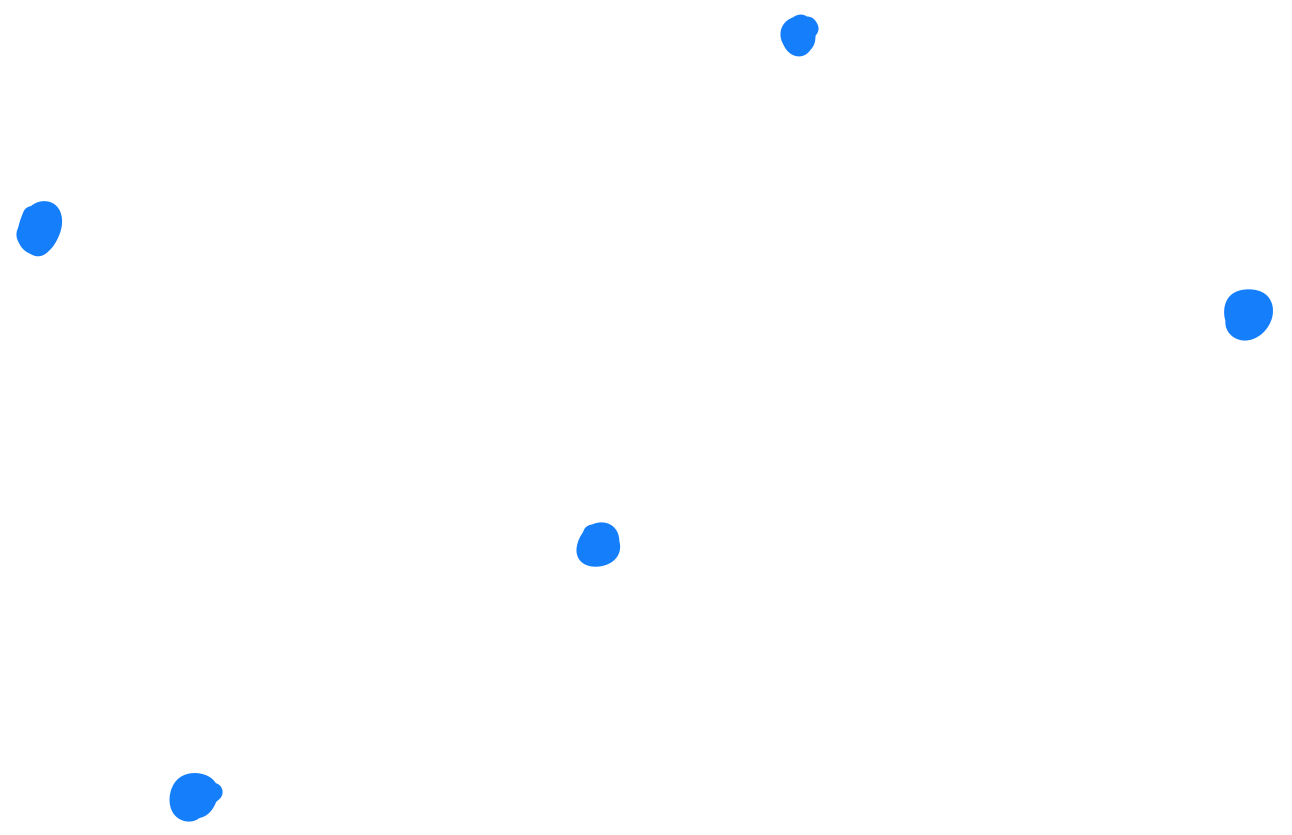
The agreement power of disagreement

Quentin Bramas, Anissa Lamani, and Sébastien Tixeuil

Strasbourg University, Strasbourg University, and Sorbonne University

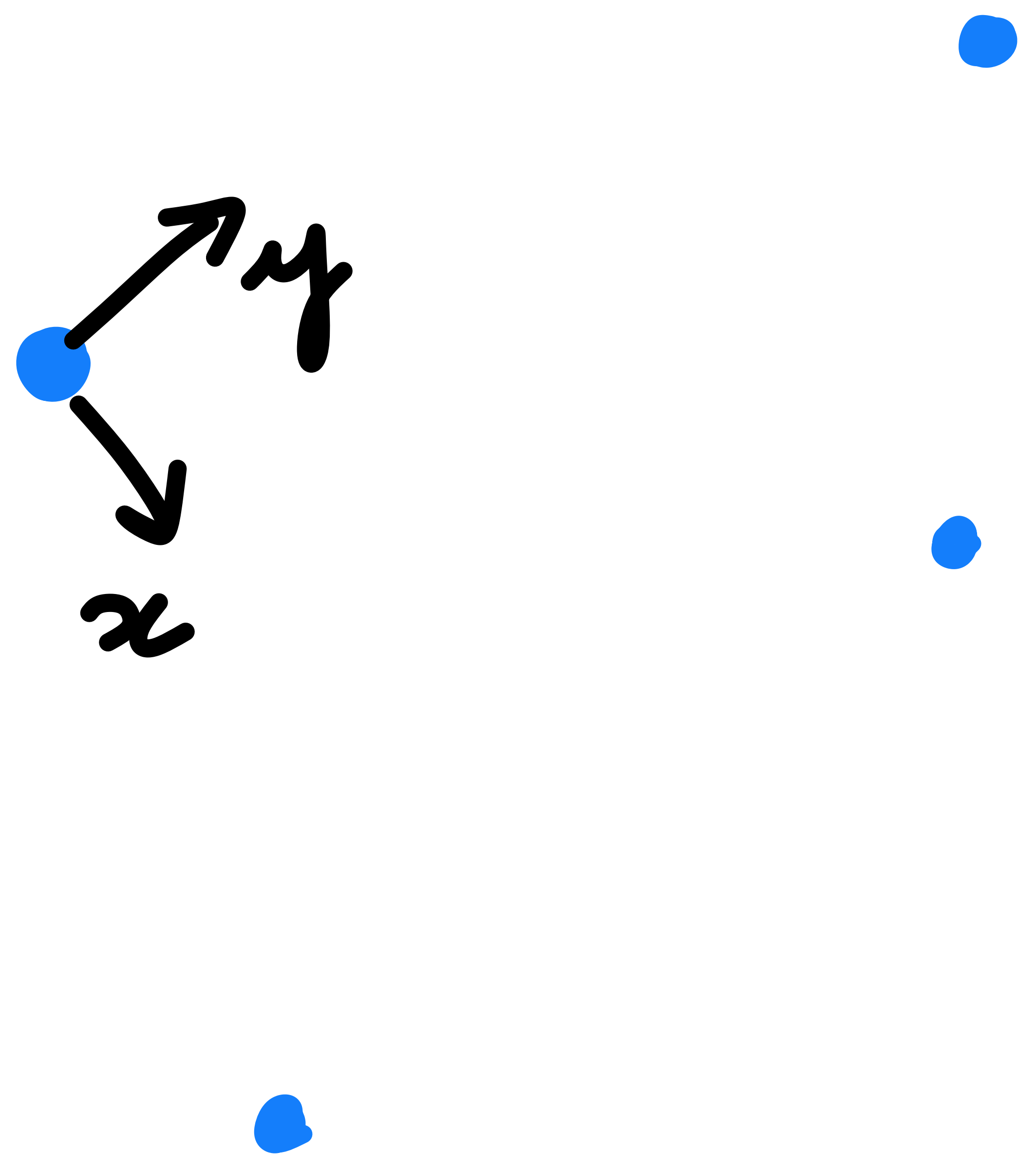
Mobile Autonomous Robots

- Anonymous
- Uniform
- Disoriented
- Silent
- Oblivious



Execution Cycle

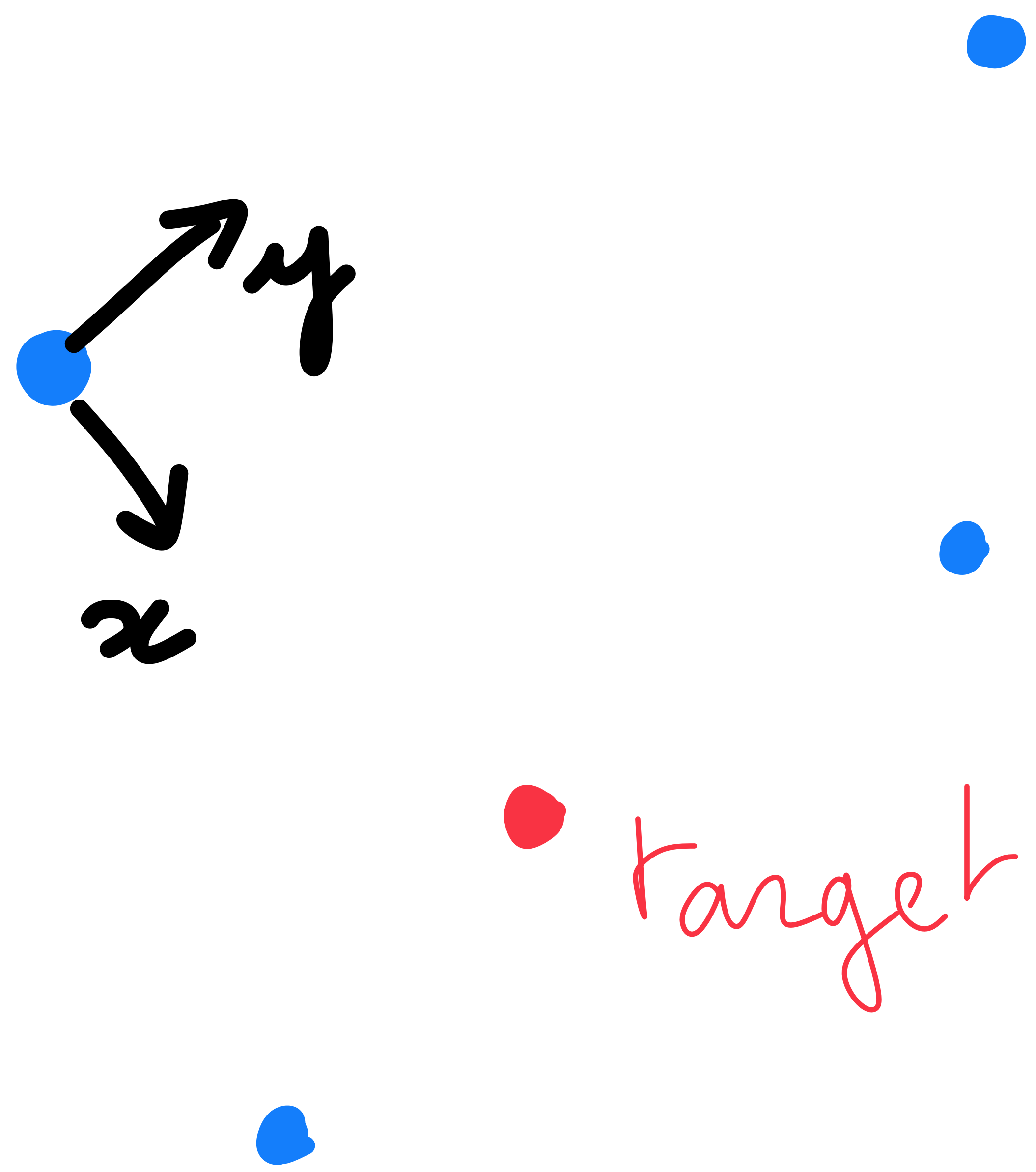
- Look
- Compute
- Move



- Fully-Synchronous
- Semi-Synchronous

Execution Cycle

- Look
- Compute
- Move

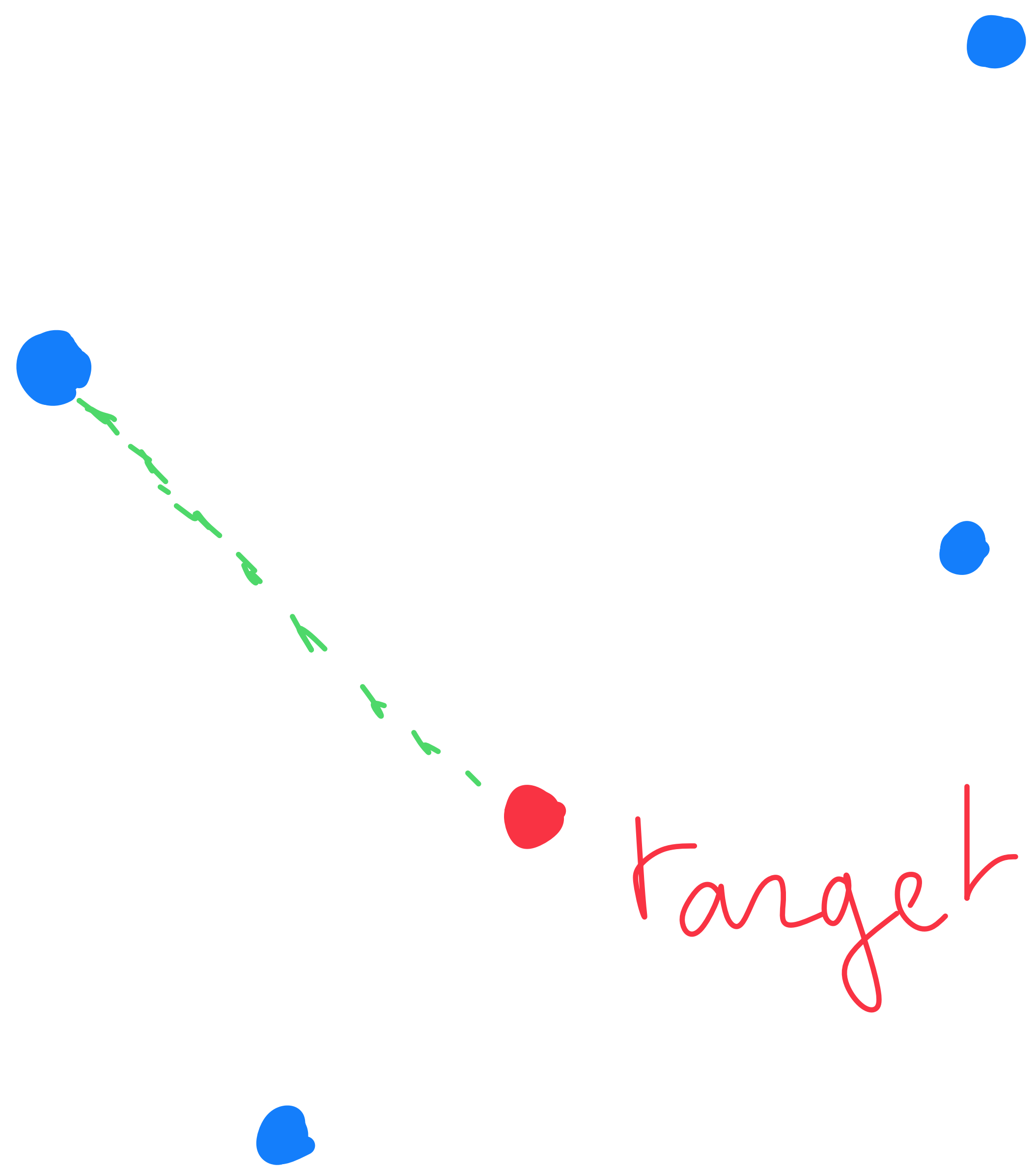


- Fully-Synchronous
- Semi-Synchronous

Execution Cycle

- Look
- Compute
- Move

- Fully-Synchronous
- Semi-Synchronous



Execution Cycle

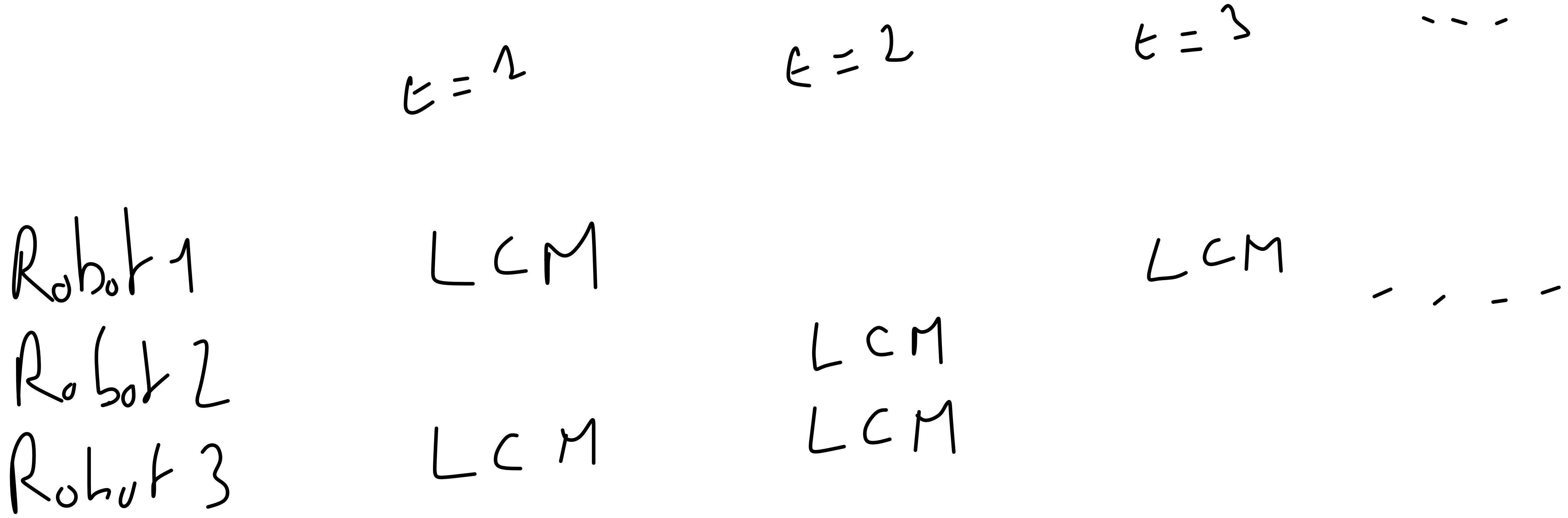
- Look
- Compute
- Move

- Fully-Synchronous
- Semi-Synchronous

	t = 1	t = 2	t = 3	...
Robot 1	LCM	LCM	LCM	...
Robot 2	LCM	LCM	LCM	
Robot 3	LCM	LCM	LCM	

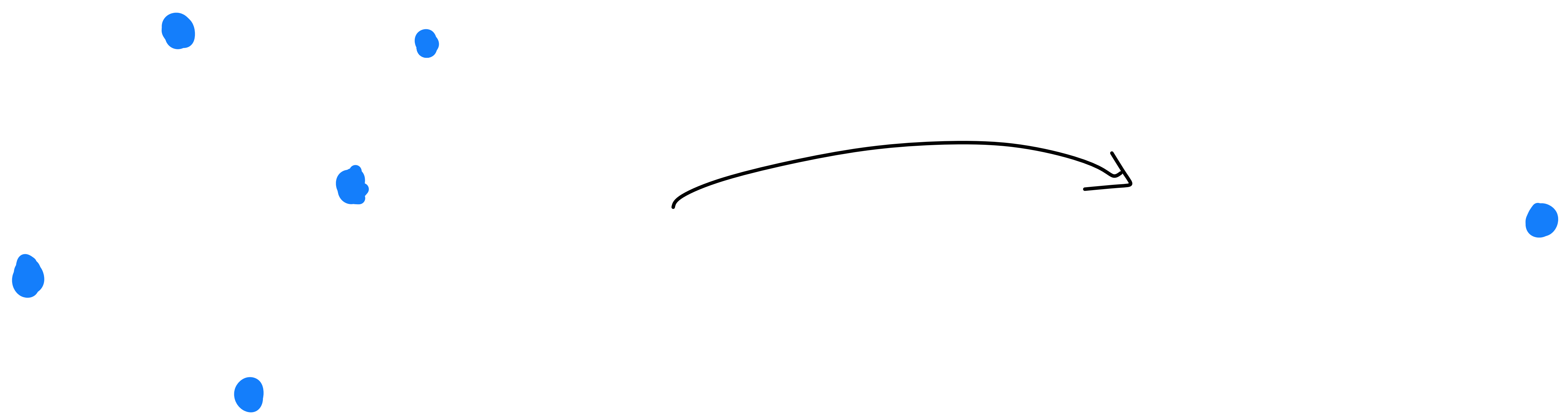
Execution Cycle

- Look
- Compute
- Move



- Fully-Synchronous
- Semi-Synchronous

The Fundamental Problem of Gathering



Related Work

Related Work

SSYNC:

Related Work

SSYNC:

Unsolvable in SSYNC without additional assumption [Suzuki & Yamashita 1999]

Related Work

SSYNC:

Unsolvable in SSYNC without additional assumption [Suzuki & Yamashita 1999]

Solvable in SSYNC when robots **agree** on a single axis [Bramas, Lamani, Tixeuil 2021]

Related Work

SSYNC:

- **Unsolvable** in SSYNC without additional assumption [Suzuki & Yamashita 1999]
- **Solvable** in SSYNC when robots **agree** on a single axis [Bramas, Lamani, Tixeuil 2021]
- **Solvable** in SSYNC when robots **disagree** on the unit distance [This work]

Related Work

SSYNC:

- **Unsolvable** in SSYNC without additional assumption [Suzuki & Yamashita 1999]
- **Solvable** in SSYNC when robots **agree** on a single axis [Bramas, Lamani, Tixeuil 2021]
- **Solvable** in SSYNC when robots **disagree** on the unit distance [This work]

FSYNC

Related Work

SSYNC:

Unsolvable in SSYNC without additional assumption [Suzuki & Yamashita 1999]

Solvable in SSYNC when robots **agree** on a single axis [Bramas, Lamani, Tixeuil 2021]

Solvable in SSYNC when robots **disagree** on the unit distance [This work]

FSYNC

Solvable in FSYNC (Move to the middle)

The Impossibility of Rendezvous

[Suzuki & Yamashita 99]

Impossibility of rendezvous in SSYNC

Impossibility of rendezvous in SSYNC

Assume an algorithm exists

Impossibility of rendezvous in SSYNC

Assume an algorithm exists

There must exists a rule in the algorithm saying: « in view V, stay idle »

Impossibility of rendezvous in SSYNC

Assume an algorithm exists

There must exists a rule in the algorithm saying: « in view V, stay idle »

Impossibility of rendezvous in SSYNC

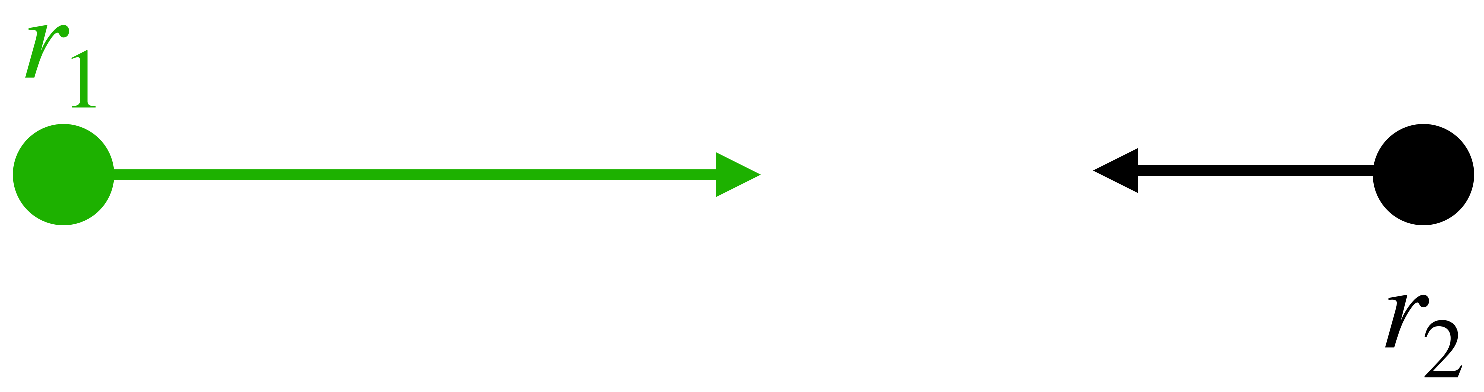
Assume an algorithm exists

There must exists a rule in the algorithm saying: « in view V, stay idle »

Both robots have view V, and stay Idle.

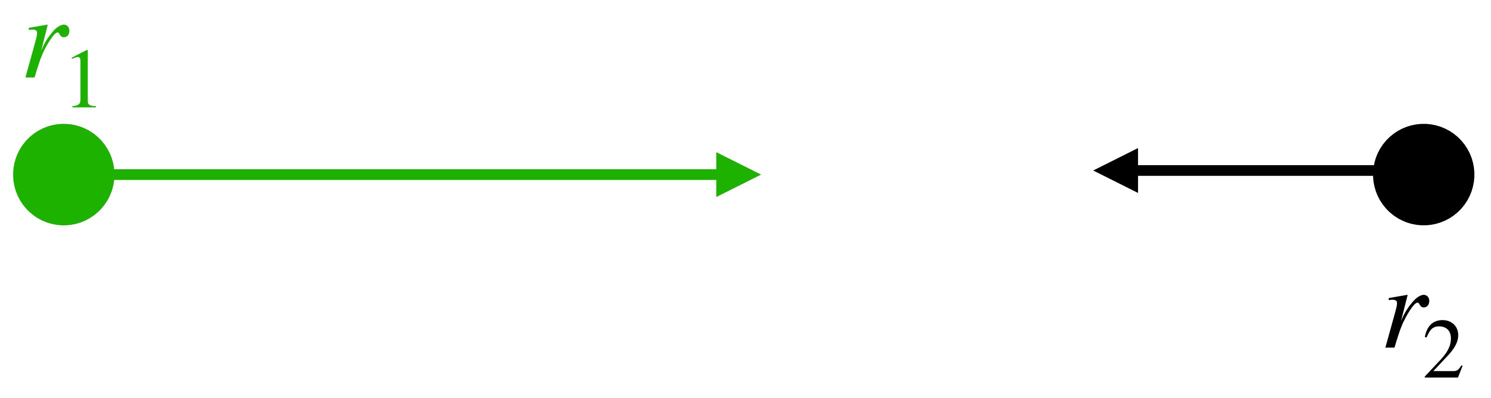
Rendezvous
Robots with different unit-distance

Notation



Notation

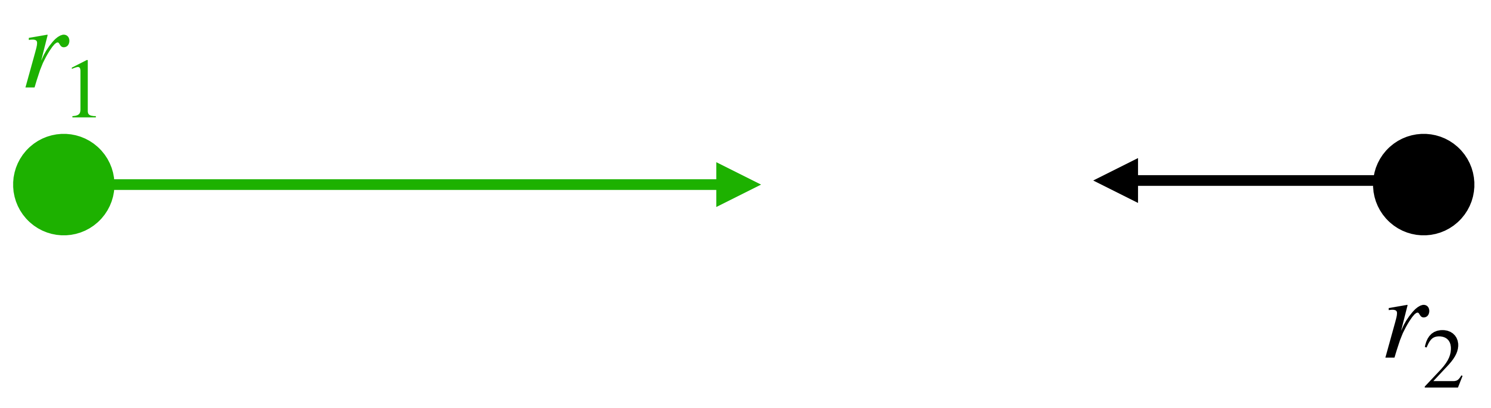
We have two robots r_1 and r_2 having unit distance $unit_1$ and $unit_2$



Notation

We have two robots r_1 and r_2 having unit distance $unit_1$ and $unit_2$

With $\dfrac{unit_1}{unit_2} = \rho > 1$ (indeed, with $\rho = 1$ the problem is unsolvable)

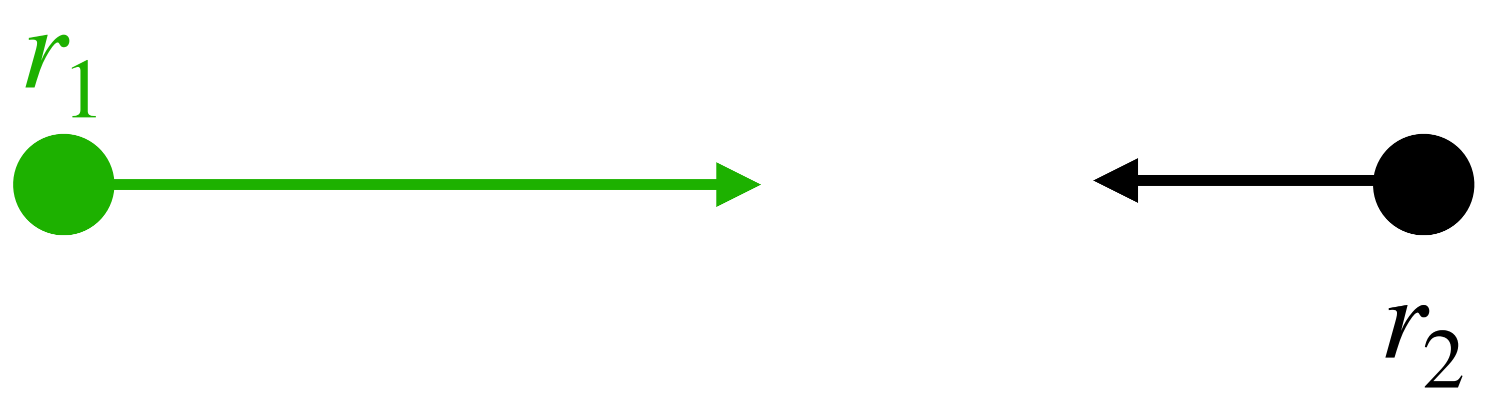


Notation

We have two robots r_1 and r_2 having unit distance $unit_1$ and $unit_2$

With $\frac{unit_1}{unit_2} = \rho > 1$ (indeed, with $\rho = 1$ the problem is unsolvable)

This implies that robots see the distance between them differently



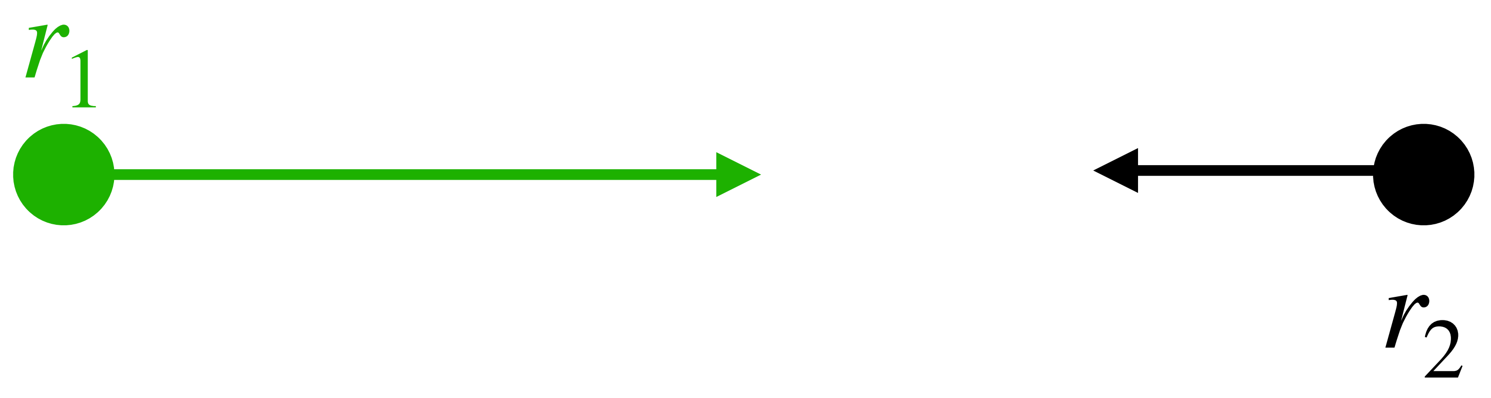
Notation

We have two robots r_1 and r_2 having unit distance $unit_1$ and $unit_2$

With $\dfrac{unit_1}{unit_2} = \rho > 1$ (indeed, with $\rho = 1$ the problem is unsolvable)

This implies that robots see the distance between them differently

r_1 sees distance d_1 and r_2 sees distance d_2 with $d_2 = \rho d_1$



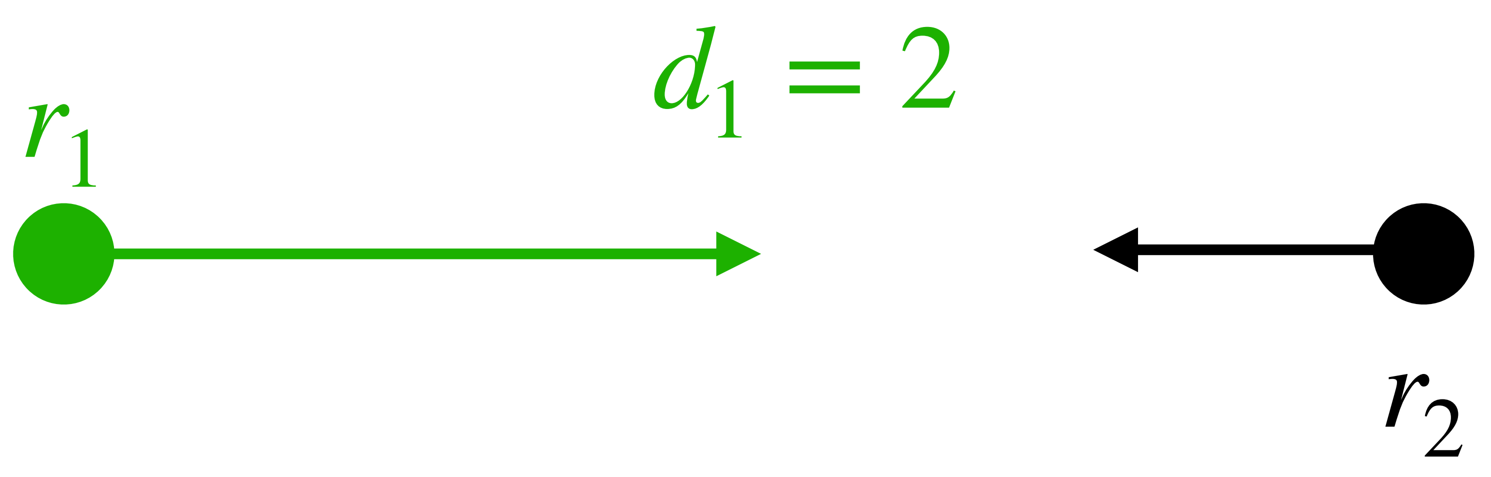
Notation

We have two robots r_1 and r_2 having unit distance $unit_1$ and $unit_2$

With $\dfrac{unit_1}{unit_2} = \rho > 1$ (indeed, with $\rho = 1$ the problem is unsolvable)

This implies that robots see the distance between them differently

r_1 sees distance d_1 and r_2 sees distance d_2 with $d_2 = \rho d_1$



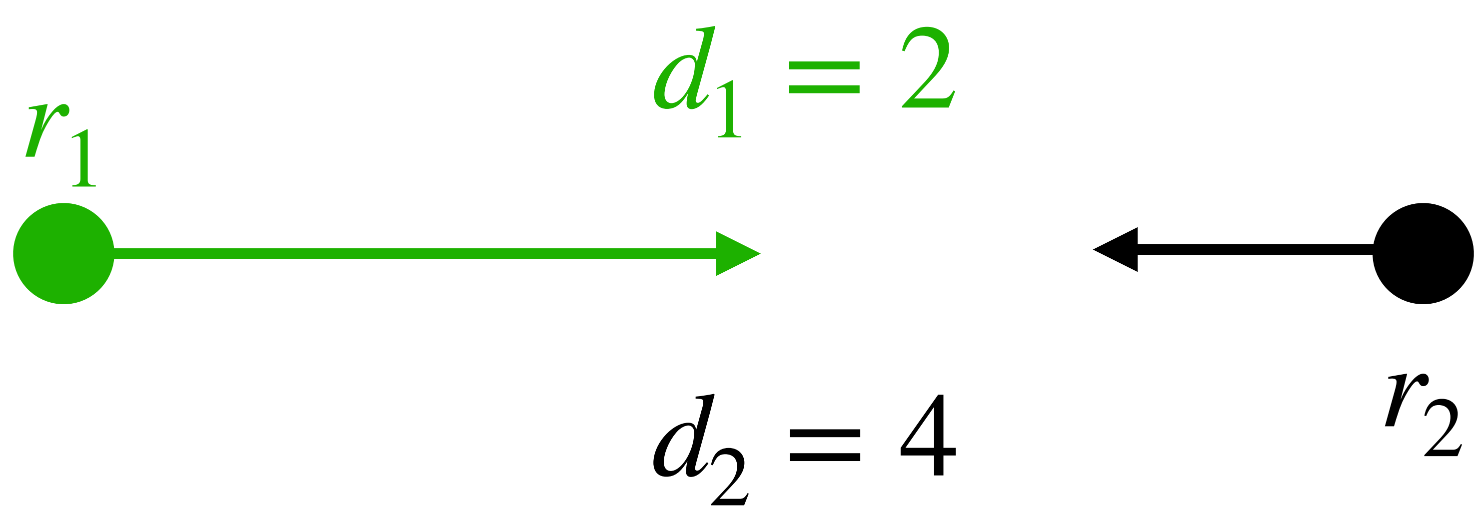
Notation

We have two robots r_1 and r_2 having unit distance $unit_1$ and $unit_2$

With $\dfrac{unit_1}{unit_2} = \rho > 1$ (indeed, with $\rho = 1$ the problem is unsolvable)

This implies that robots see the distance between them differently

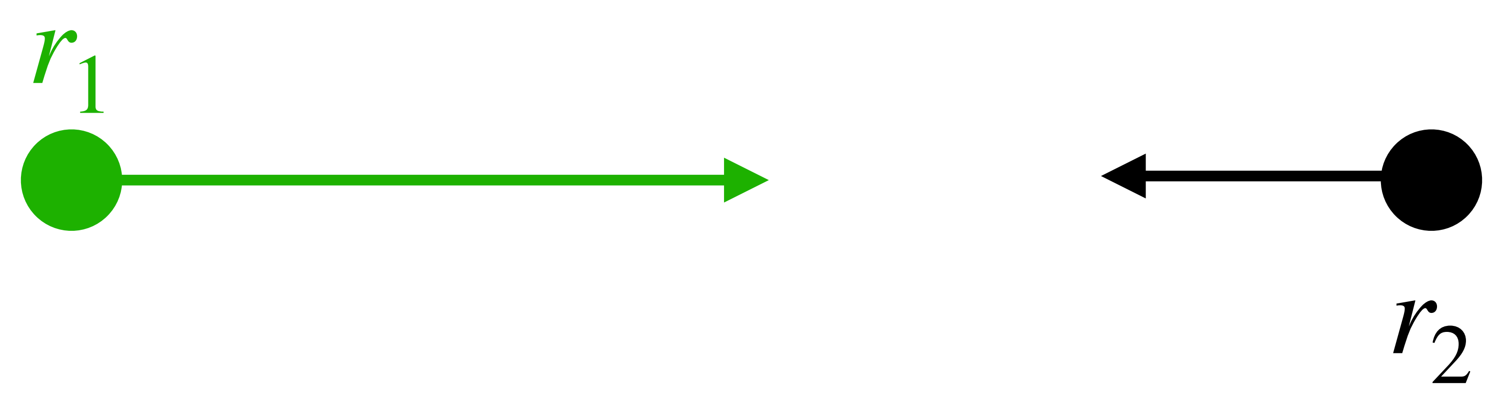
r_1 sees distance d_1 and r_2 sees distance d_2 with $d_2 = \rho d_1$



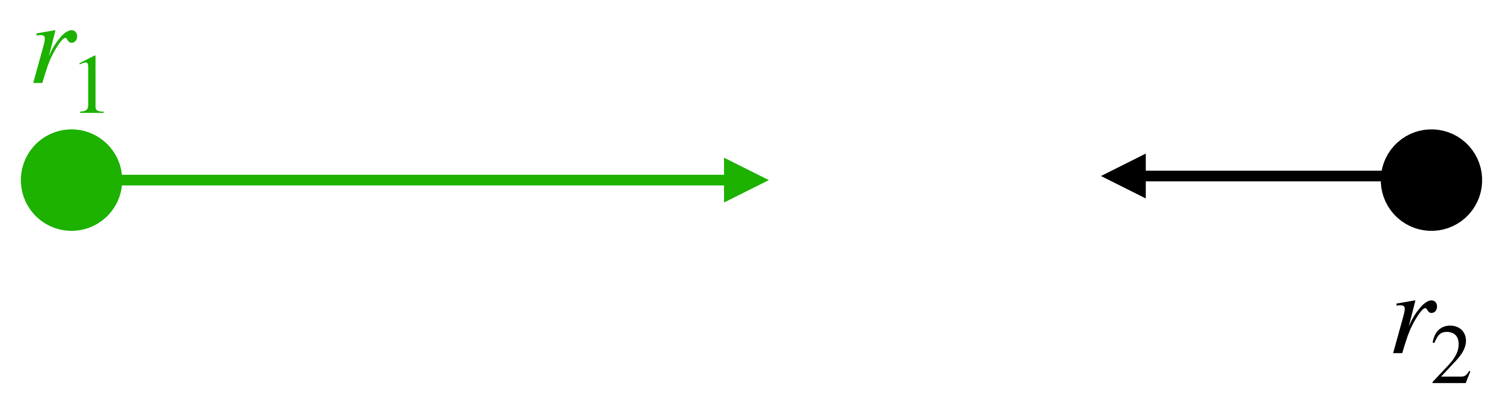
Rendezvous

$$\rho = 2$$

Rendezvous when $\rho = 2$



Rendezvous when $\rho = 2$



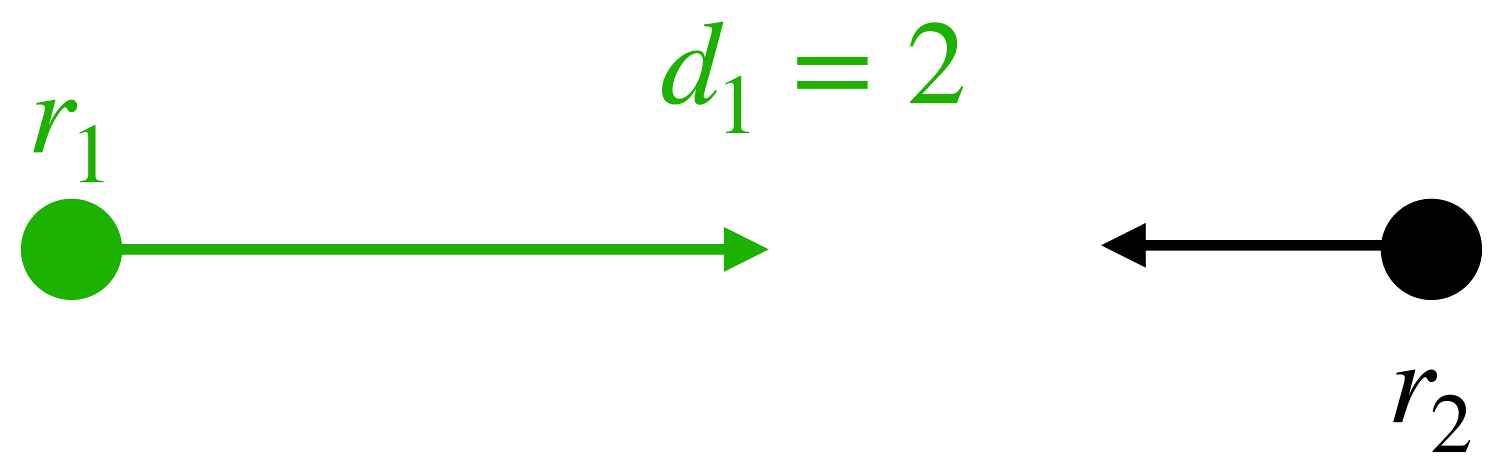
Definition:

We define ***the level of robot*** r_i

the unique number l_i

such that $d_i \in [2^{-l_i}, 2^{-l_i+1})$

Rendezvous when $\rho = 2$



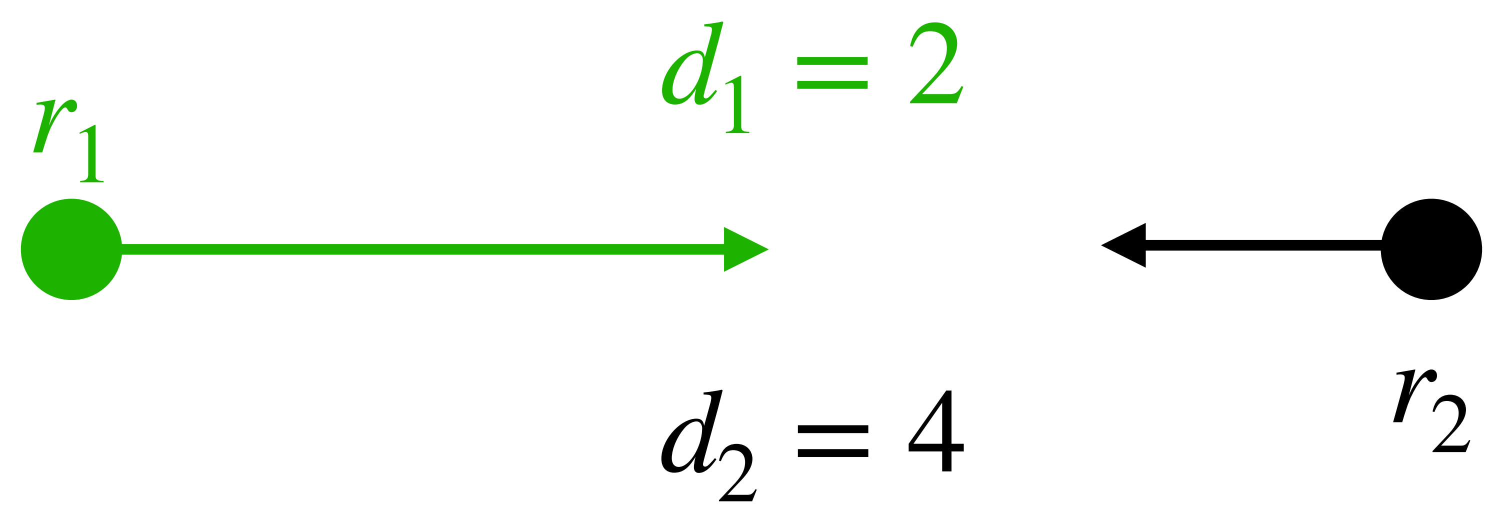
Definition:

We define ***the level of robot*** r_i

the unique number l_i

such that $d_i \in [2^{-l_i}, 2^{-l_i+1})$

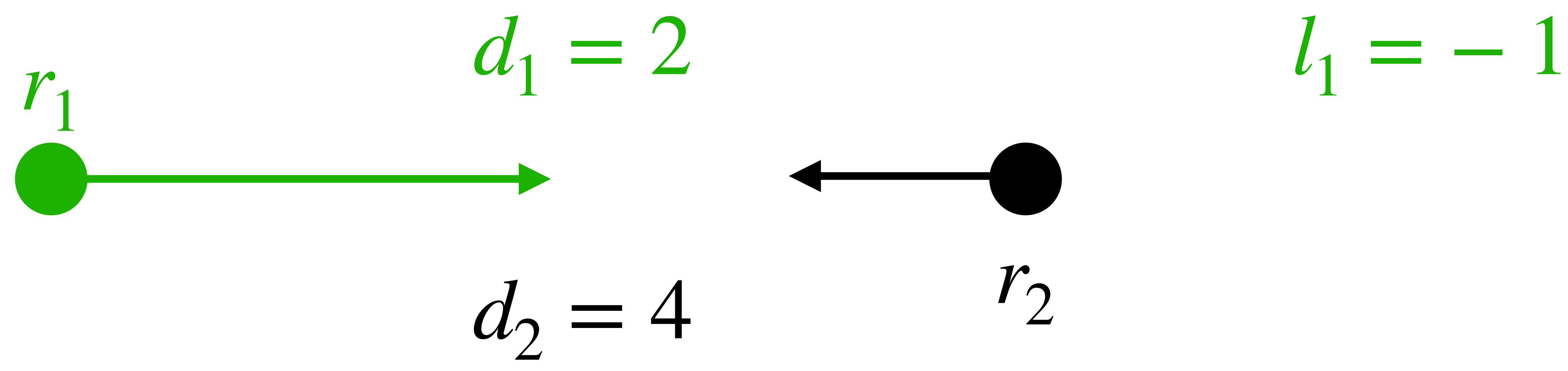
Rendezvous when $\rho = 2$



Definition:

We define ***the level of robot*** r_i
the unique number l_i
such that $d_i \in [2^{-l_i}, 2^{-l_i+1})$

Rendezvous when $\rho = 2$



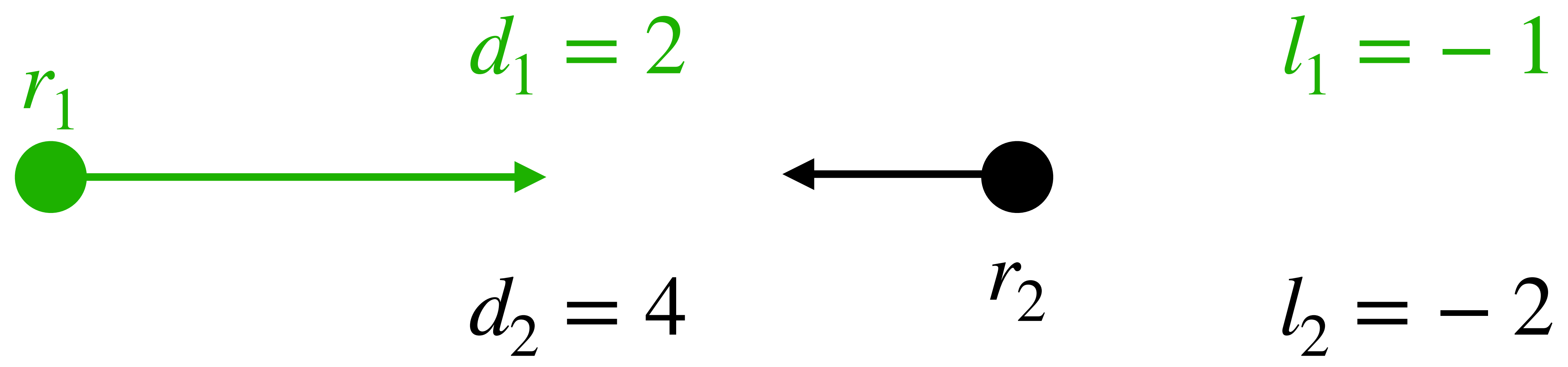
Definition:

We define ***the level of robot*** r_i

the unique number l_i

such that $d_i \in [2^{-l_i}, 2^{-l_i+1})$

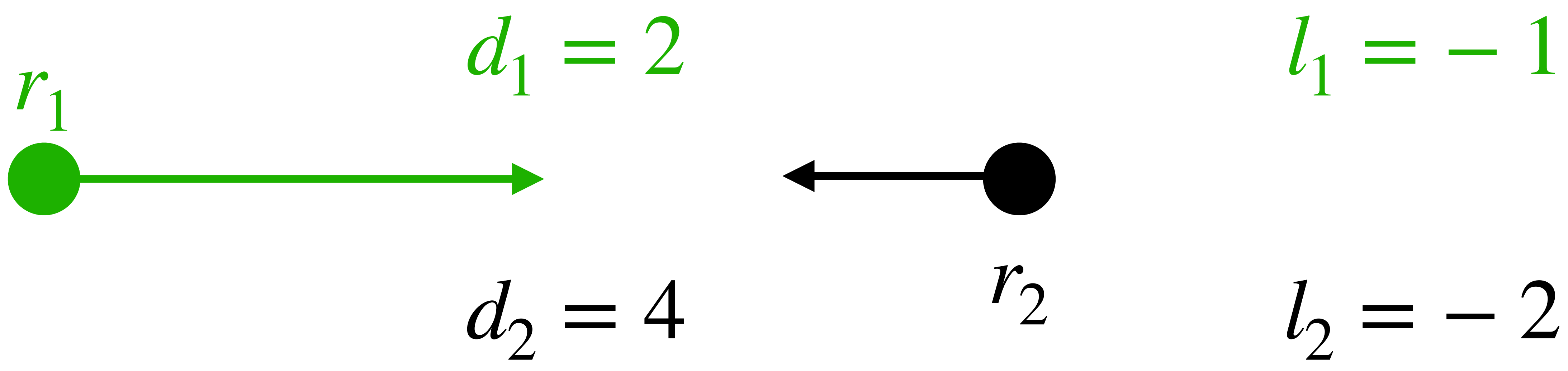
Rendezvous when $\rho = 2$



Definition:

We define ***the level of robot*** r_i
the unique number l_i
such that $d_i \in [2^{-l_i}, 2^{-l_i+1})$

Rendezvous when $\rho = 2$



Definition:

We define *the level of robot* r_i
the unique number l_i
such that $d_i \in [2^{-l_i}, 2^{-l_i+1})$

Property:

We have $l_1 = l_2 + 1$

Rendezvous when $\rho = 2$

Definition:

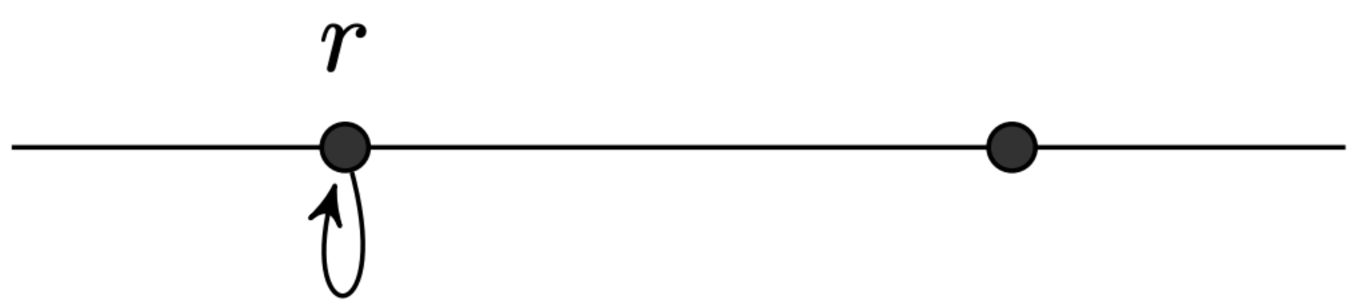
We define *the level of robot* r_i
the unique number l_i
such that $d_i \in [2^{-l_i}, 2^{-l_i+1})$

Property:

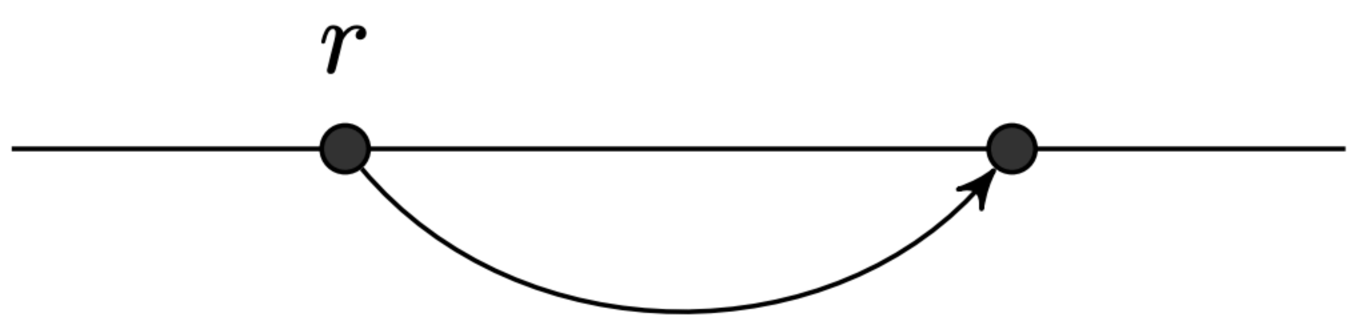
We have $l_1 = l_2 + 1$

Algo1:

$\text{case } l_r \equiv 0 \mod 2$



$\text{case } l_r \equiv 1 \mod 2$

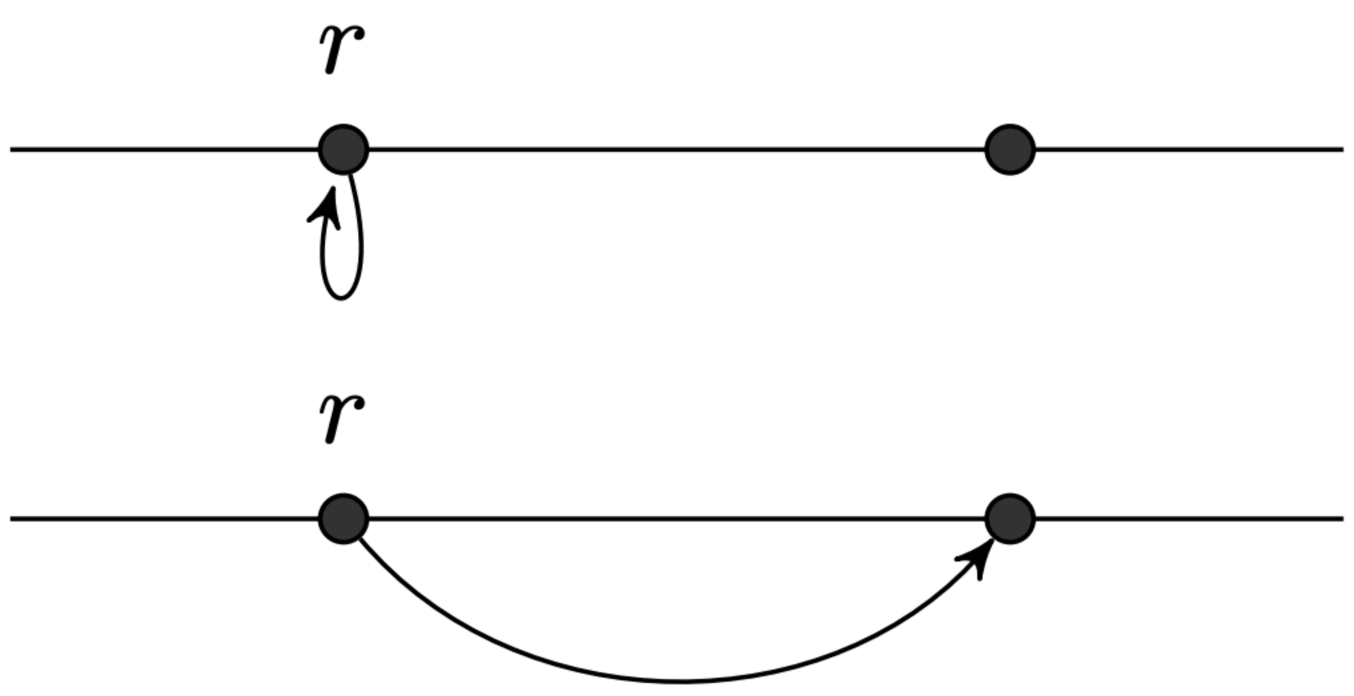


Rendezvous when $\rho = 2$

Algo1:

case $l_r \equiv 0 \mod 2$

case $l_r \equiv 1 \mod 2$



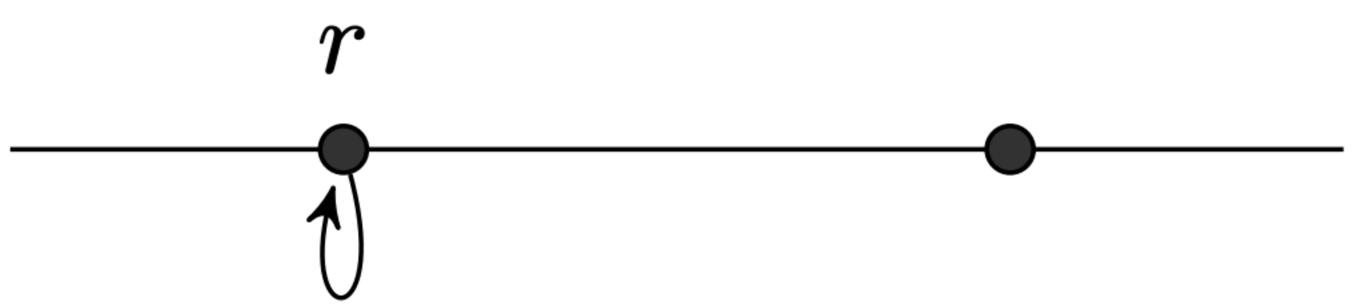
Theorem:

Algo 1 solves rendezvous in SSYNC

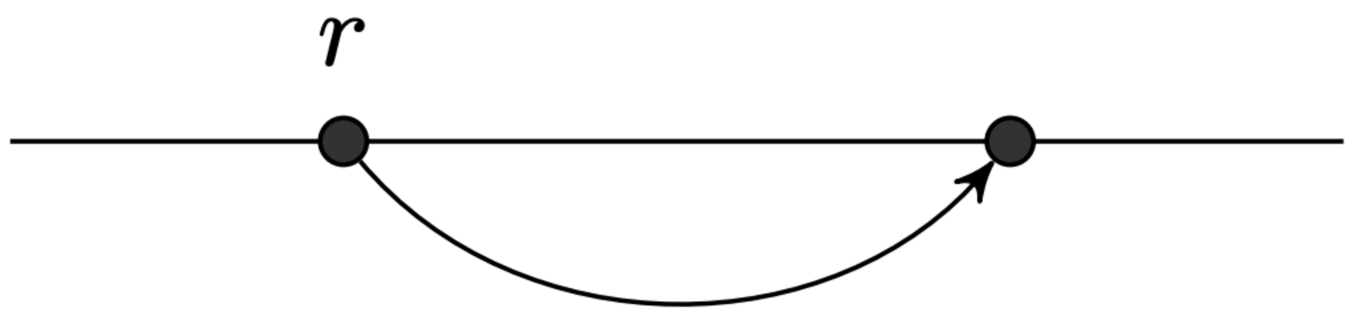
Rendezvous when $\rho = 2$

Algo1:

case $l_r \equiv 0 \mod 2$



case $l_r \equiv 1 \mod 2$



Theorem:

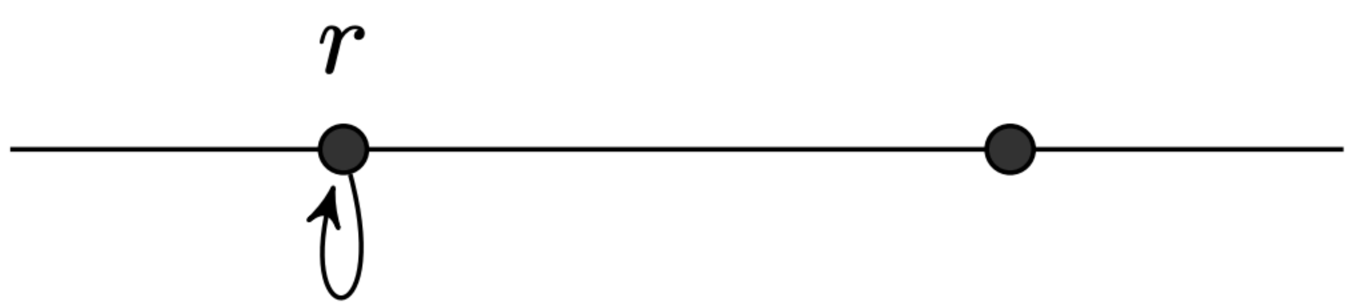
Algo 1 solves rendezvous in SSYNC

Proof:

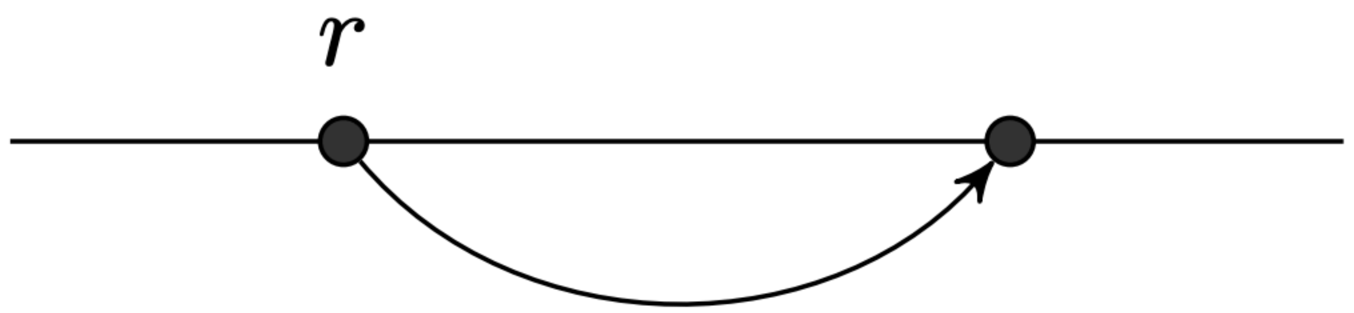
Rendezvous when $\rho = 2$

Algo1:

$\text{case } l_r \equiv 0 \mod 2$



$\text{case } l_r \equiv 1 \mod 2$



Theorem:

Algo 1 solves rendezvous in SSYNC

Proof:

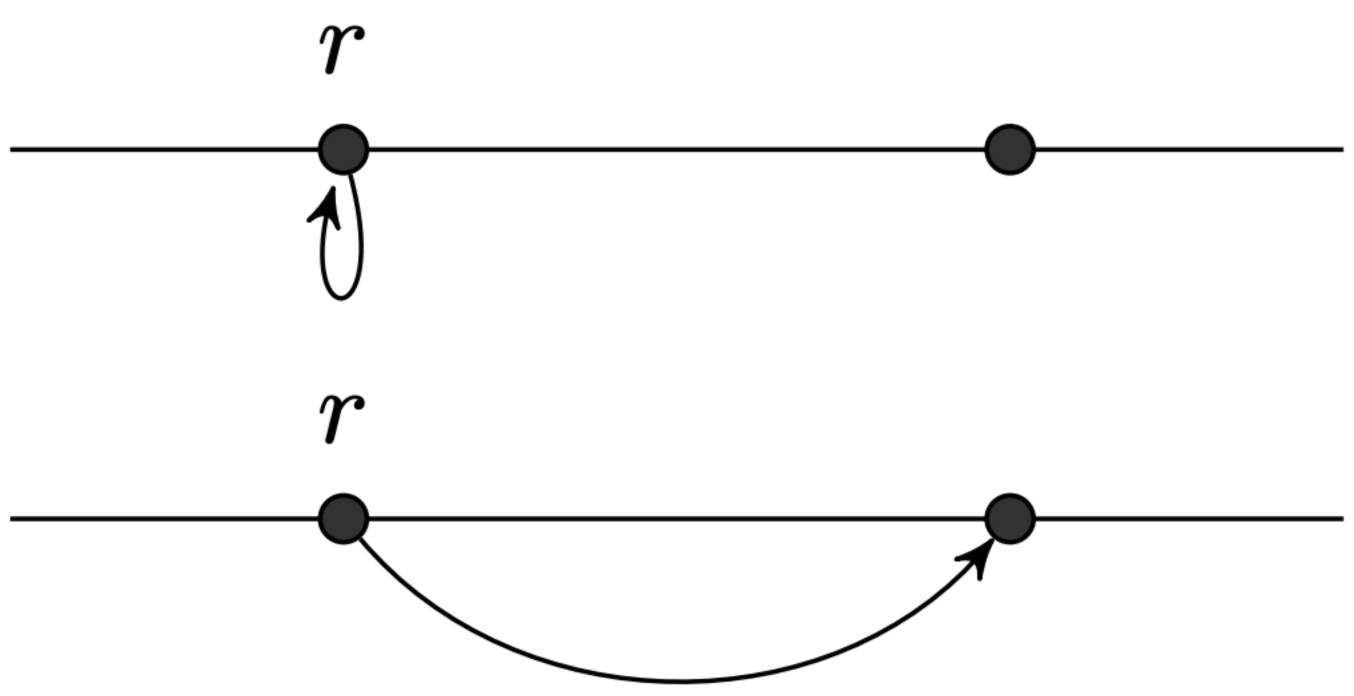
One robot moves to the other robot.

Rendezvous when $\rho = 2$

Algo1:

$$\text{case } l_r \equiv 0 \mod 2$$

$$\text{case } l_r \equiv 1 \mod 2$$



Theorem:

Algo 1 solves rendezvous in SSYNC

Proof:

One robot moves to the other robot.

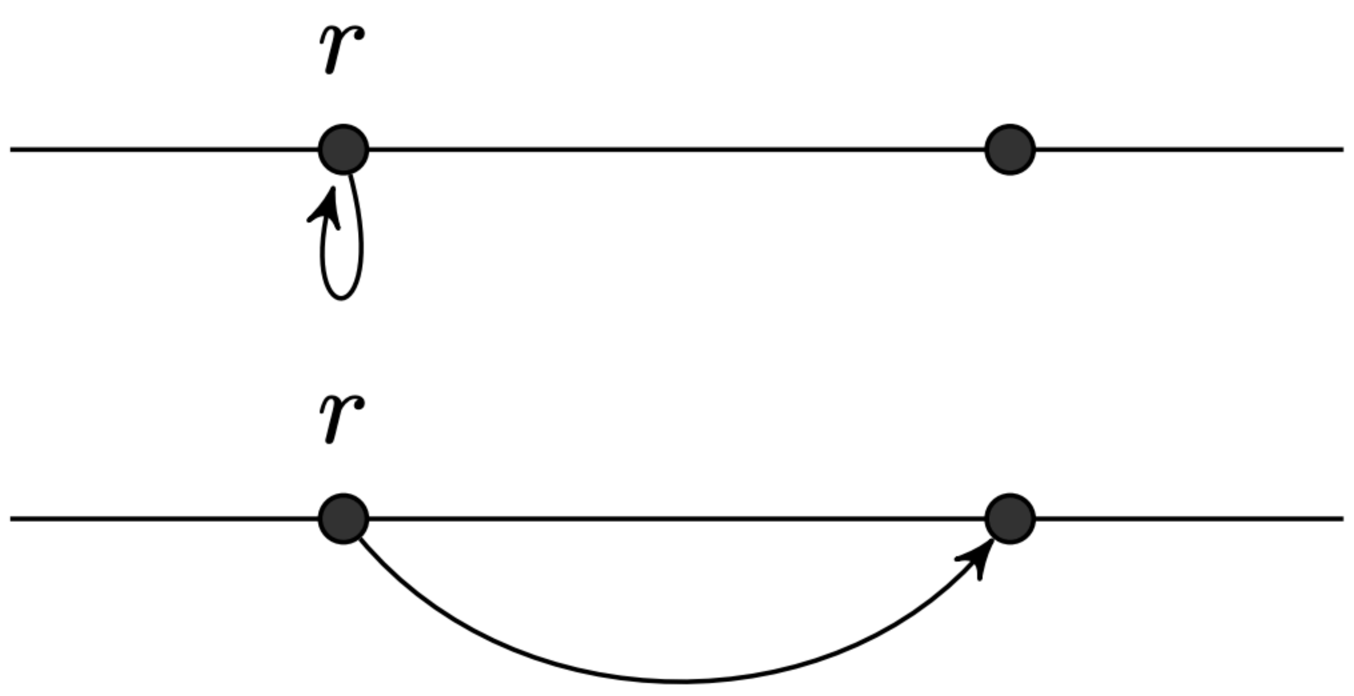
If it does not reach its target, then the distance between the robots decreases by a fixed amount.

Rendezvous when $\rho = 2$

Algo1:

$$\text{case } l_r \equiv 0 \mod 2$$

$$\text{case } l_r \equiv 1 \mod 2$$



Theorem:

Algo 1 solves rendezvous in SSYNC

Proof:

One robot moves to the other robot.

If it does not reach its target, then the distance between the robots decreases by a fixed amount.

Eventually, the distance between the robots is smaller than δ, hence robots eventually reach their target.

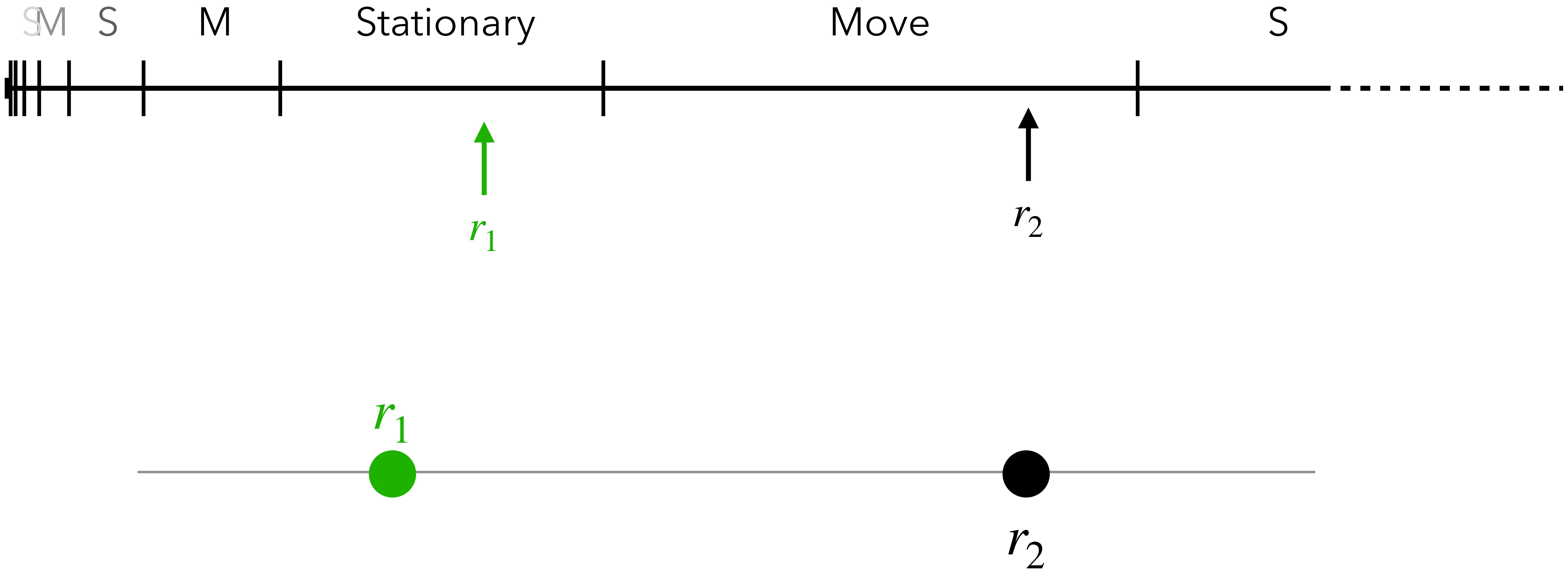
Rendezvous

$$\rho = [\rho_{min}, \rho_{\max}]$$

Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Understanding Levels

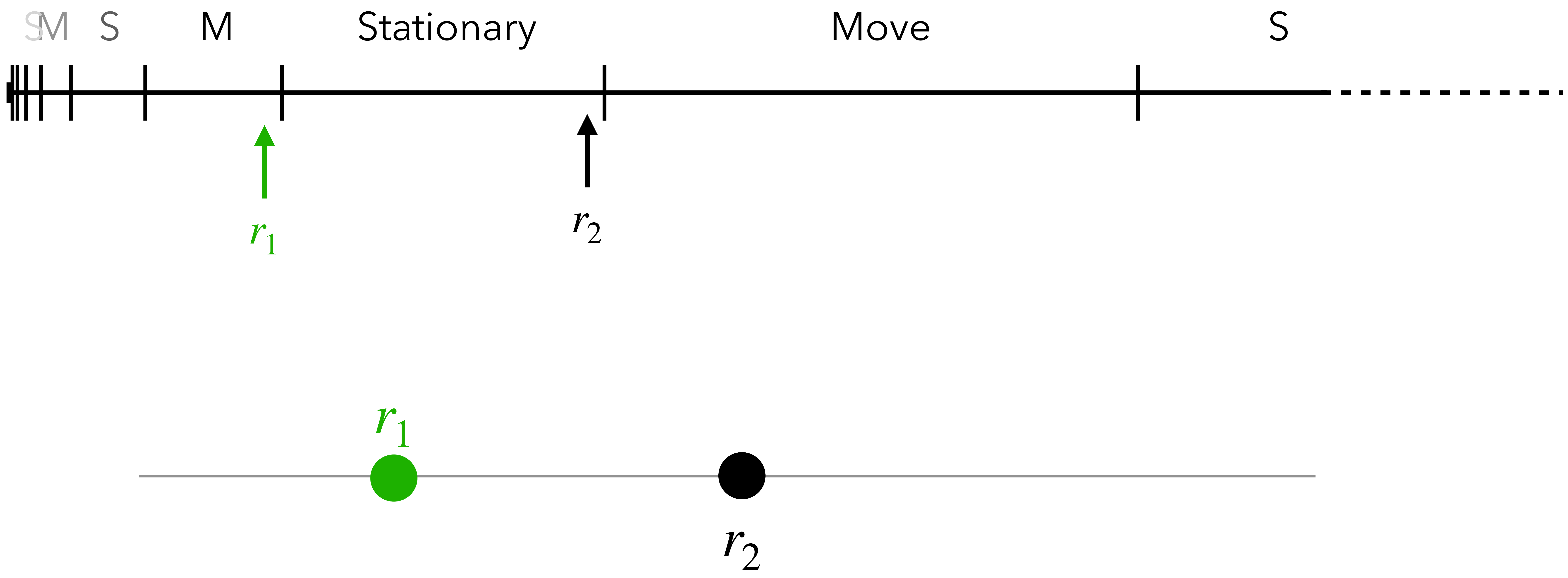
When $\rho = 2$



Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Understanding Levels

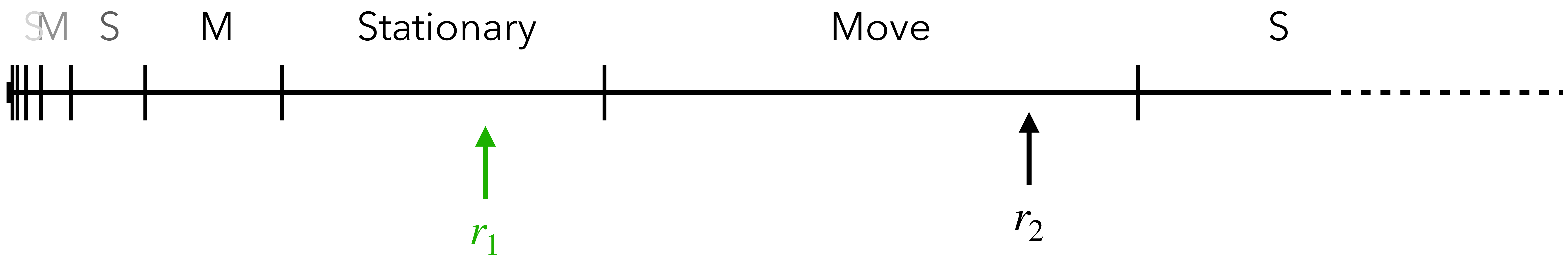
When $\rho = 2$



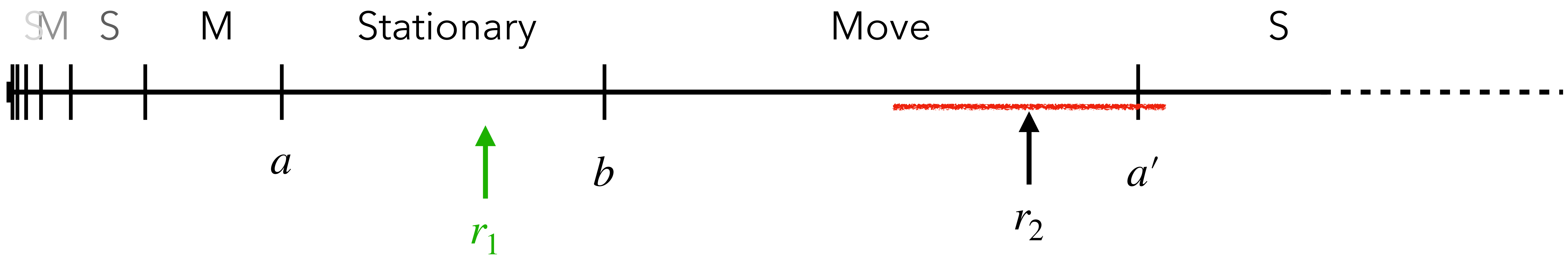
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Understanding Levels

When $\rho = 2$



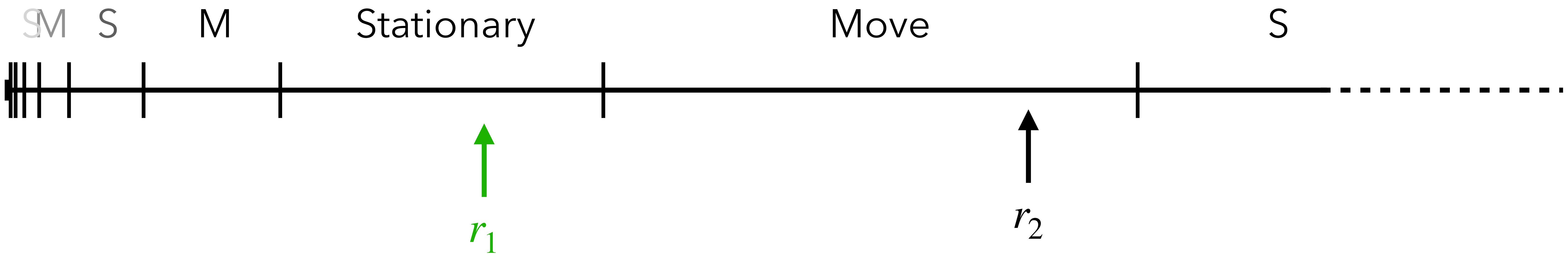
When $\rho = [\rho_{min}, \rho_{\max}]$



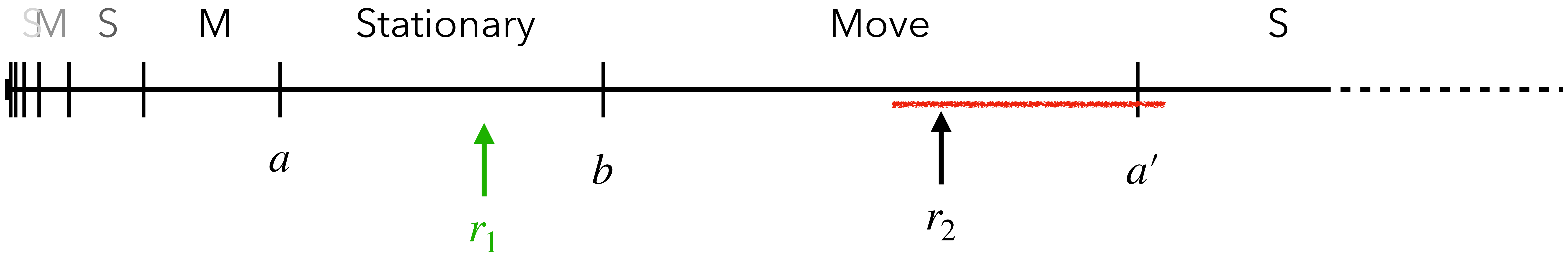
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Understanding Levels

When $\rho = 2$



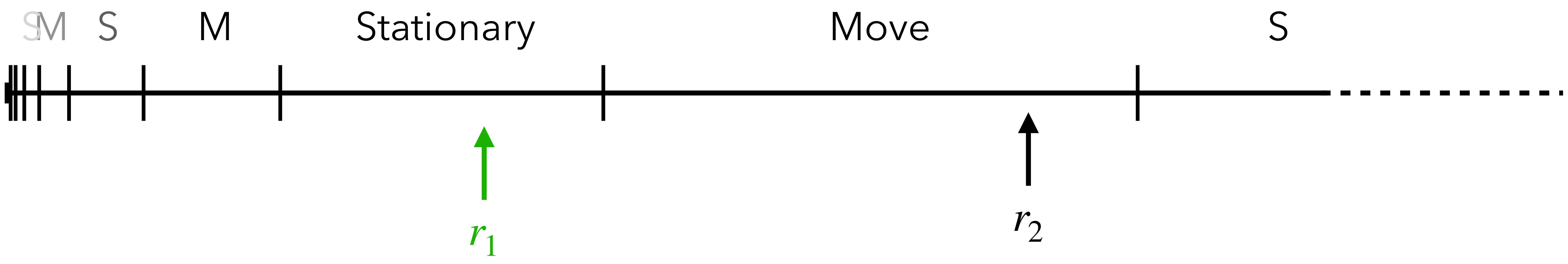
When $\rho = [\rho_{min}, \rho_{\max}]$



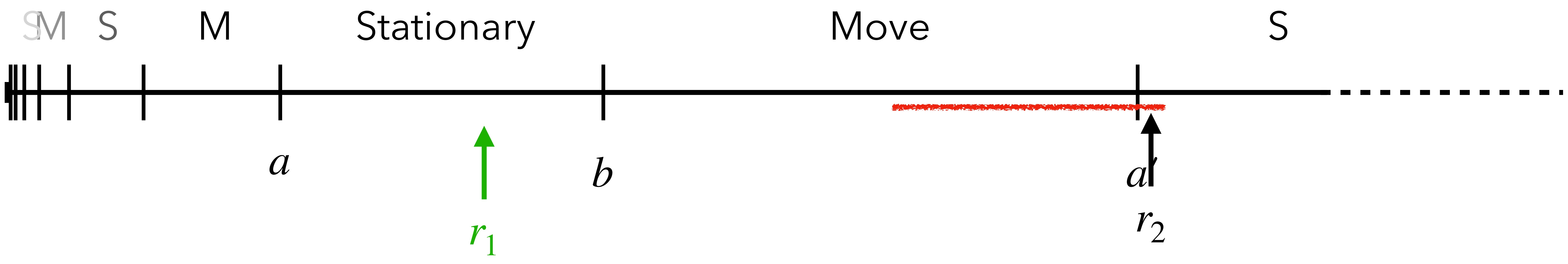
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Understanding Levels

When $\rho = 2$



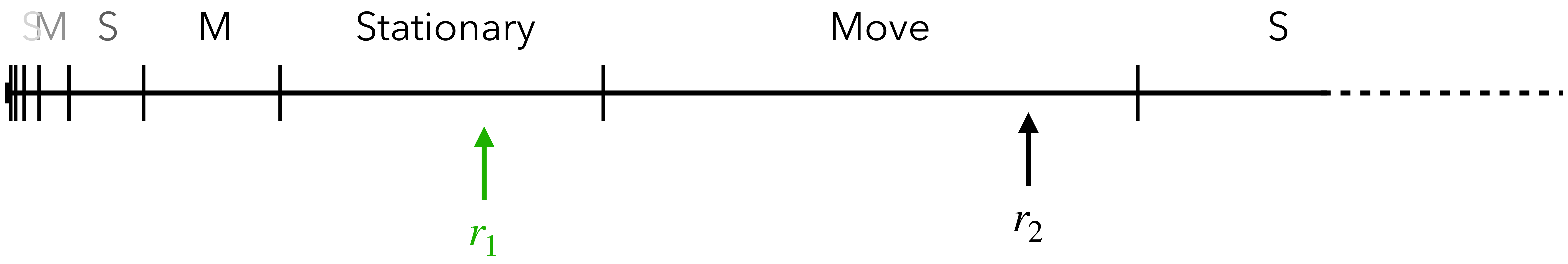
When $\rho = [\rho_{min}, \rho_{\max}]$



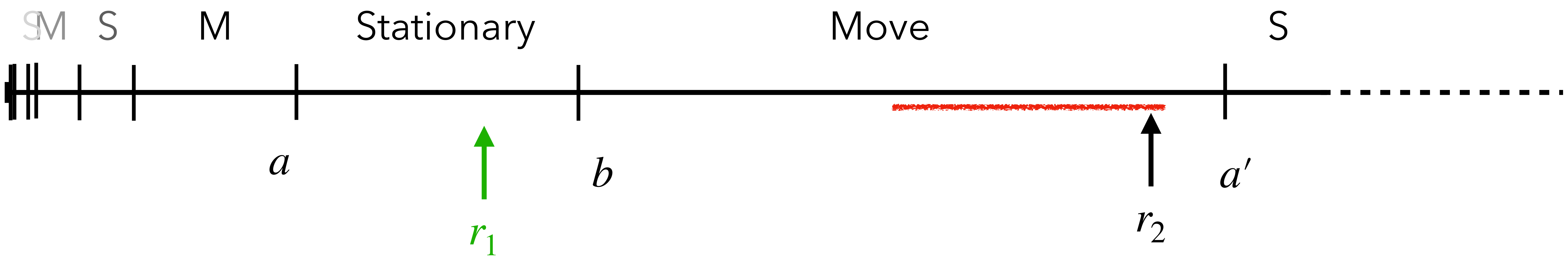
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Understanding Levels

When $\rho = 2$

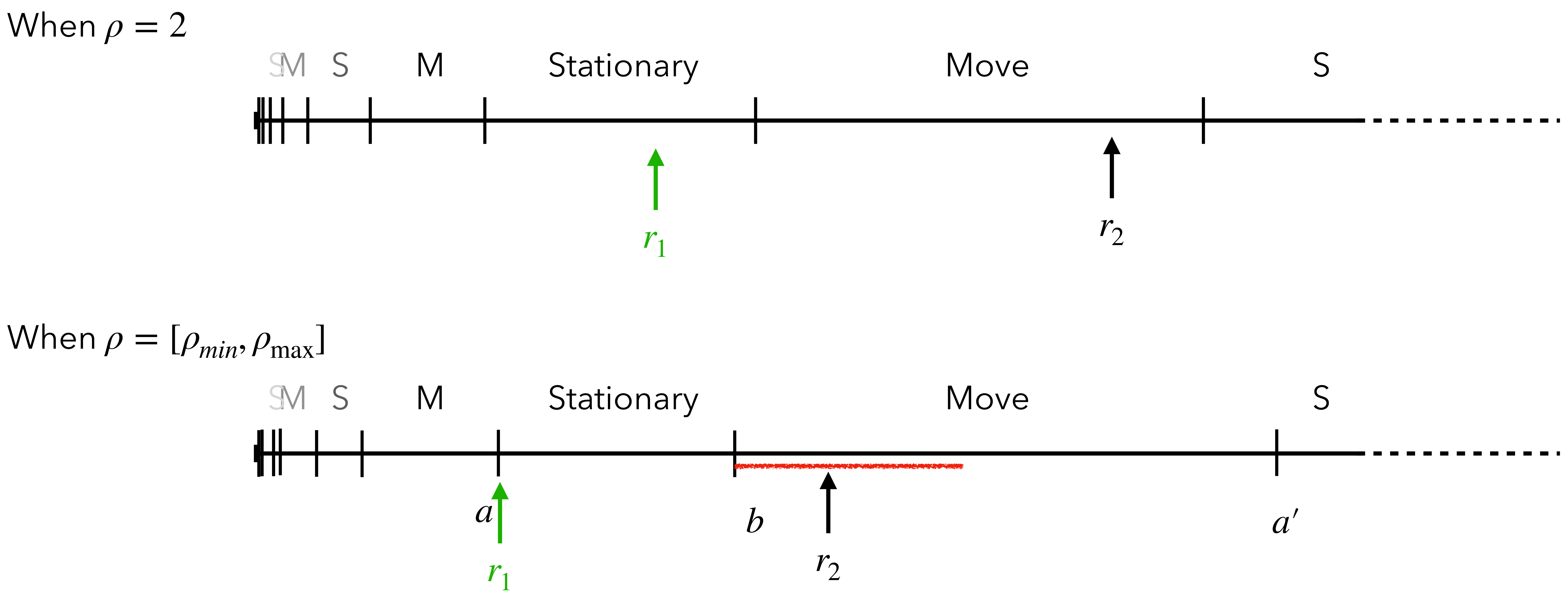


When $\rho = [\rho_{min}, \rho_{\max}]$



Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

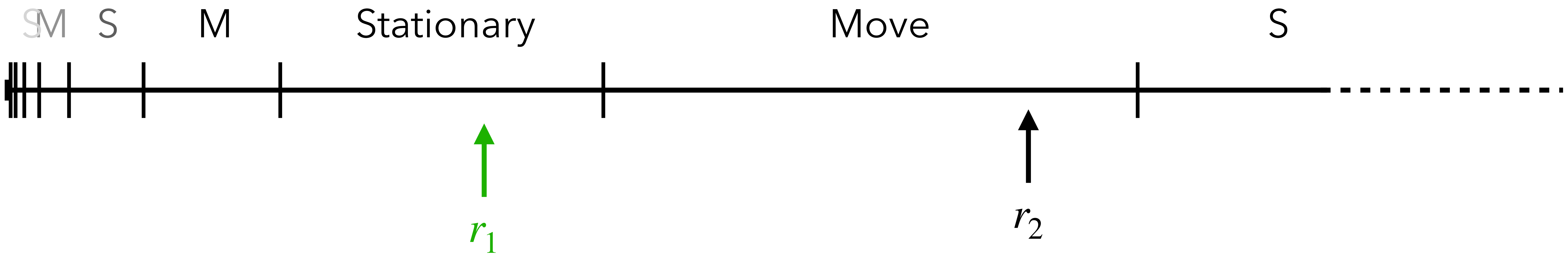
Understanding Levels



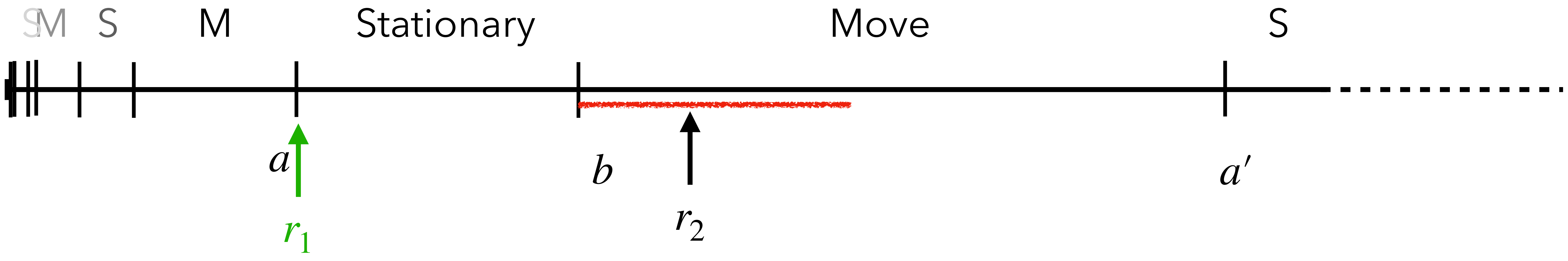
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Understanding Levels

When $\rho = 2$



When $\rho = [\rho_{min}, \rho_{\max}]$



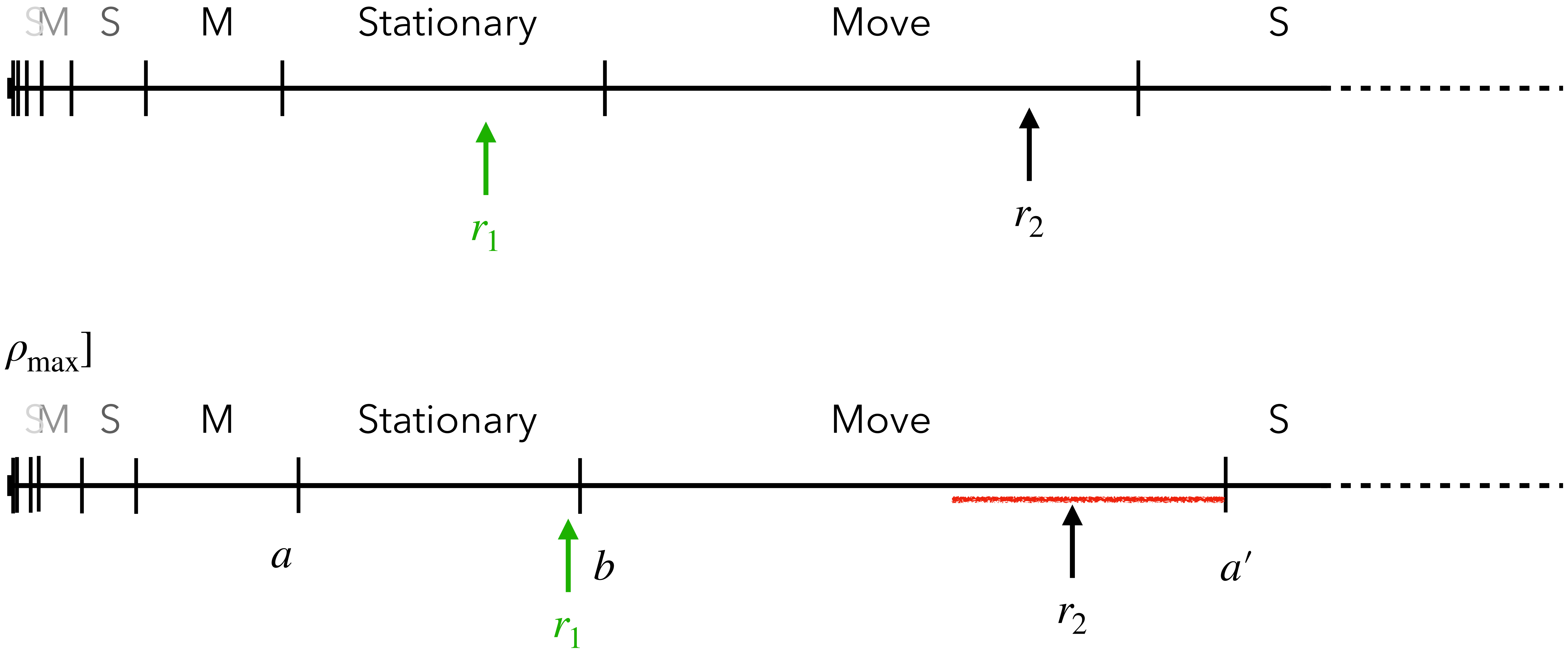
$$b = a \times \rho_{\min}$$

Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Understanding Levels

When $\rho = 2$

When $\rho = [\rho_{min}, \rho_{\max}]$

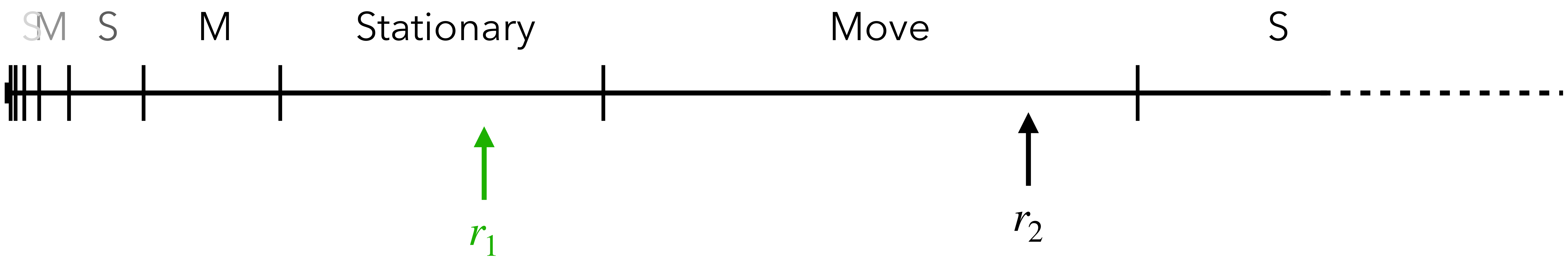


$b = a \times \rho_{\min}$

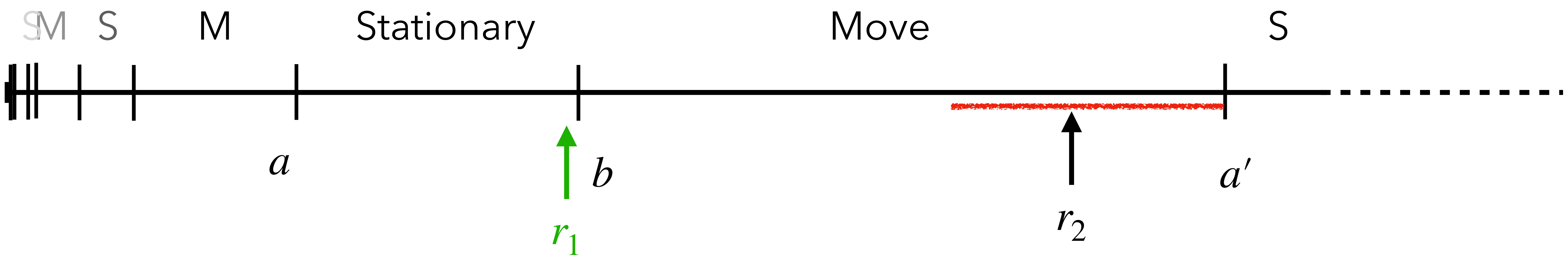
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Understanding Levels

When $\rho = 2$



When $\rho = [\rho_{min}, \rho_{\max}]$



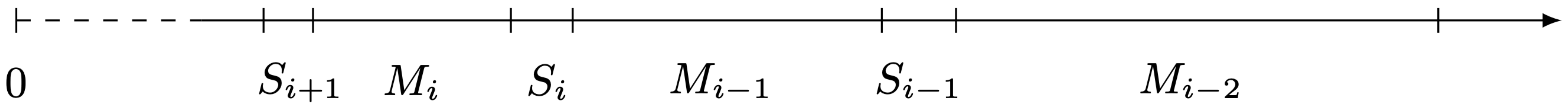
$b = a \times \rho_{\min}$ $a' = b \times \rho_{\max}$

Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Robots Levels

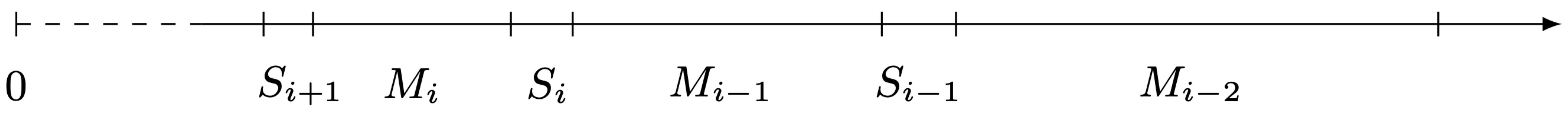
$$\forall i \in \mathbb{Z} \qquad S_i = [\rho_{\min}^{-i}\rho_{\max}^{-i}, \rho_{\min}^{-(i-1)}\rho_{\max}^{-i})$$

$$M_i = [\rho_{\min}^{-i}\rho_{\max}^{-(i+1)}, \rho_{\min}^{-i}\rho_{\max}^{-i})$$



Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

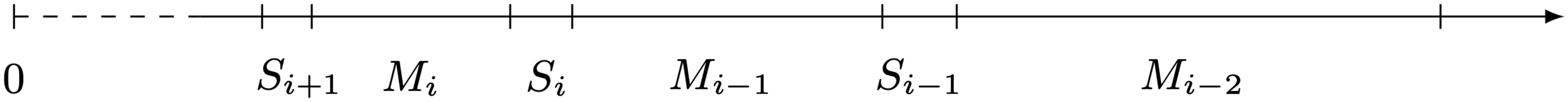
Robots Levels



Lemma 1: if a robot is in S_i then the other robot is in M_i or in M_{i-1}

Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Robots Levels



Lemma 1: if a robot is in S_i then the other robot is in M_i or in M_{i-1}

Algo2:

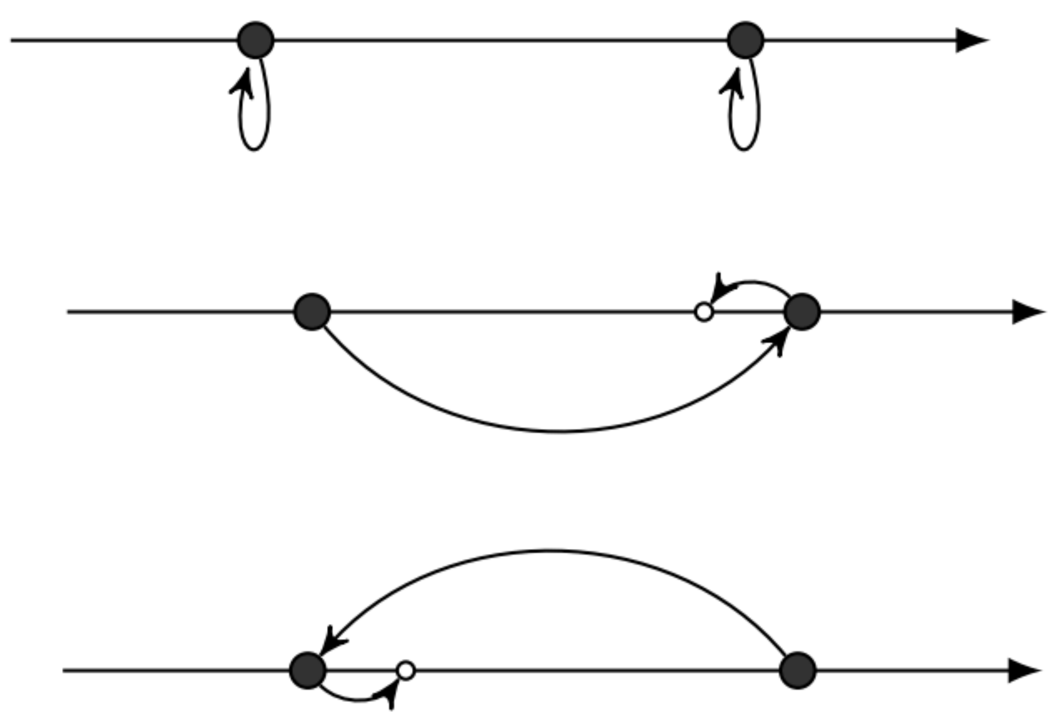
$d \in \mathcal{S}_0 \cup \mathcal{S}_1$

$d \in \mathcal{M}_0$

The right robot moves a distance $\mathfrak{s}(d)$

$d \in M_1$

The left robot moves a distance $\mathfrak{s}(d)$



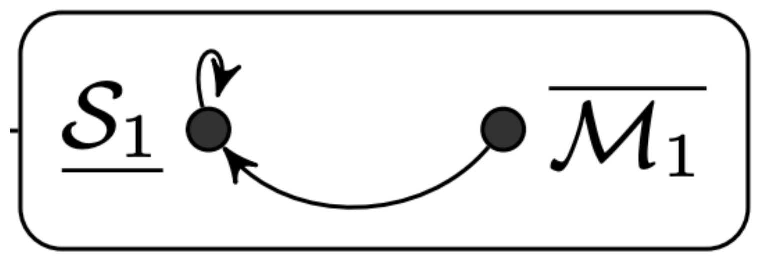
With $\mathfrak{s}(d) \in \mathcal{S}_0$

Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution

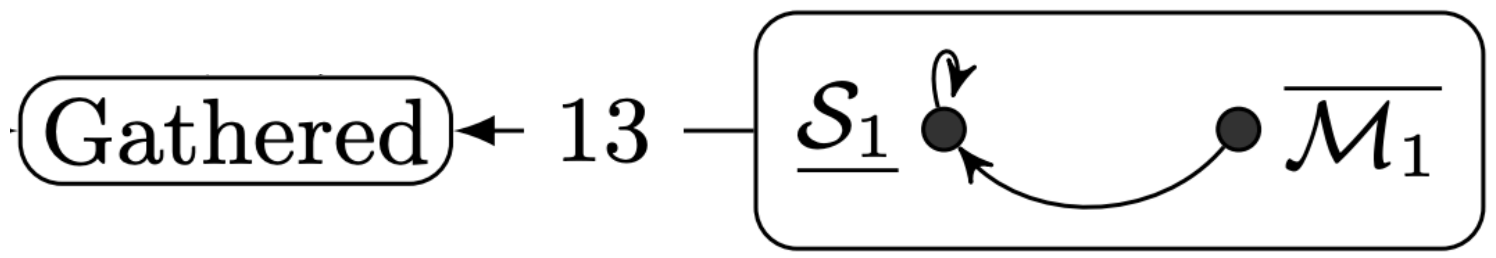
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution



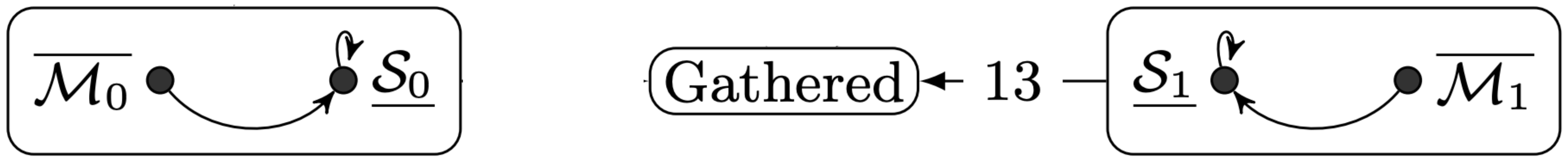
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution



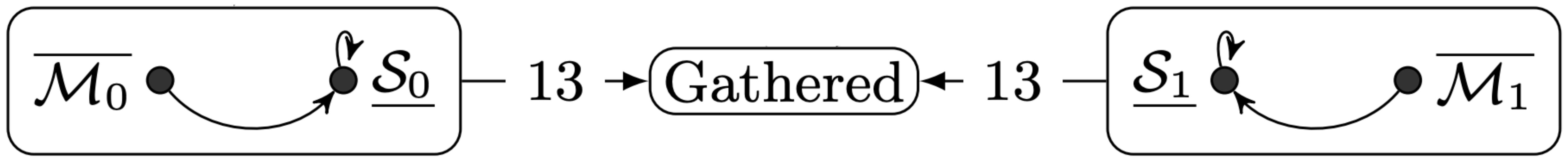
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution



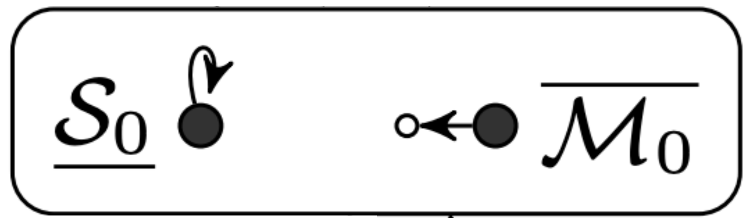
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

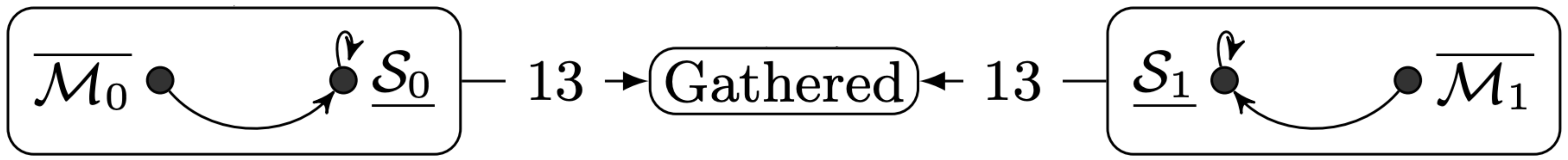
Algo2 execution



Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

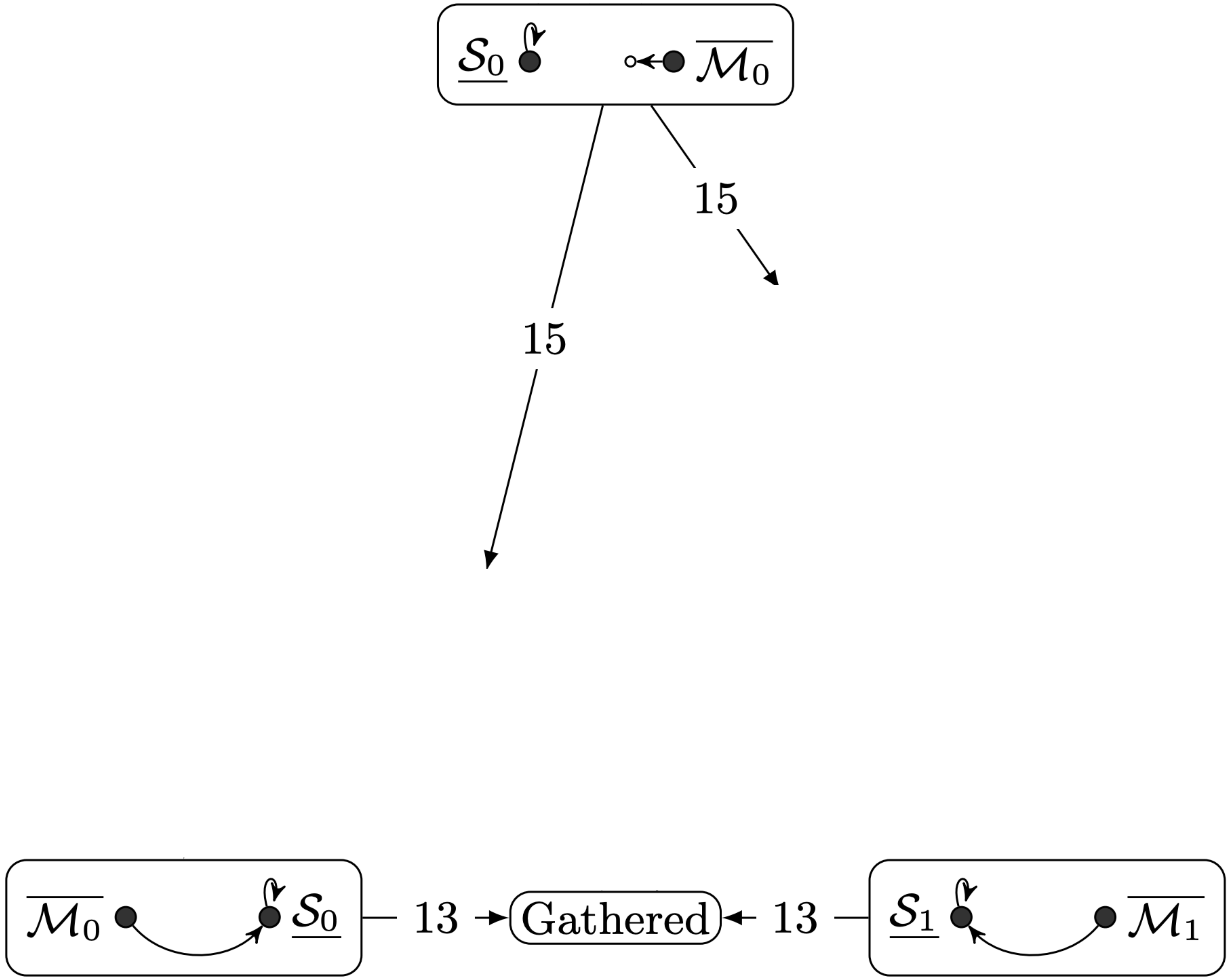
Algo2 execution





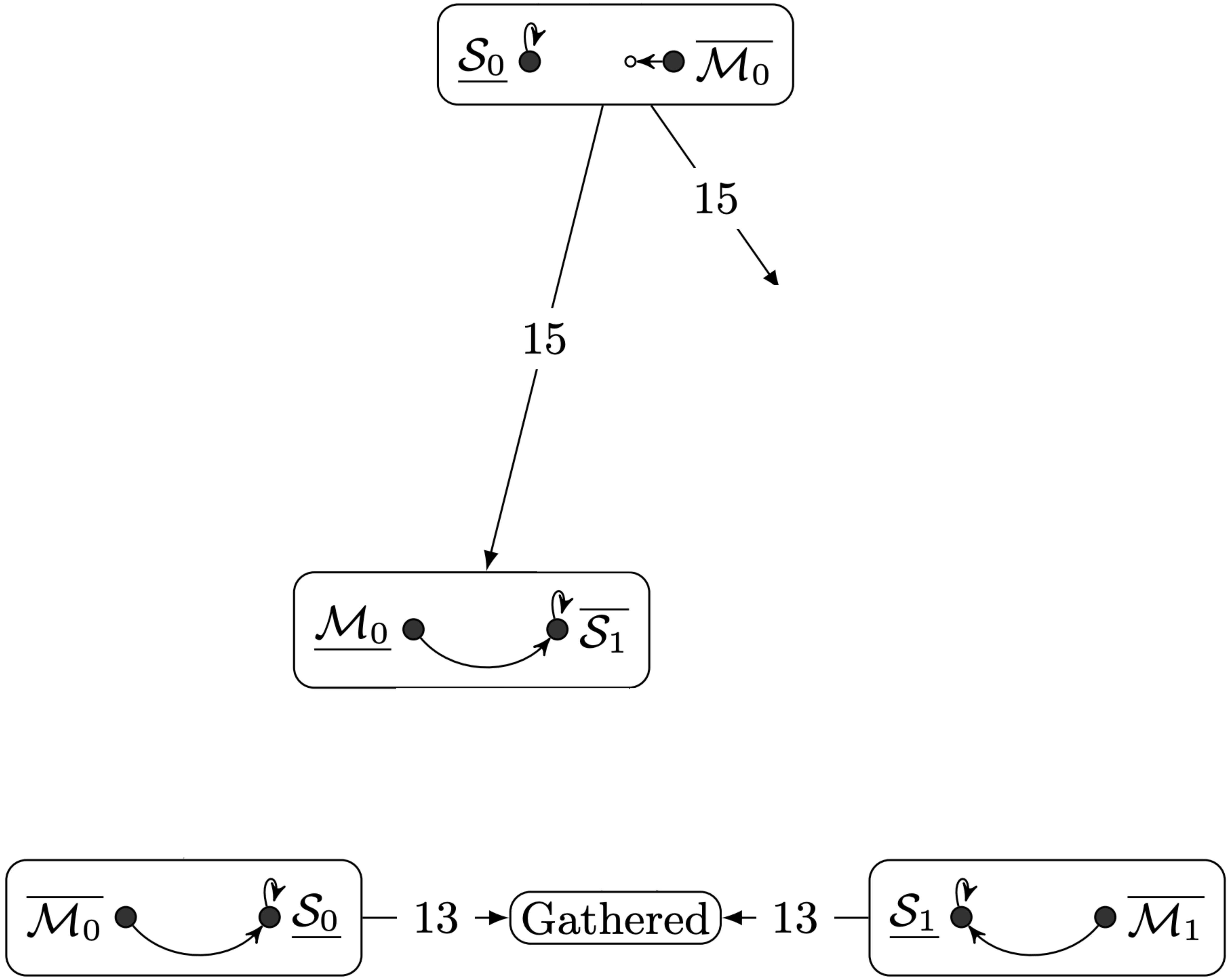
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution



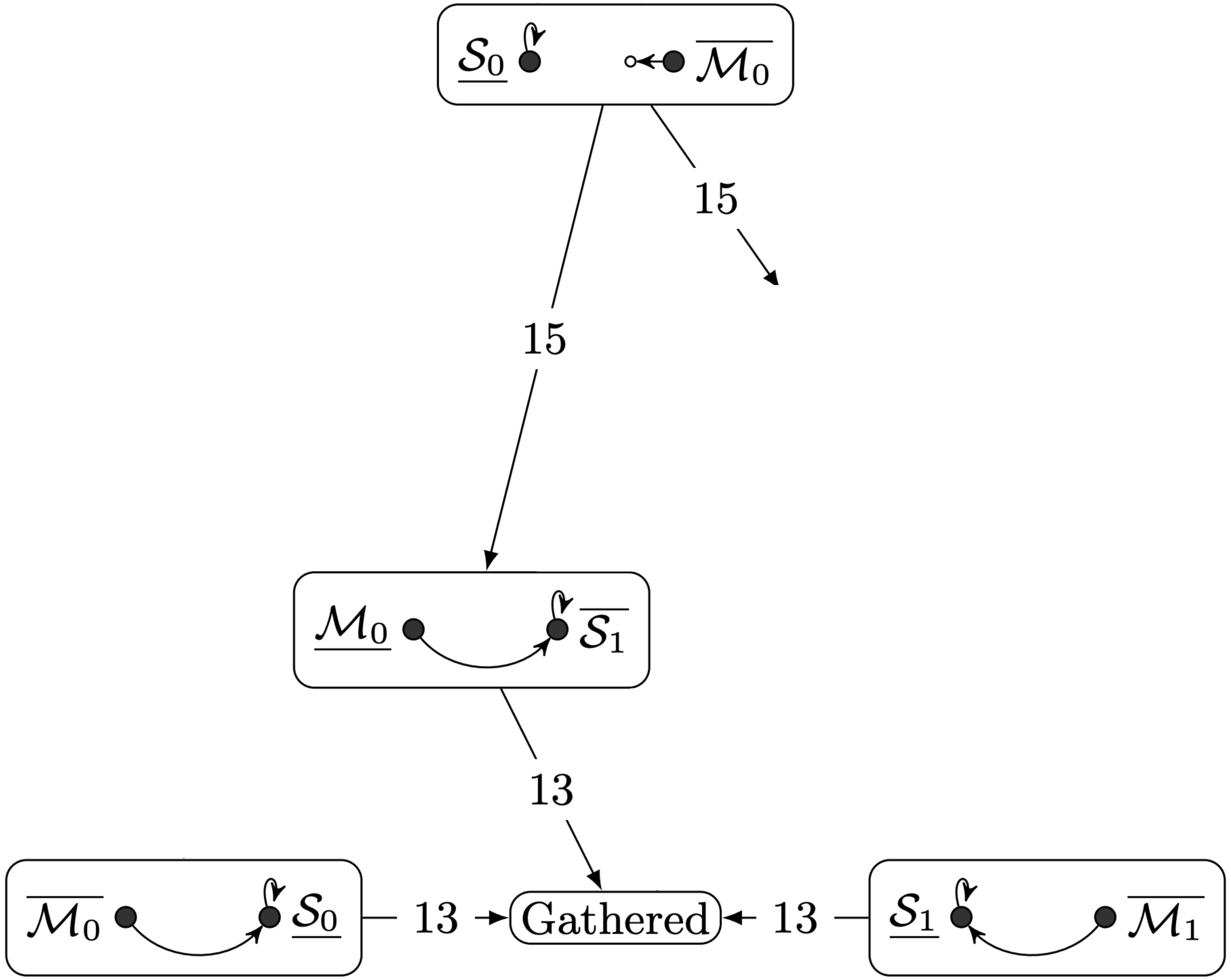
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution



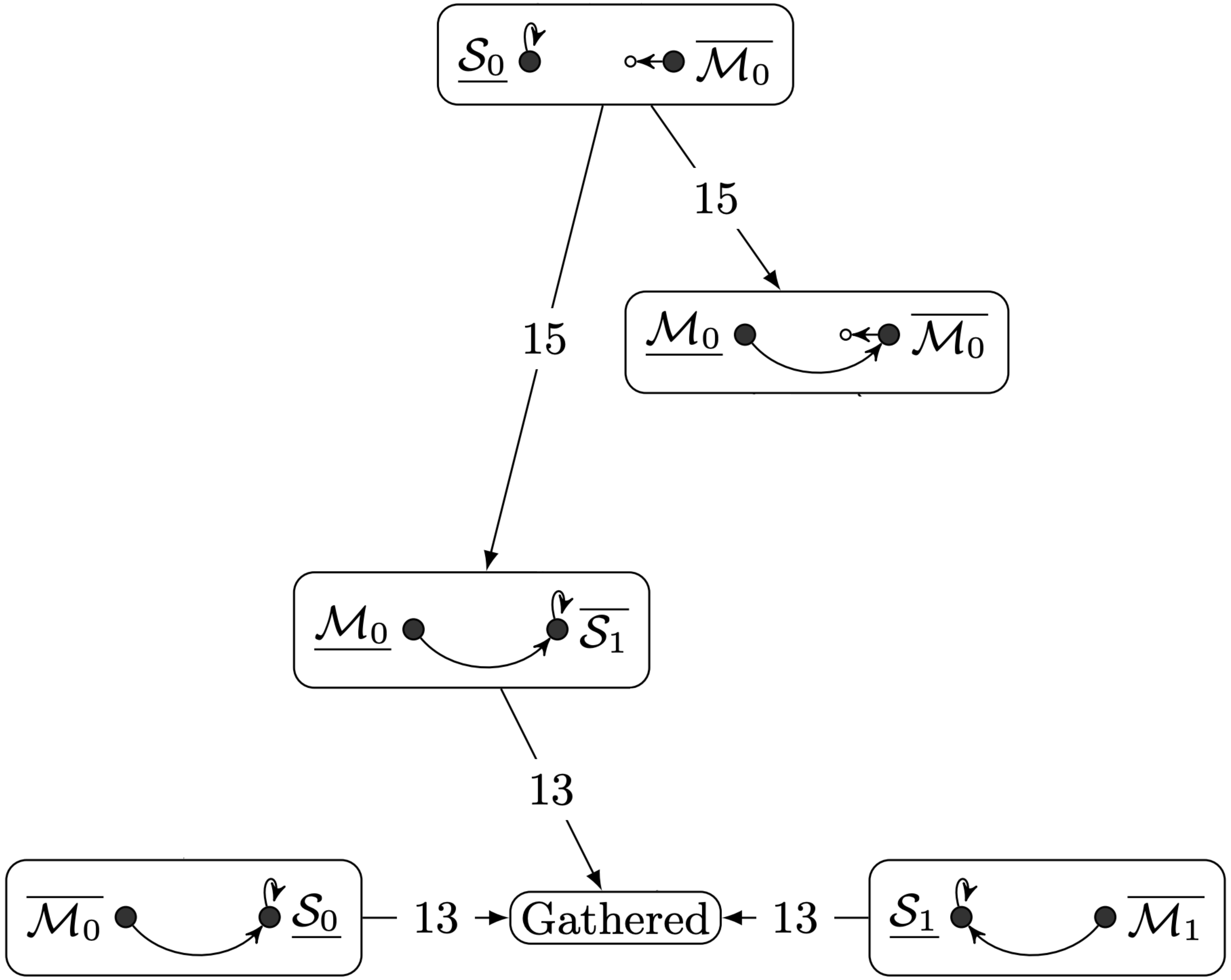
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution



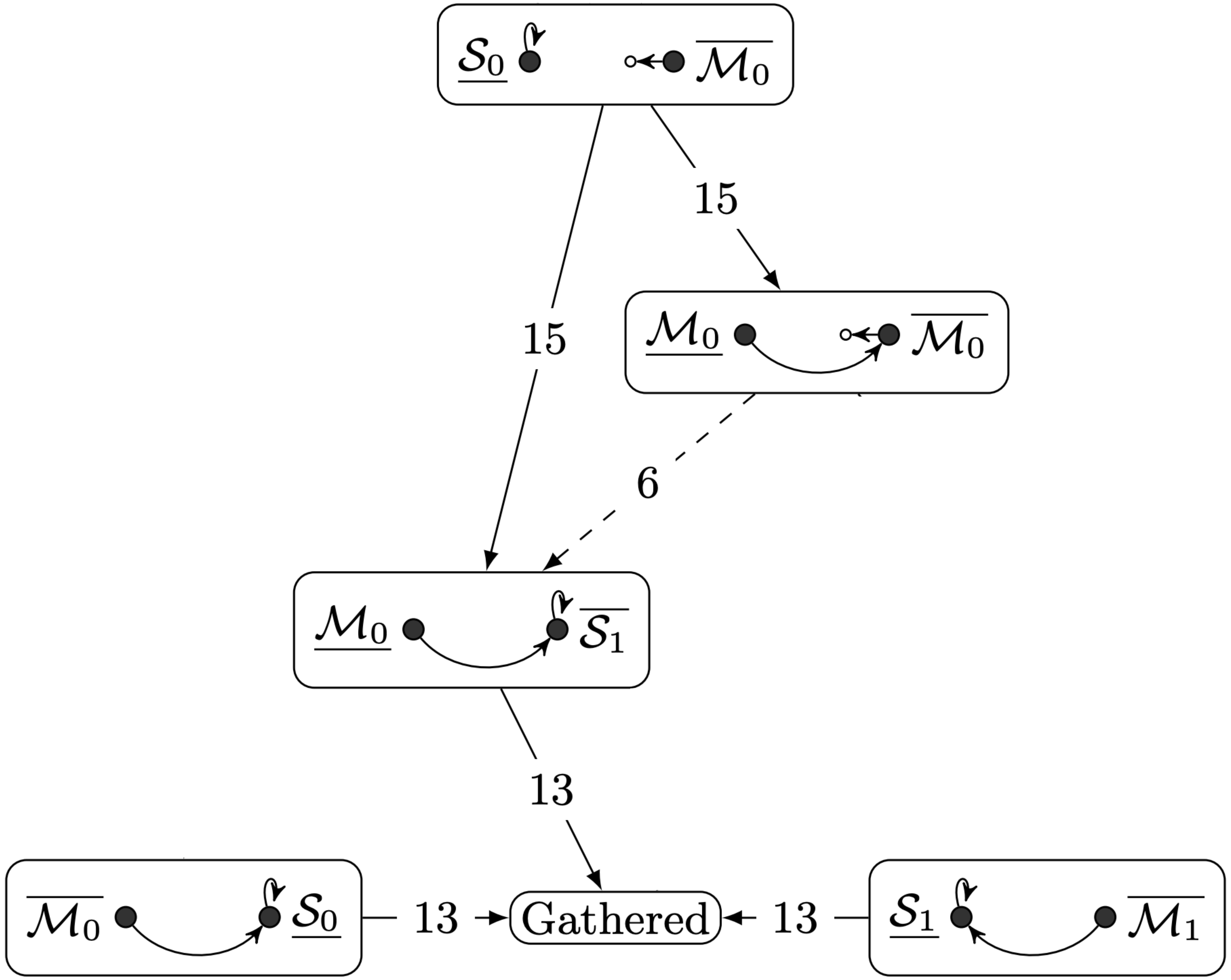
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution



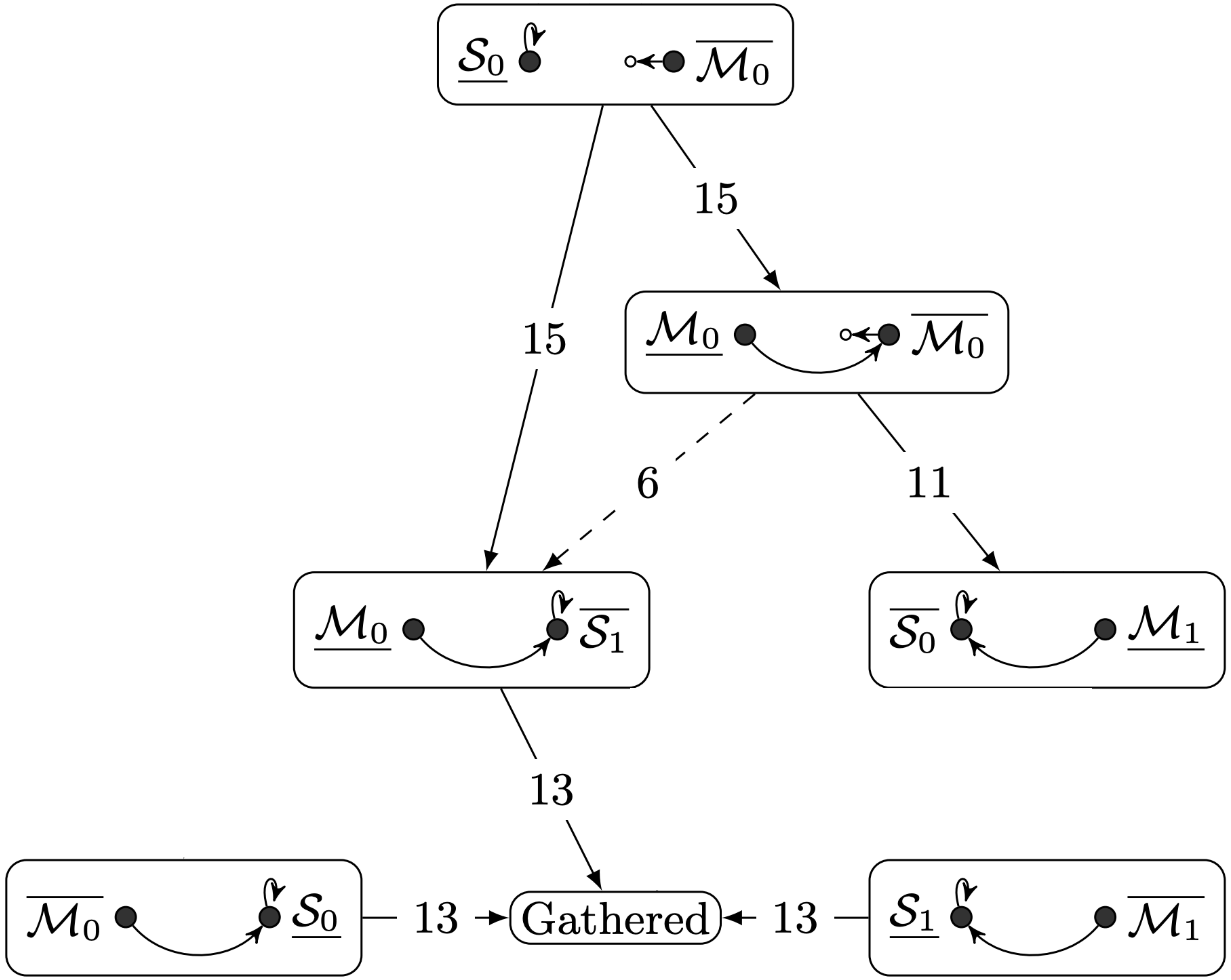
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution



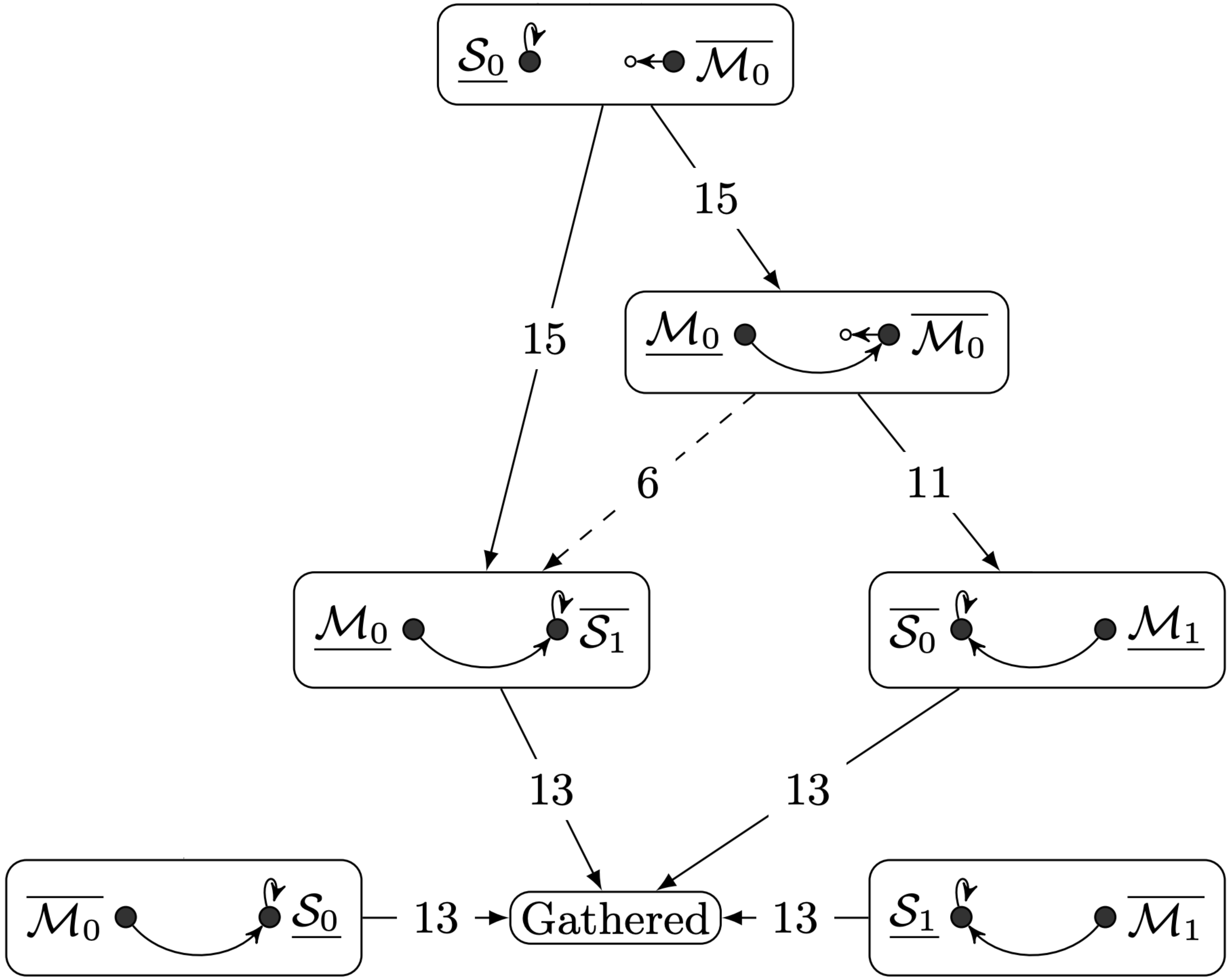
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution

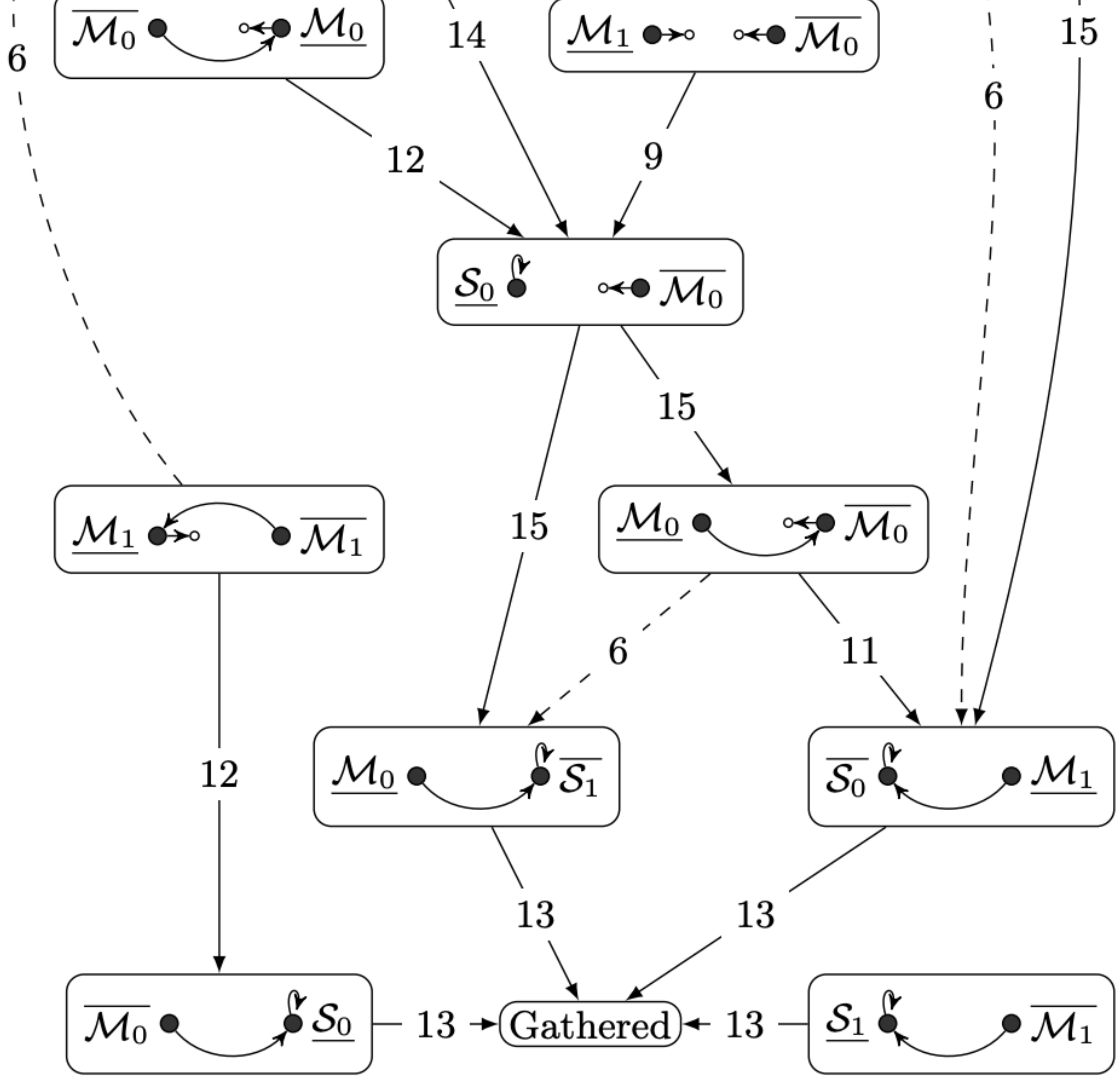


Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution

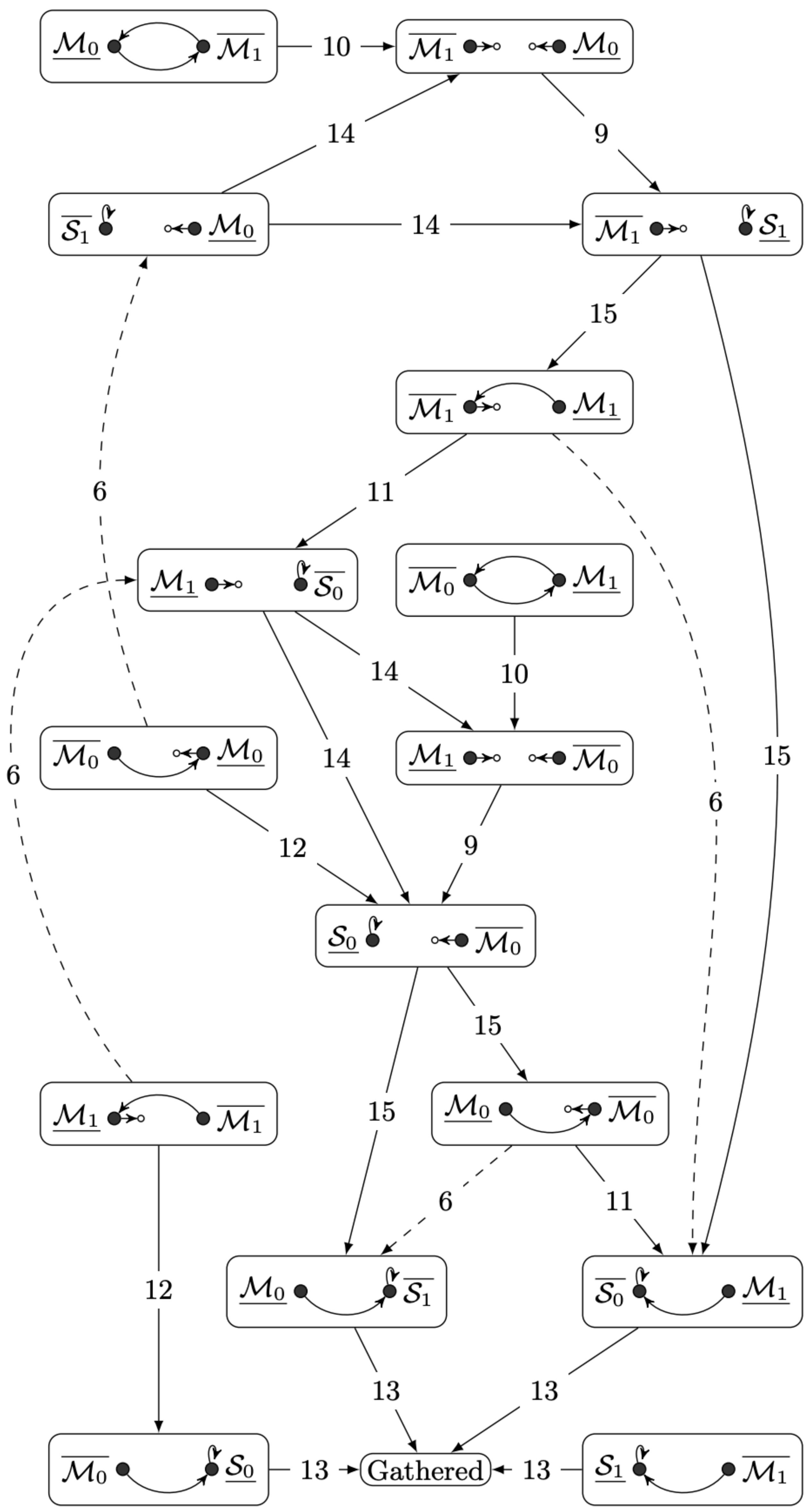


Rendezvous wh



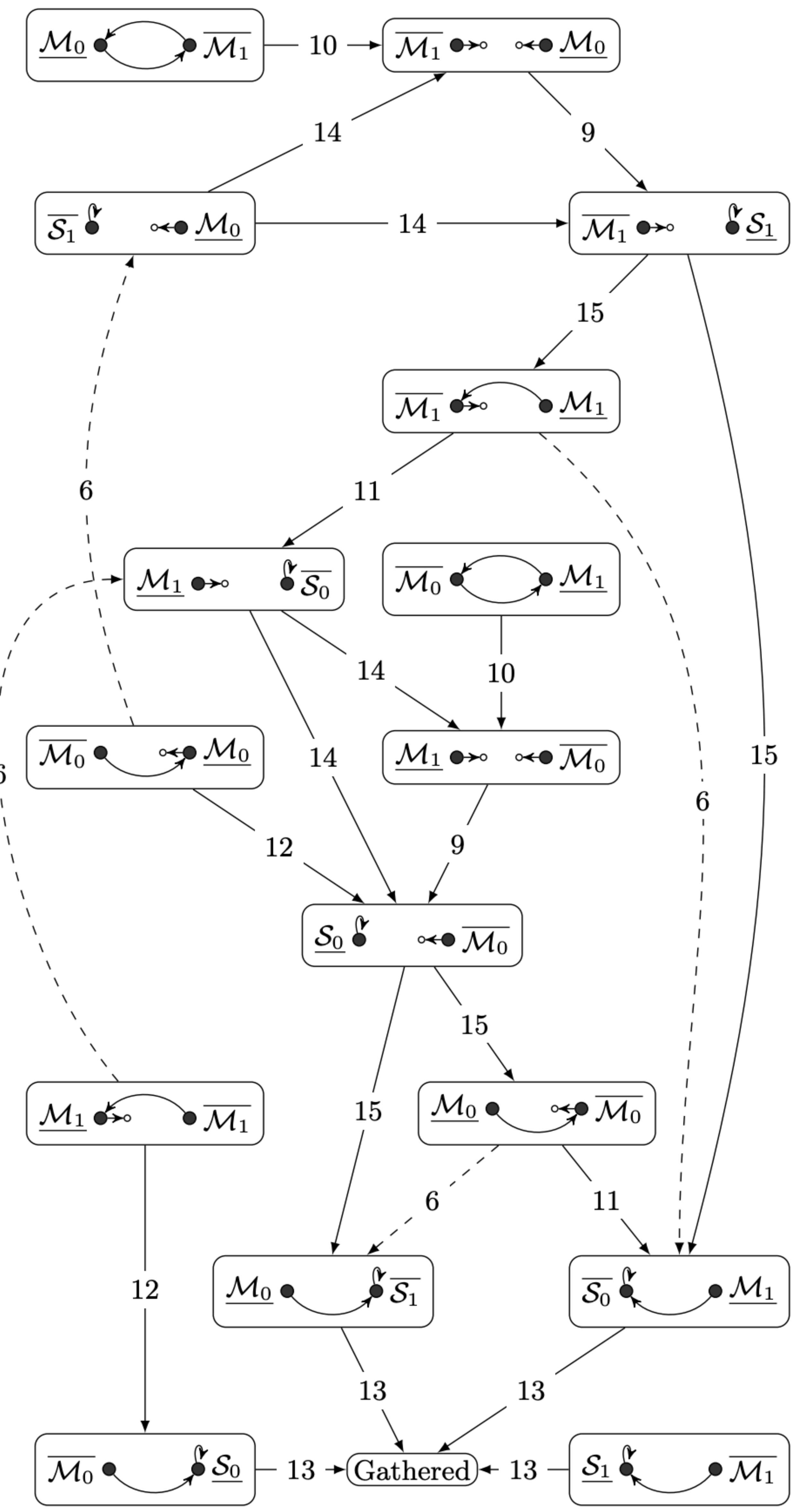
Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

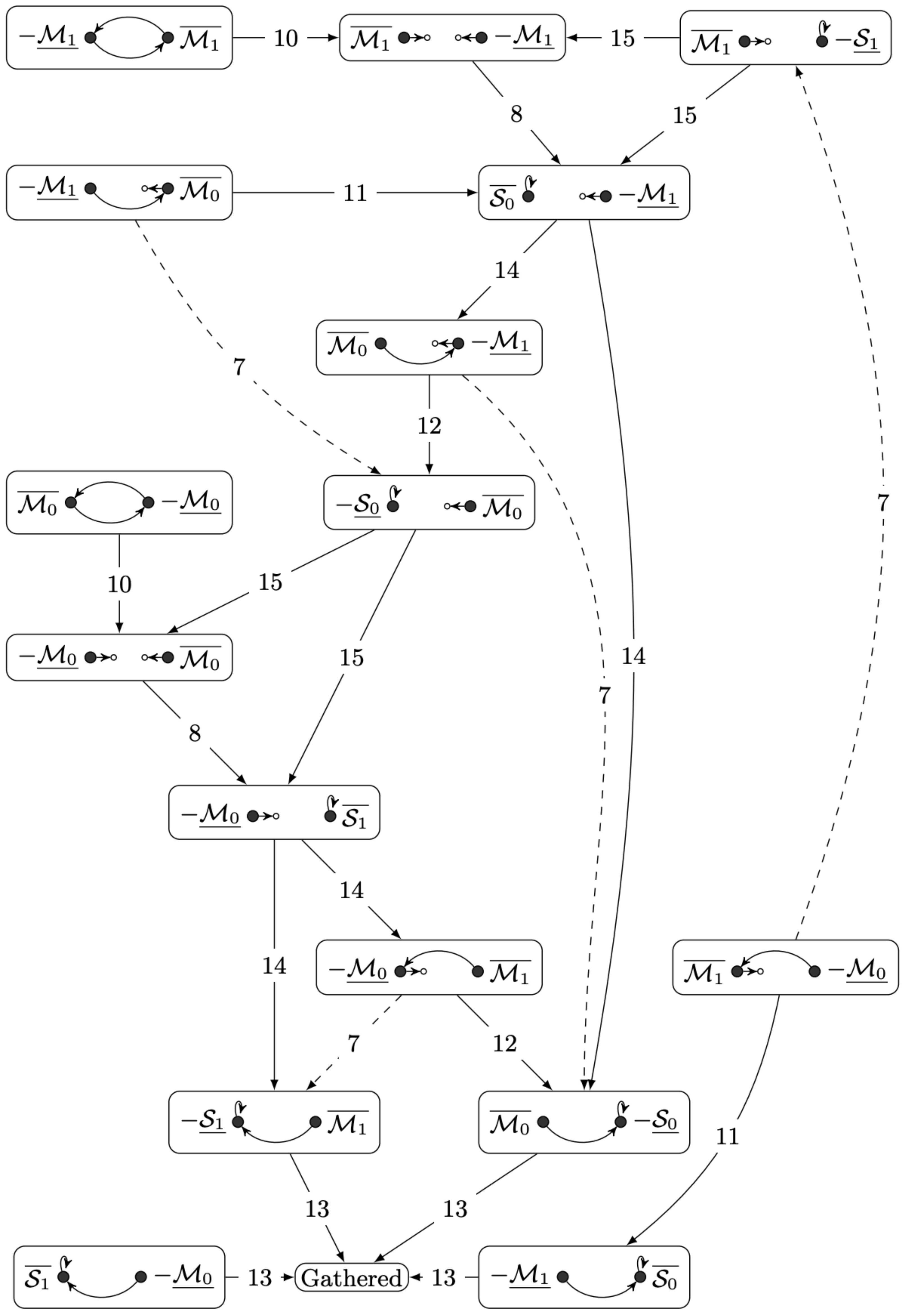
Algo2 execution



Rendezvous when $\rho = [\rho_{min}, \rho_{\max}]$

Algo2 execution





Conclusion

Conclusion

- We solved SSYNC rendez-vous with no agreement on the axis direction and with a disagreement on the unit distance.

Conclusion

- We solved SSYNC rendez-vous with no agreement on the axis direction and with a disagreement on the unit distance.

Conclusion

- We solved SSYNC rendez-vous with no agreement on the axis direction and with a disagreement on the unit distance.
- What if we **do not** have known bounds on ρ ?
  We conjecture that the problem is unsolvable, but it is still open.

Conclusion

- We solved SSYNC rendez-vous with no agreement on the axis direction and with a disagreement on the unit distance.

- What if we **do not** have known bounds on ρ ?
  We conjecture that the problem is unsolvable, but it is still open.

Thank you!