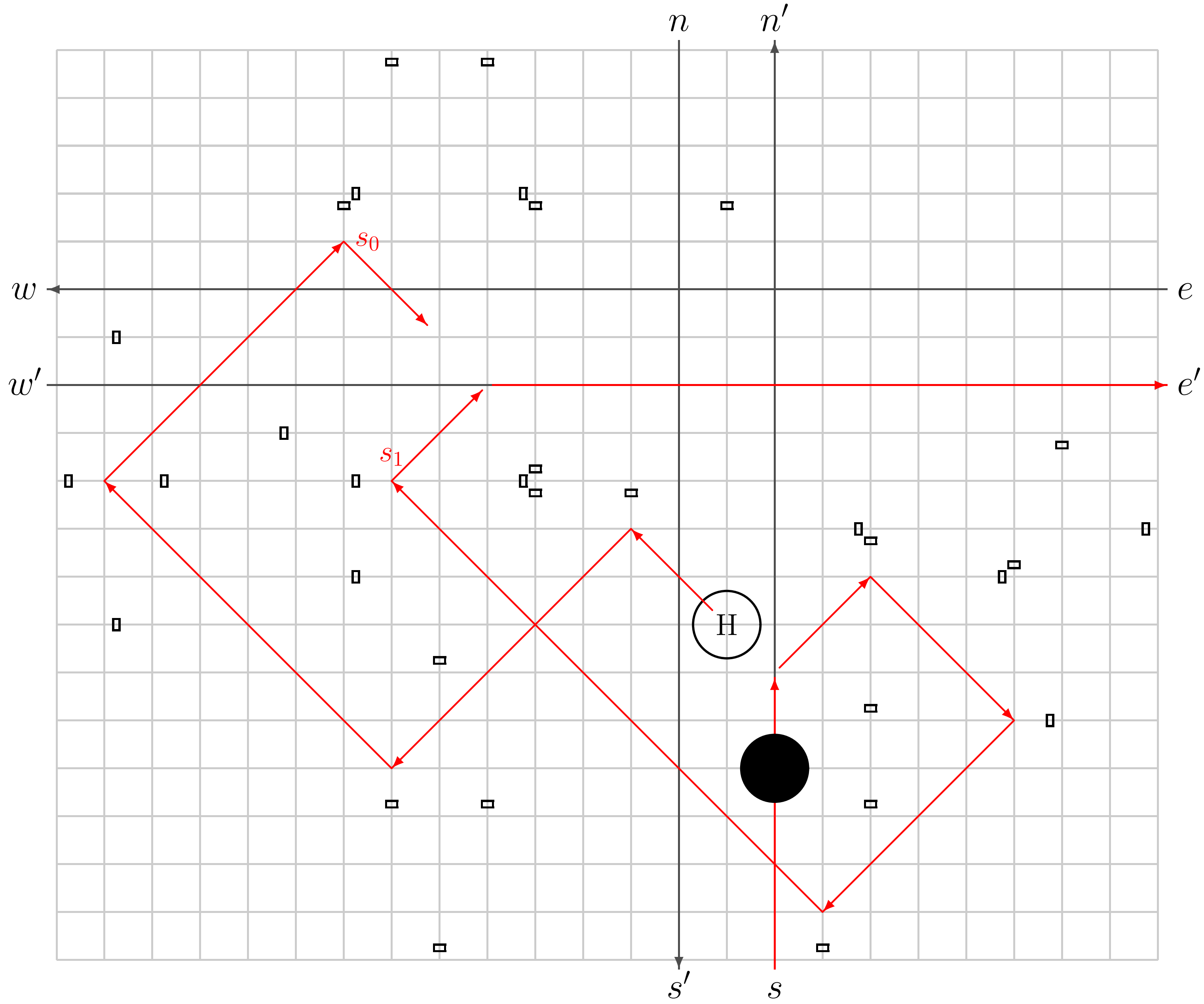
Movements of Balls (State: H, Input: s)



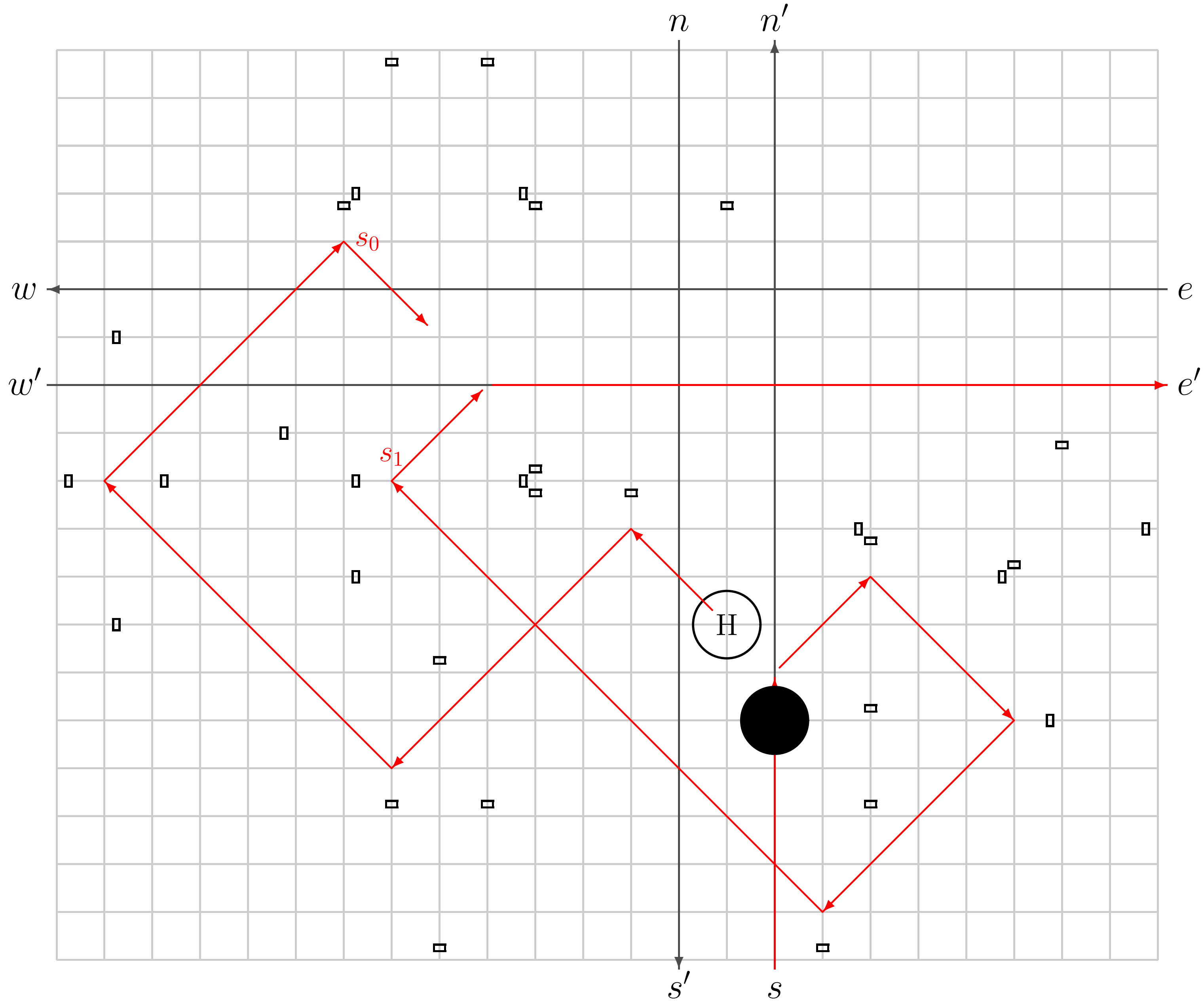
Movements of Balls (State: H, Input: s)



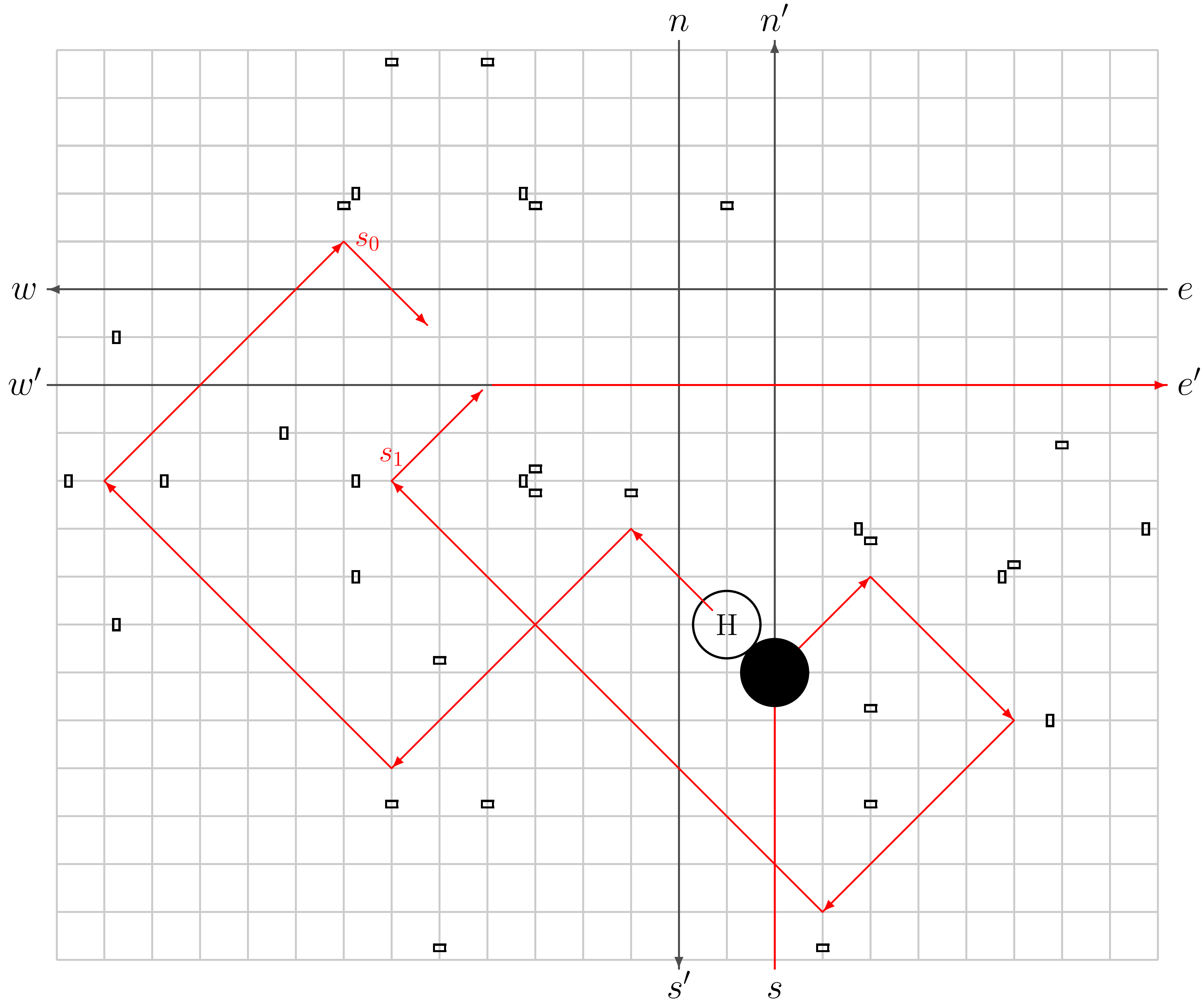
Movements of Balls (State: H, Input: s)



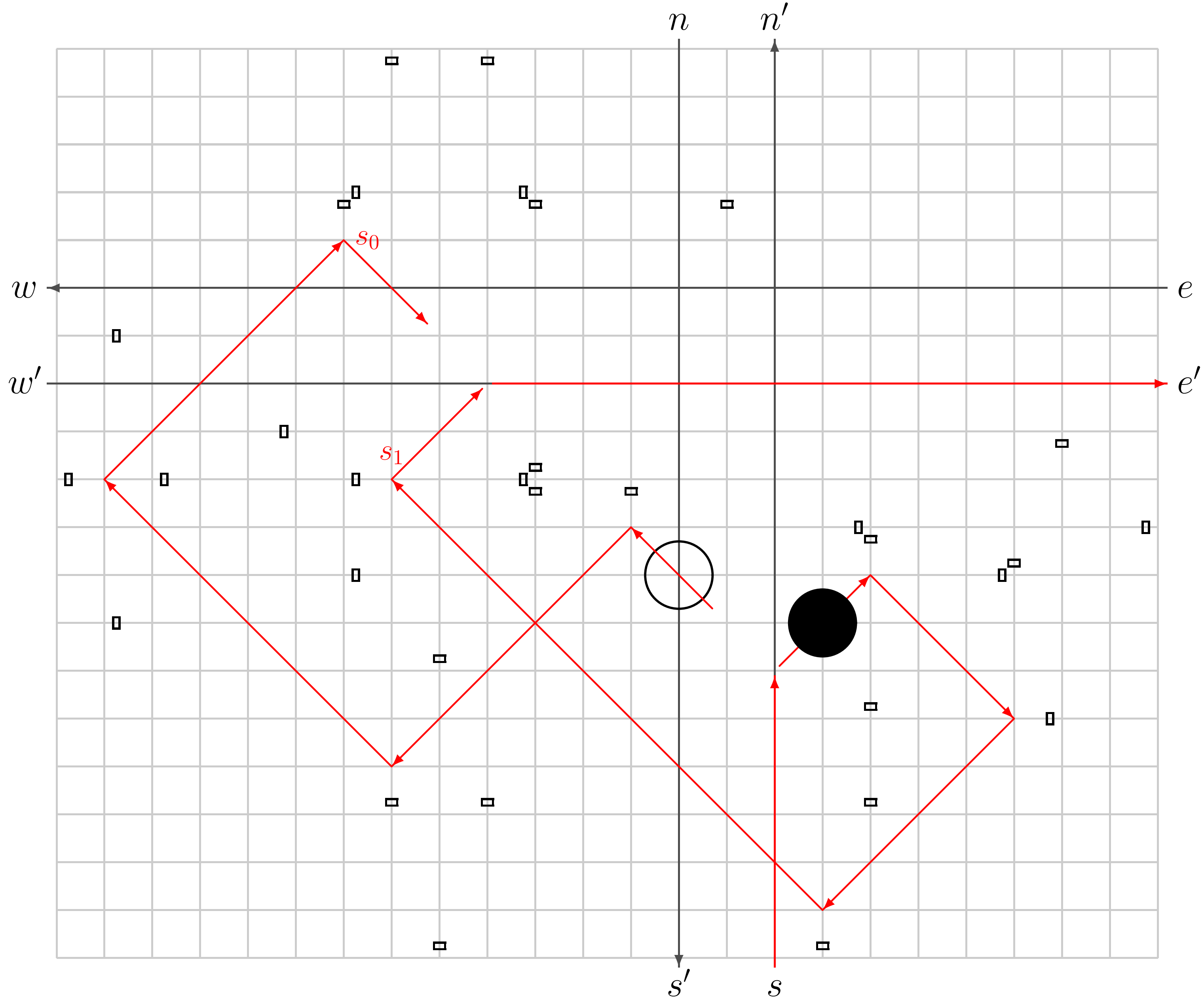
Movements of Balls (State: H, Input: s)



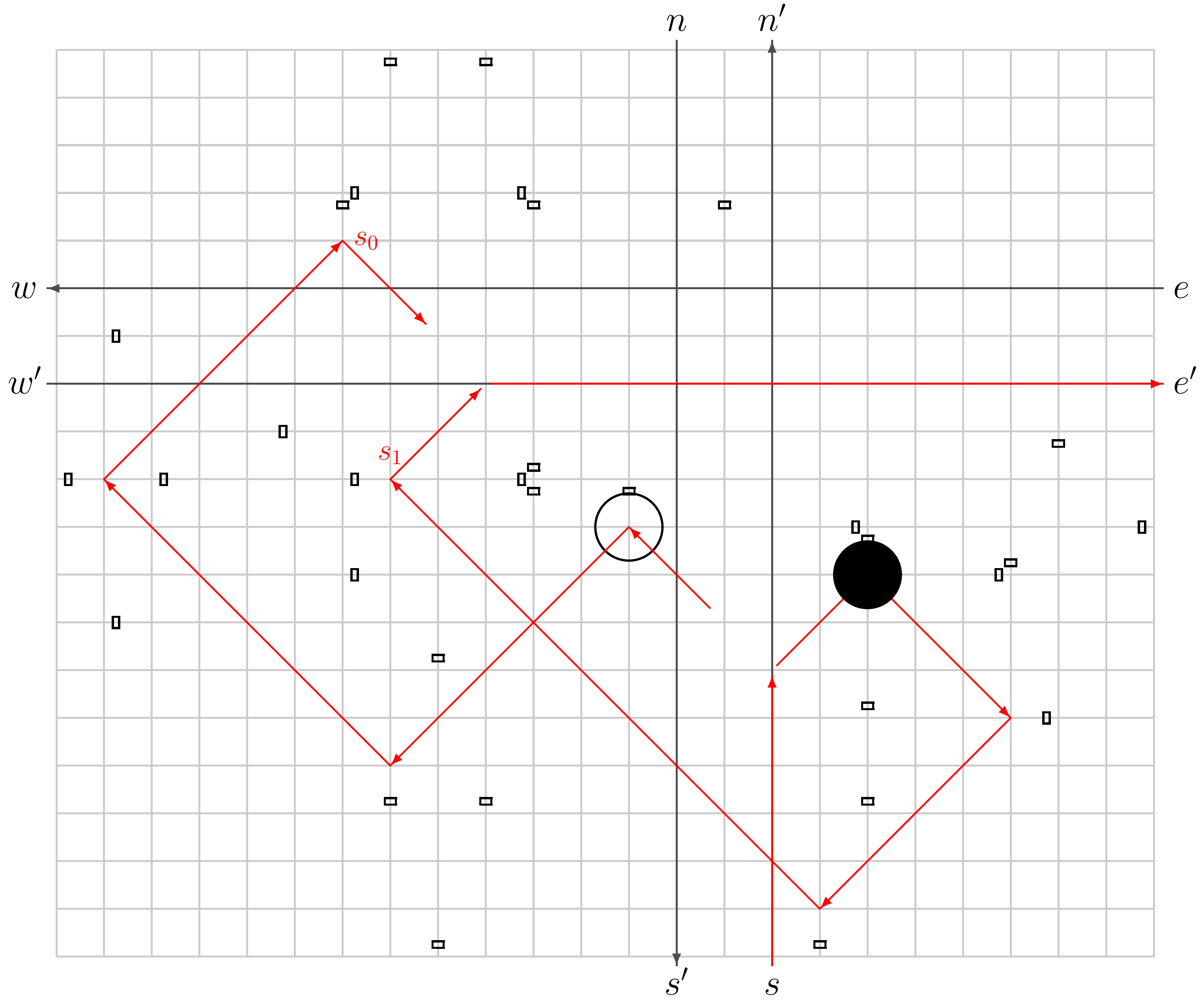
Movements of Balls (State: H, Input: s)



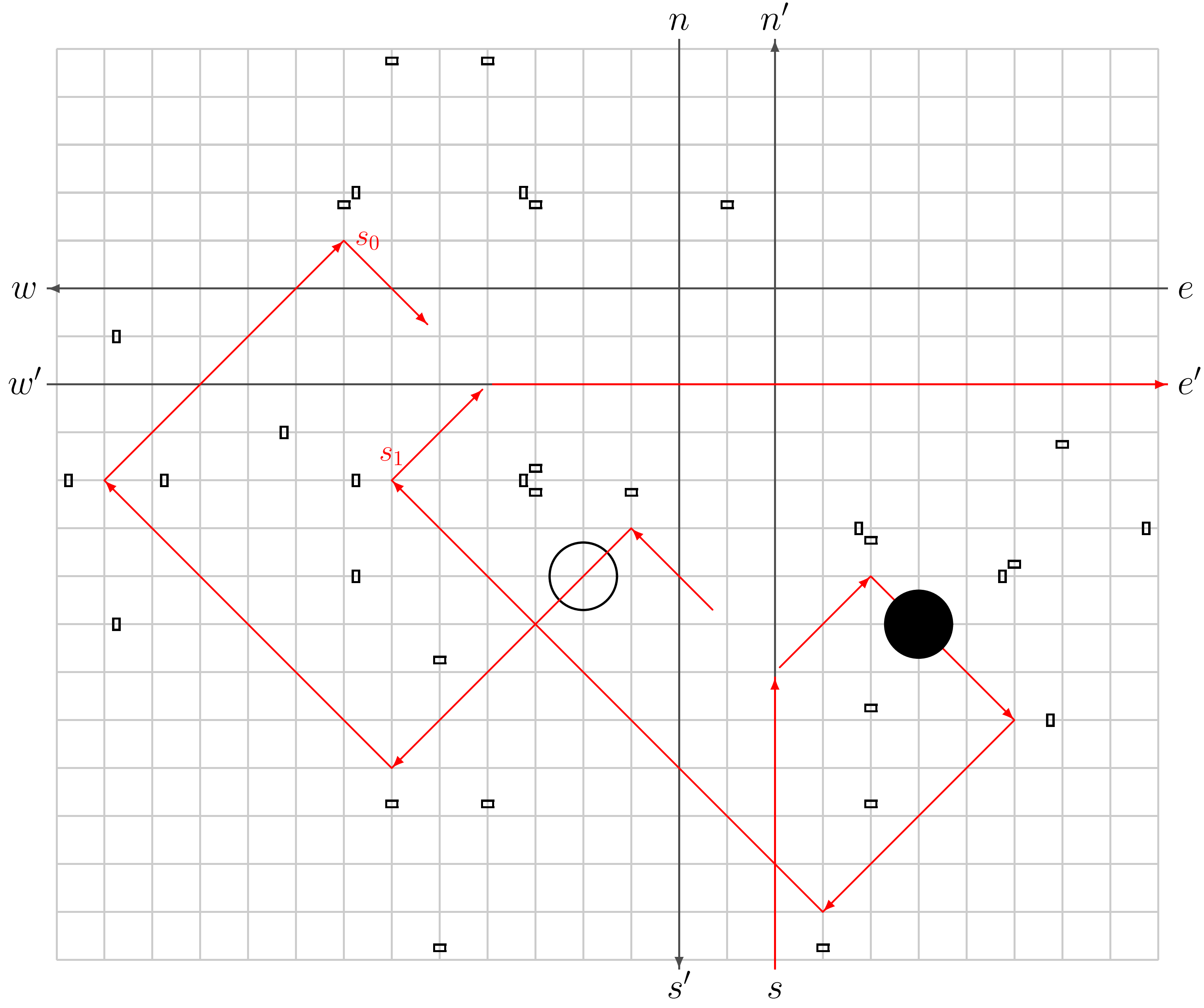
Movements of Balls (State: H, Input: s)



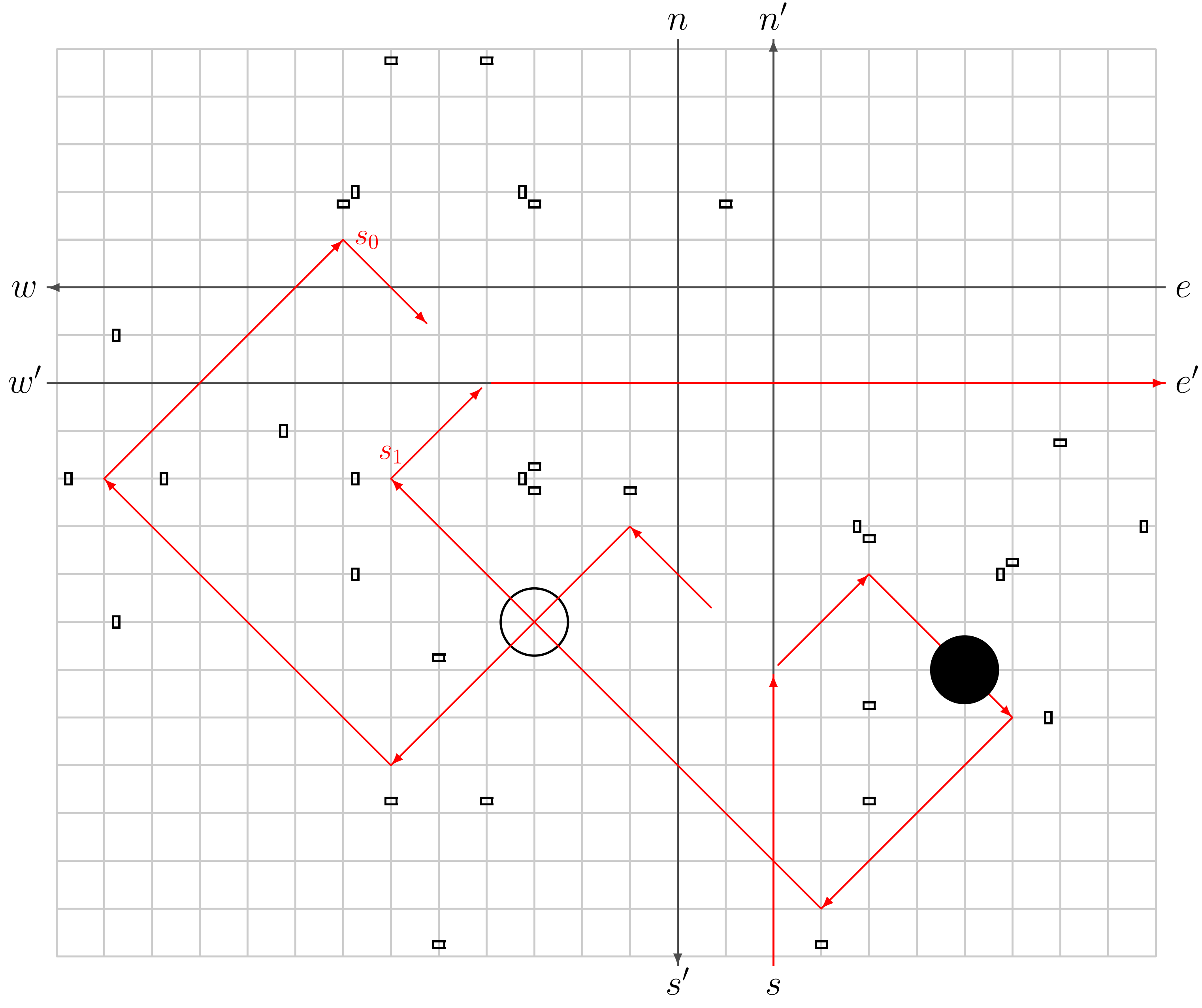
Movements of Balls (State: H, Input: s)



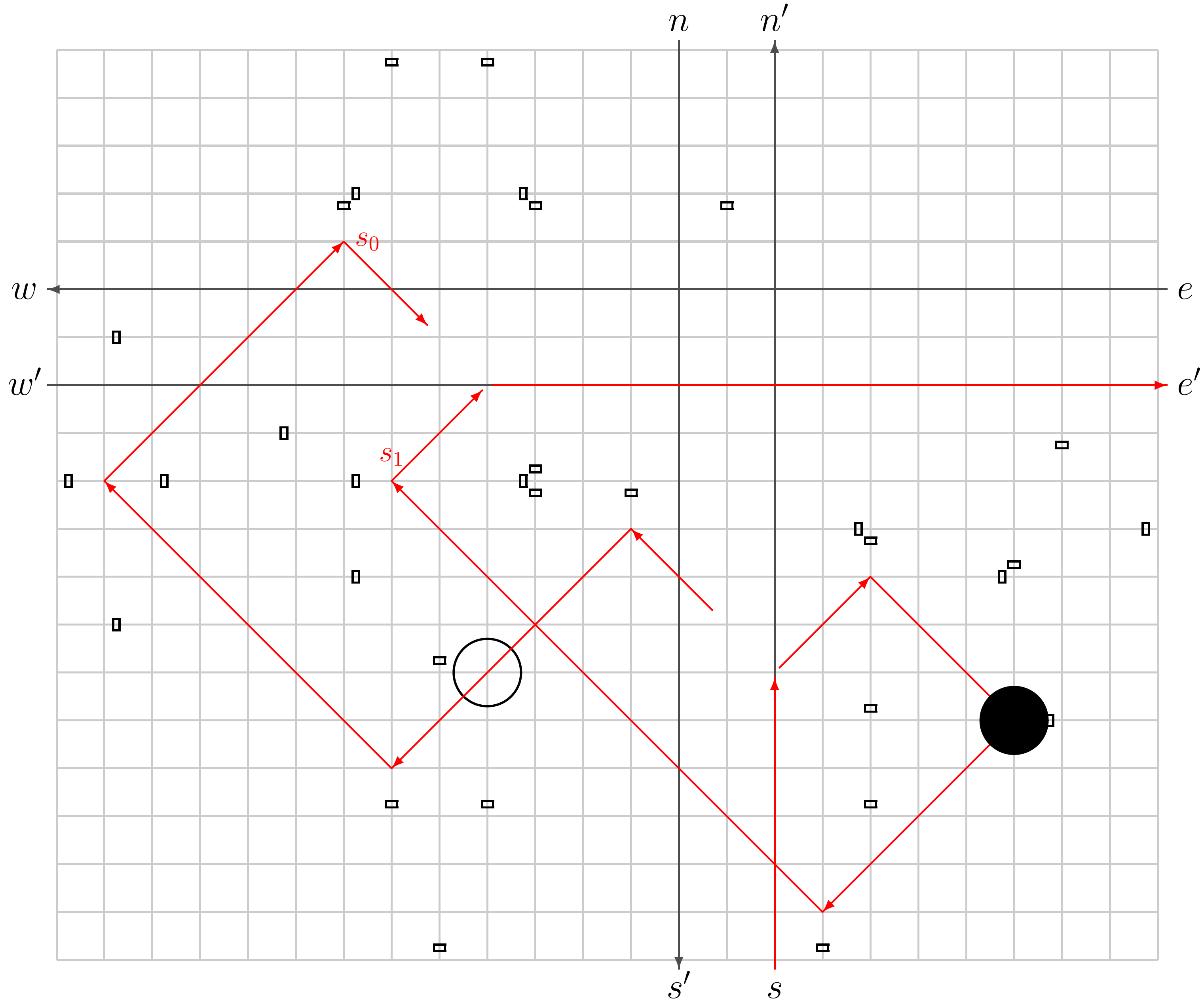
Movements of Balls (State: H, Input: s)



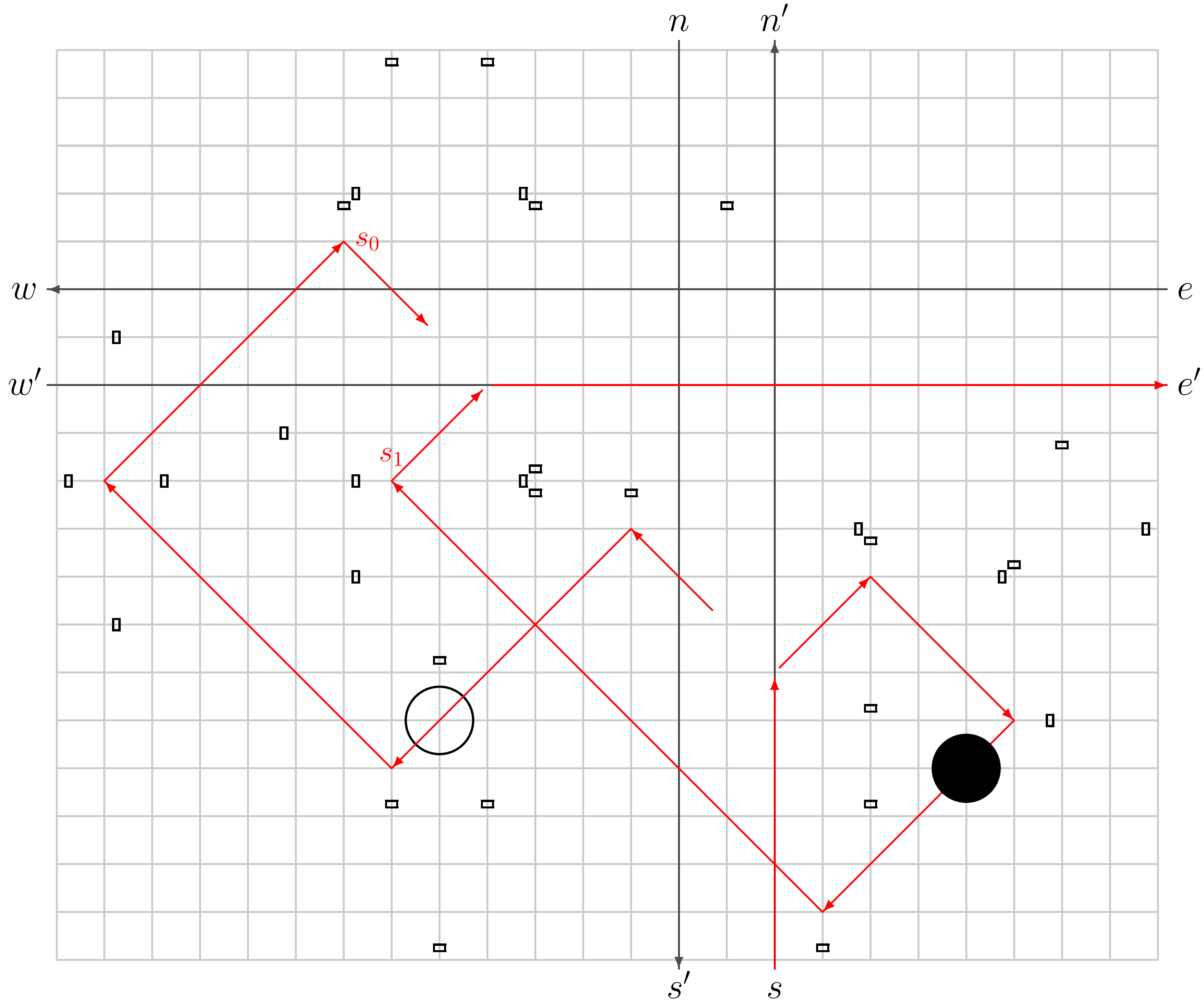
Movements of Balls (State: H, Input: s)



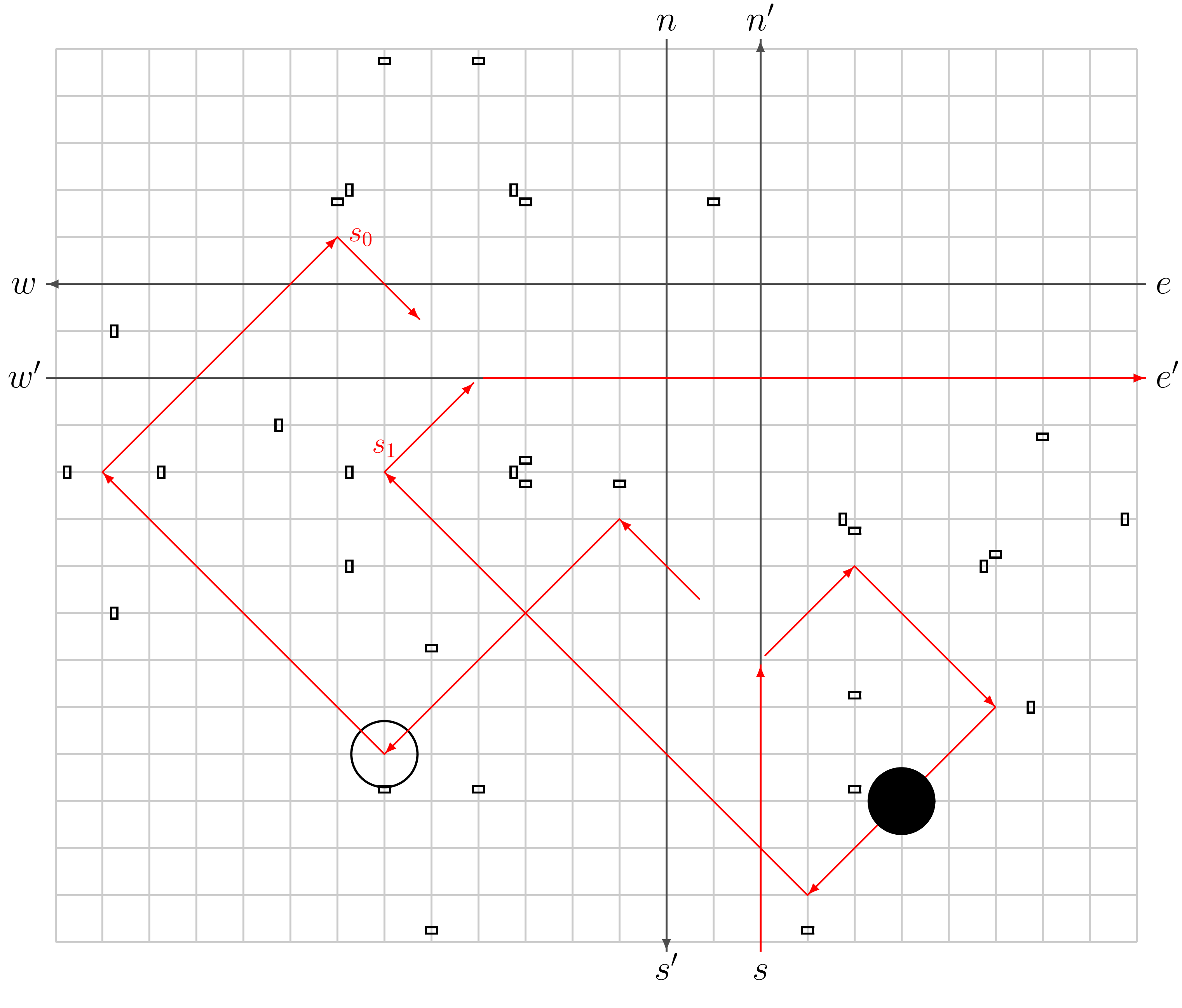
Movements of Balls (State: H, Input: s)



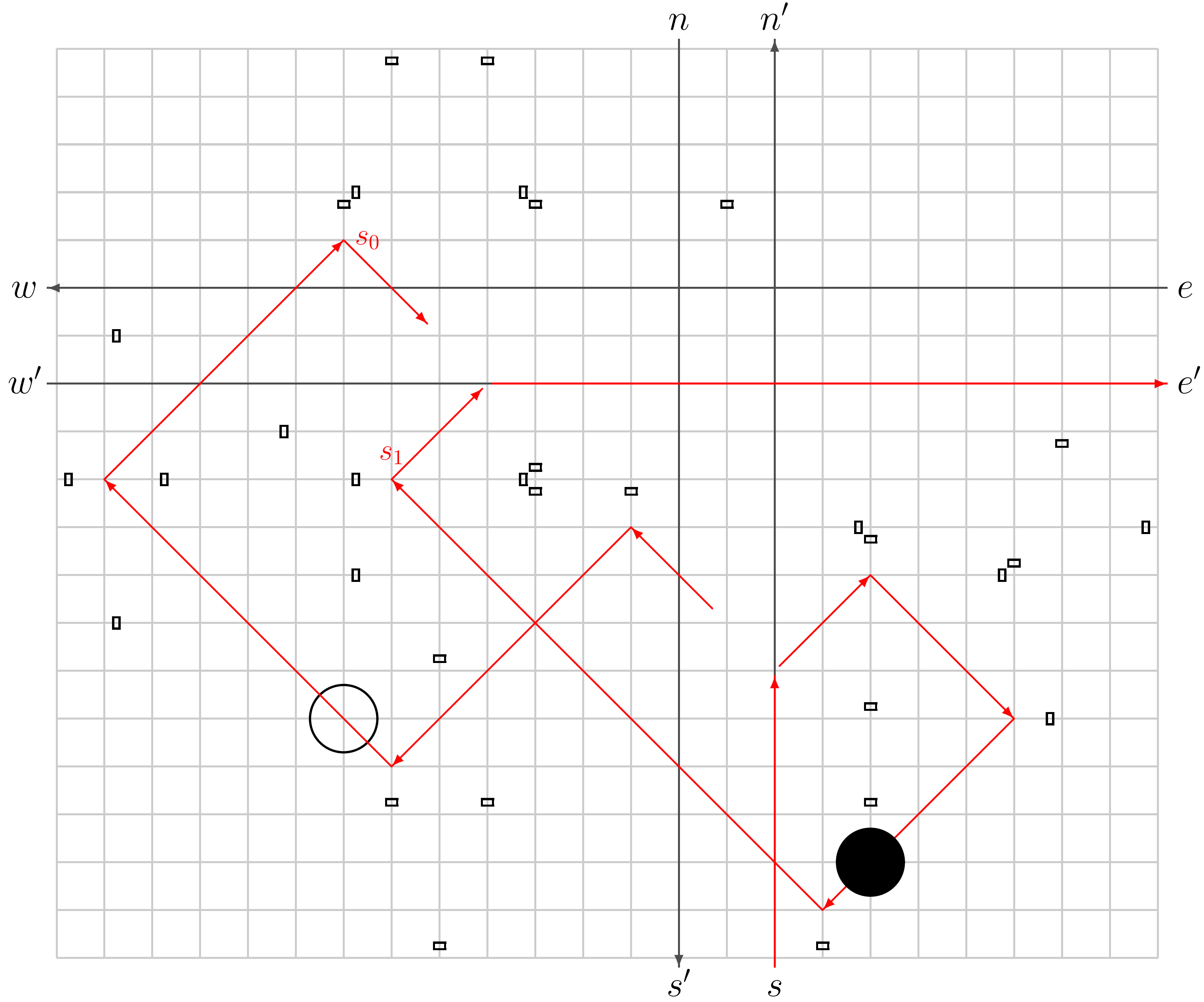
Movements of Balls (State: H, Input: s)



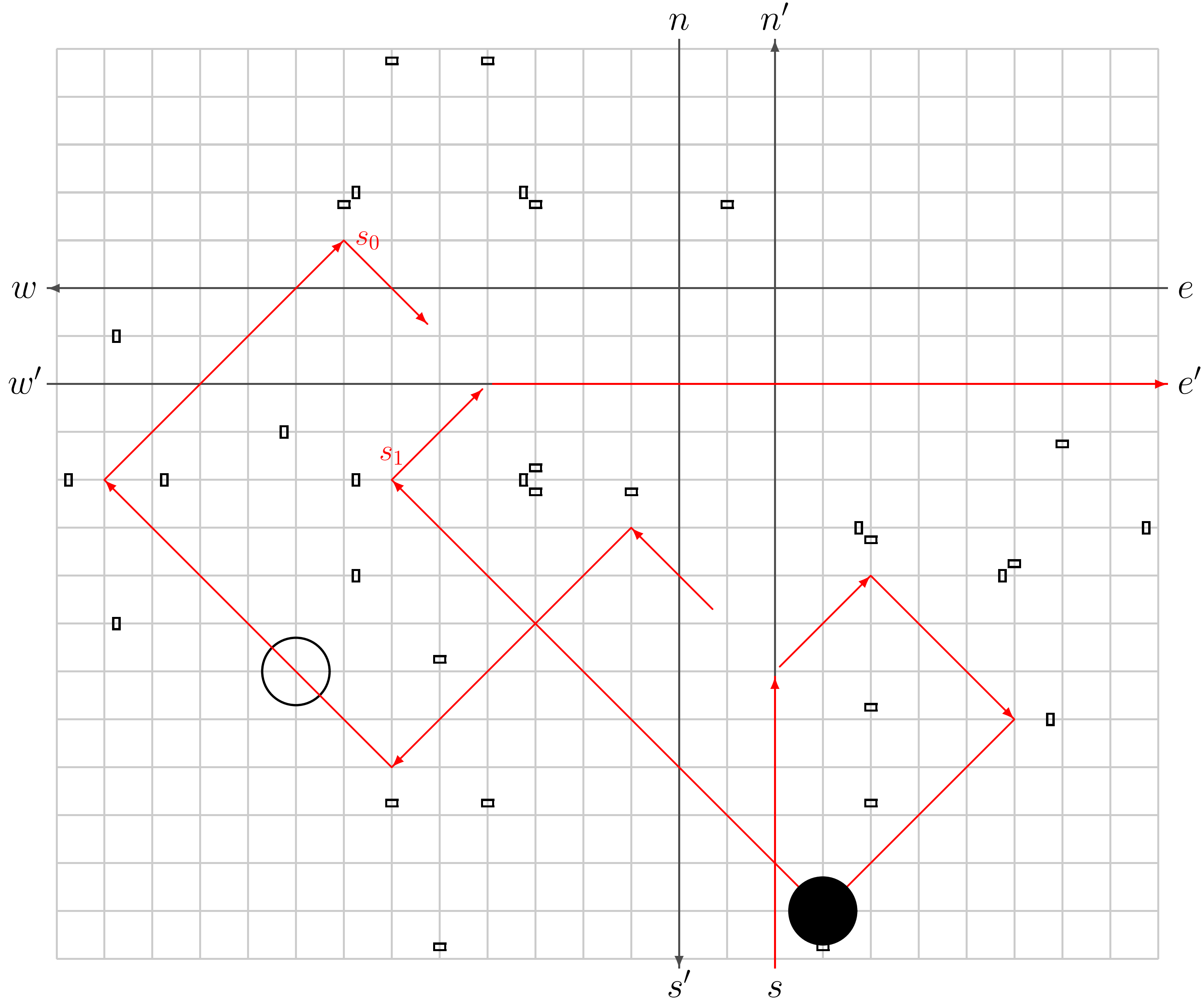
Movements of Balls (State: H, Input: s)



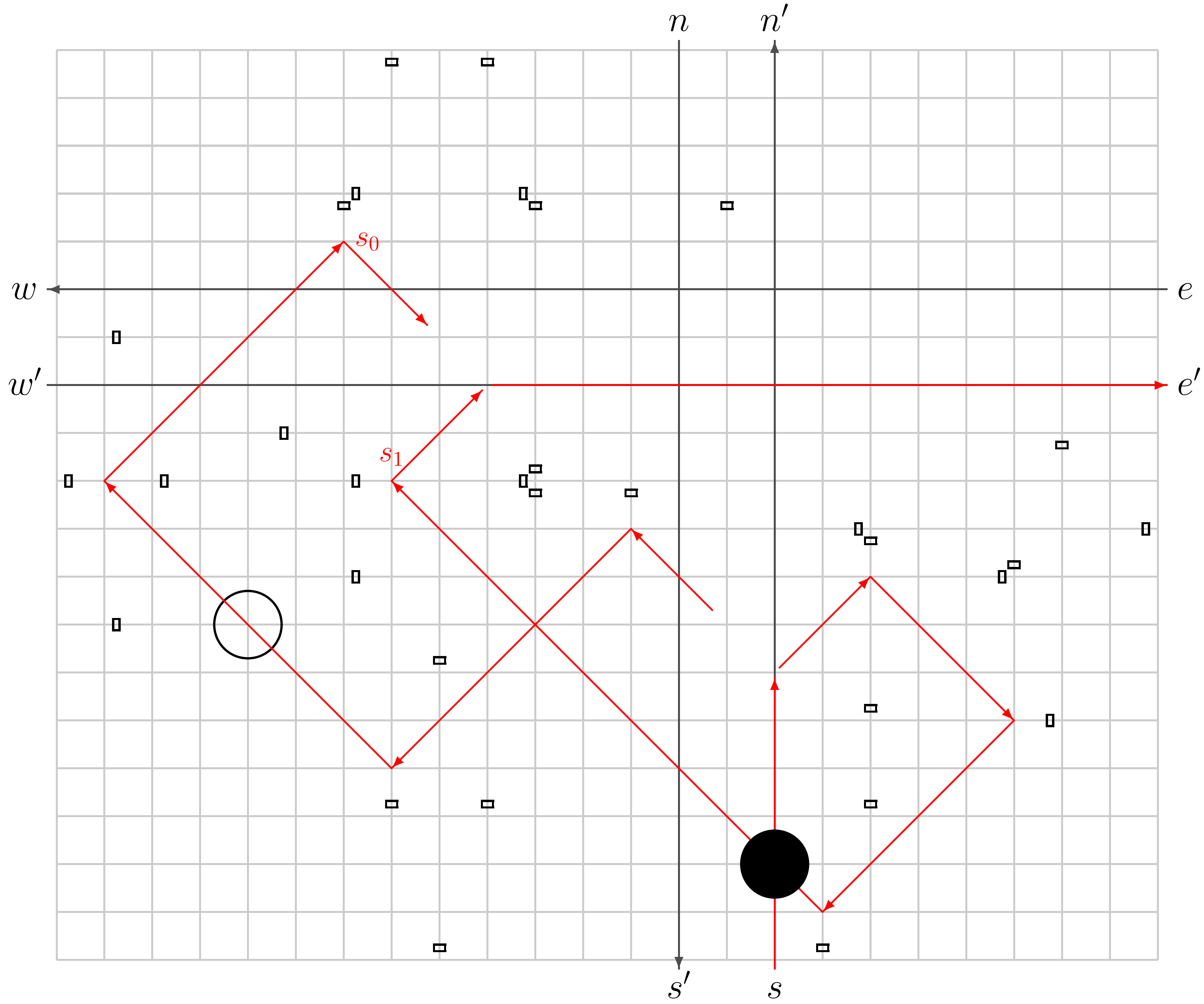
Movements of Balls (State: H, Input: s)



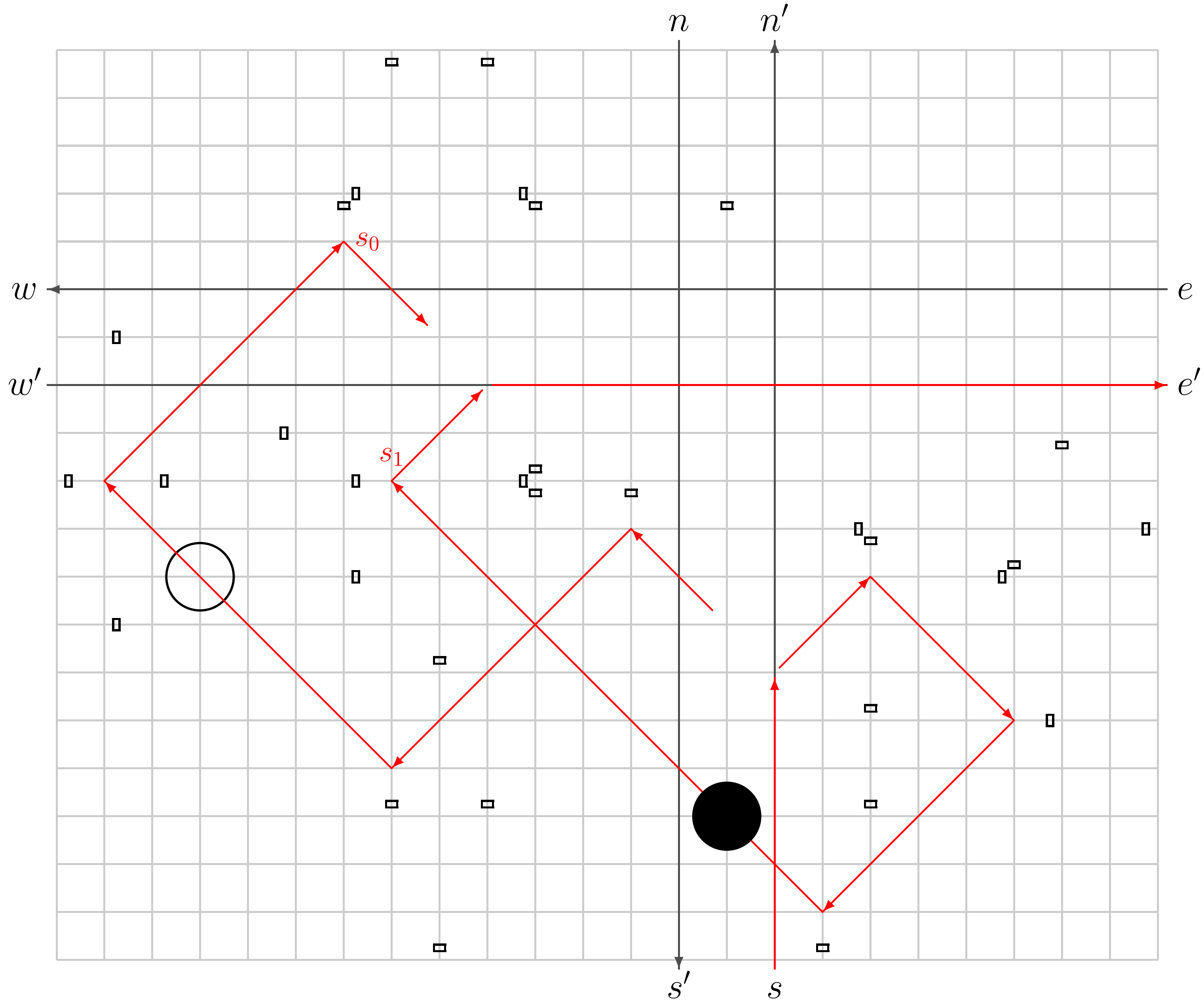
Movements of Balls (State: H, Input: s)



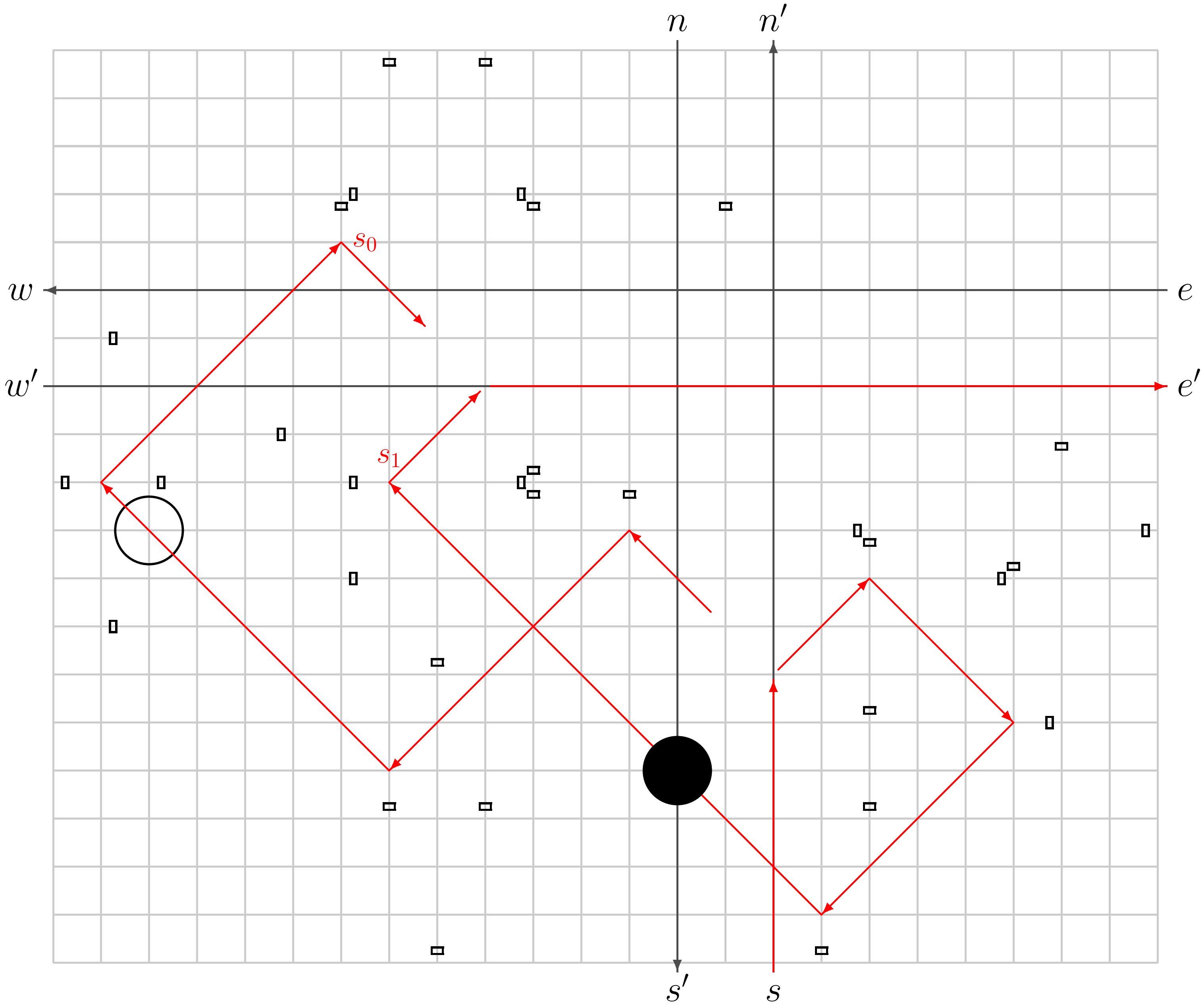
Movements of Balls (State: H, Input: s)



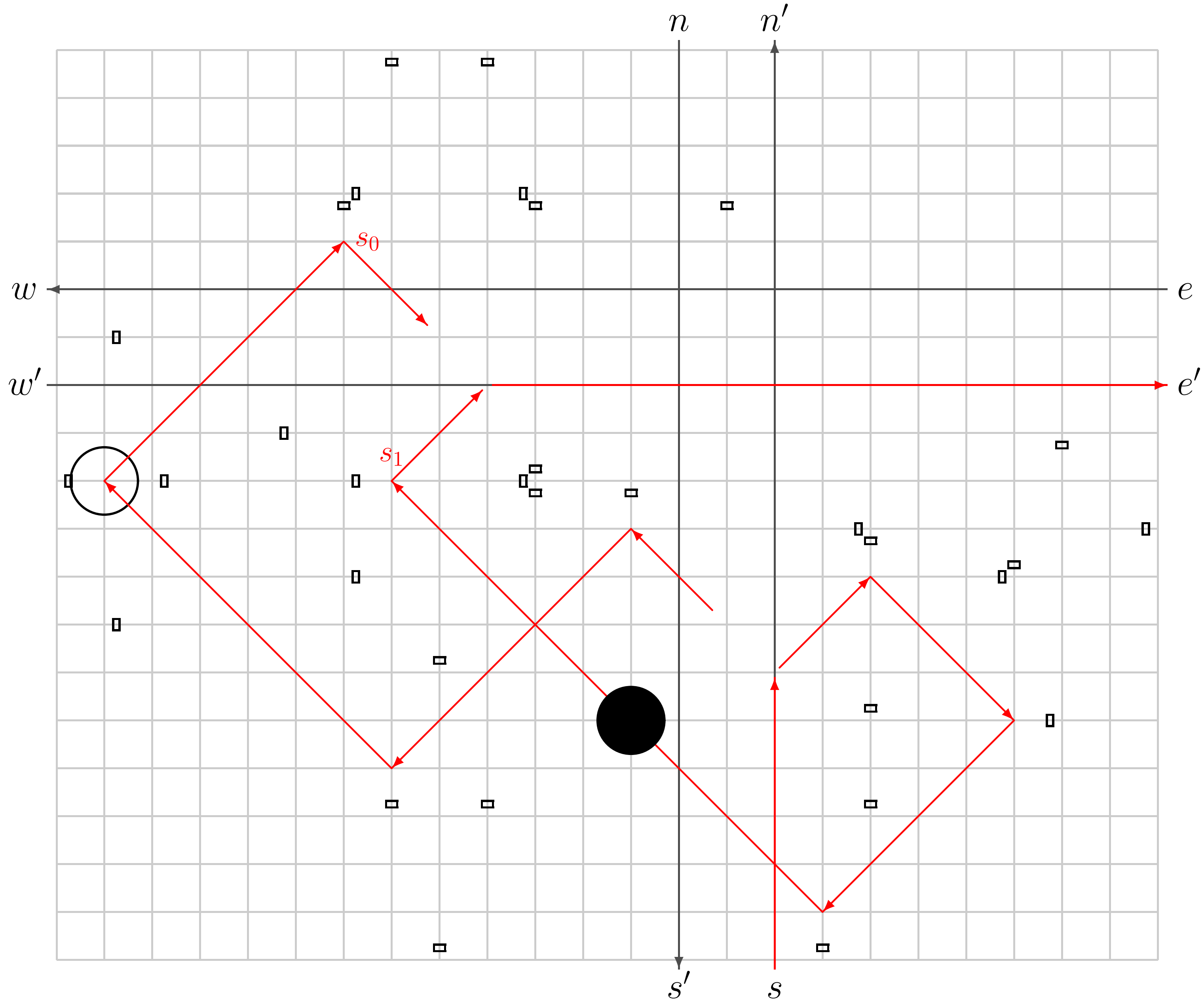
Movements of Balls (State: H, Input: s)



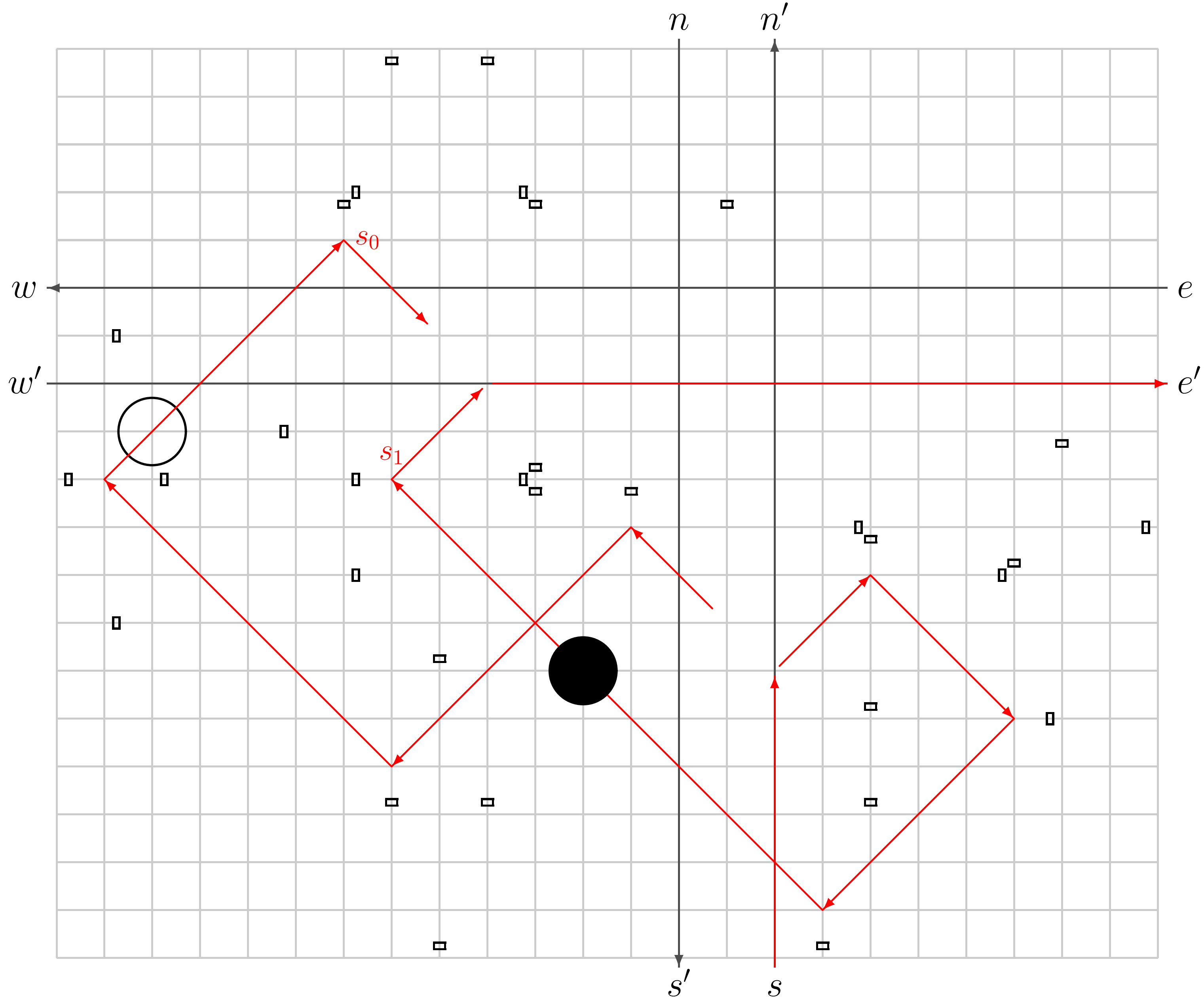
Movements of Balls (State: H, Input: s)



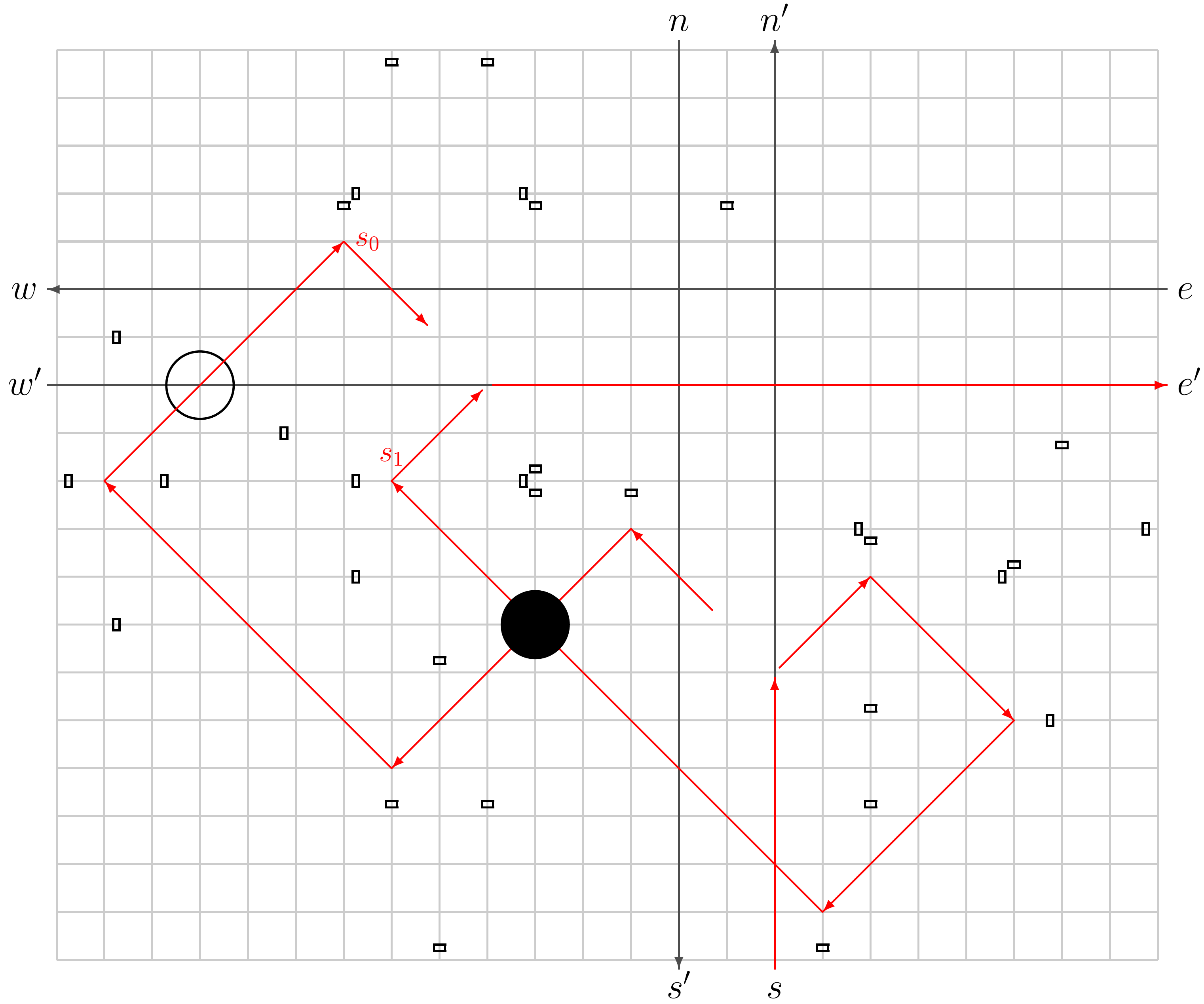
Movements of Balls (State: H, Input: s)



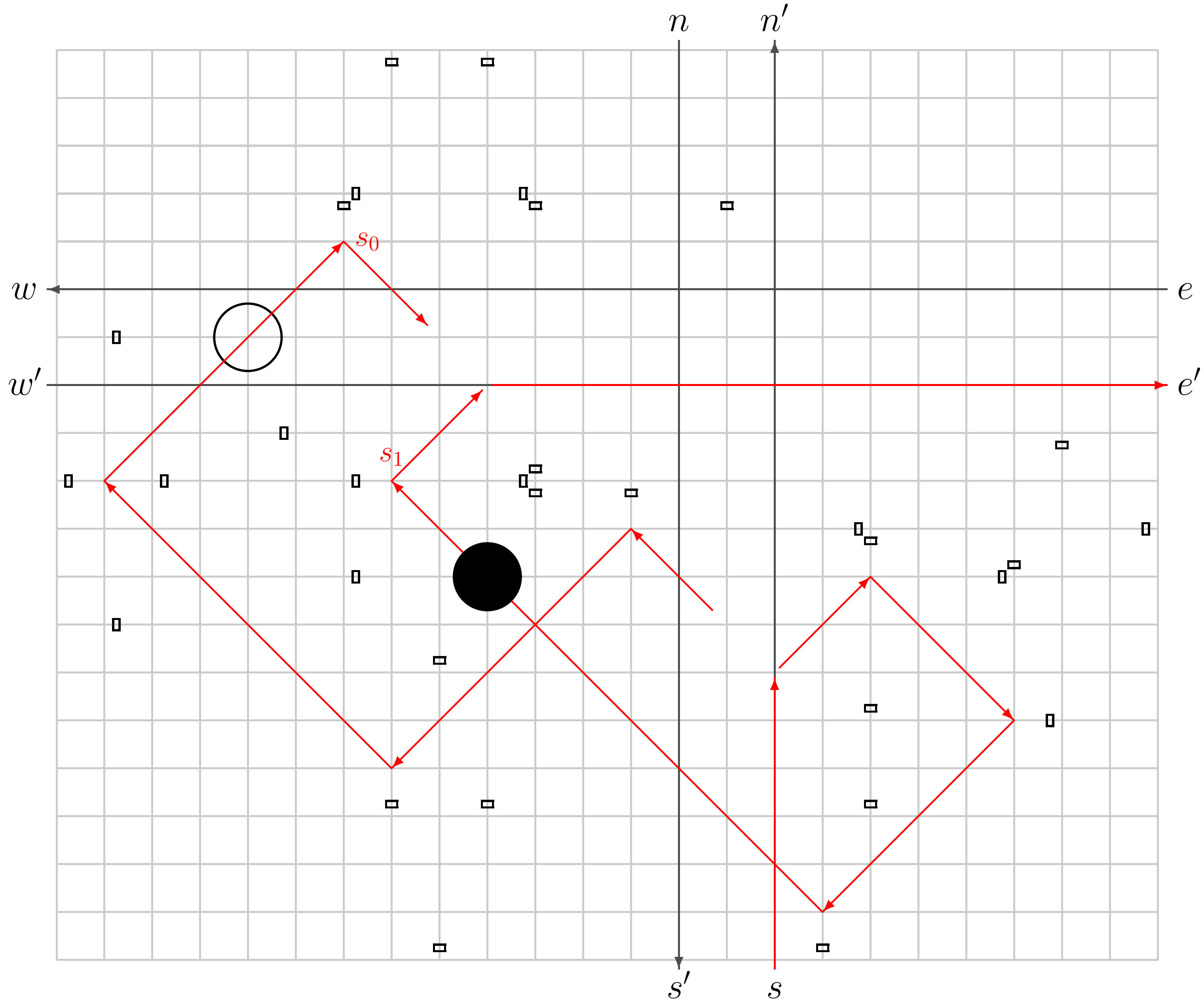
Movements of Balls (State: H, Input: s)



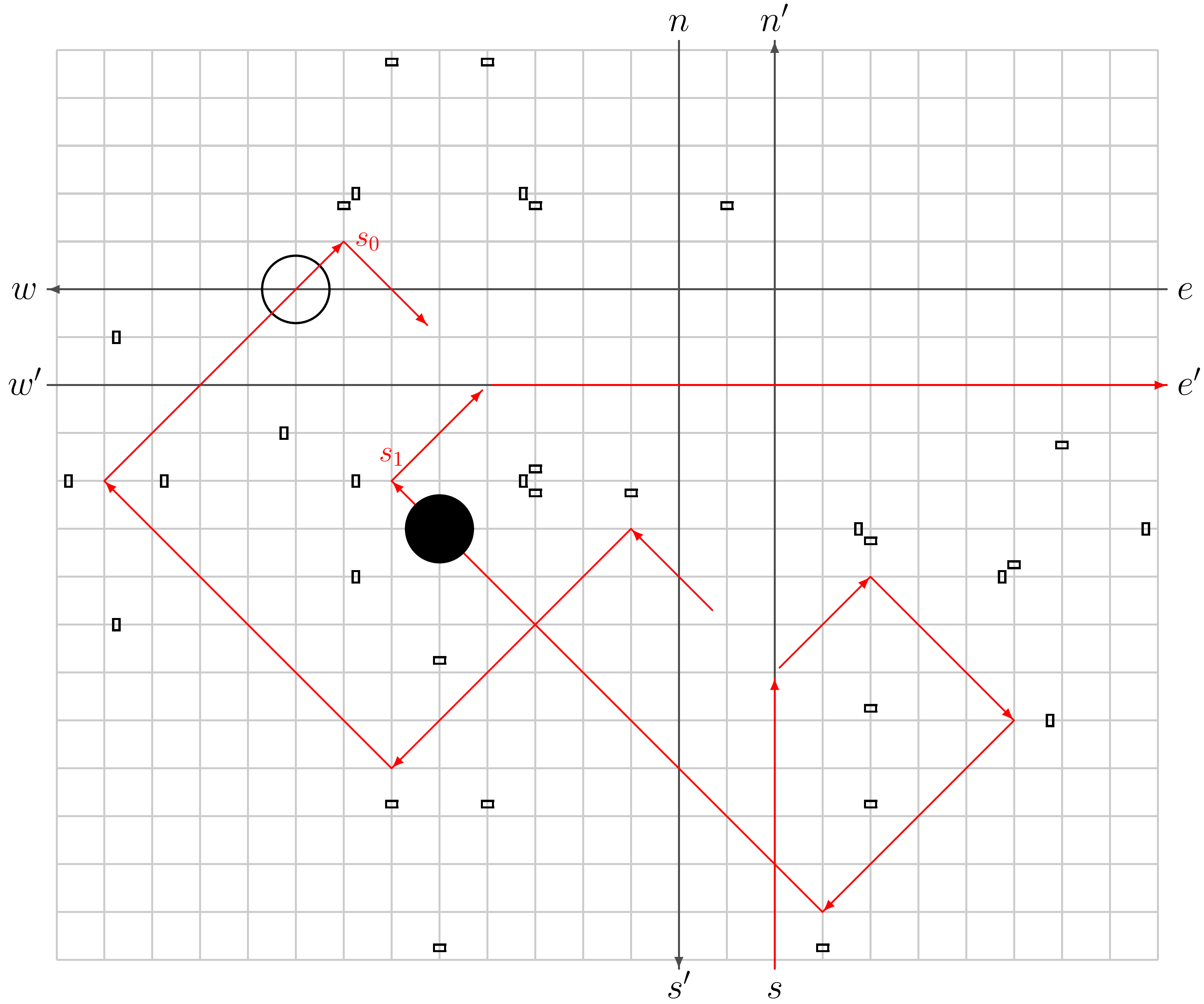
Movements of Balls (State: H, Input: s)



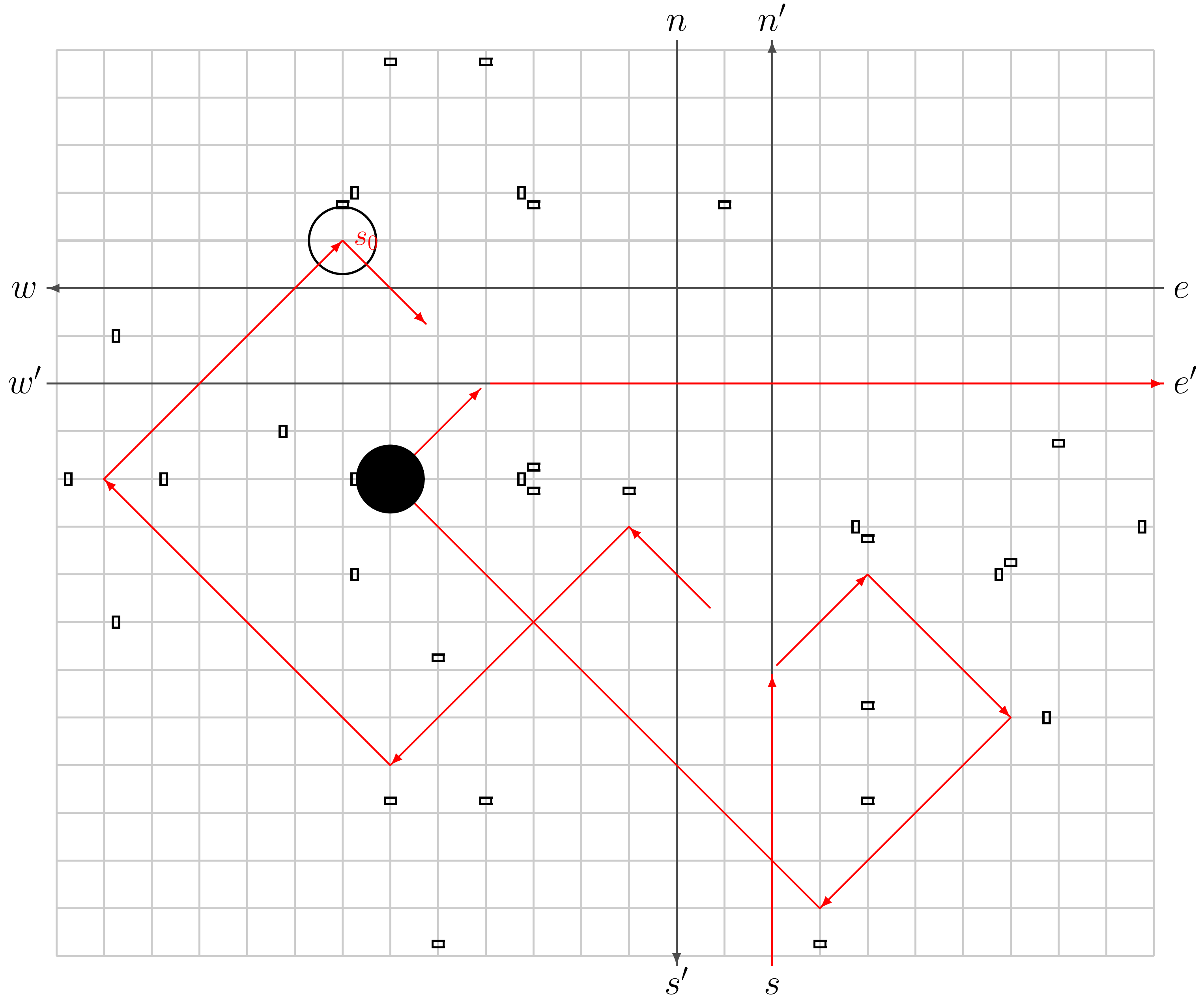
Movements of Balls (State: H, Input: s)



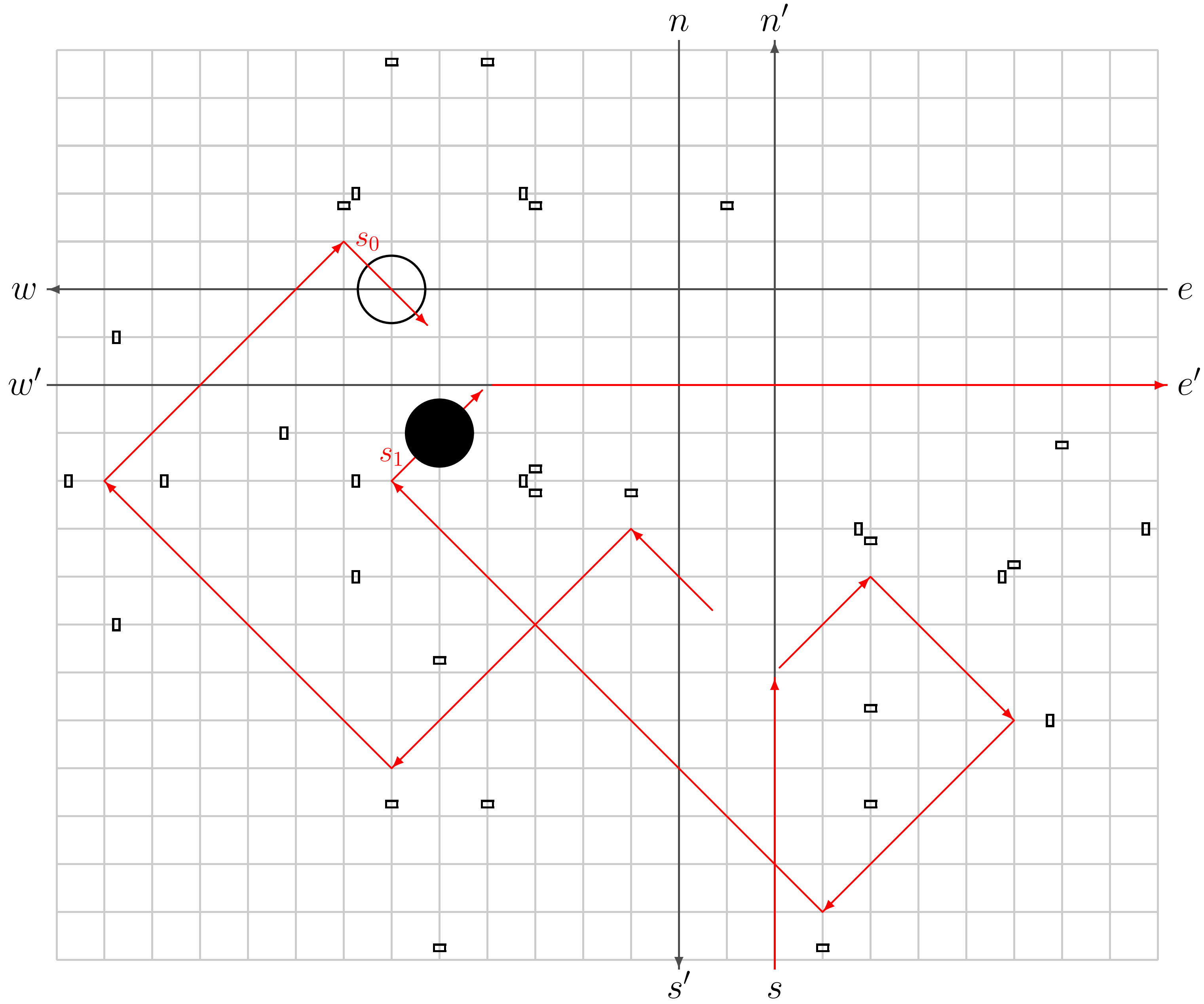
Movements of Balls (State: H, Input: s)



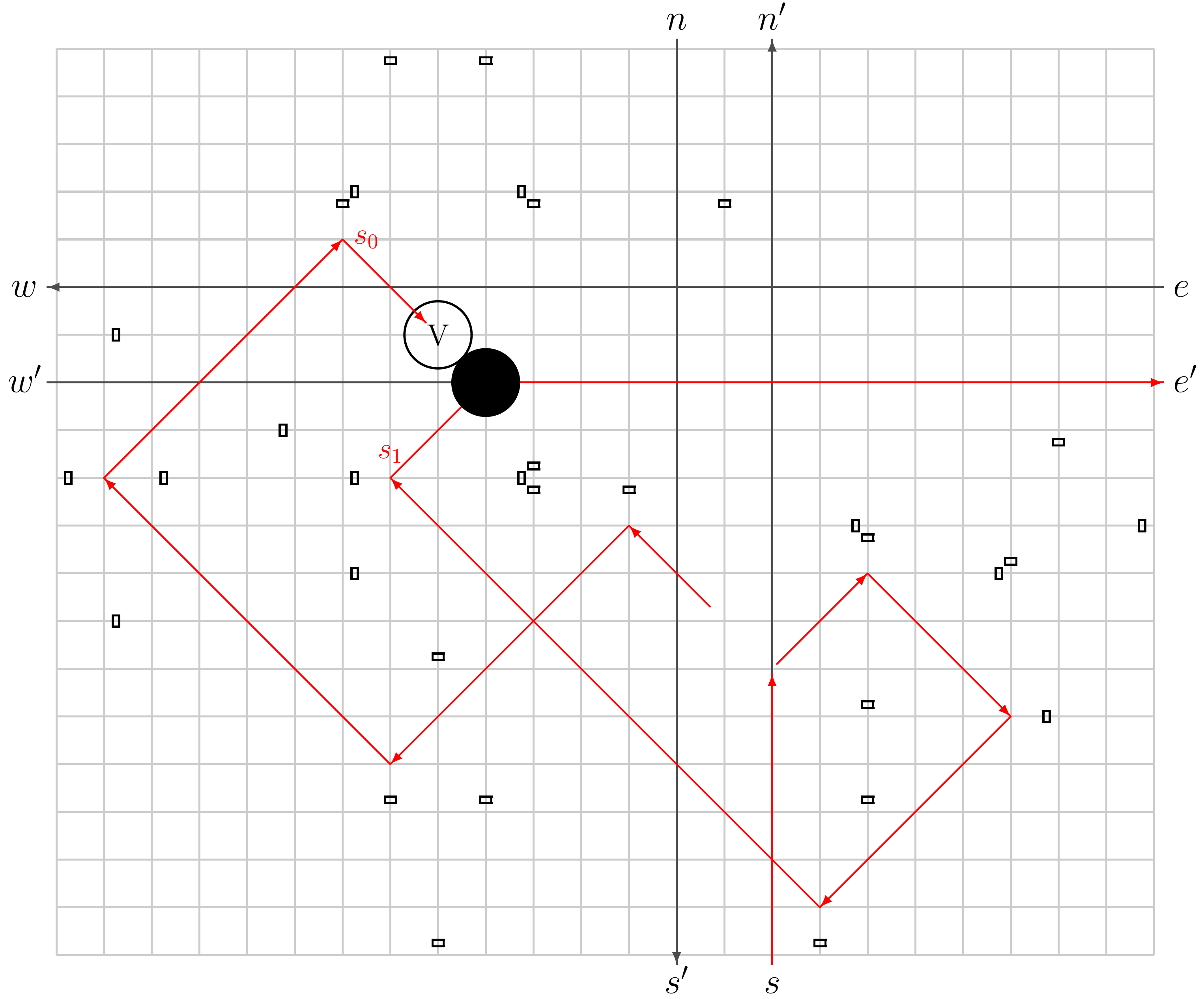
Movements of Balls (State: H, Input: s)



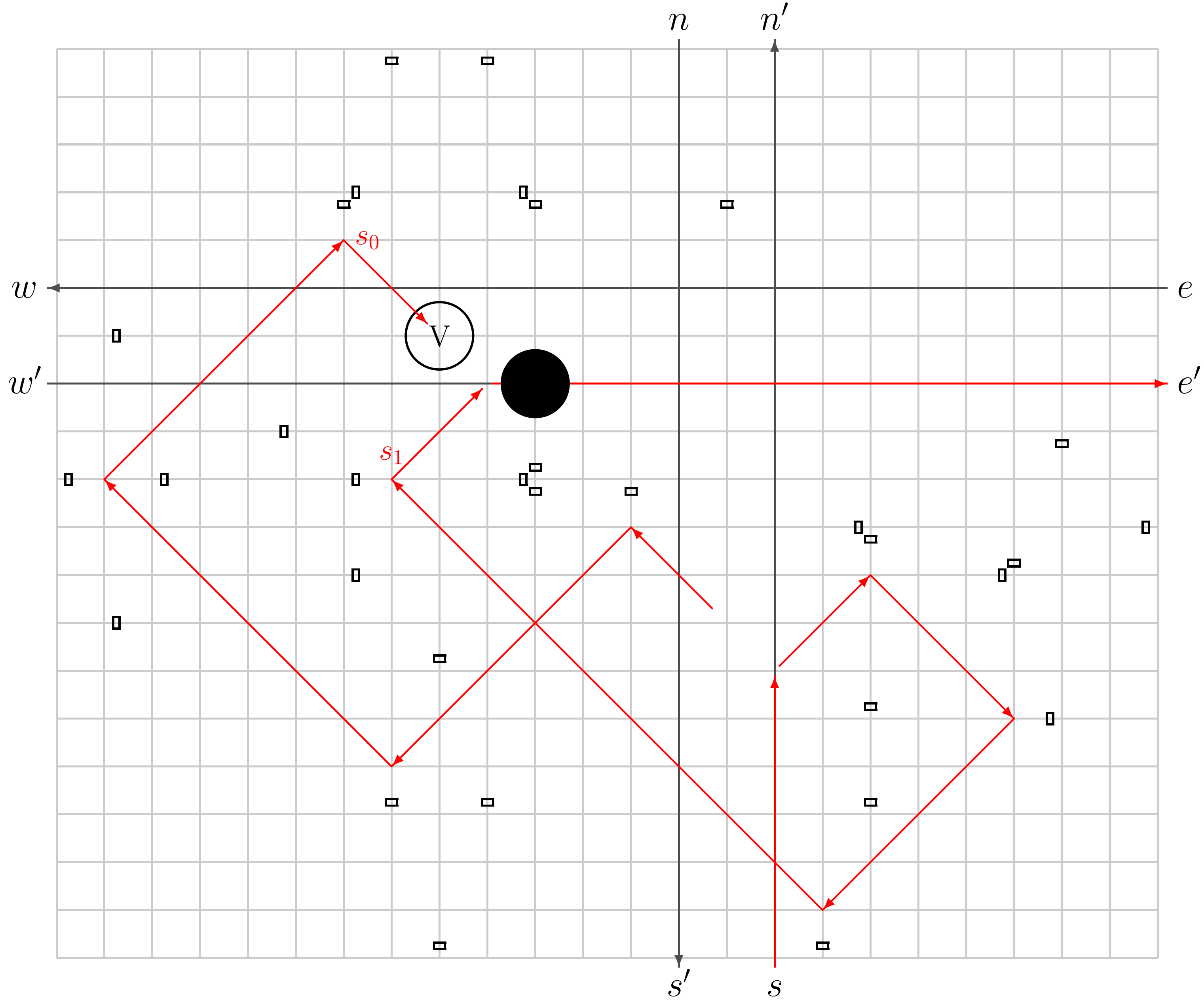
Movements of Balls (State: H, Input: s)



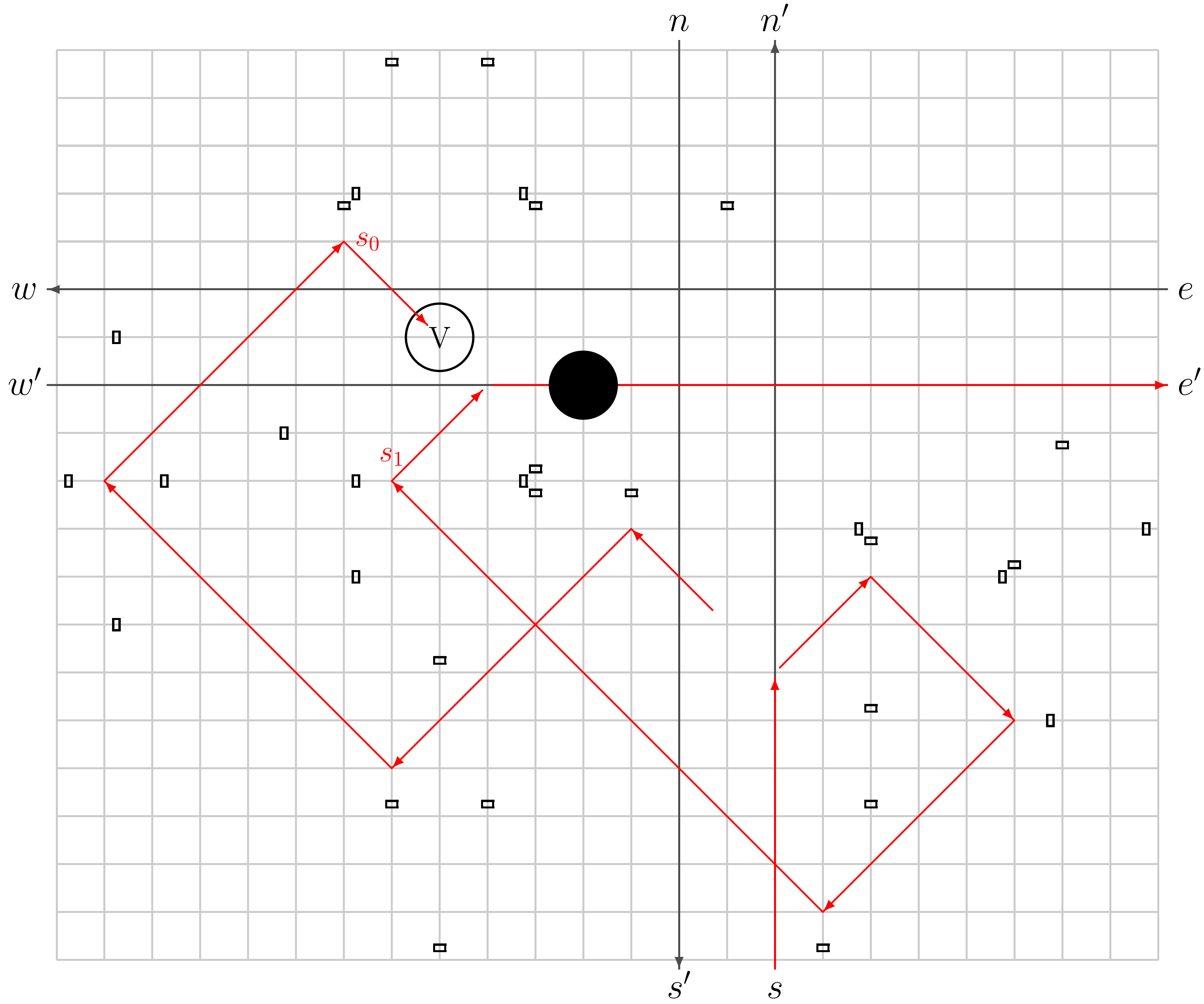
Movements of Balls (State: H, Input: s)



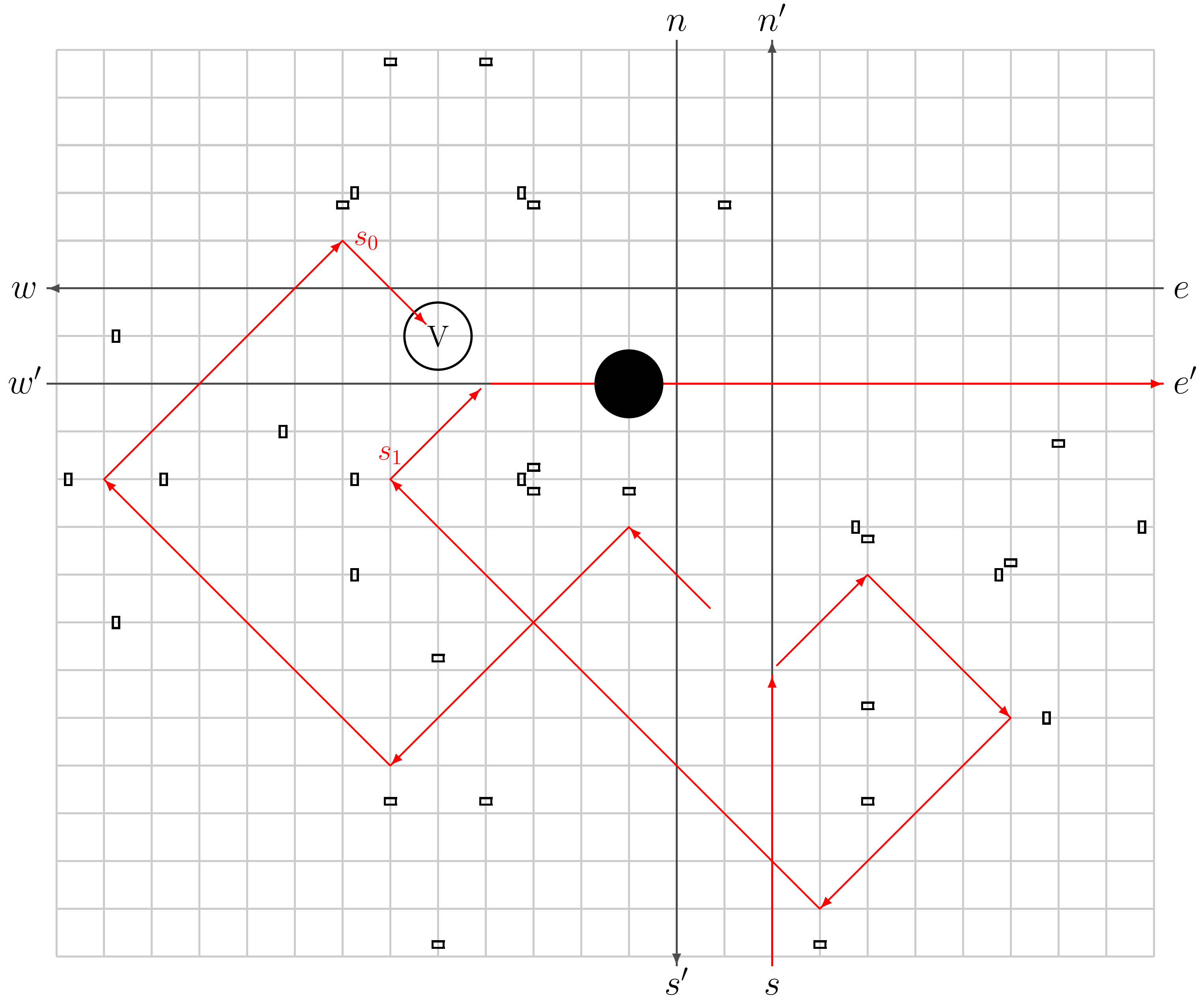
Movements of Balls (State: H, Input: s)



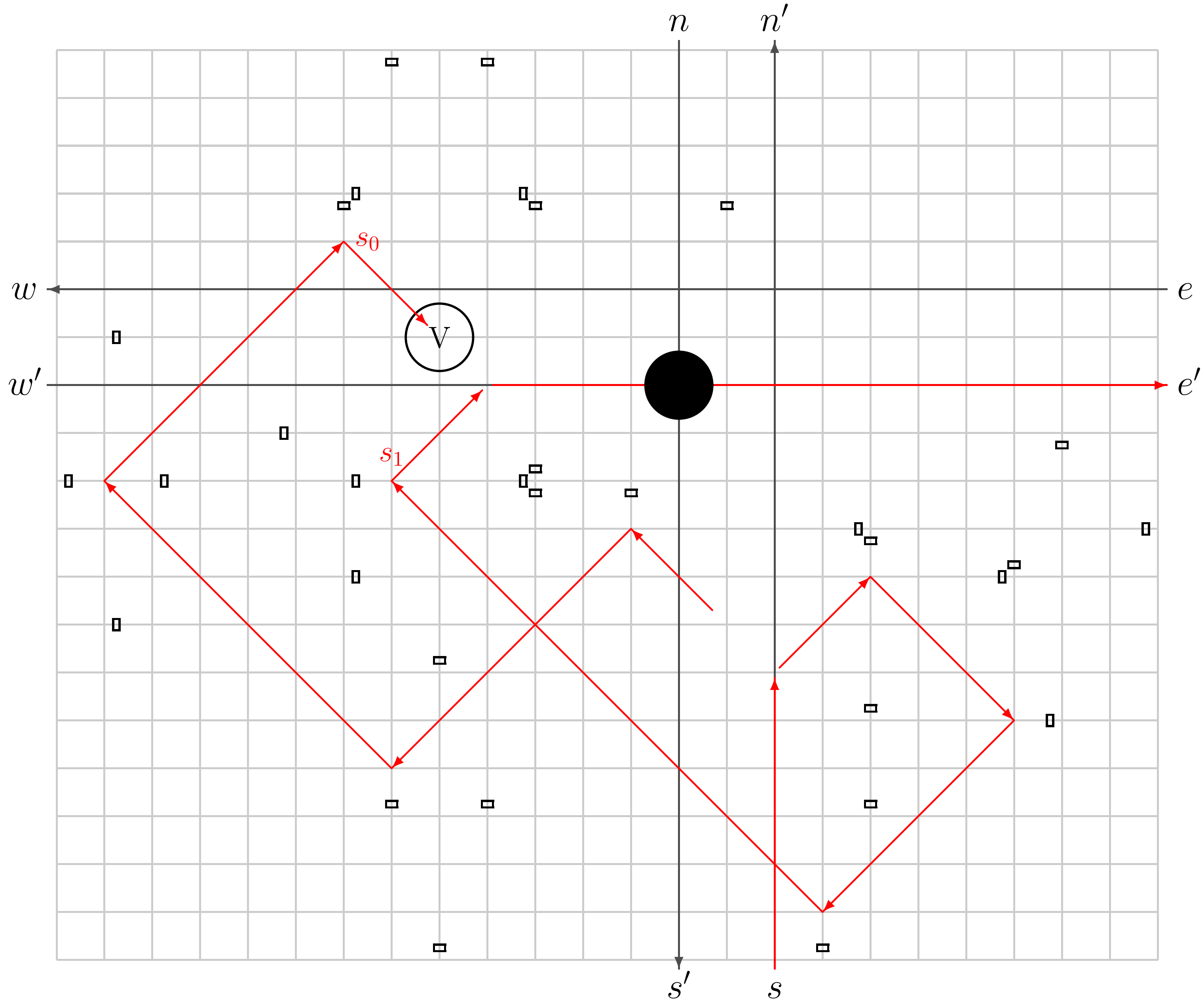
Movements of Balls (State: H, Input: s)



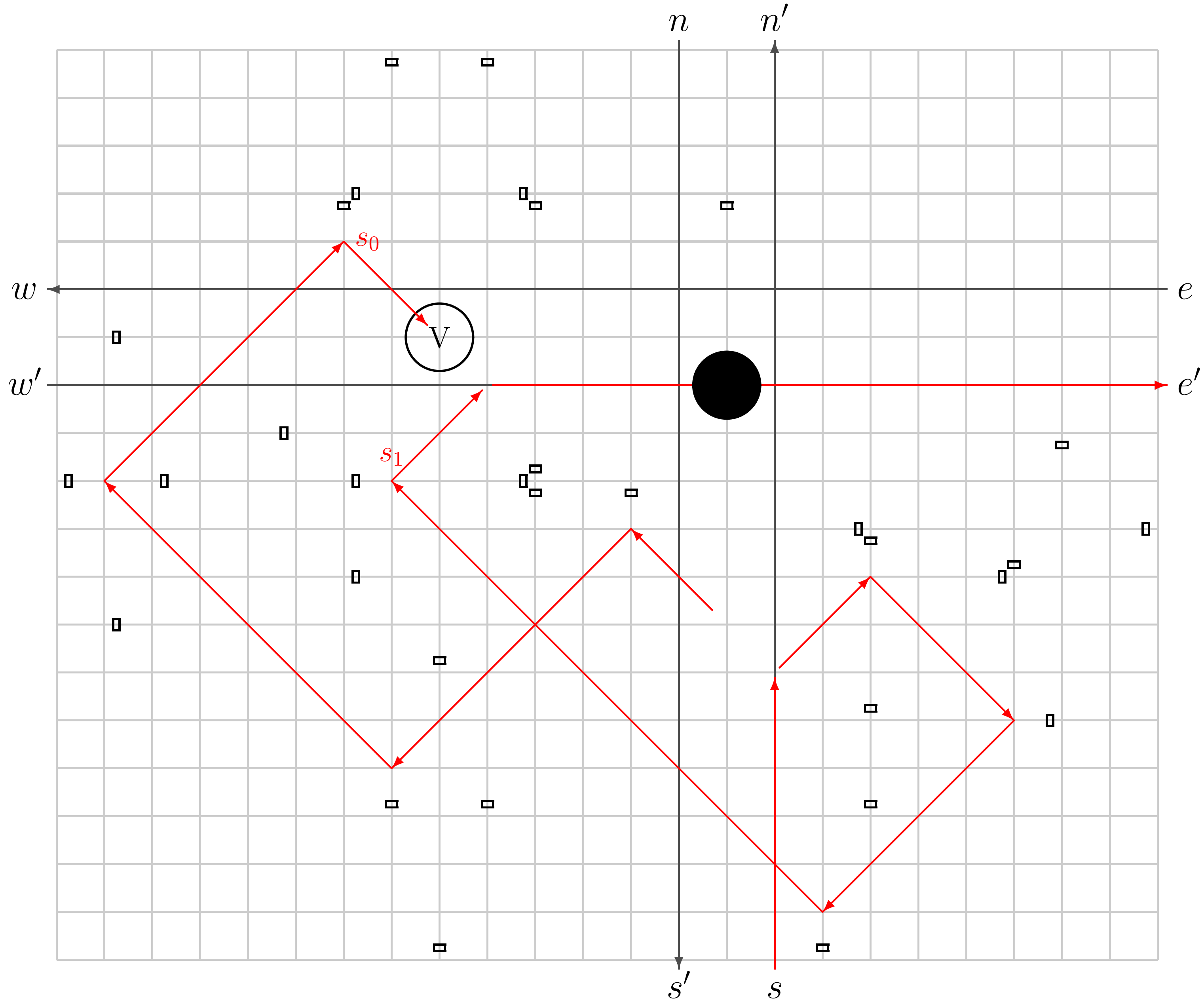
Movements of Balls (State: H, Input: s)



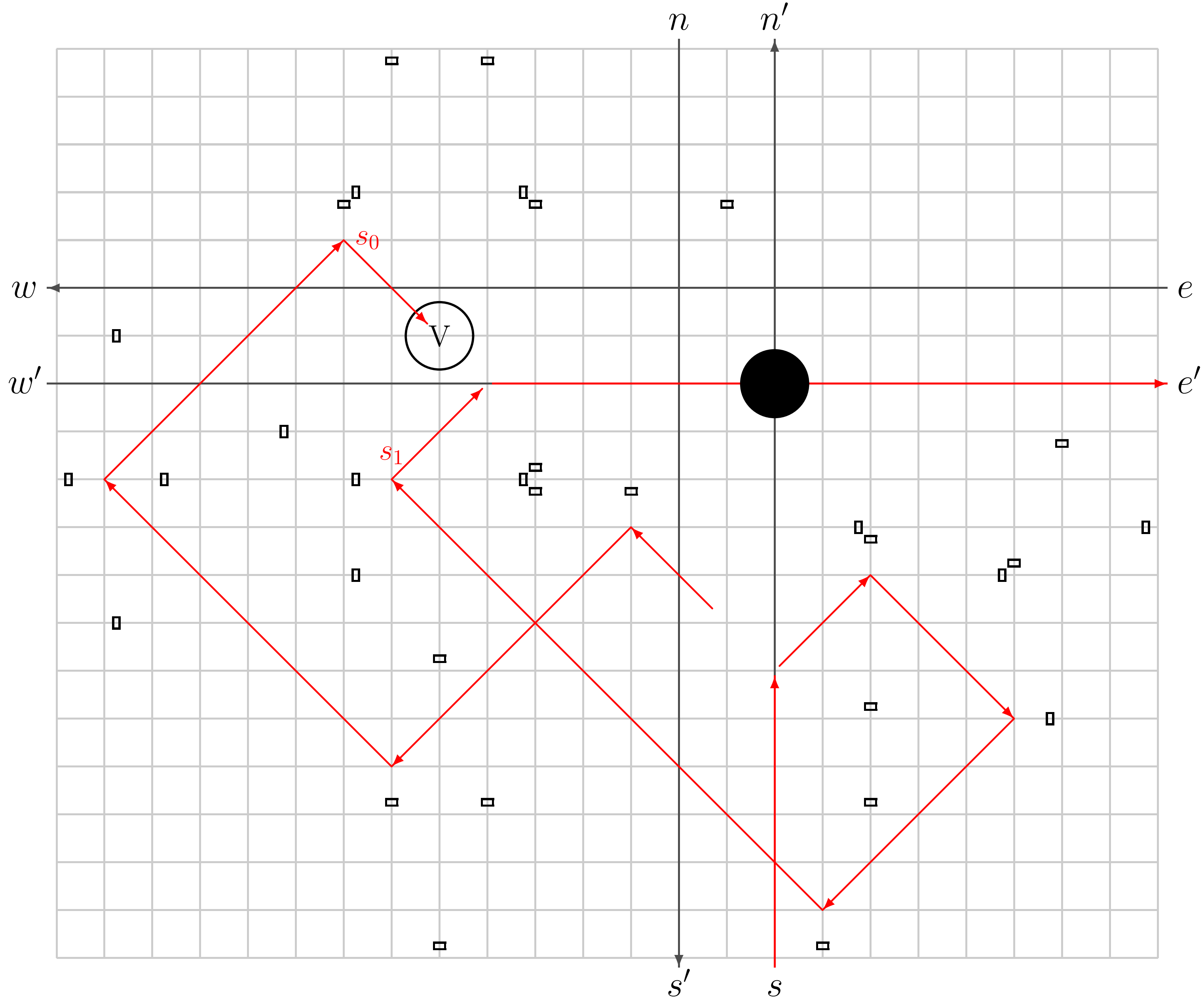
Movements of Balls (State: H, Input: s)



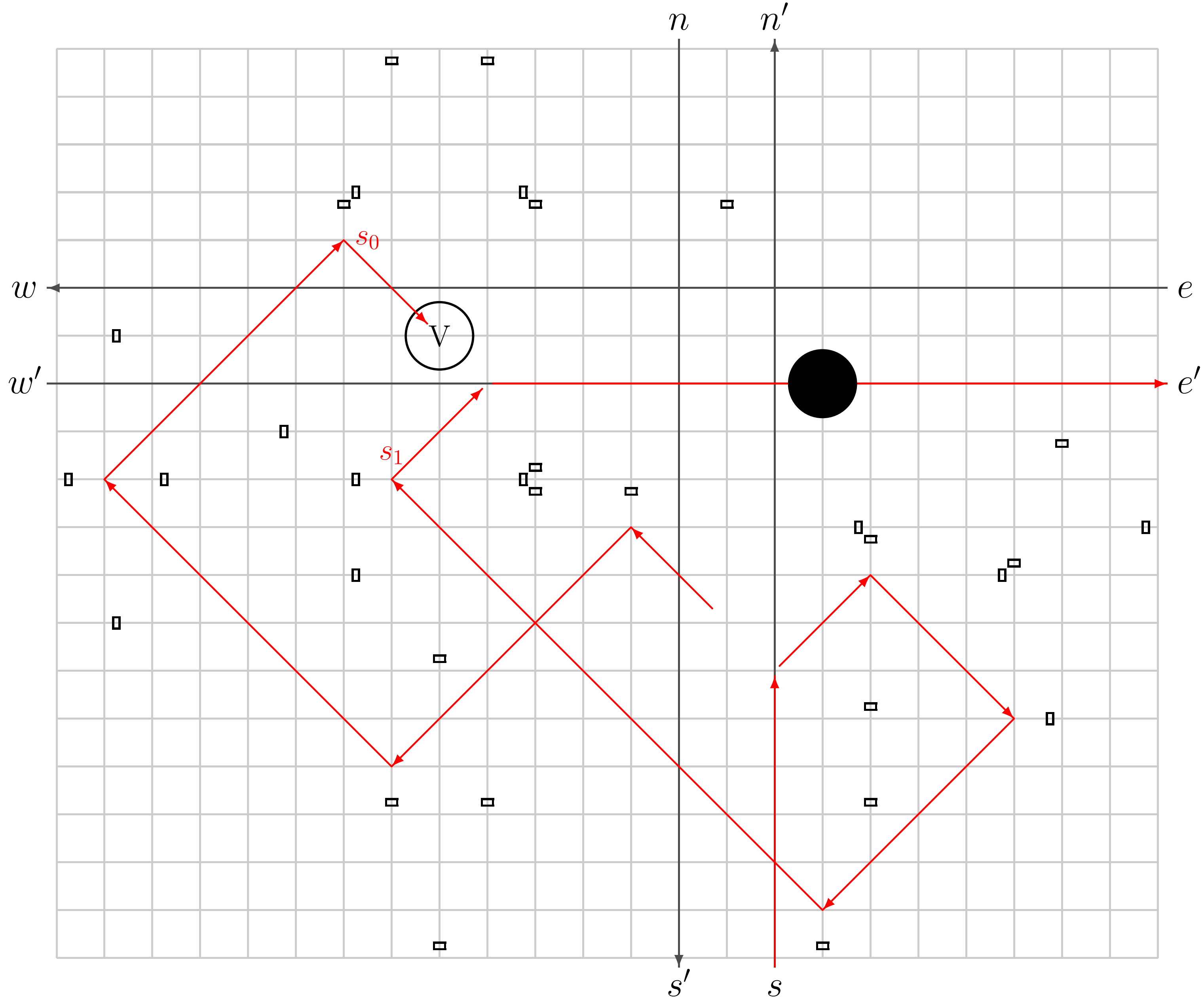
Movements of Balls (State: H, Input: s)



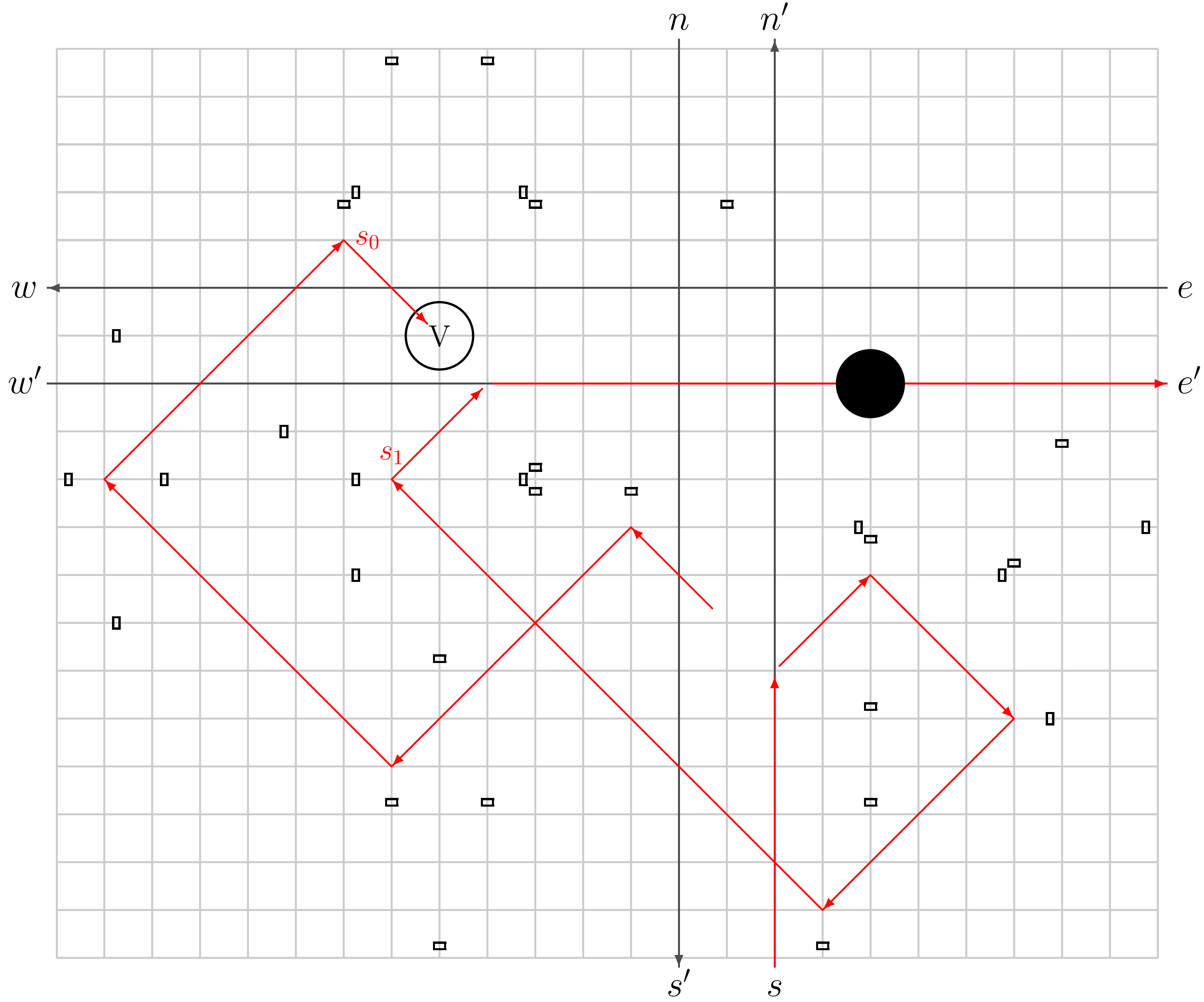
Movements of Balls (State: H, Input: s)



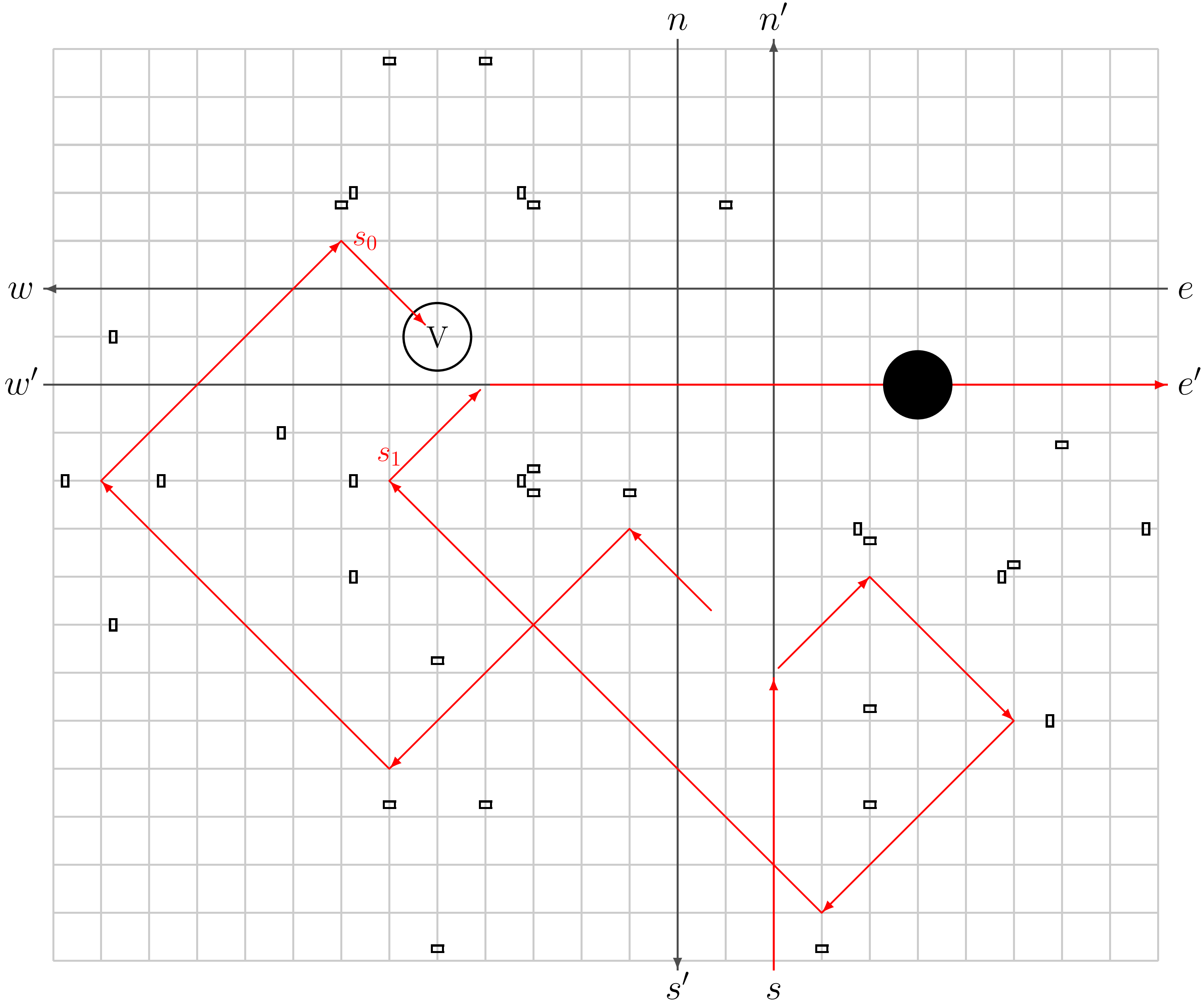
Movements of Balls (State: H, Input: s)



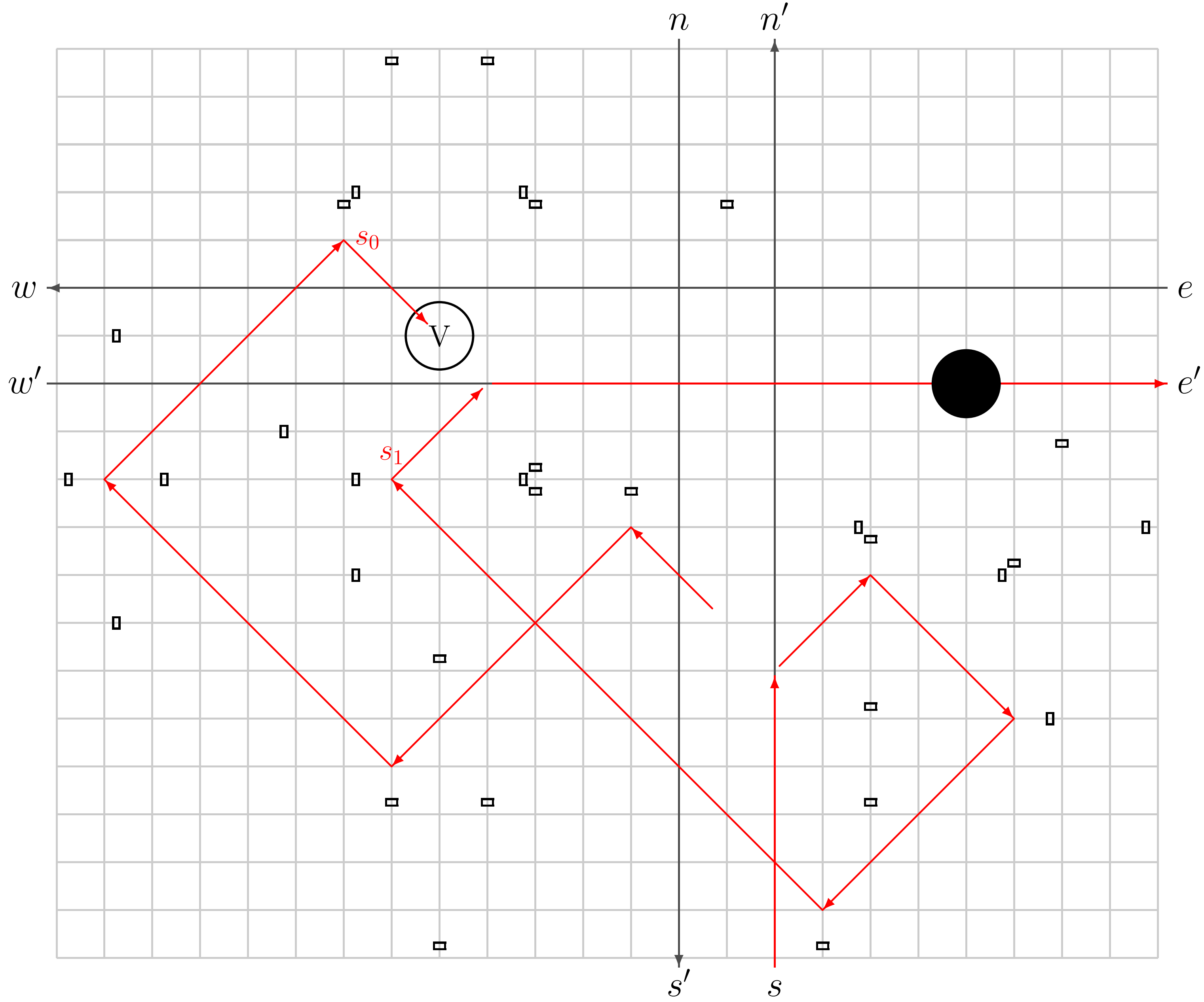
Movements of Balls (State: H, Input: s)



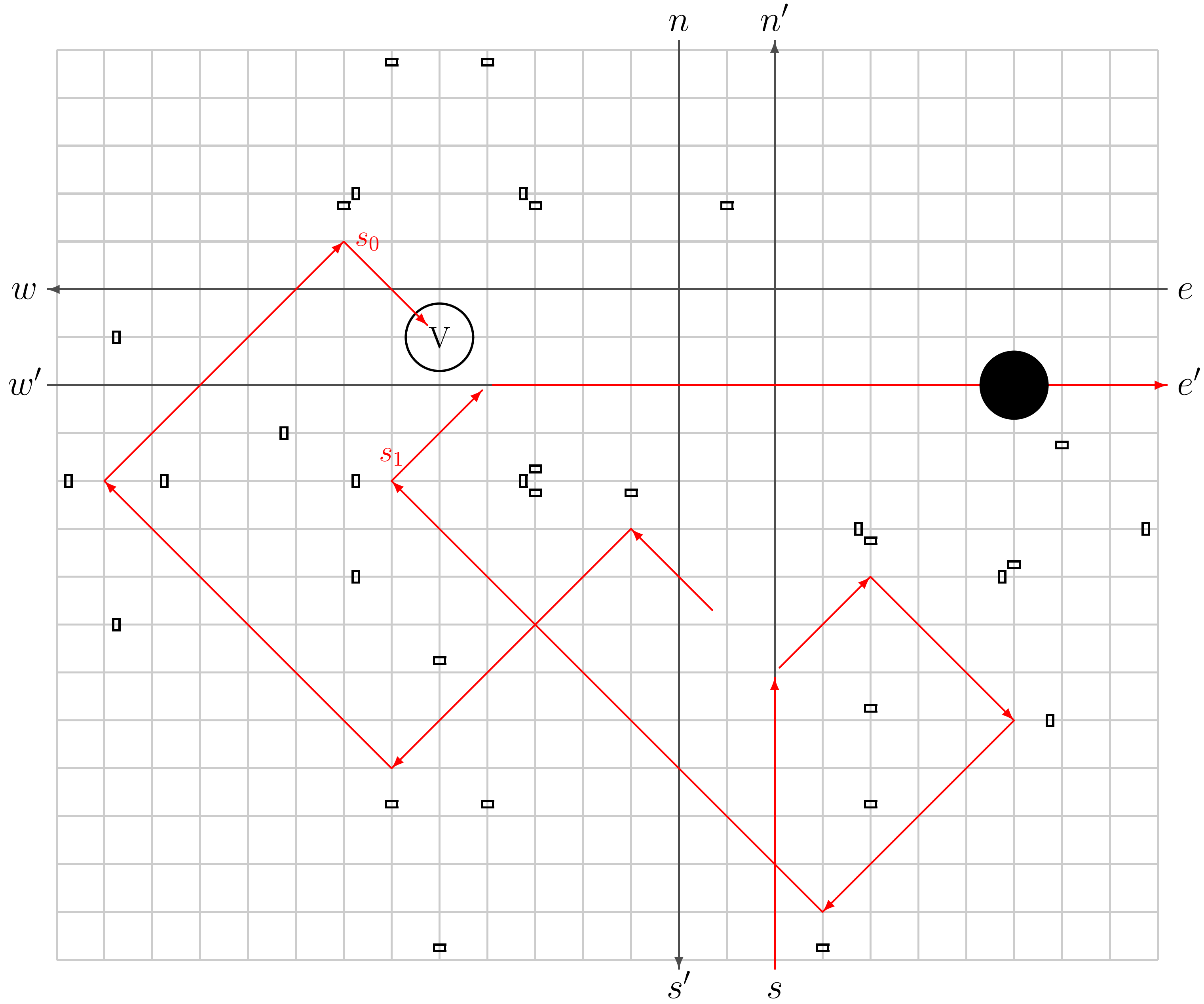
Movements of Balls (State: H, Input: s)



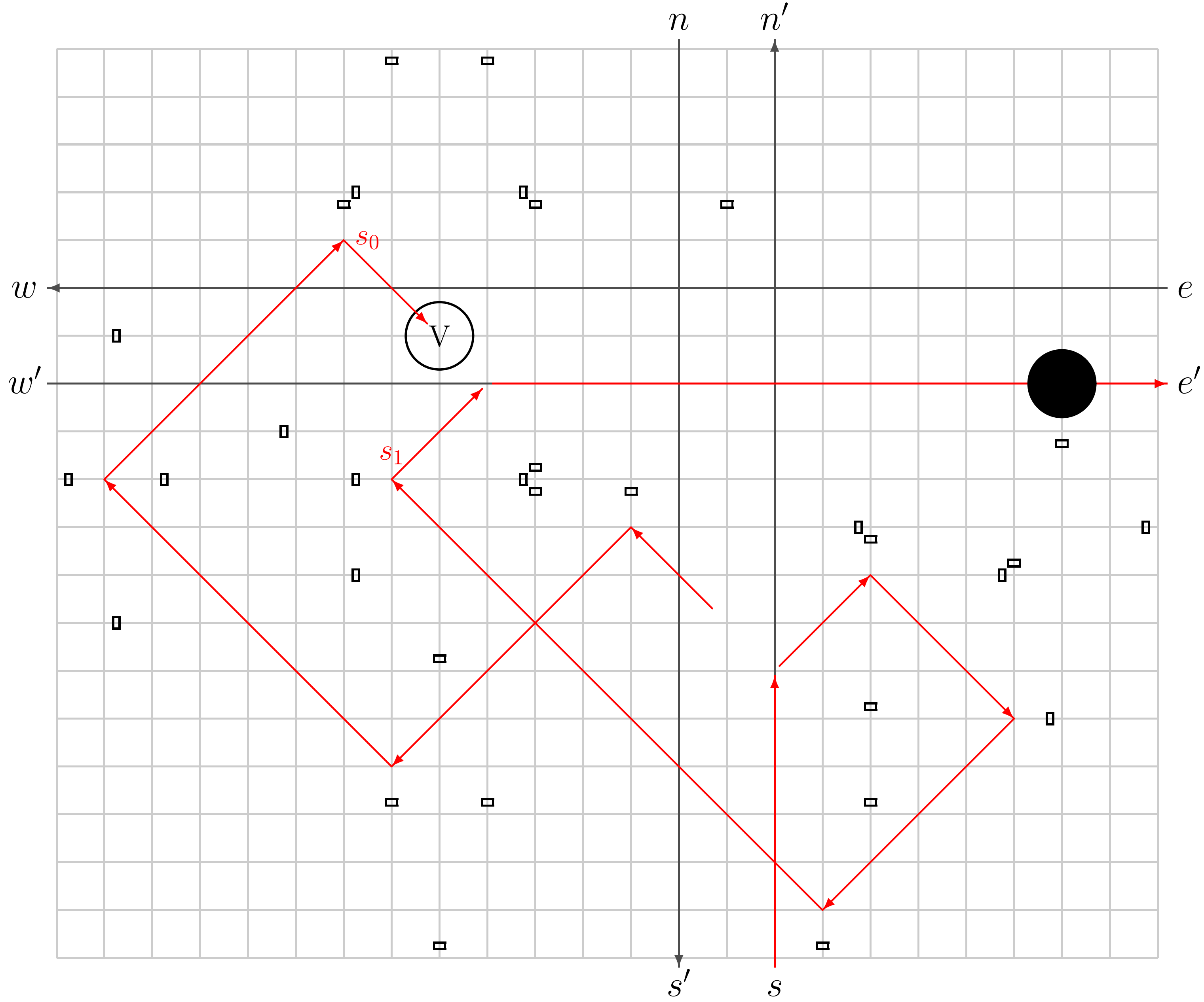
Movements of Balls (State: H, Input: s)



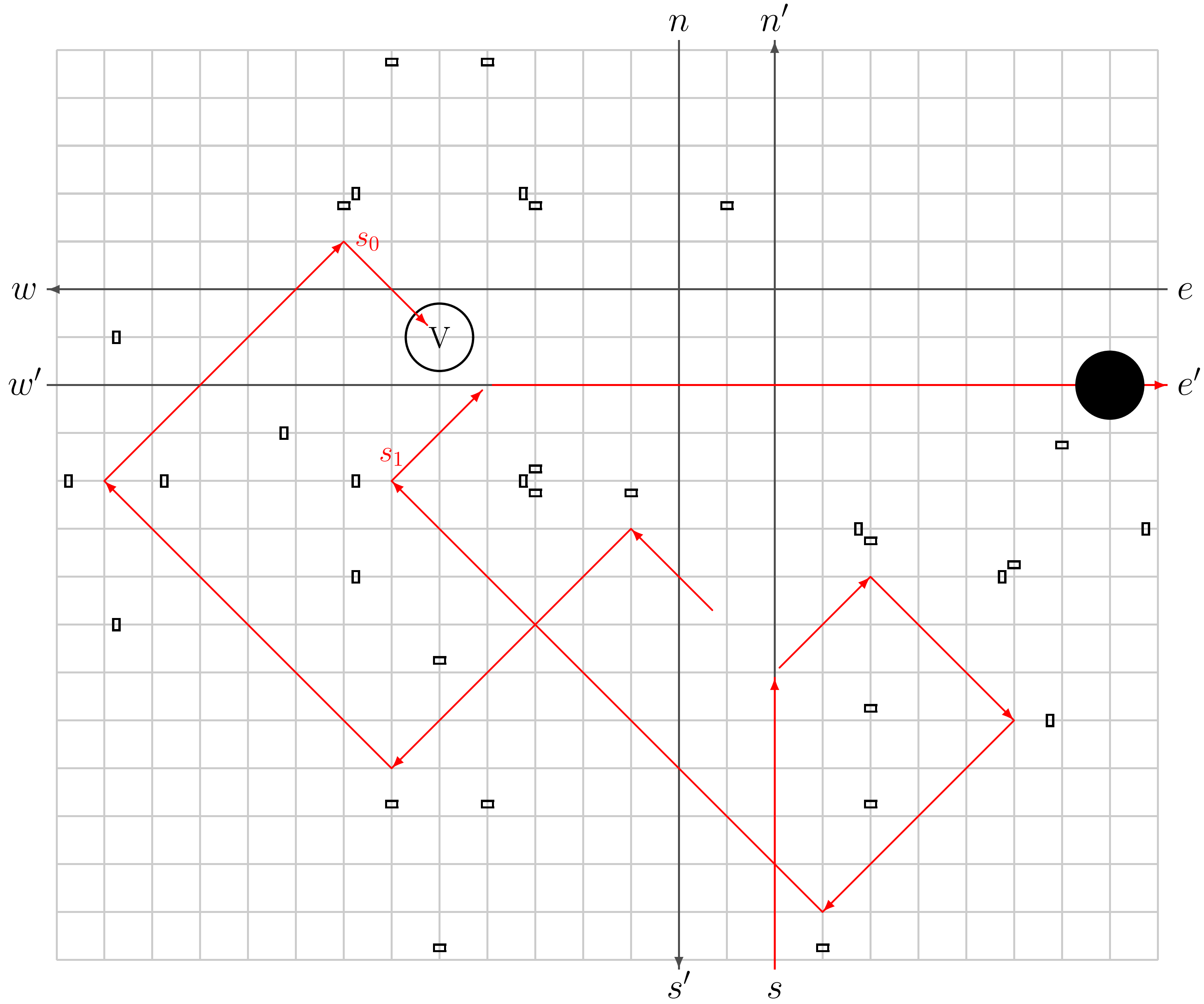
Movements of Balls (State: H, Input: s)



Movements of Balls (State: H, Input: s)



Movements of Balls (State: H, Input: s)



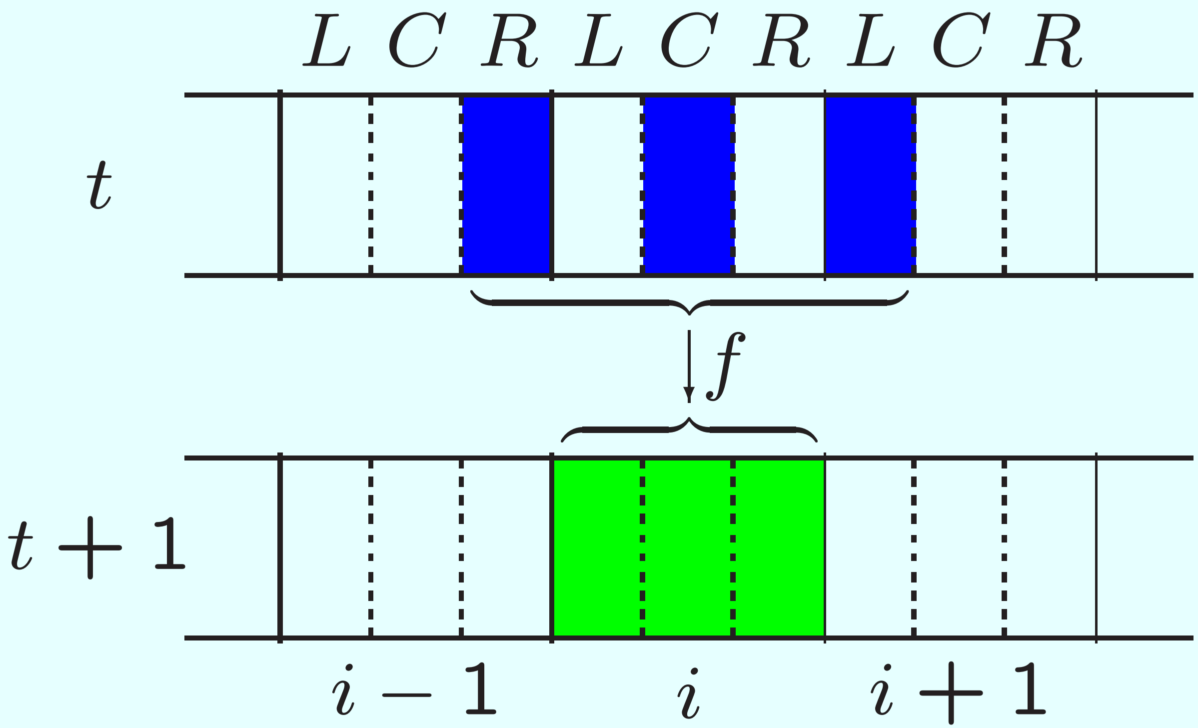
3. Reversible Cellular Automata

Reversible Cellular Automata (RCAs)

- It is a CA whose global function is one-to-one.
- A kind of spatio-temporal model of a physically reversible space.
- In spite of the strong restriction of reversibility, they have rich ability of computing.
  - Computation-universality
  - Self-reproduction
  - Synchronization
  - etc.

Partitioned Cellular Automata

- 1D Partitioned CA (PCA)



A local function f of a 1D PCA.

- We can design RCAs easily using PCAs.

Universal Reversible CAs
— 1D Case —

- On infinite configurations:
  24-state RPCA [Morita, 2008]
- On finite configurations:
  98-state RPCA [Morita, 2007]

cf. 1D Universal Irreversible CAs:

- On infinite configurations:
  2-state CA (ECA of rule 110) [Cook, 2004]
- On finite configurations:
  7-state CA (a modified model) [Lindgren et al., 1990]

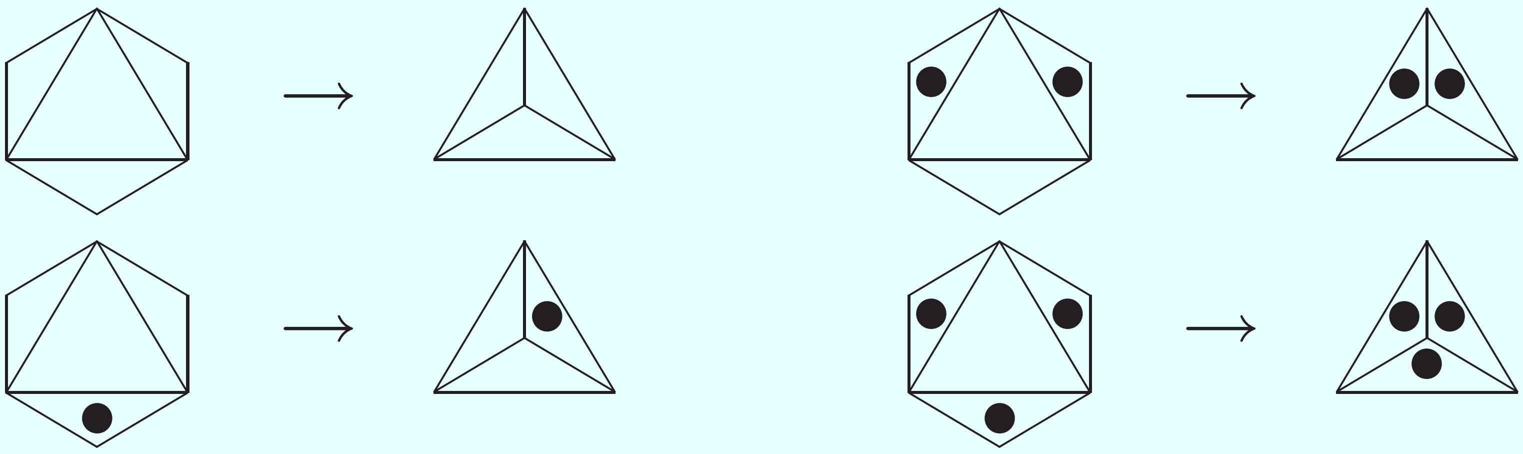
Universal Reversible CAs
— 2D Case —

- On infinite configurations:
  - 2-state Margolus-neighbor RCA [Margolus, 1984]
  - 16-state RPCAs [Morita and Ueno, 1992]
  - 8-state triangular RPCA [Imai and Morita, 1998]

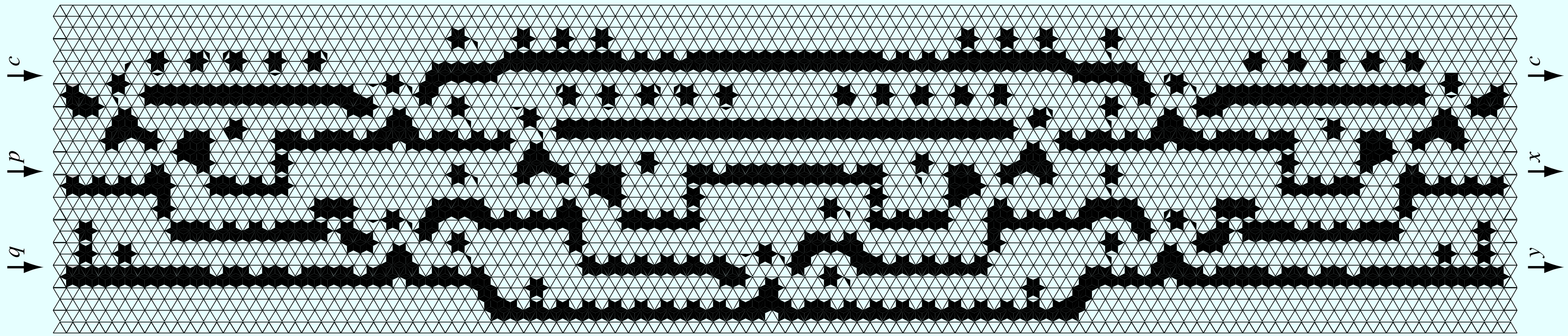
An 8-State Triangular RPCA T_1

[Imai and Morita, 1998]

- It has an extremely simple local function:



A Fredkin Gate in a Triangular 8-State RPCA T_1

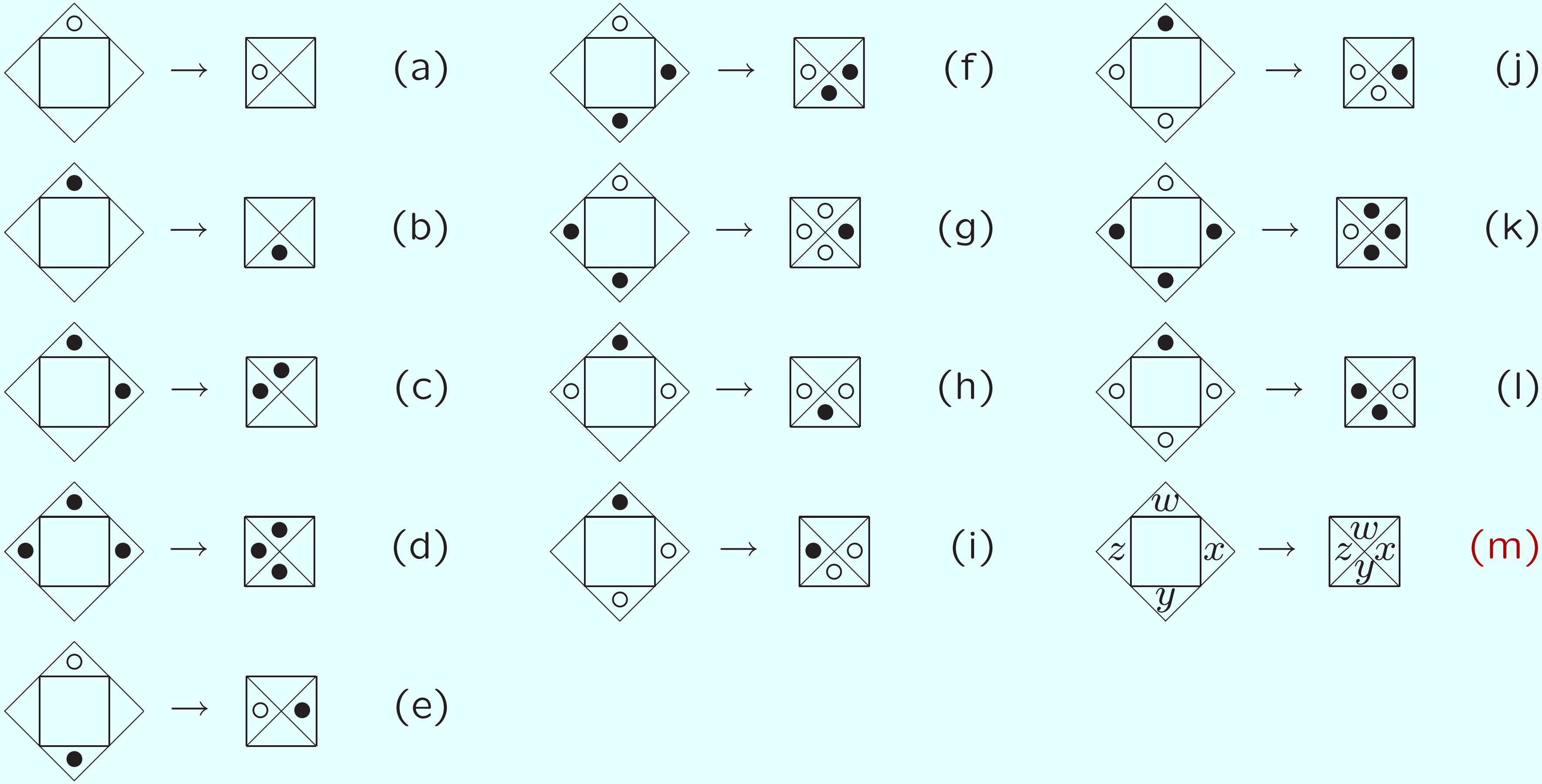


Universal Reversible CAs
— 2D Case —

- On infinite configurations:
  - 2-state Margolus-neighbor RCA [Margolus, 1984]
  - 16-state RPCAs [Morita and Ueno, 1992]
  - 8-state triangular RPCA [Imai and Morita, 1998]
- On finite configurations:
  - 81-state RPCA [Morita and Ogiro, 2001]

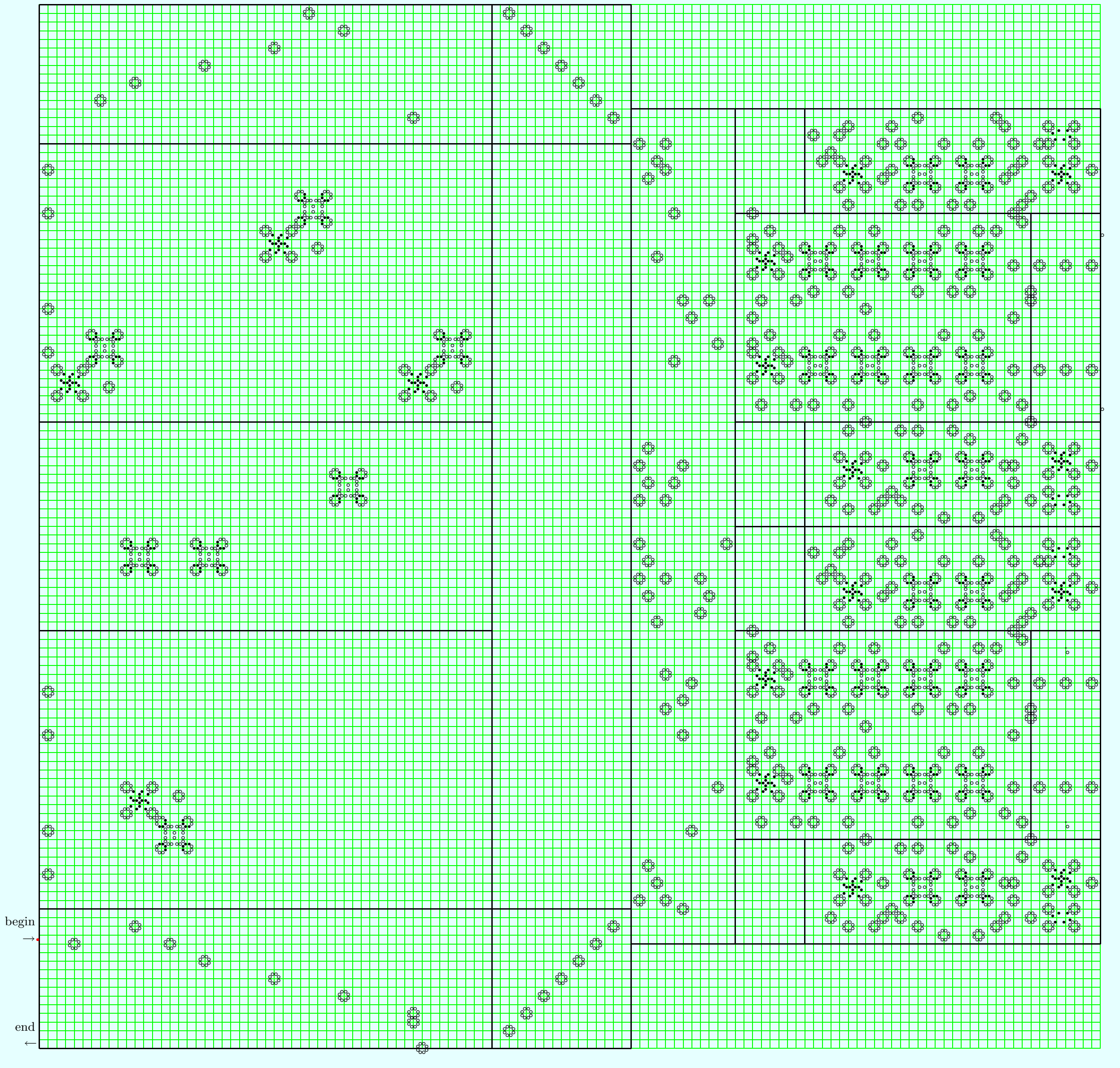
A 3^4-State Universal RPCA P_3

$$P_3 = (\mathrm{Z}^2, \{0, 1, 2\}^4, g_3, (0, 0, 0, 0))$$



The rule scheme (m) represents 33 rules not specified by (a)–(l)
$(w, x, y, z \in \{$ blank, ○, ● $\} = \{0, 1, 2\})$.

Reversible Counter Machine in P_3 Space



Movie of an RCM(2) in P_3

Self-Reproduction of a Worm in 2D RCA

[Morita and Imai, 1996]

Self-Reproduction of a Loop in 3D RCA

[Imai, Hori and Morita, 2002]

Concluding Remarks

- We saw even very simple reversible systems have computation-universality.
- Computation can be carried out in a very different way from that of conventional computers.
- We expect that further studies on them will give new insights for future computing.

Thank you for your attention!