History-Deterministic One-Counter Nets

Aditya Prakash, K.S. Thejaswini

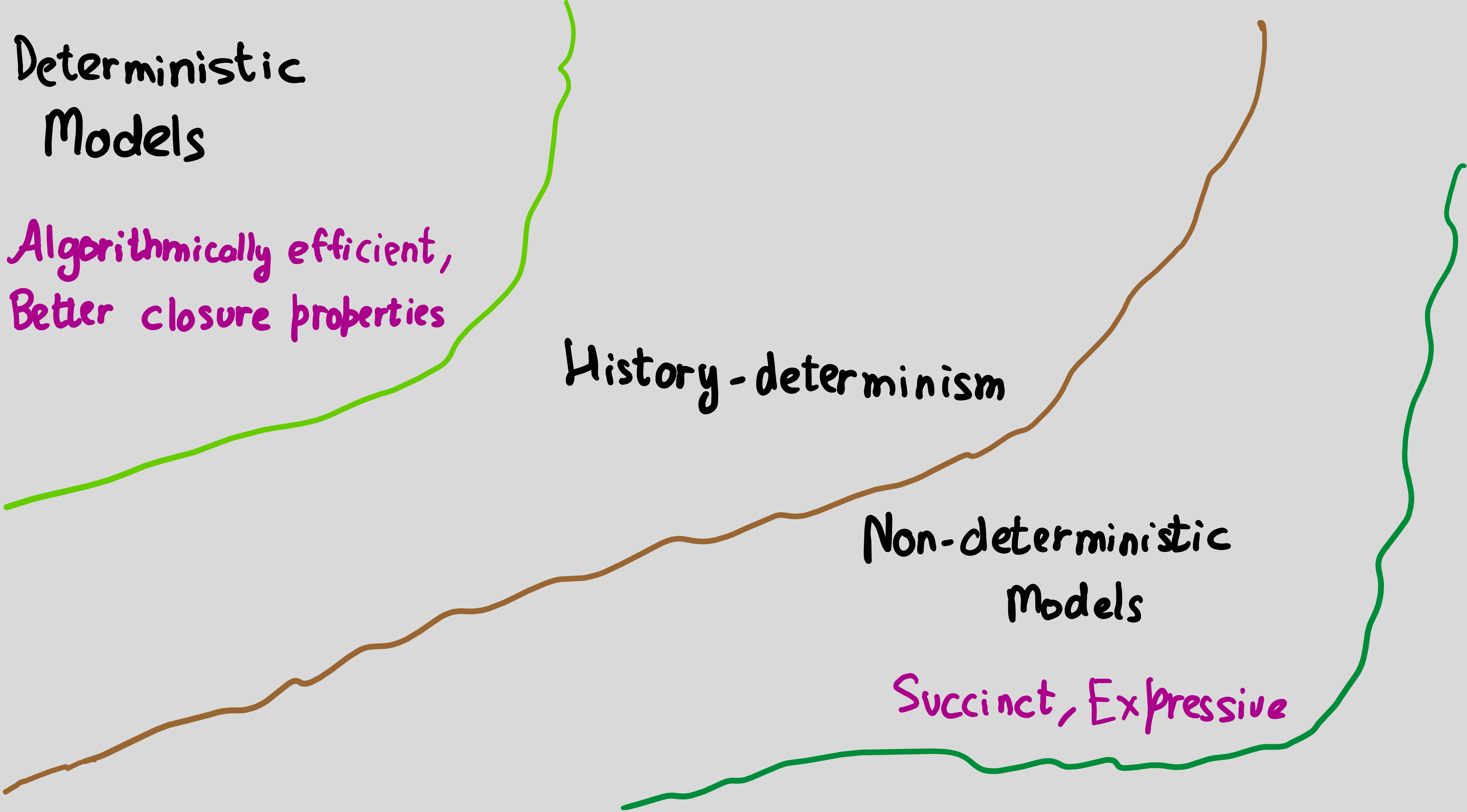
University of Warwick, UK

Deterministic Models

Algorithmically efficient,
Better closure properties

Non-deterministic Models

Succinct, Expressive



Deterministic Models

Algorithmically efficient,
Better closure properties

History-determinism

Non-deterministic Models

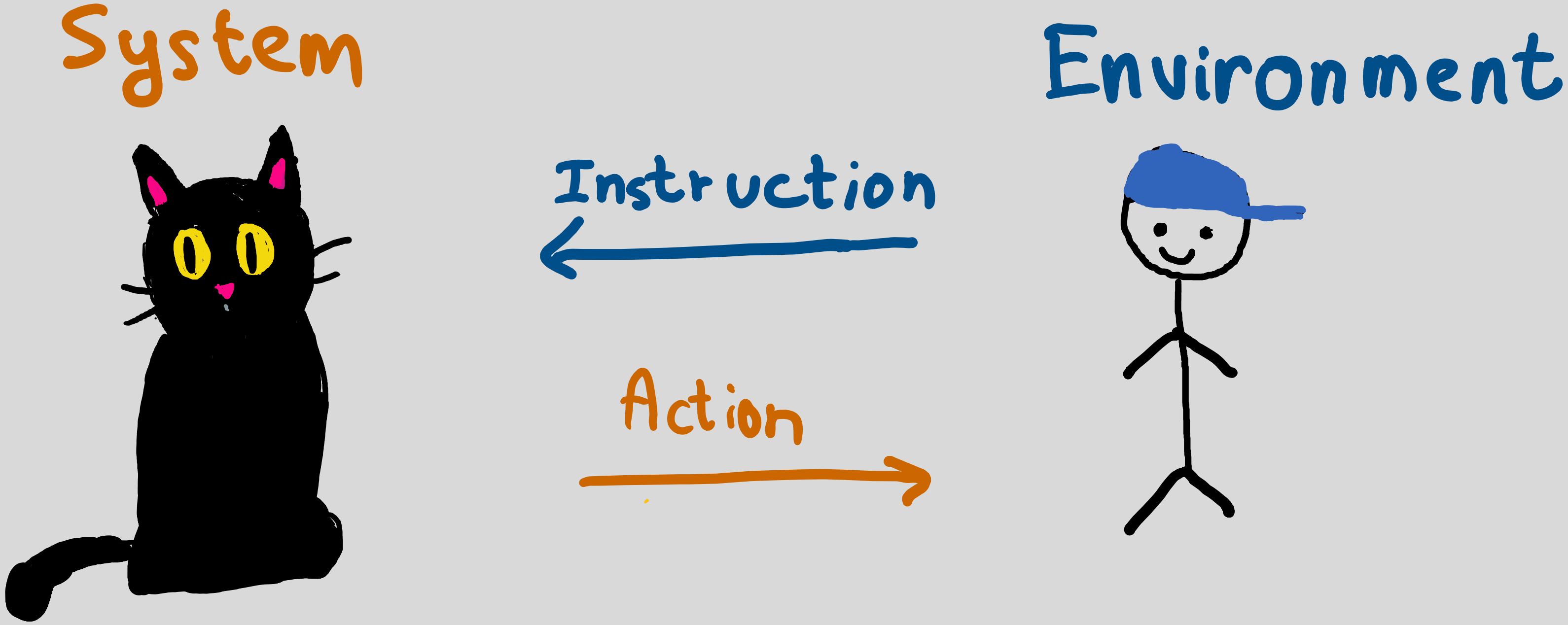
Succinct, Expressive

History-Deterministic One-Counter Nets

- What is history-determinism?
- The resolvers in an HD-OCN are semilinear

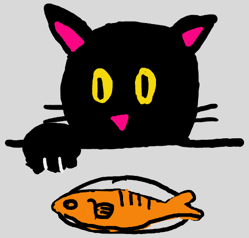
  Corollary: Every HD-OCN can be converted to a deterministic one-counter automaton.
- Complexity of checking history-determinism.

1. History - Determinism


System
Environment
Instruction
Action

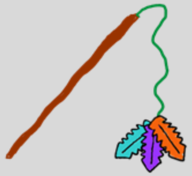
1.

x n



x n

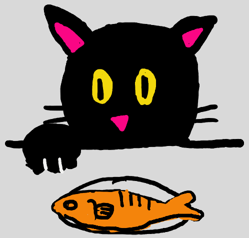
2.

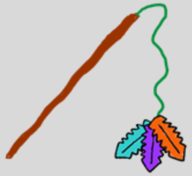


x m

x m

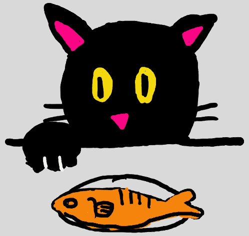
1.  $\times n$

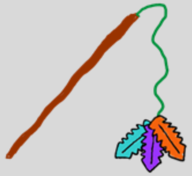
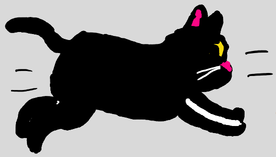
 , $+1 \times n$

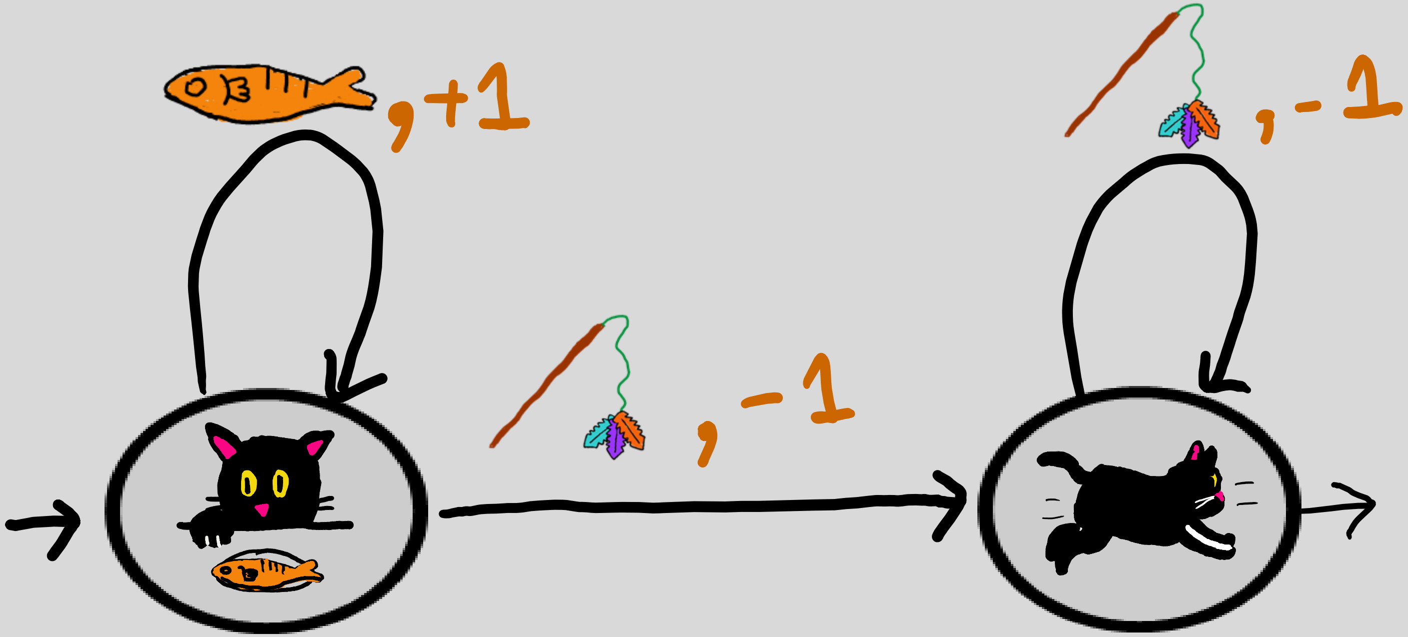
2.  $\times m$

 , $-1 \times m$

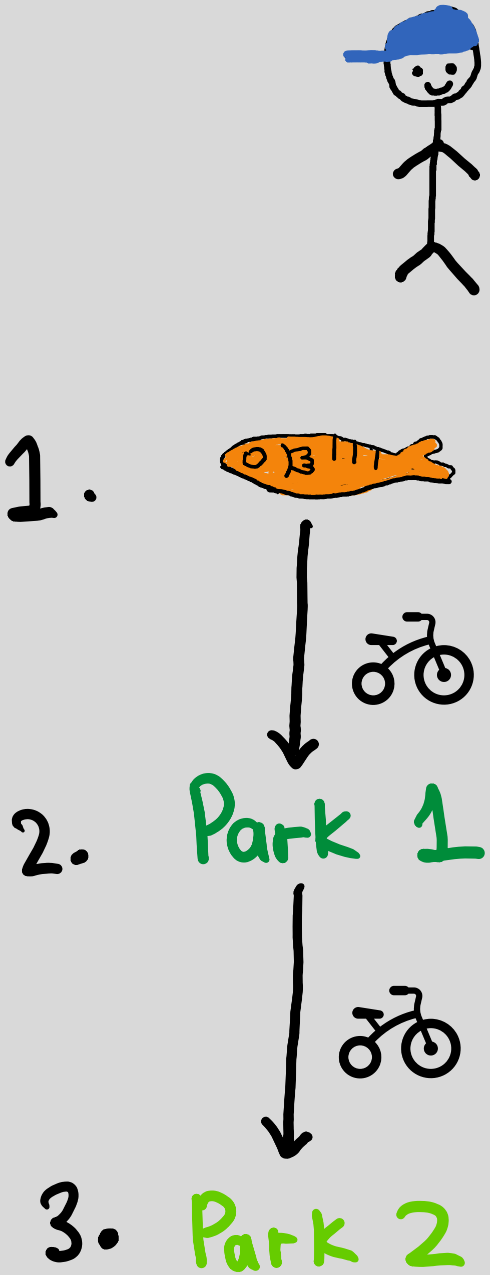
$n \geq m$

1.  × n  , +1 × n

2.  × m  , -1 × m



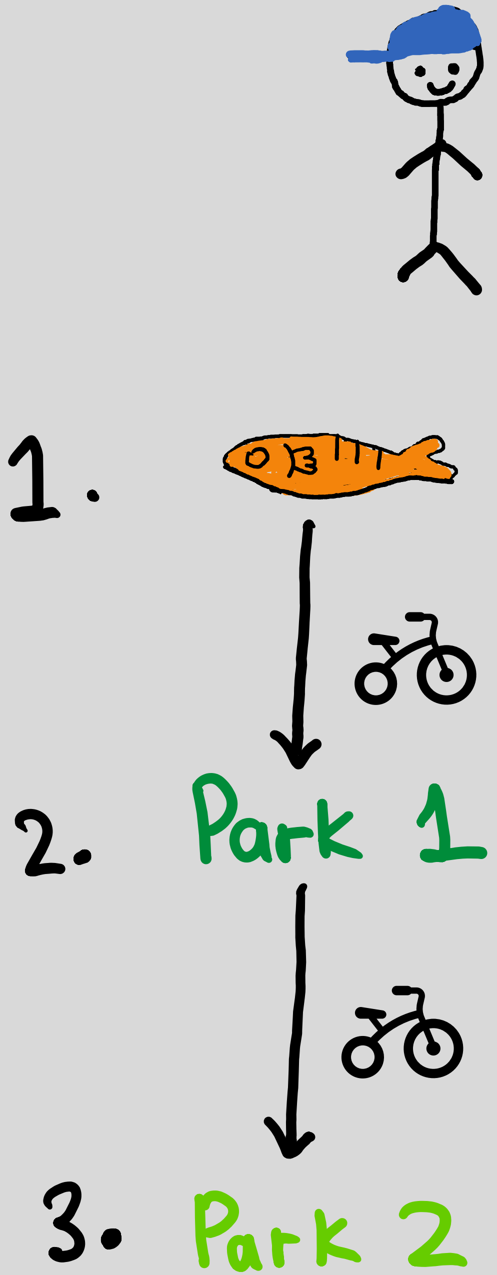
One-Counter Net

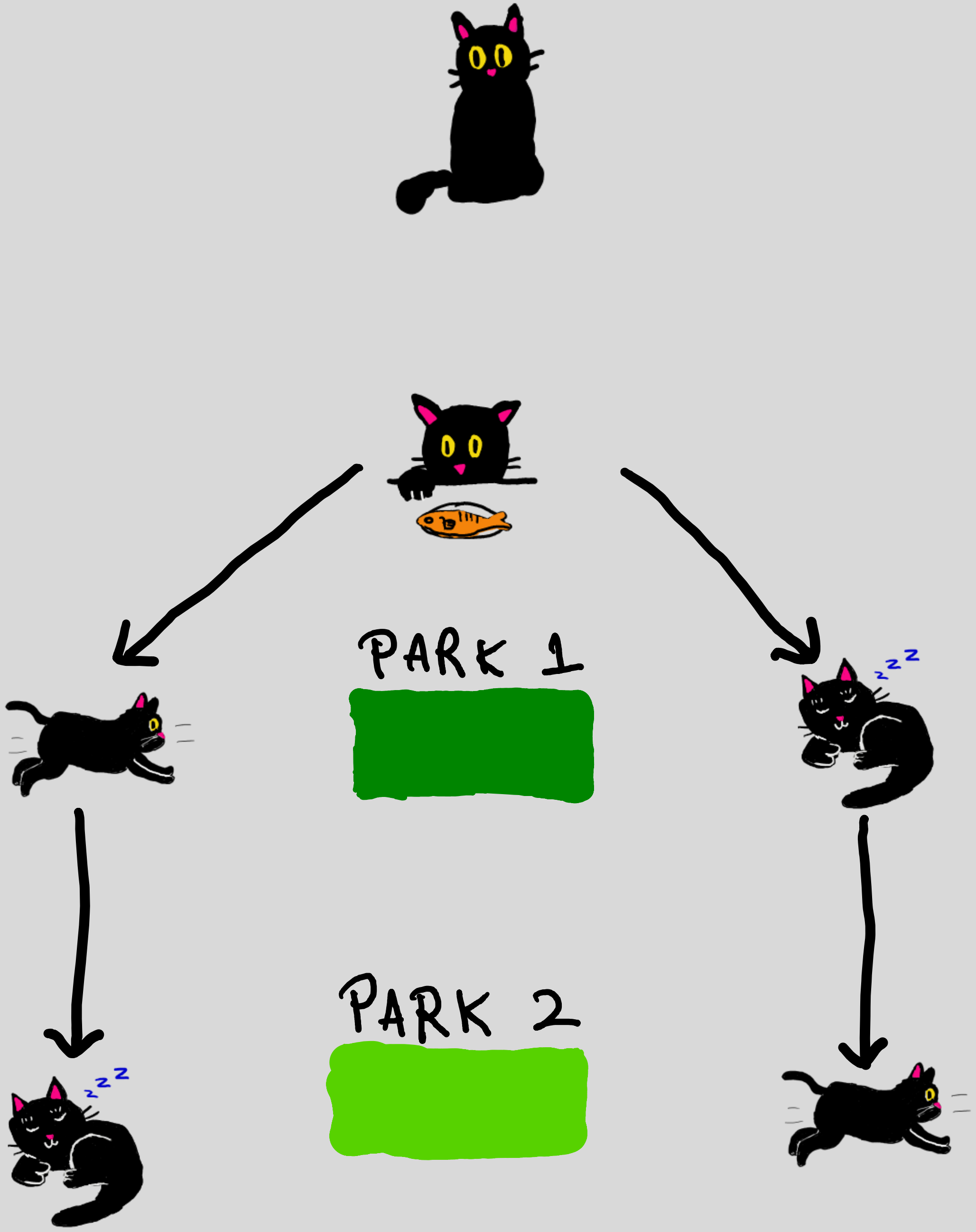


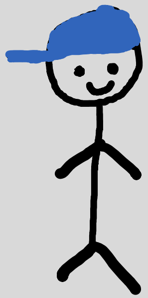
1.

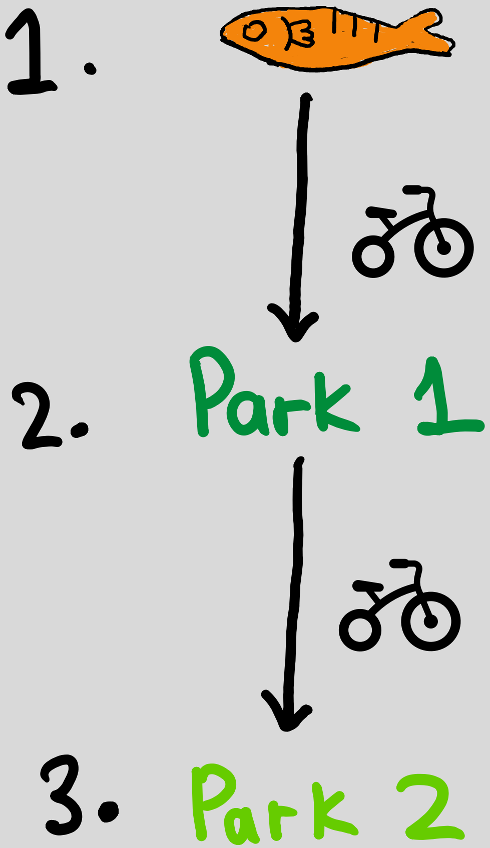
2. Park 1

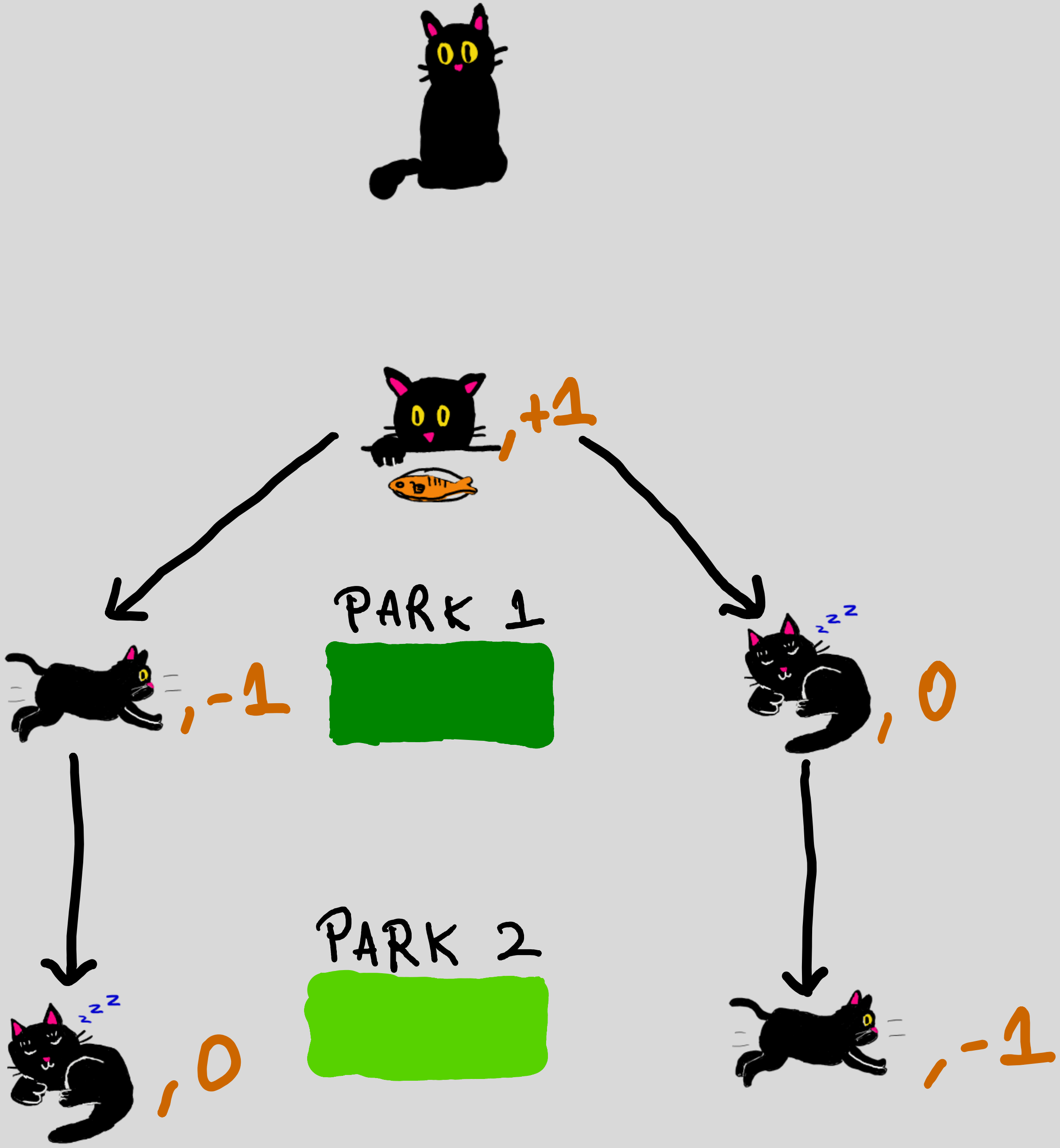
3. Park 2

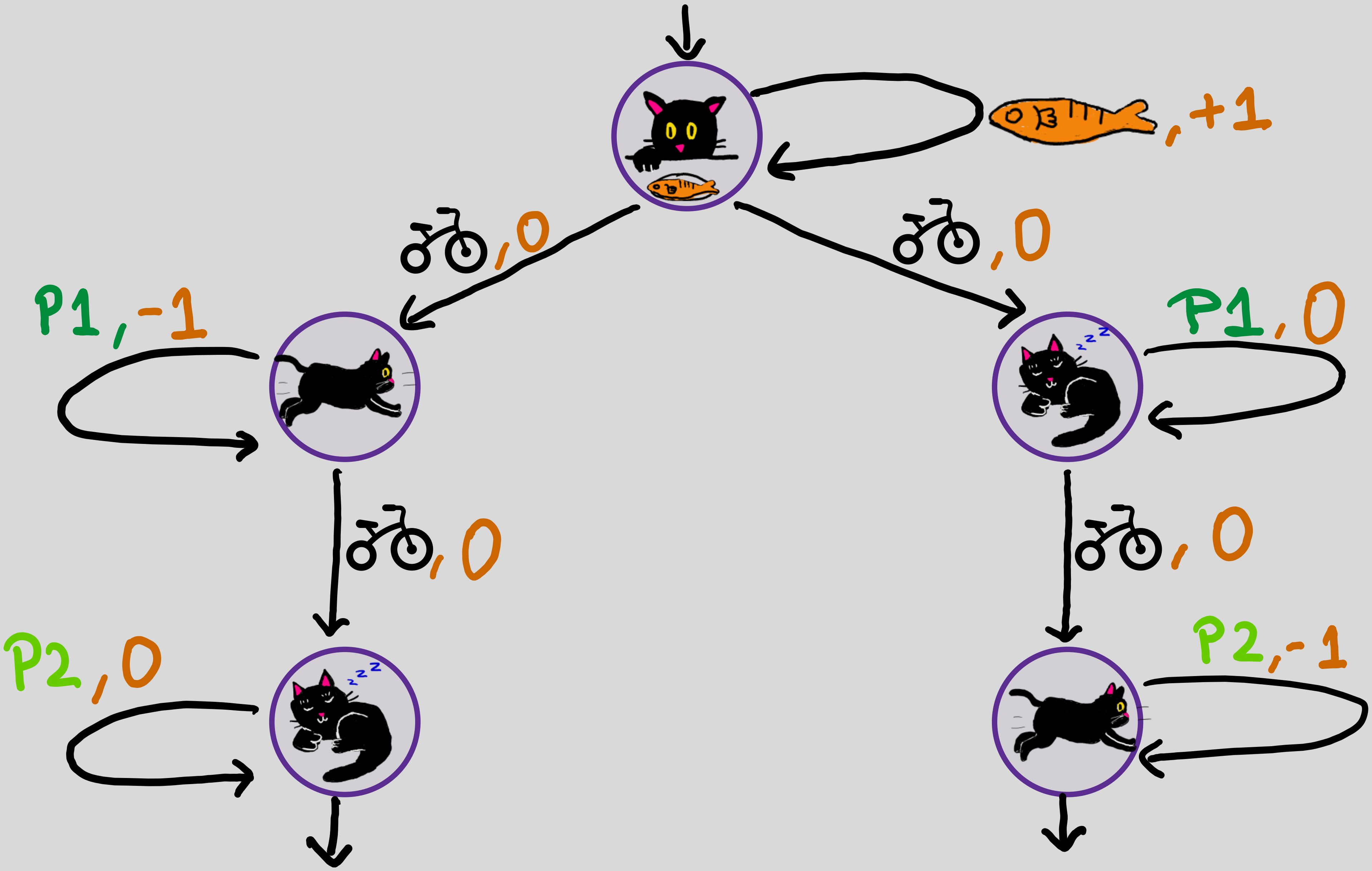

1.
2. Park 1
3. Park 2

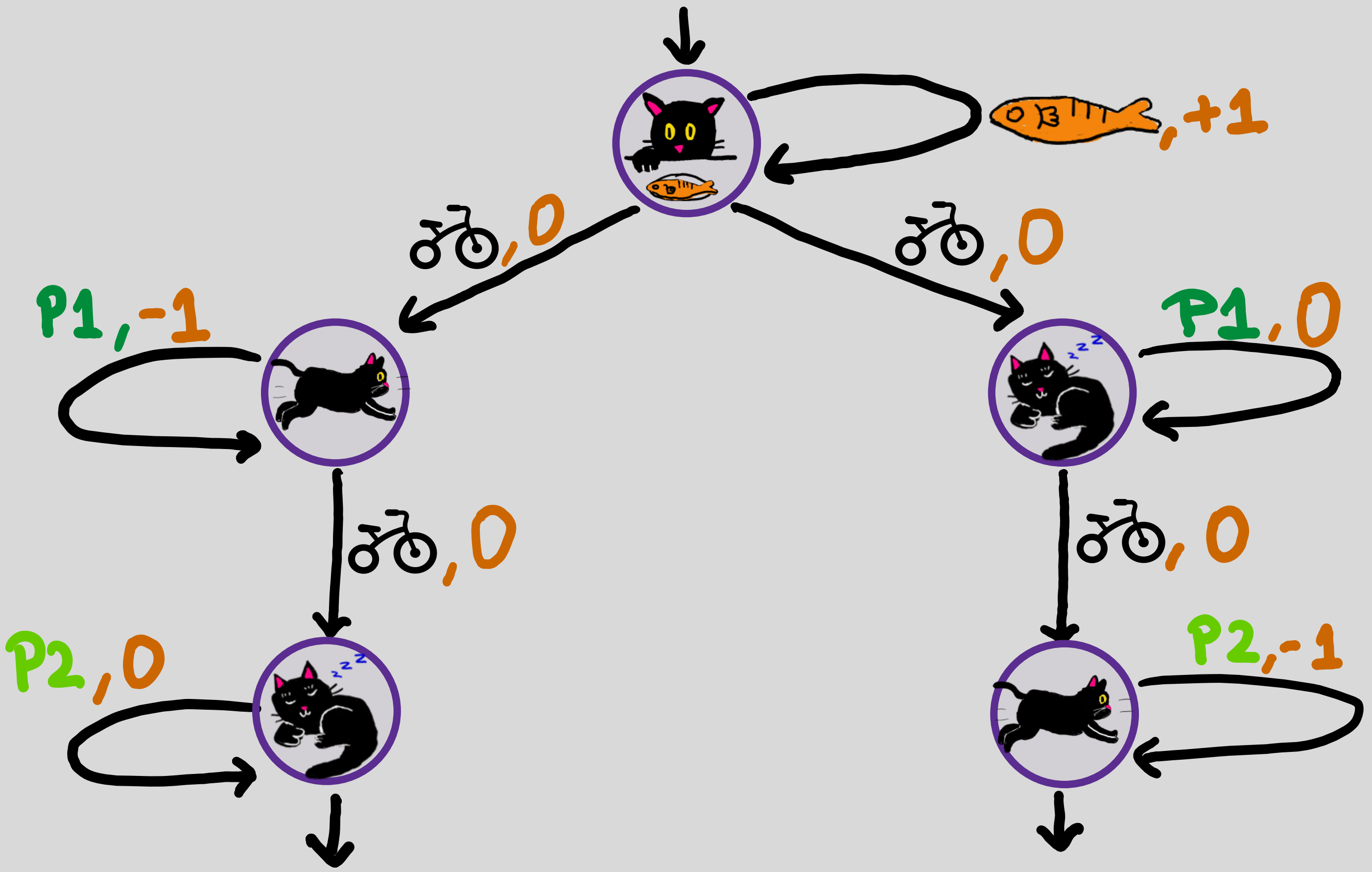

PARK 1
PARK 2




1.
2. Park 1
3. Park 2


, +1
PARK 1
, -1
, 0
PARK 2
, 0
, -1


, +1
, 0
, 0
P1, -1
P1, 0
, 0
, 0
P2, 0
P2, -1



L = { 🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$ }

(with 5 )


I will spend 4 hours in Park 1, 7 hours in Park 2.
(with 5)


I will spend 4 hours in Park 1, 7 hours in Park 2.
Ok, I will play in Park 1.
(with 5 🐟)

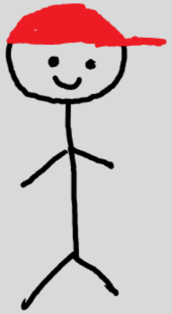

I will spend 4 hours in Park 1, 7 hours in Park 2.
OK, I will play in Park 1.
(with 5)


I will spend 4 hours in Park 1, 3 hours in Park 2.
(with 5)

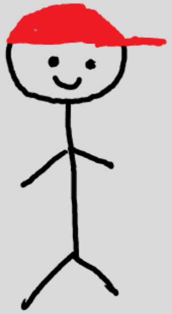

I will spend 4 hours in Park 1, 7 hours in Park2.
Ok, I will play in Park 1.
(with 5)

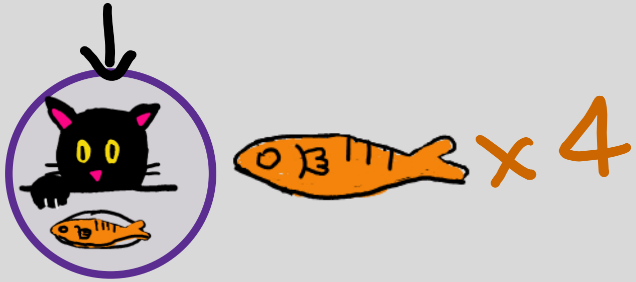

I will spend 4 hours in Park 1, 3 hours in Park 2.
I can play in either parks!!
(with 5)

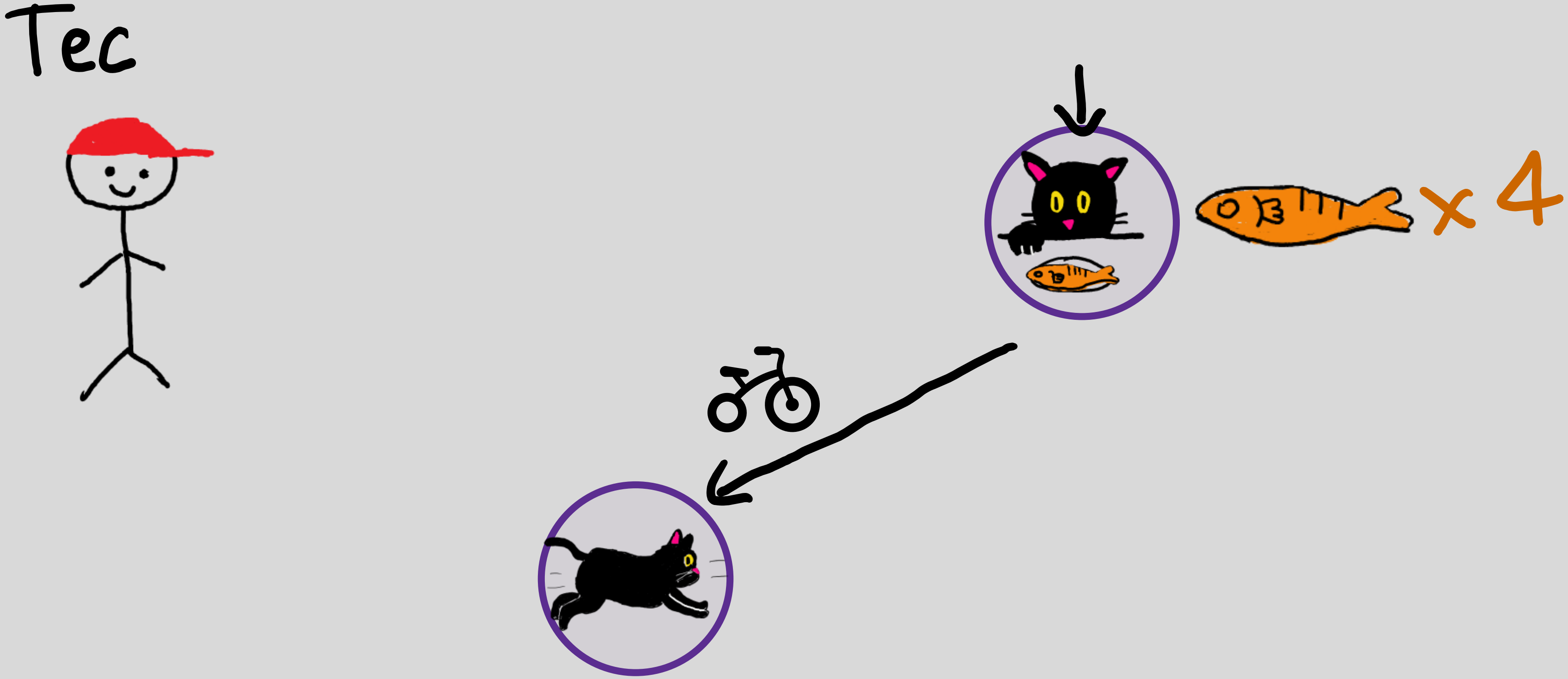
Tec

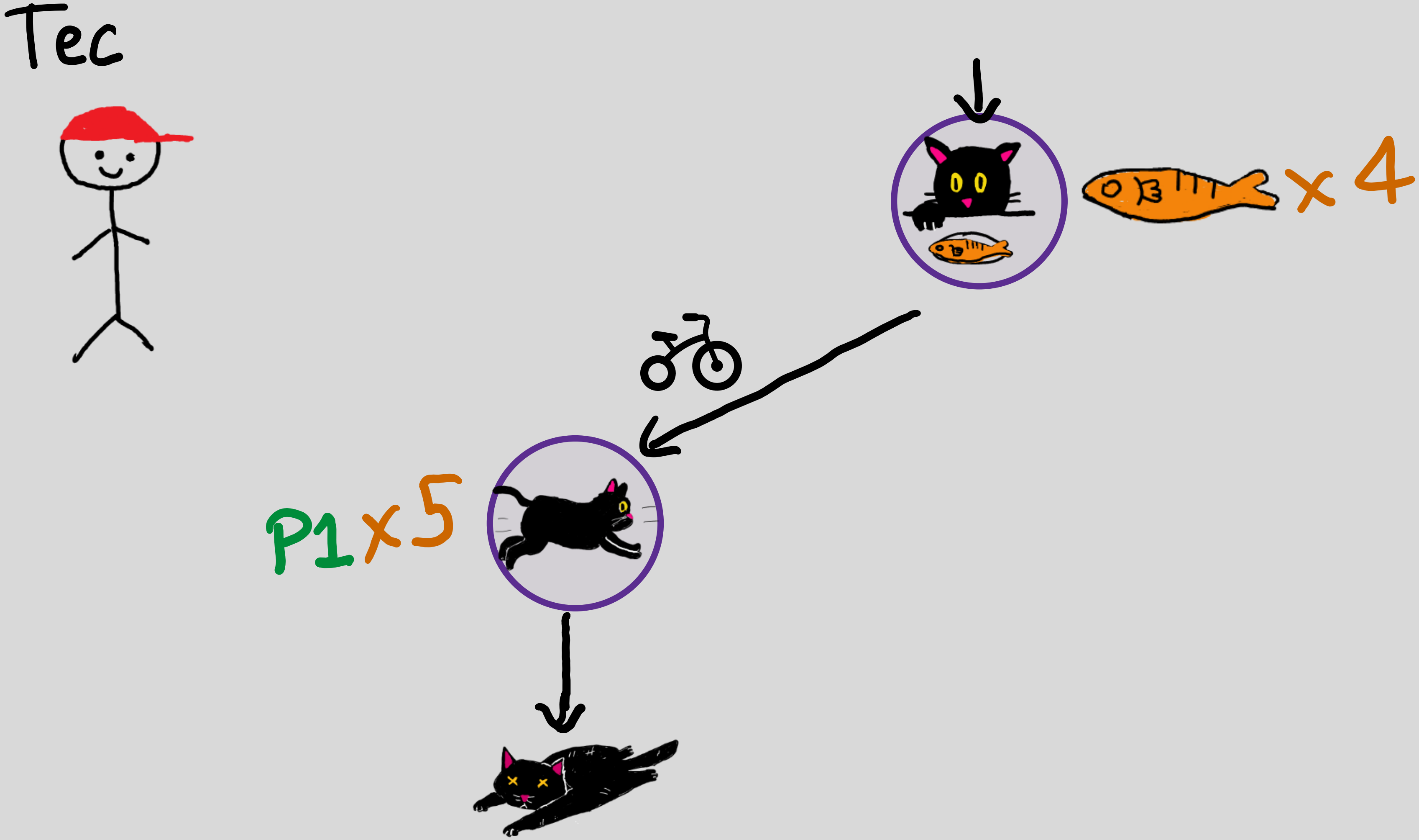


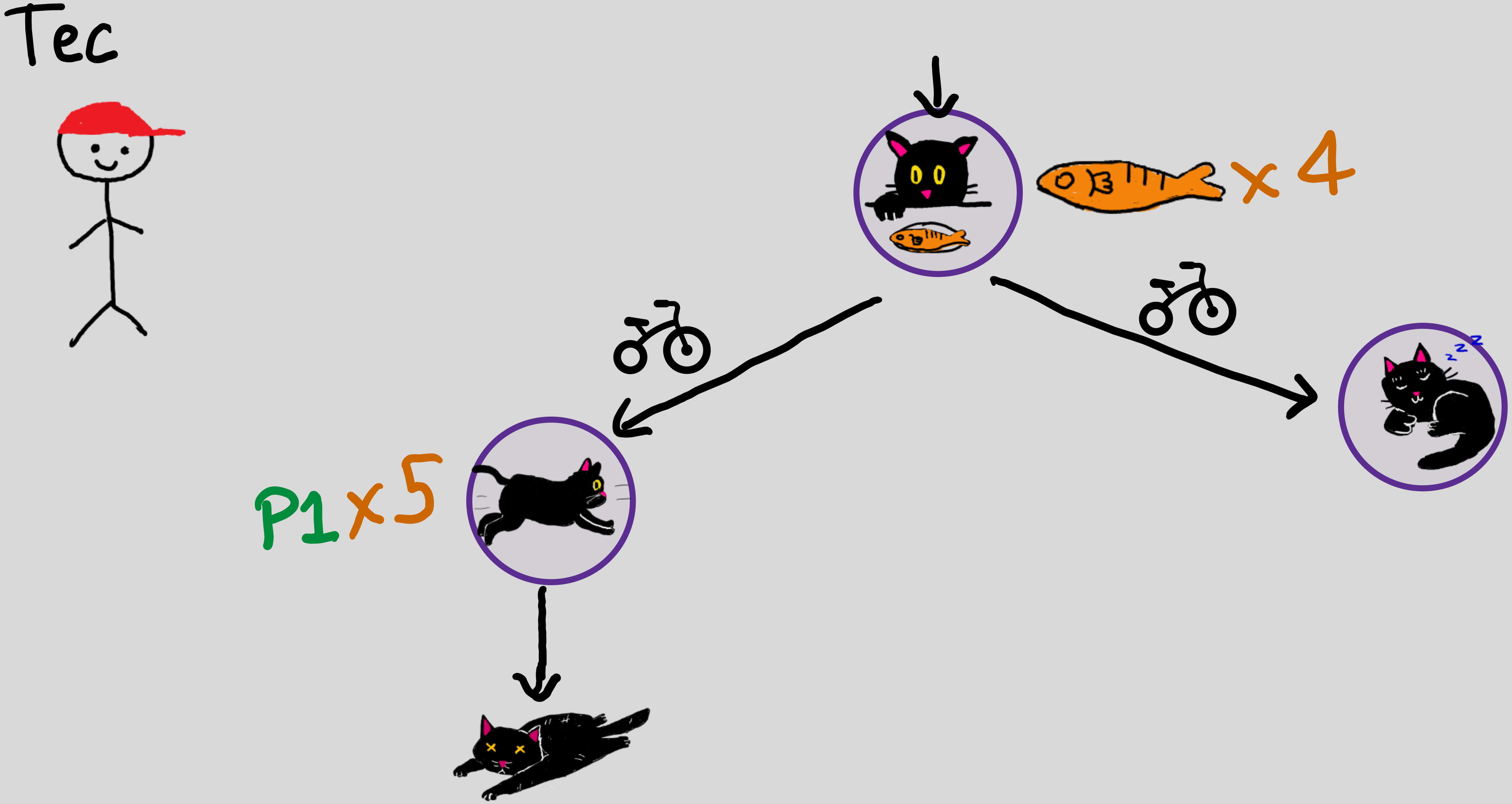
Tec



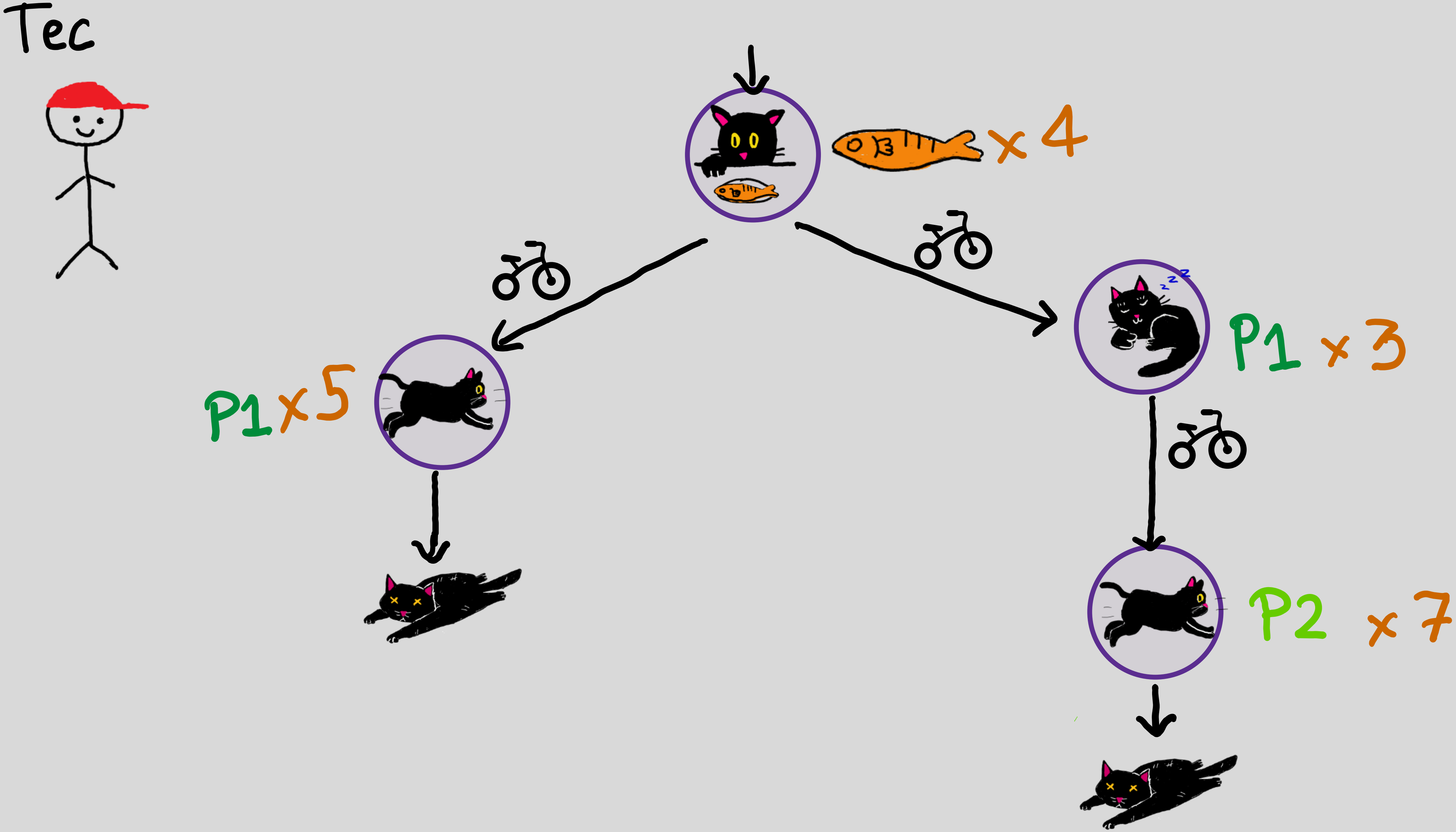



Tec
× 4

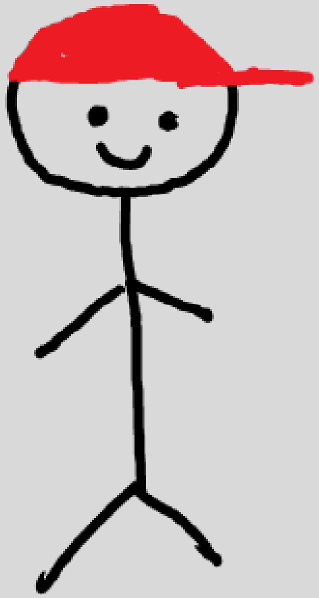

Tec
x 4
P1 x 5


Tec
x 4
P1 x 5

Tec

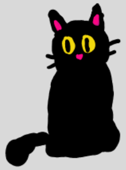


Tec




Tec

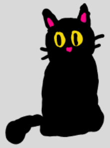

The Evil
Catsitter

Definition: We say an one-counter net is history-deterministic if  has a strategy that produces an accepting run whenever  gives an accepting word.

History - Determinism Game:

History-Determinism Game:

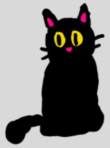
Starts at ( , 0):

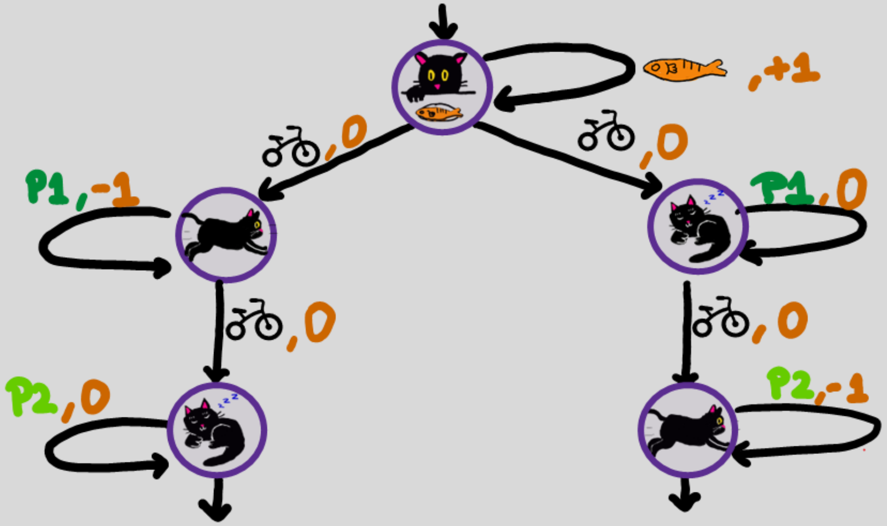
1.  selects letter
2.  selects transition

History-Determinism Game:

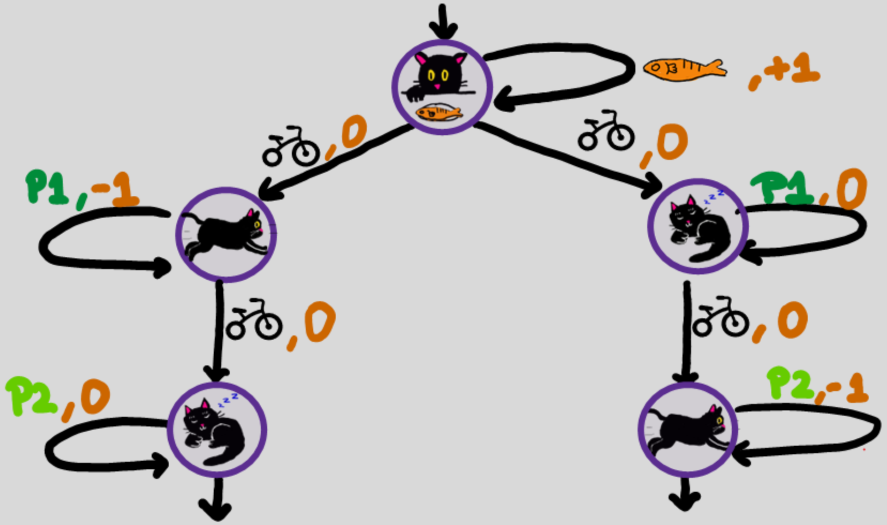
Starts at ( , 0):

1.  selects letter
2.  selects transition

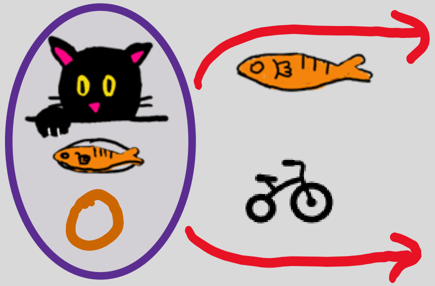
Winning Condition for  : If 's word is accepting, and 's run is rejecting.

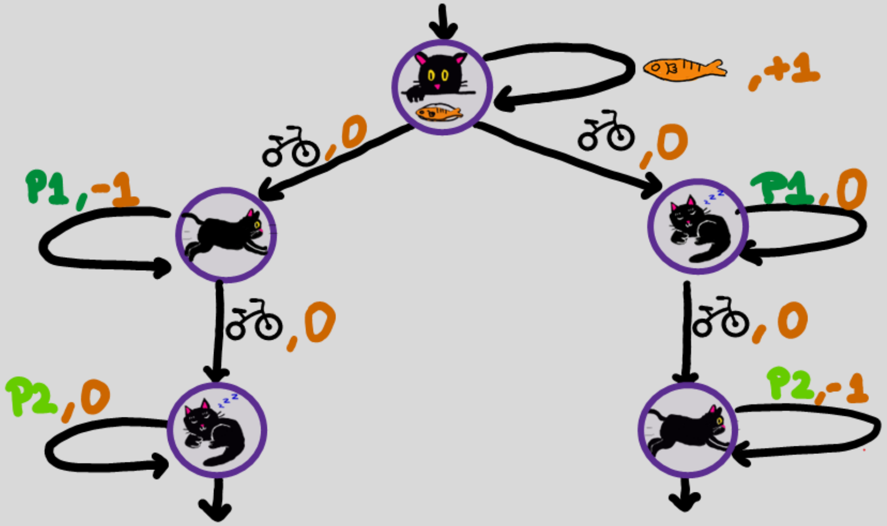


L = { 🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$ }

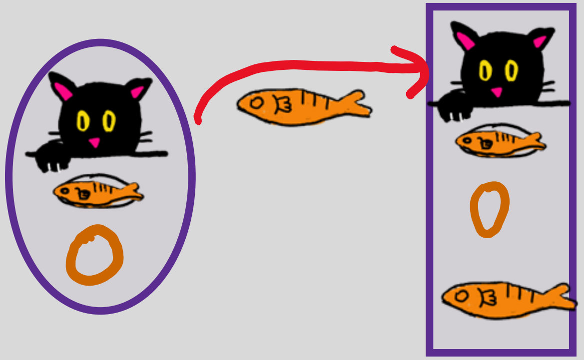


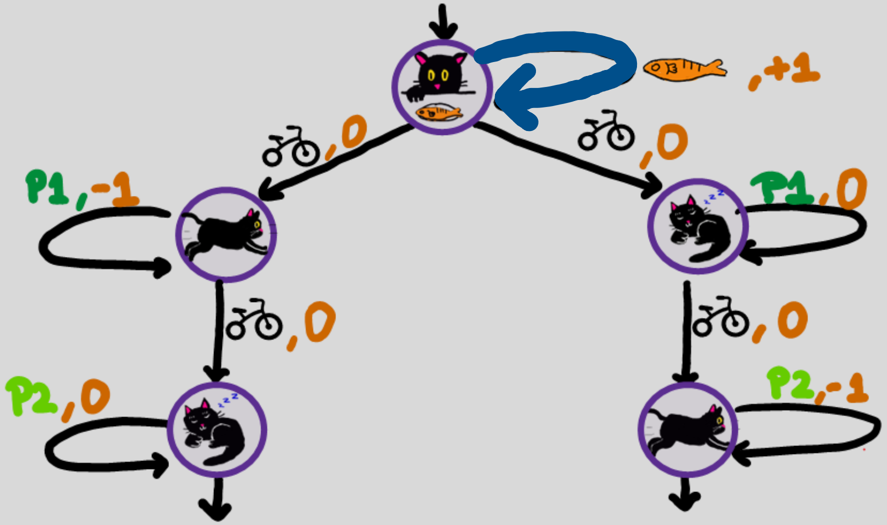
$$L = \{ \text{🐟}^{i} \; \text{🚲} \; P1^{j} \; \text{🚲} \; P2^{k} \mid i \geq j \text{ or } i \geq k \}$$



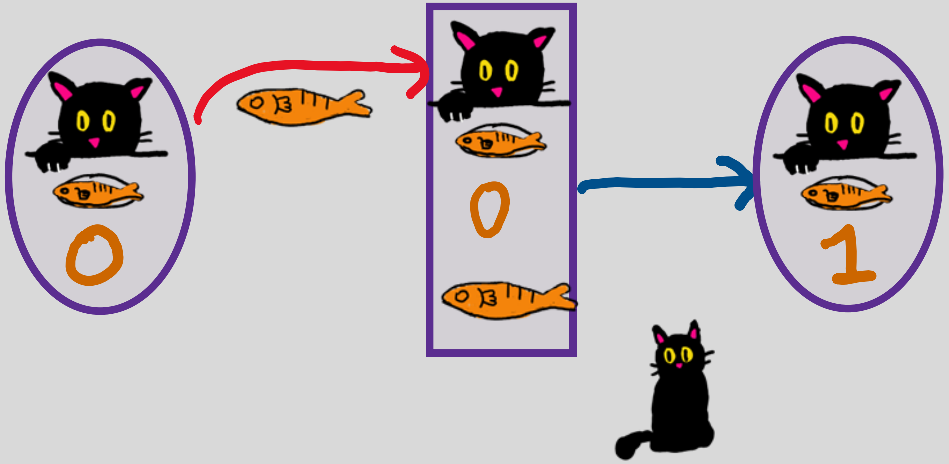


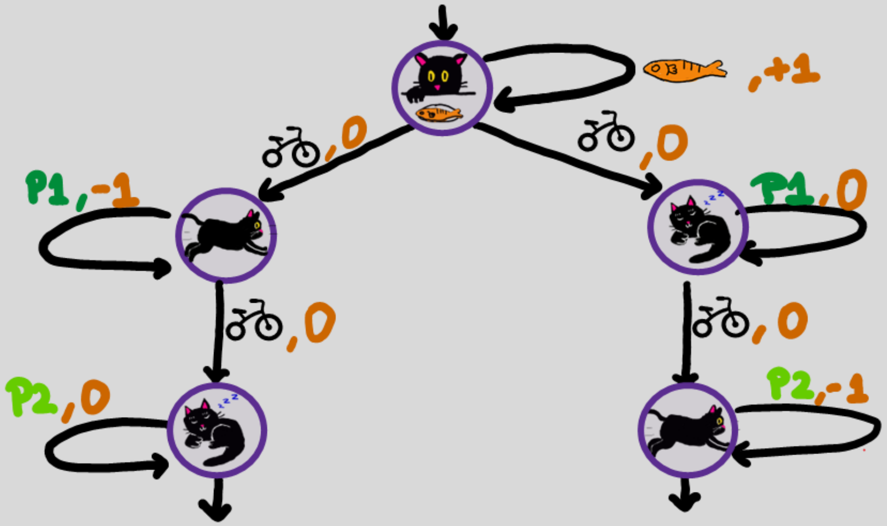
L = { 🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$ }



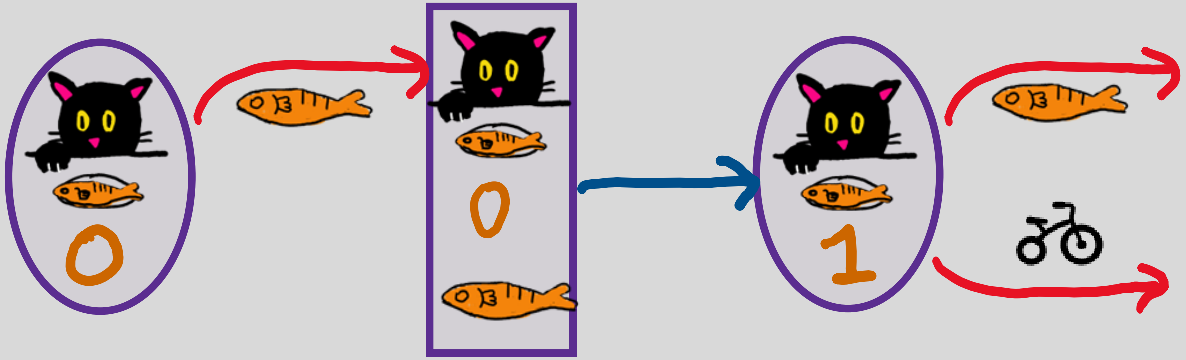


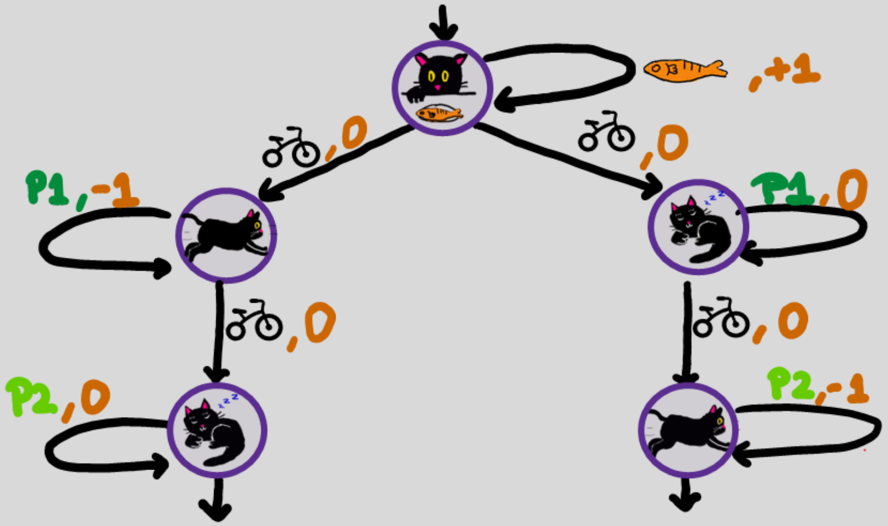
L = {🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$}



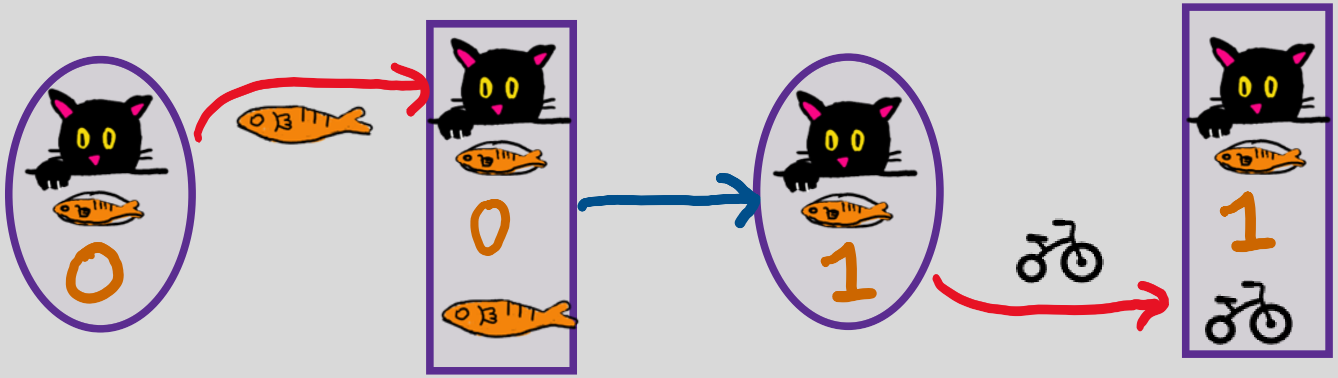


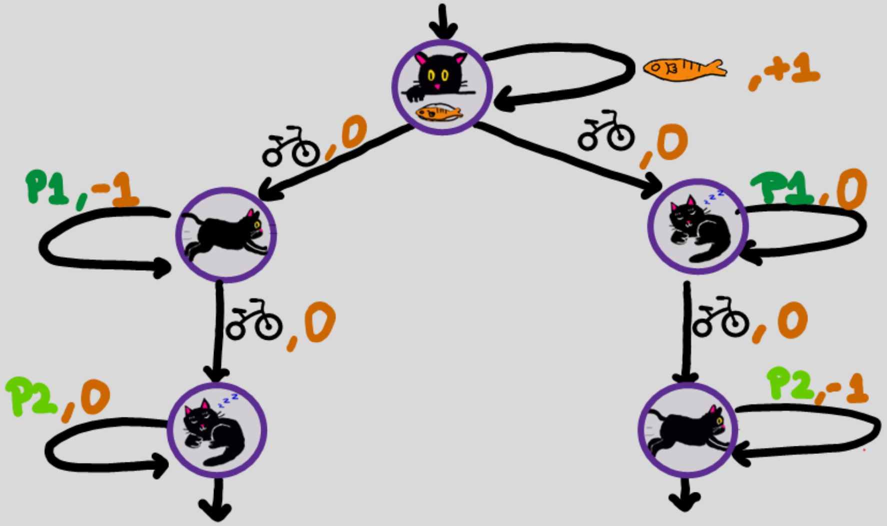
$$L = \{ \text{🐟}^{i} \; \text{🚲} \; P1^{j} \; \text{🚲} \; P2^{k} \mid i \geqslant j \text{ or } i \geqslant k \}$$



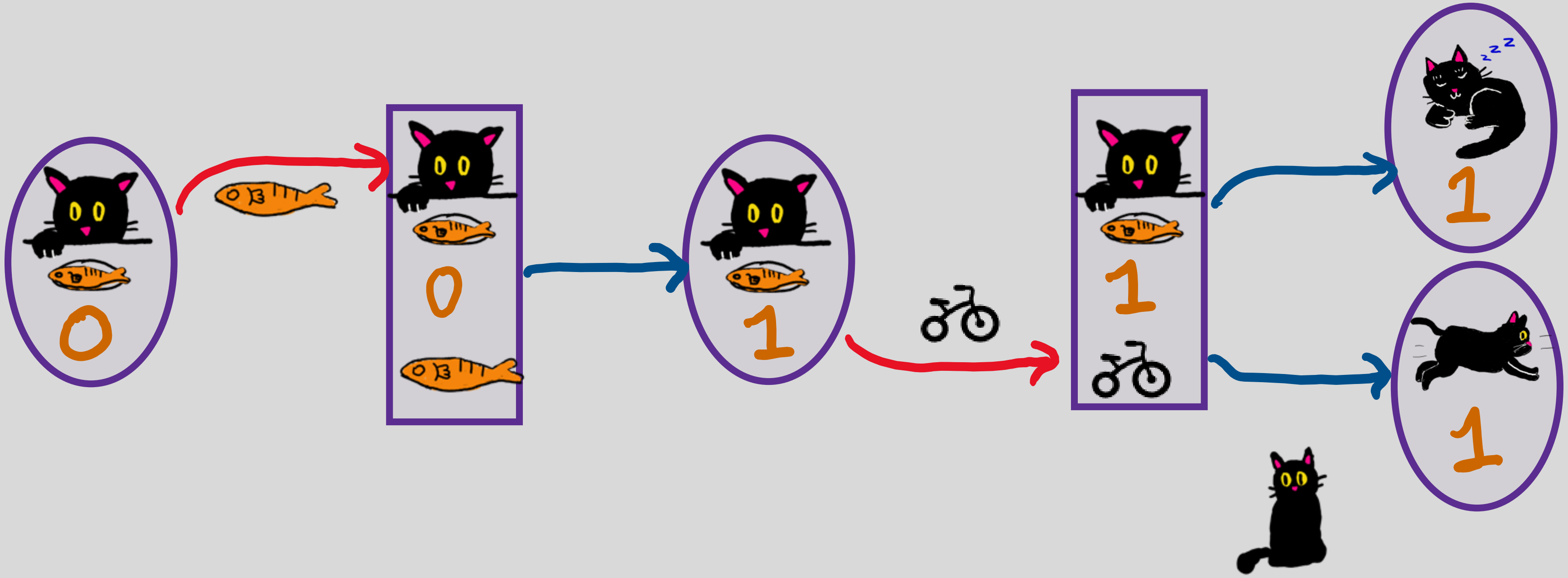


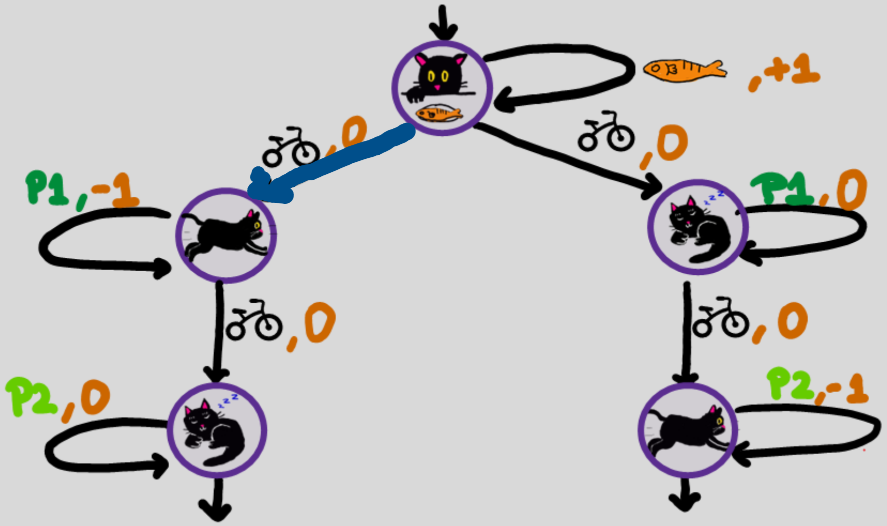
$L = \{ \text{🐟}^{i} \; \text{🚲} \; P1^{j} \; \text{🚲} \; P2^{k} \mid i \geq j \text{ or } i \geq k \}$



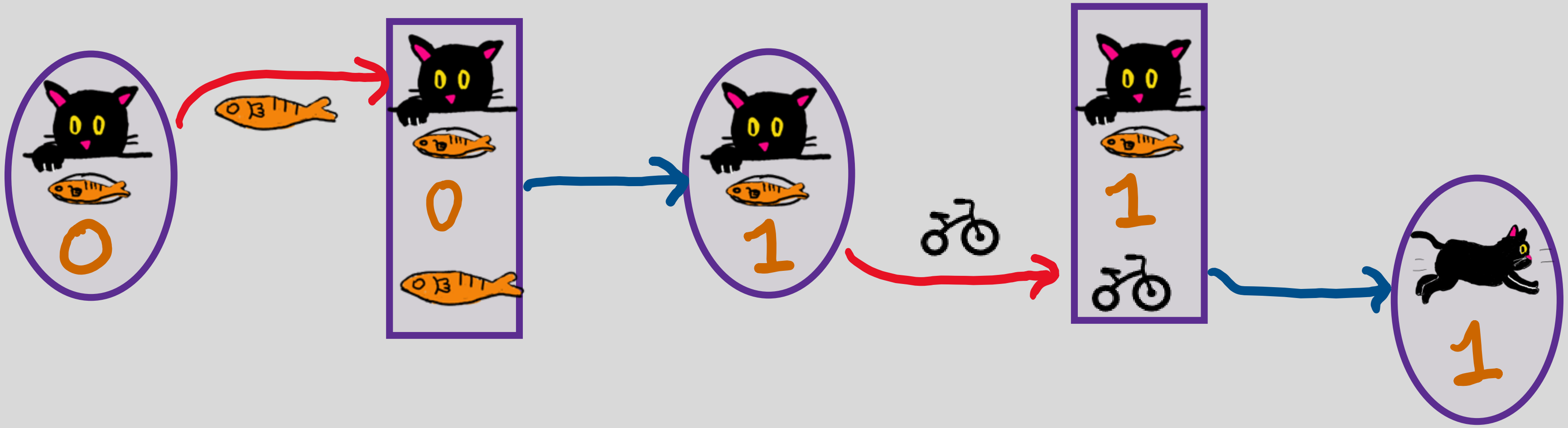


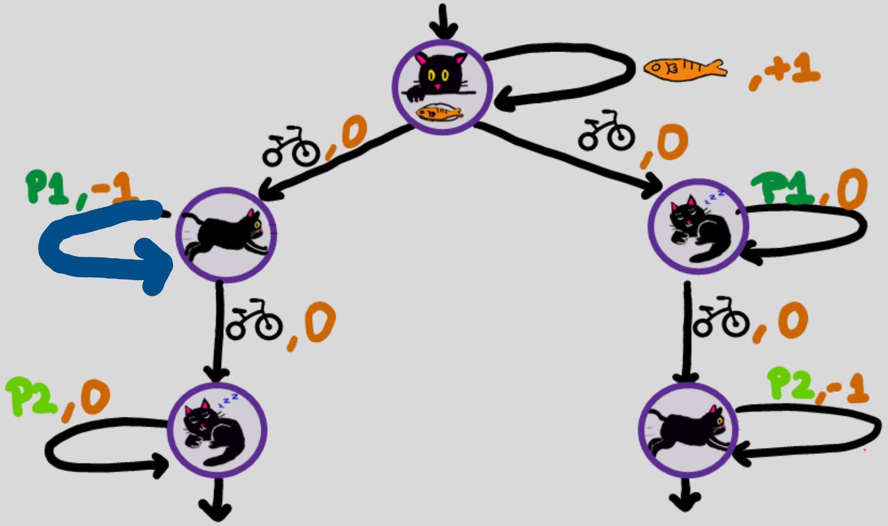
L = {🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$}



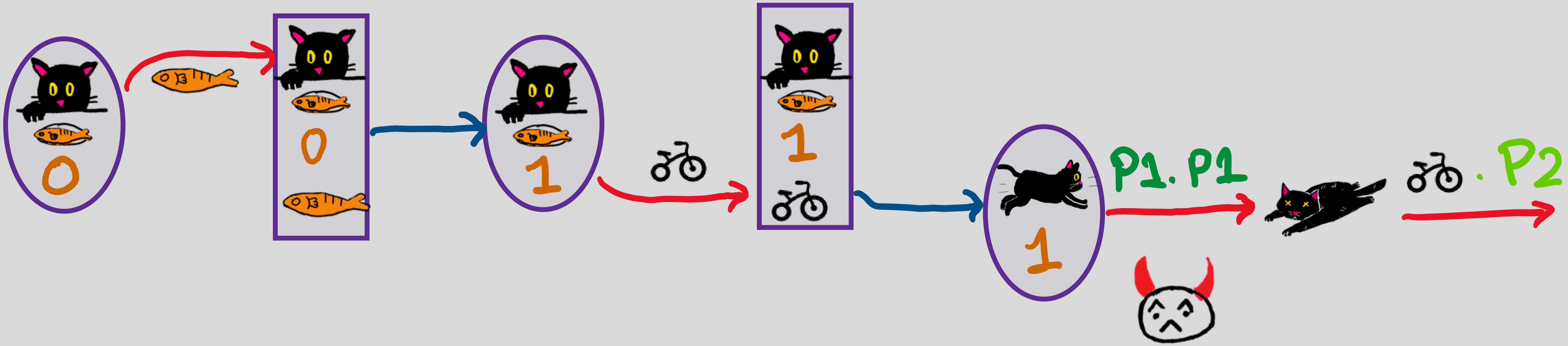


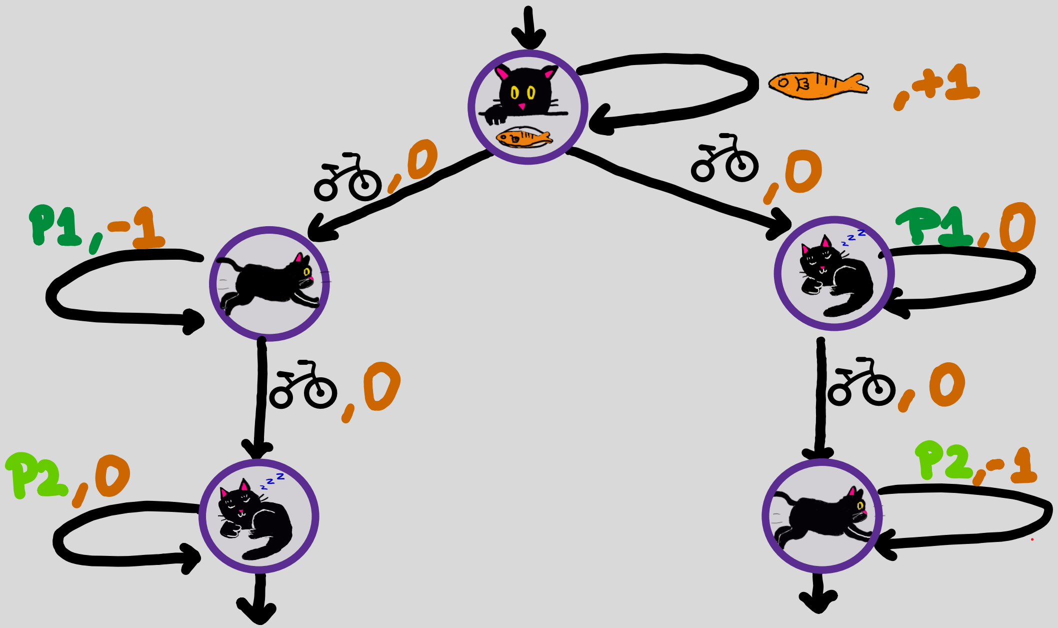
L = {🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$}



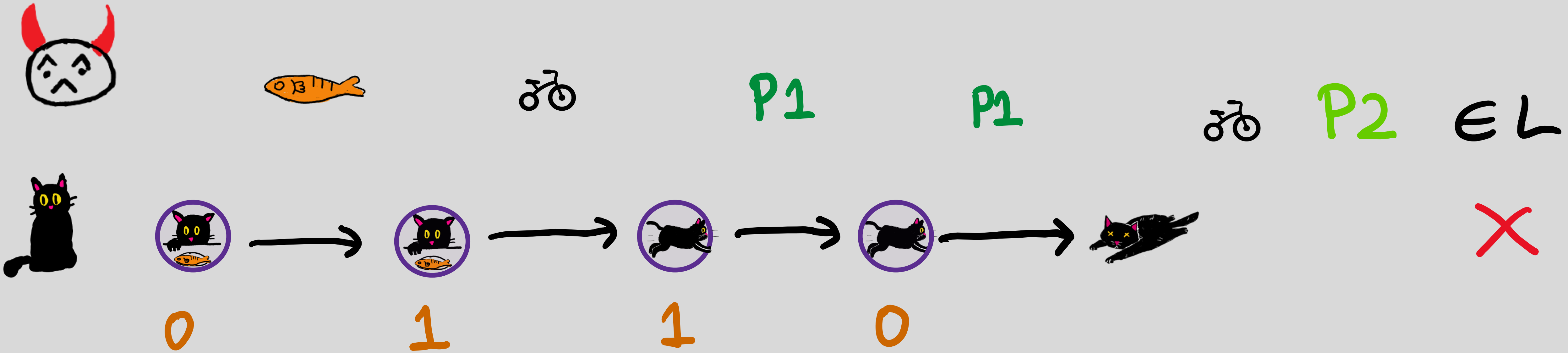


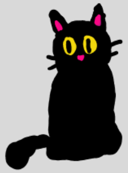
L = { 🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$ }





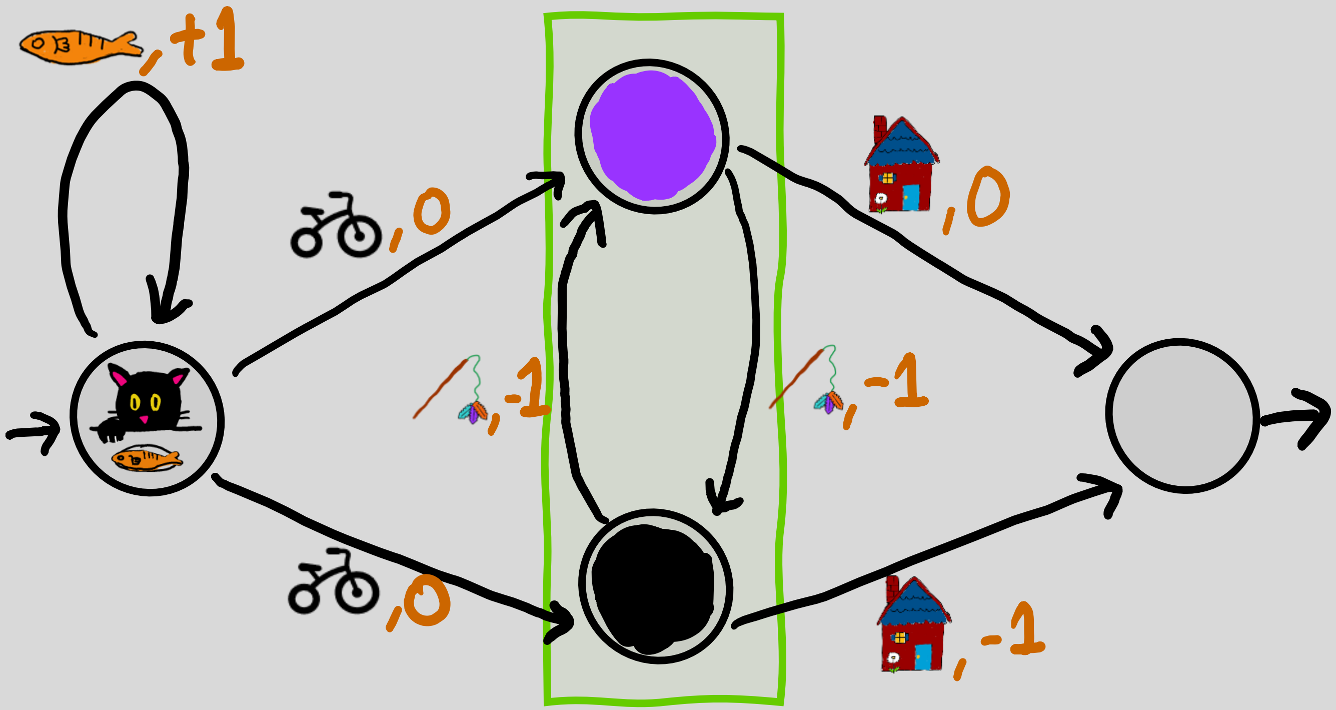
L = {🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$}



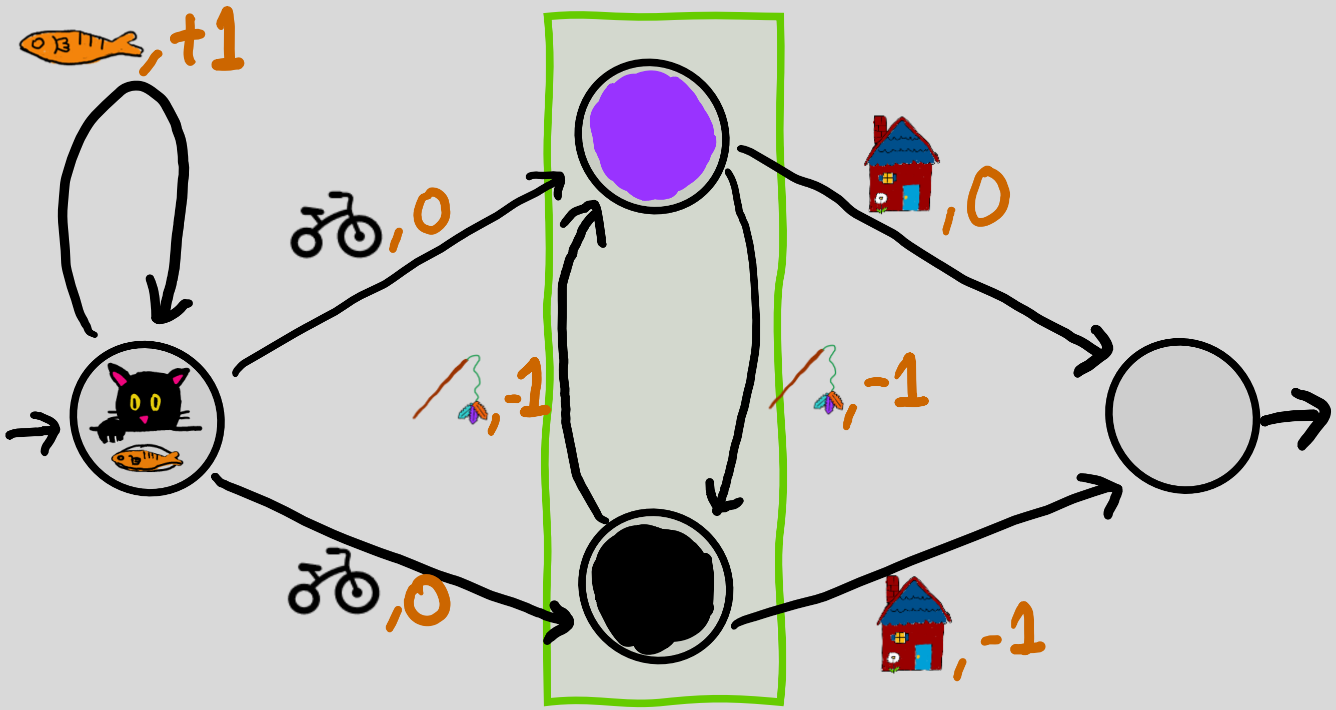
Definition: An one-counter net N is HD iff  wins HD-game.

Resolver: A winning strategy of  .

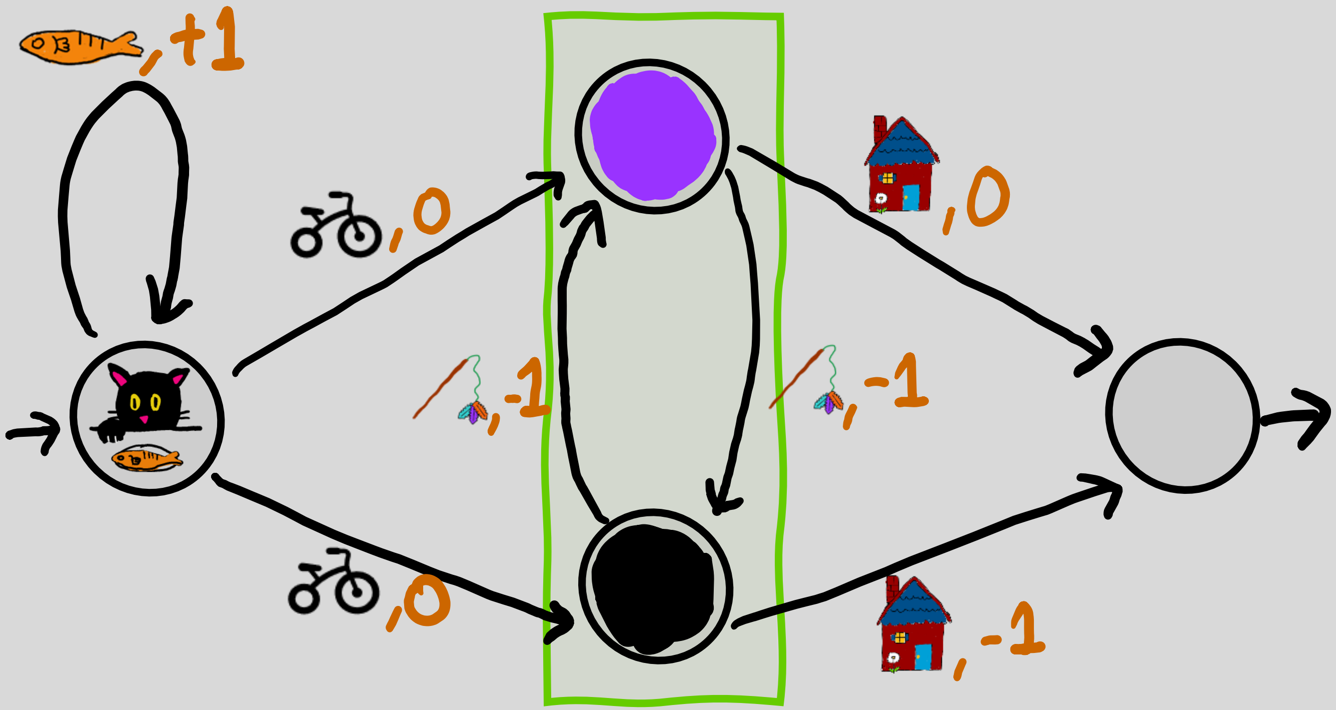
2. Resolvers



,+1
,0
,-1
,0
,-1
,0
,0
,-1

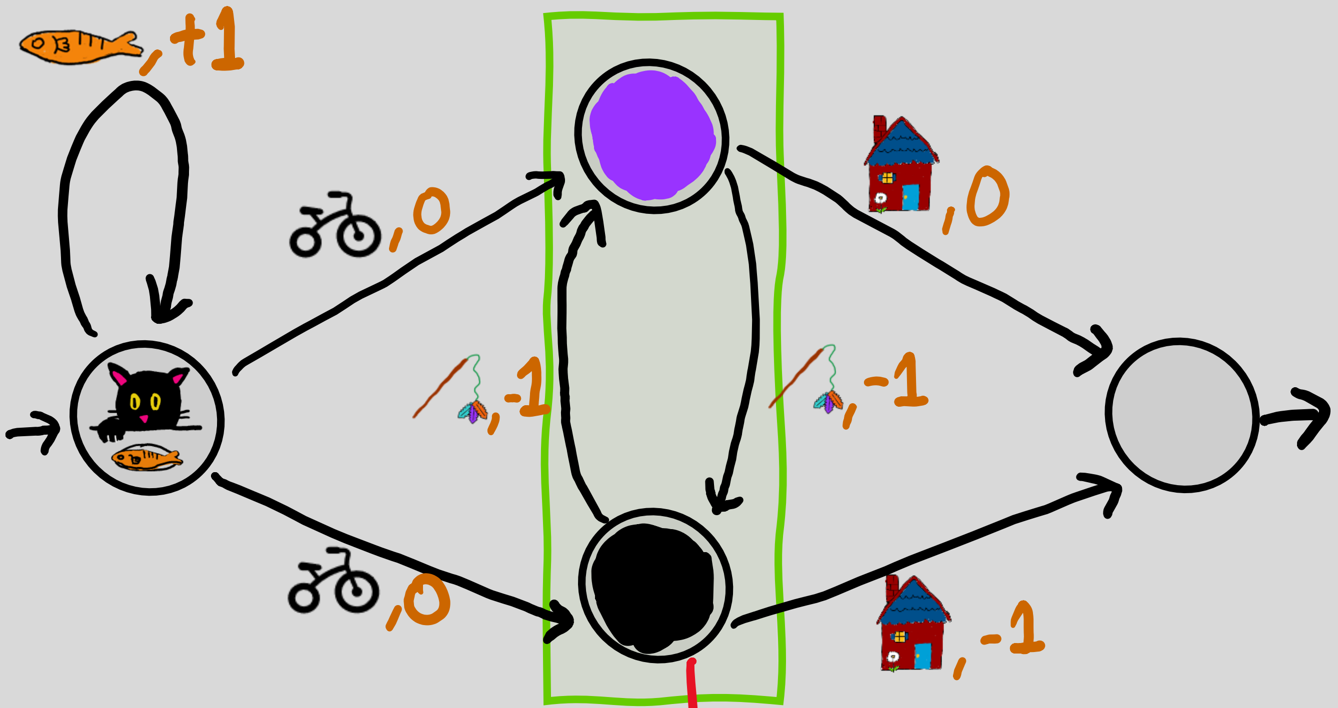


$L = \{$ 🐟^{n} 🚲 🎣^{m} 🏠 $\mid n \geq m \}$



At 🐱 , on 🚲 :

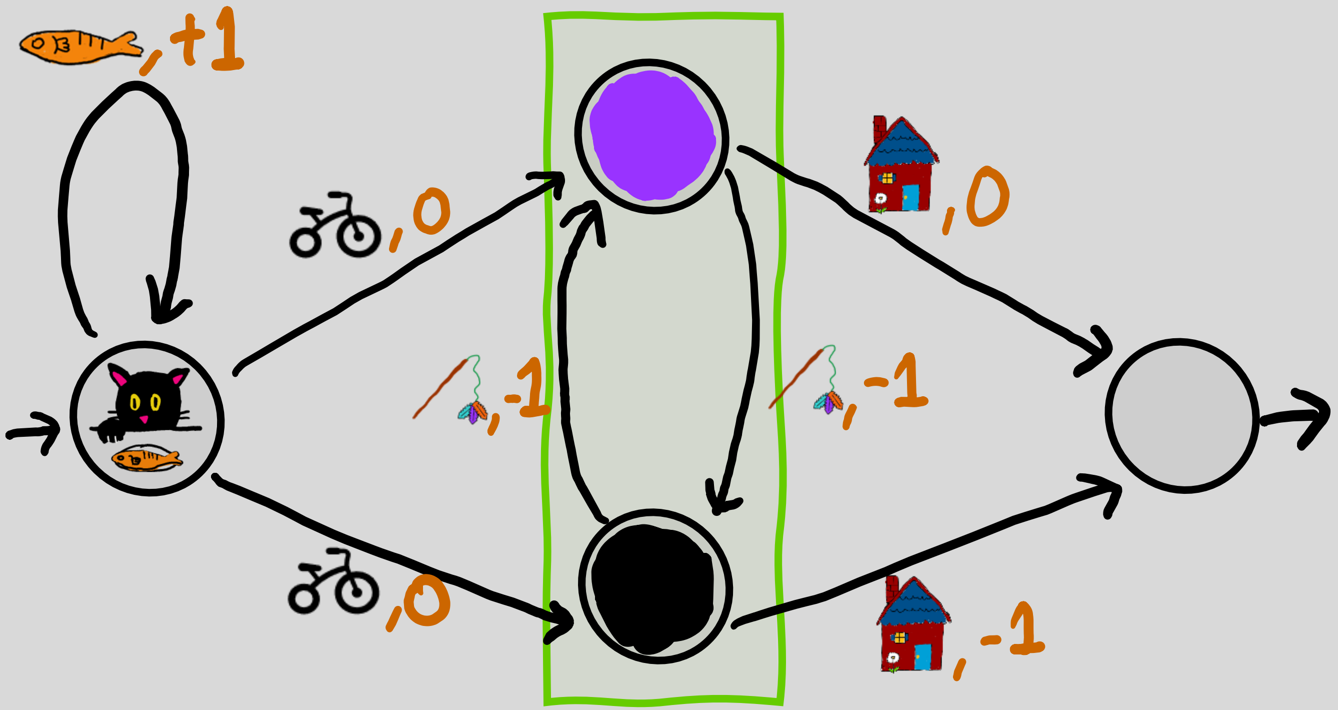
$L = \{$ 🐟^{n} 🚲 🎣^{m} 🏠 $\mid n \geq m \}$



Don't be here with counter 0

At 🐈, on 🚲 :

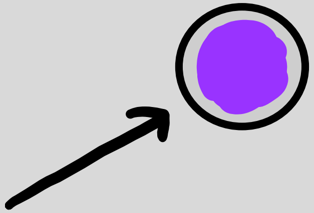
$L = \{$ 🐟^{n} 🚲 🎣^{m} 🏠 $| n \geq m \}$



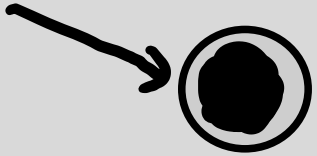
At

, on 🚲 :

Counter = 2n:



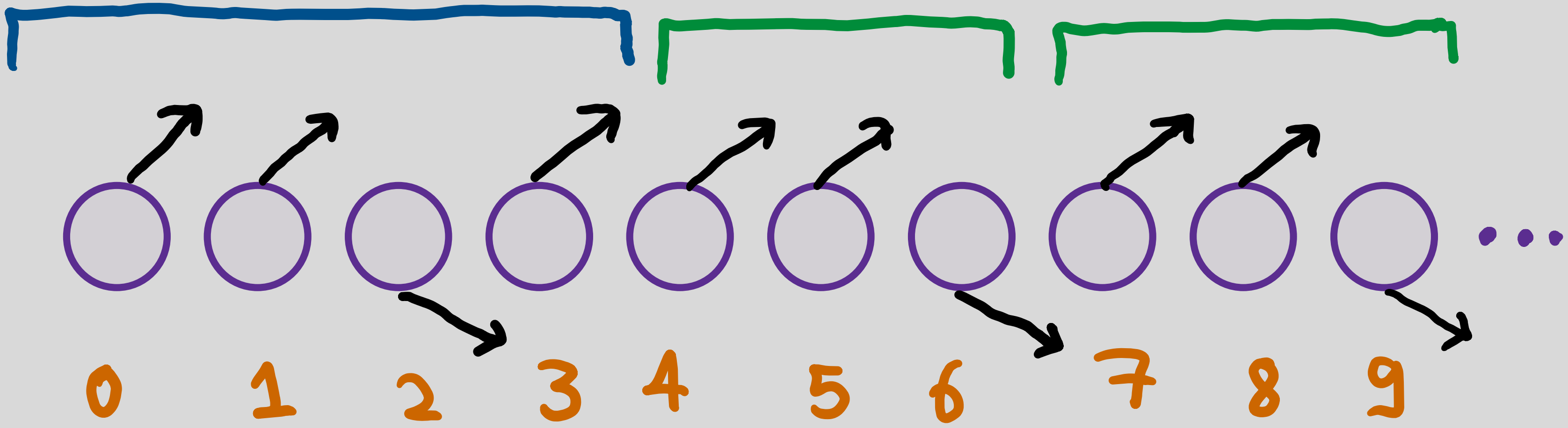
Counter = 2n + 1:



$L = \{$ 🐟^{n} 🚲 🎣^{m} 🏠 $\mid n \geq m \}$

Theorem: Given an history-deterministic OCN, there is a semilinear strategy for  that is effectively computable.

Theorem: Given an history-deterministic OCN, there is a semilinear strategy for  that is effectively computable.



Theorem: Given an history-deterministic OCN, there is a semilinear strategy for  that is effectively computable.

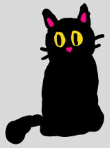
Proof:

HD Game $\Longleftrightarrow$ Token Game $\Longrightarrow$ Simulation Game

[Boker, Lehtinen'22] [Our work!!]

Token Game:

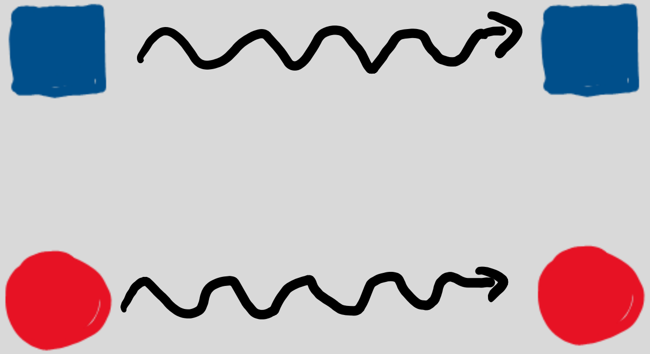
 : 🔴

 : 🟦

Token Game:

😈 : 🔴 🐈‍⬛ : 🟦

1. 😈 selects letter
2. 🐈‍⬛ selects transition on 🟦
3. 😈 selects transition on 🔴



Token Game:

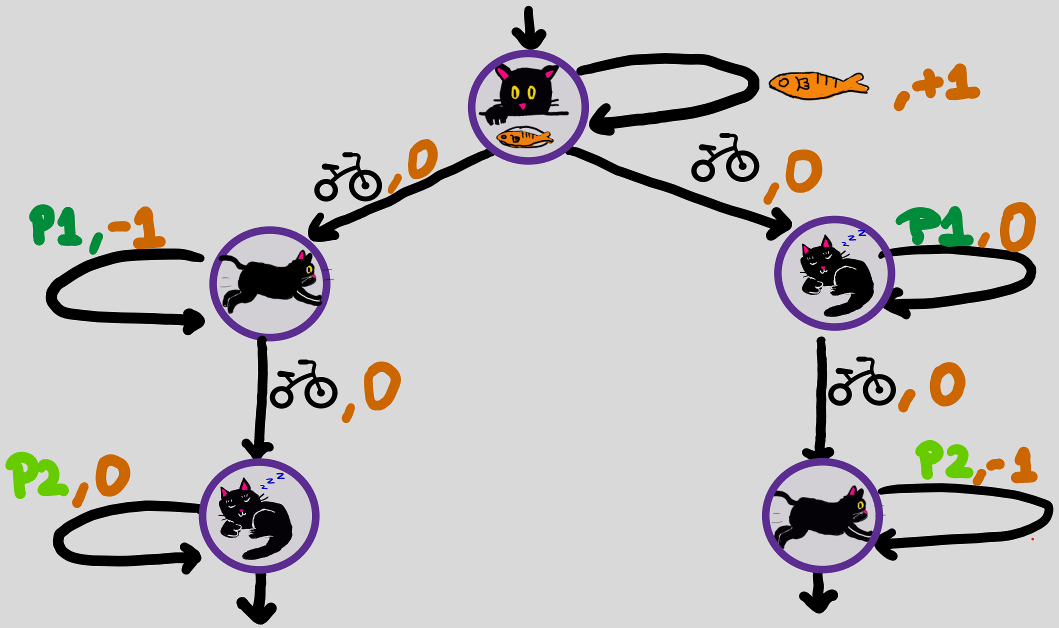
😈 : 🔴   🐈‍⬛ : 🟦

🟦 ⇝ 🟦

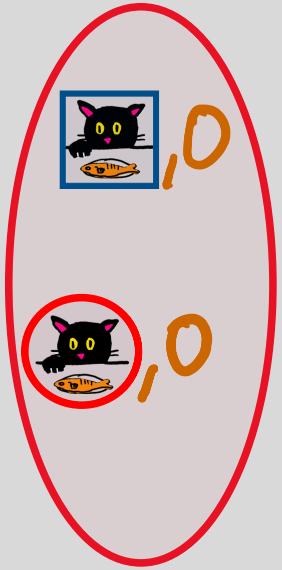
🔴 ⇝ 🔴

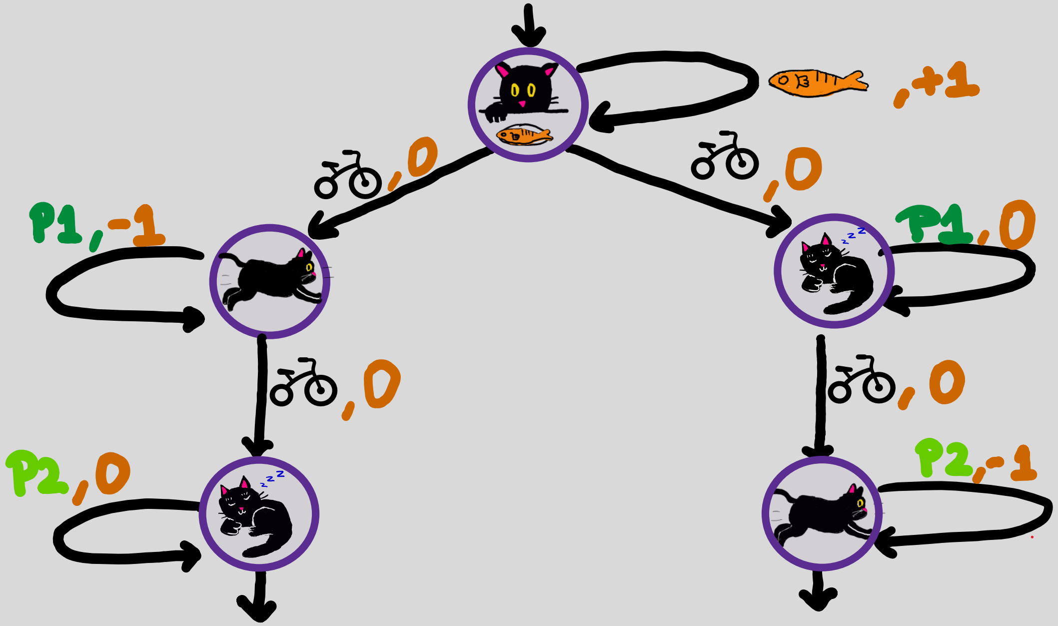
1. 😈 selects letter
2. 🐈‍⬛ selects transition on 🟦
3. 😈 selects transition on 🔴

Winning condition of 😈: If 😈's run on 🔴 is accepting and 🐈‍⬛'s run on 🟦 is rejecting.

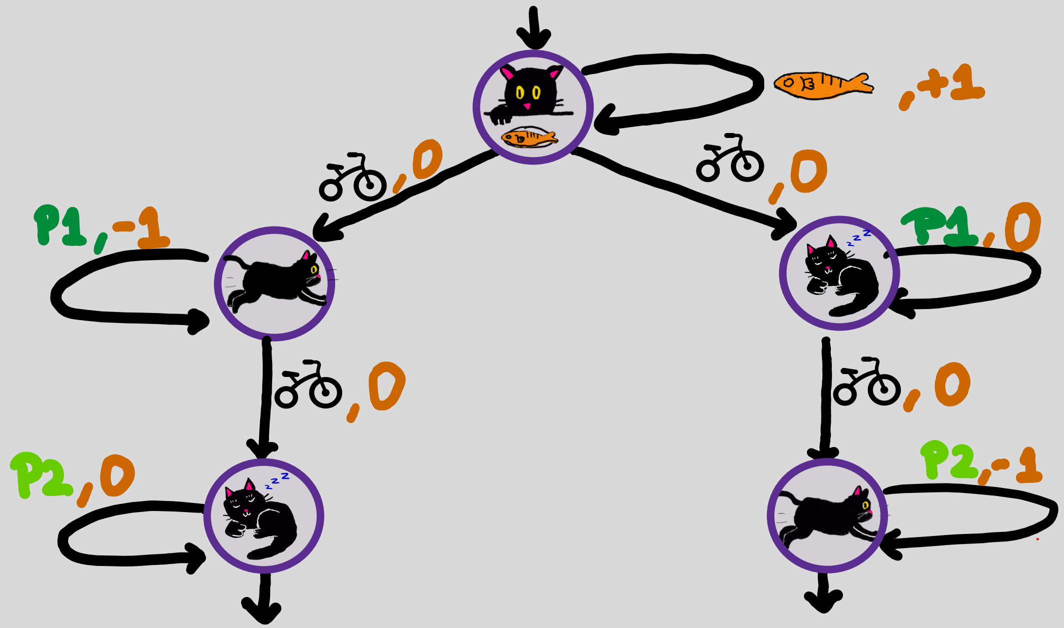


$$L = \{ 🐟^{i}\ 🚲\ P1^{j}\ 🚲\ P2^{k} \mid i \geq j \text{ or } i \geq k \}$$

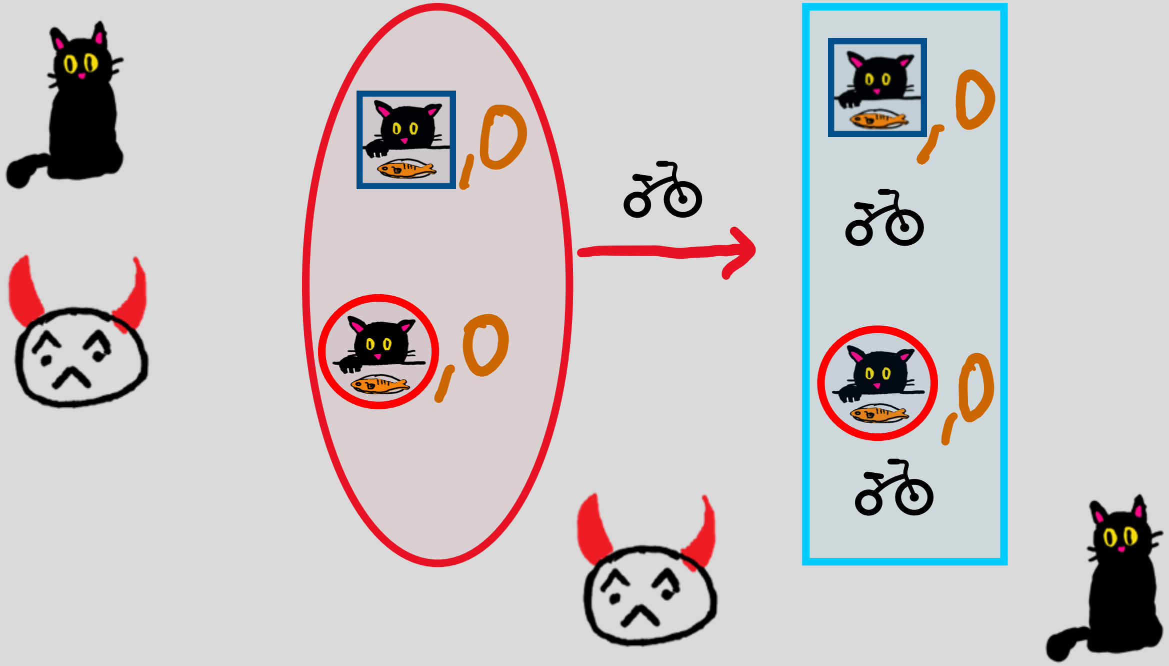


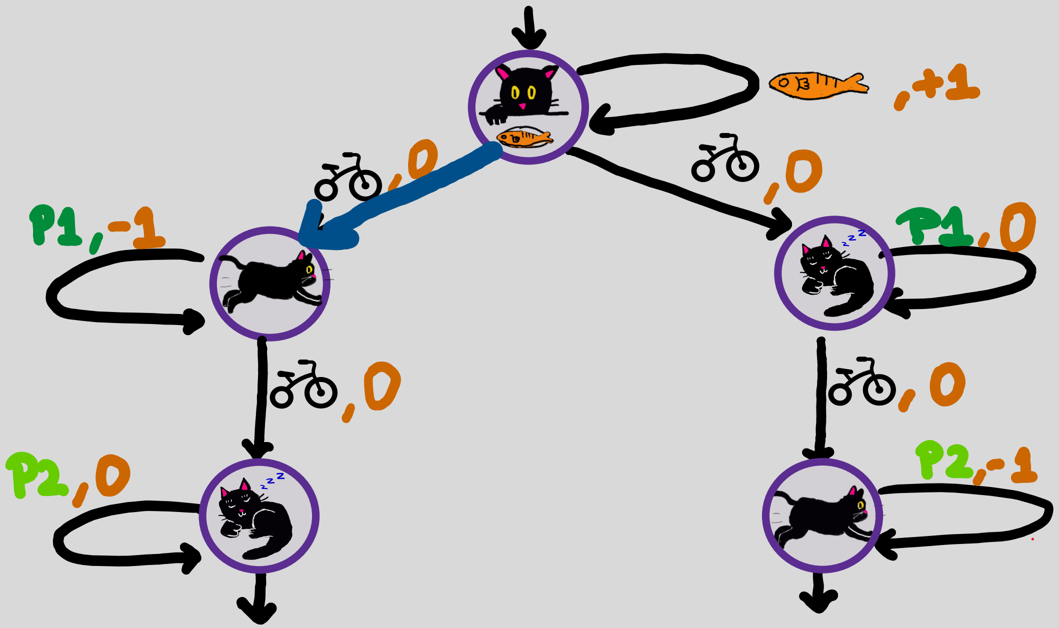


$L = \{$ 🐟^{i} 🚲 $P1^j$ 🚲 $P2^k \mid i \geq j$ or $i \geq k \}$

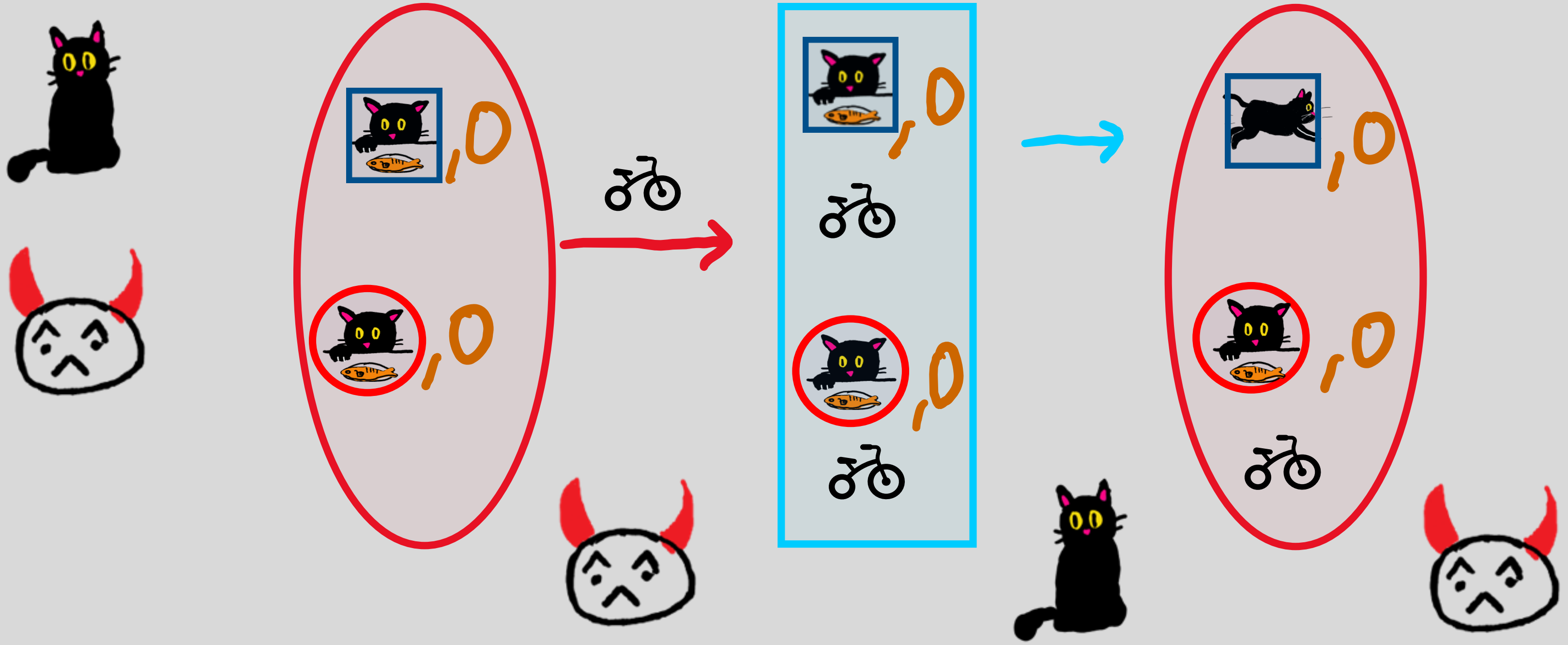


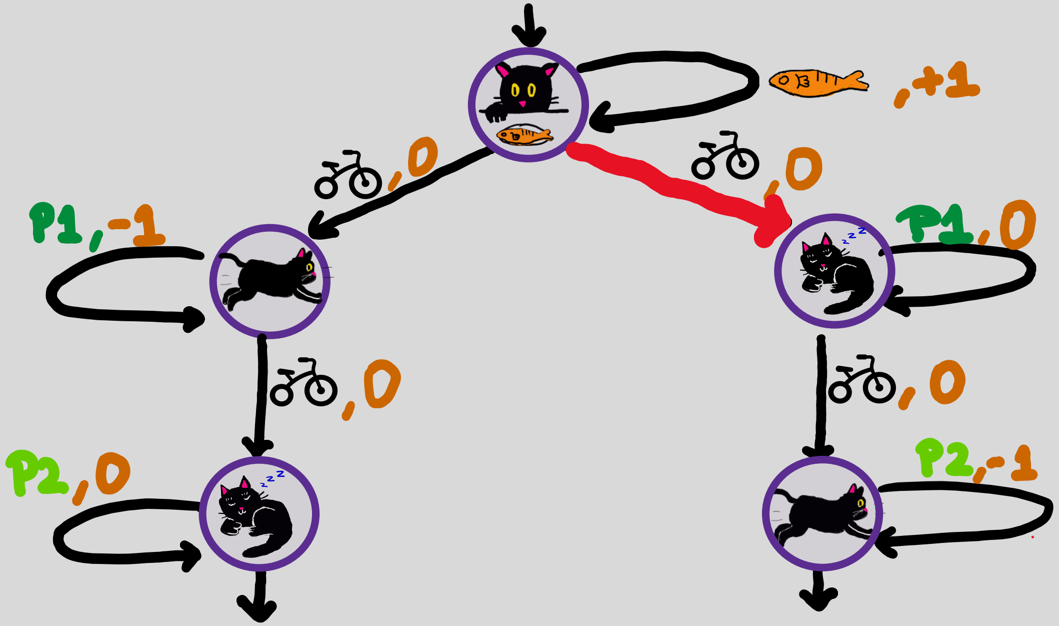
$L = \{$🐟^{i} 🚲 $P1^{j}$ 🚲 $P2^{k} \mid i \geq j$ or $i \geq k\}$



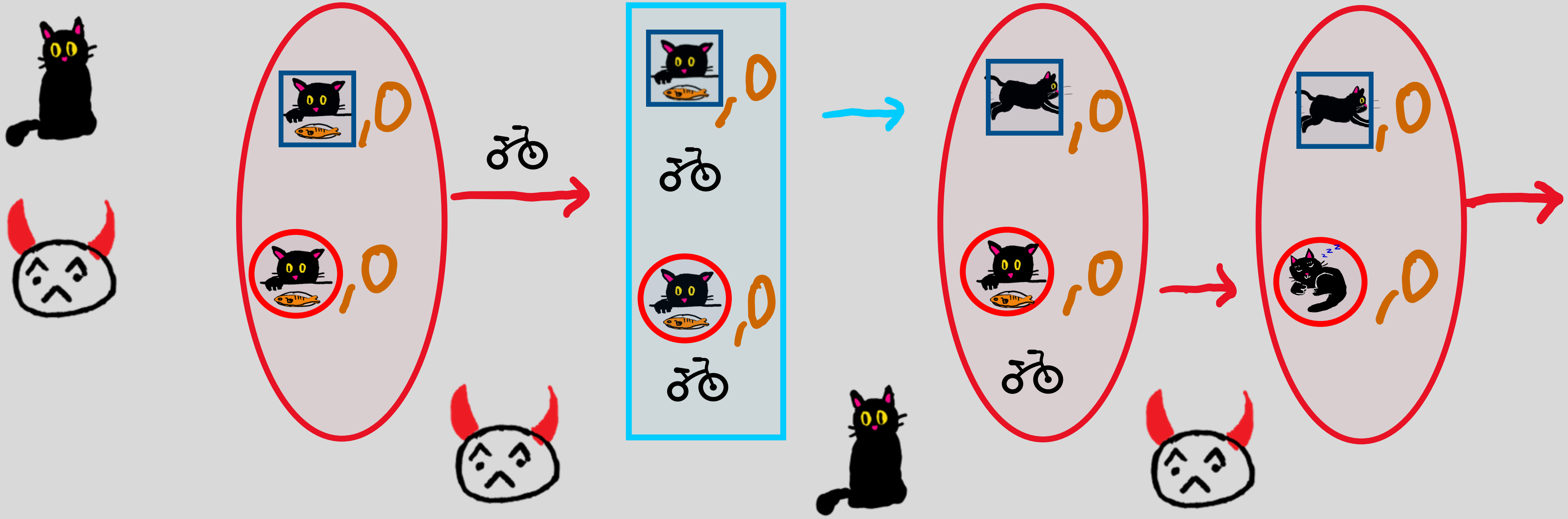


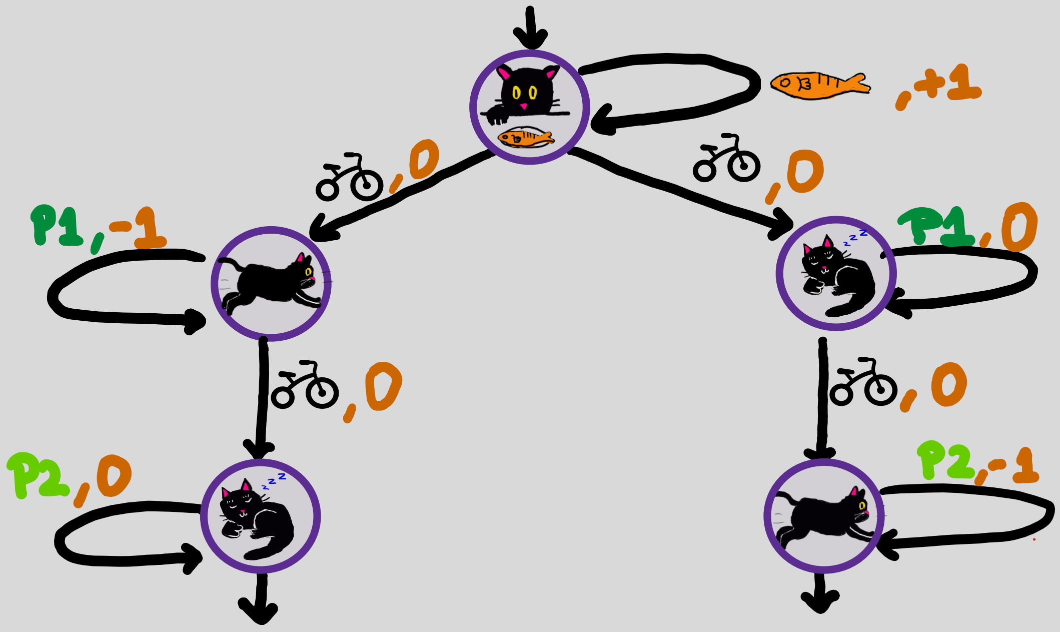
L = { 🐟^{i} 🚲 P1^{j} 🚲 P2^{k} | i ≥ j or i ≥ k }



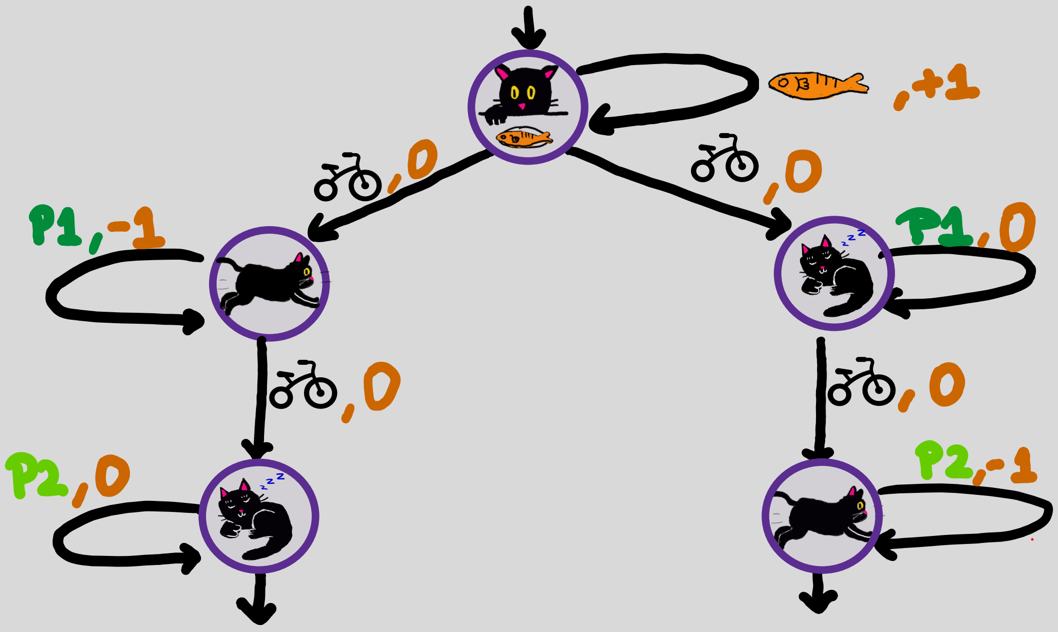


$L = \{$ 🐟^{i} 🚲 P1^{j} 🚲 P2^{k} $\mid i \geq j$ or $i \geq k \}$

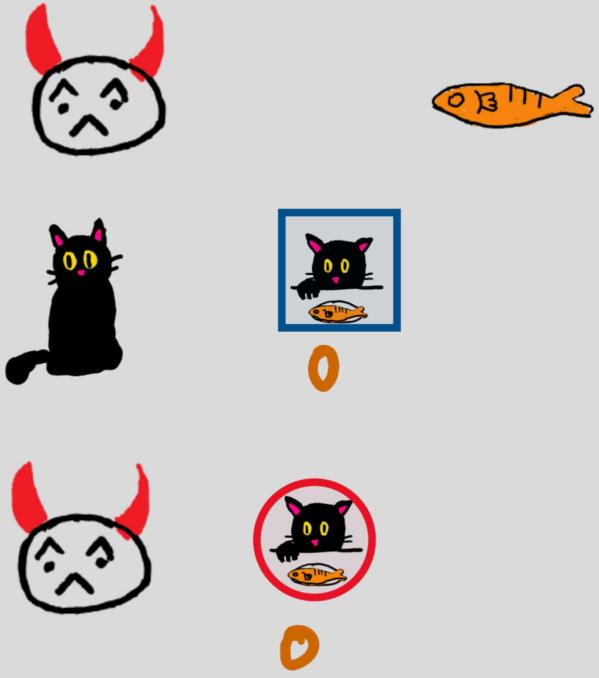


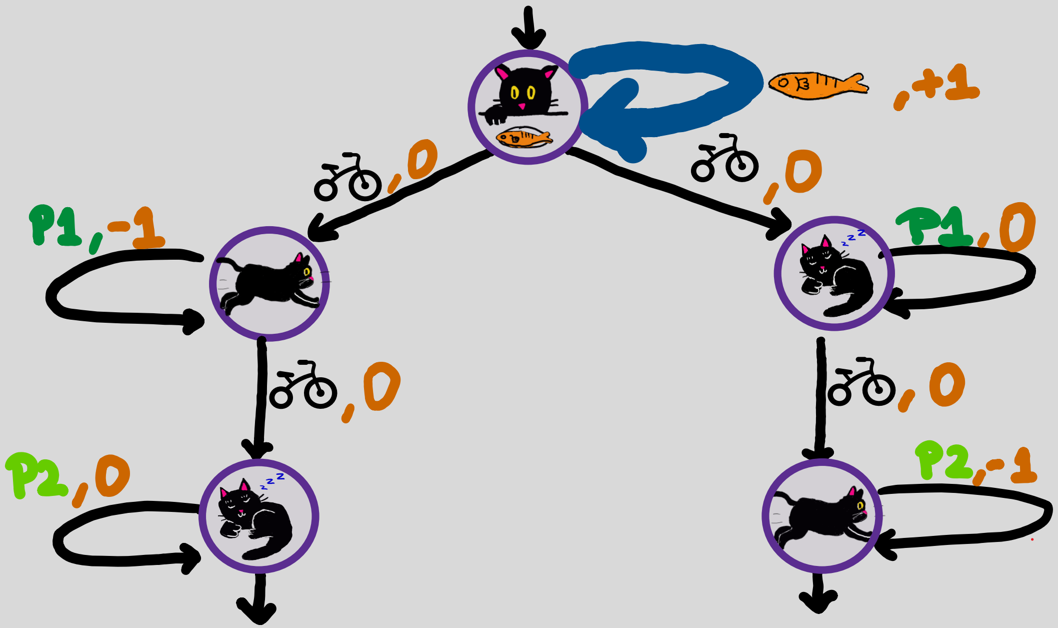


$L = \{ 🐟^{i} \; 🚲 \; P1^{j} \; 🚲 \; P2^{k} \mid i \geq j \text{ or } i \geq k \}$

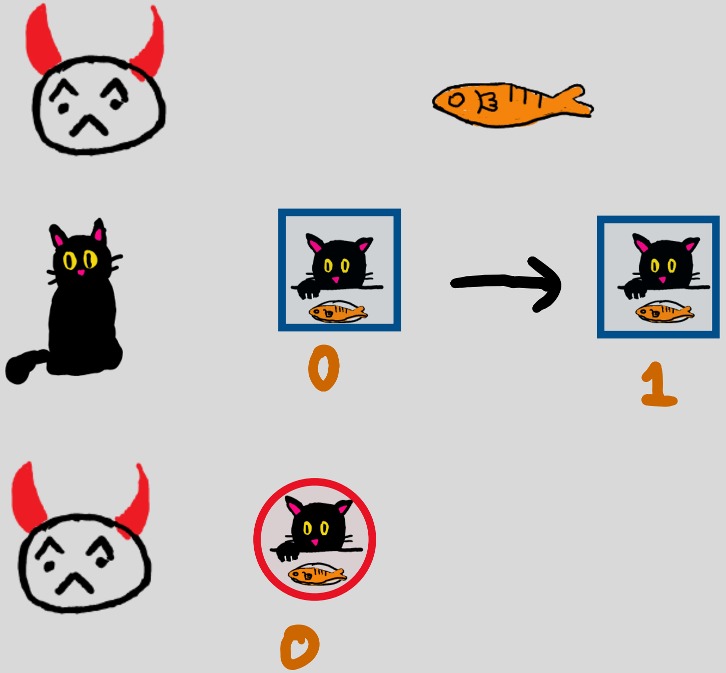


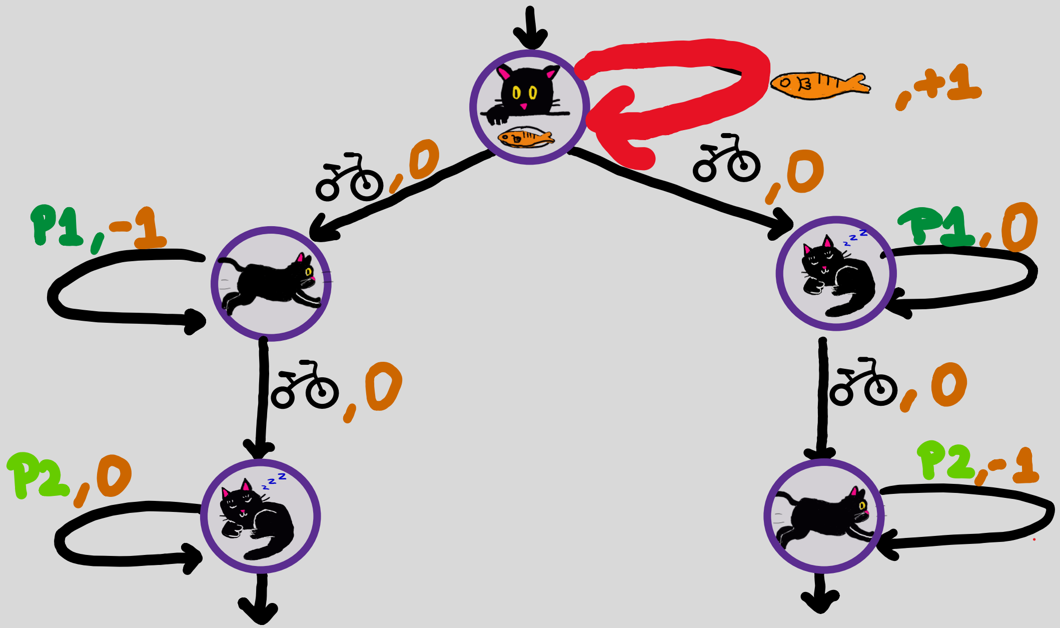
$L = \{ \text{🐟}^{i}\ \text{🚲}\ P1^{j}\ \text{🚲}\ P2^{k} \mid i \geq j \text{ or } i \geq k \}$

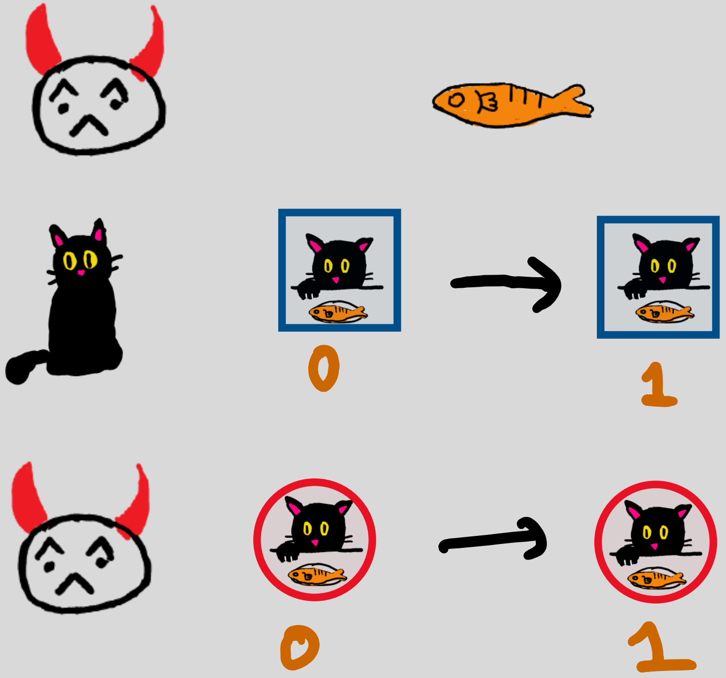




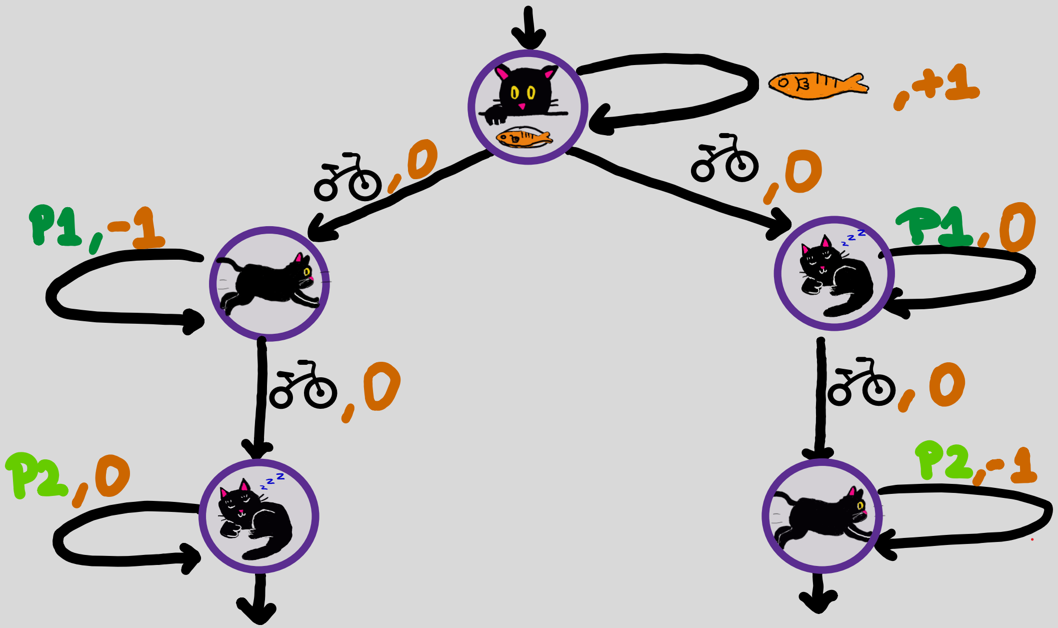
$L = \{$ 🐟^{i} 🚲 P1^{j} 🚲 P2^{k} $\mid i \geq j$ or $i \geq k \}$



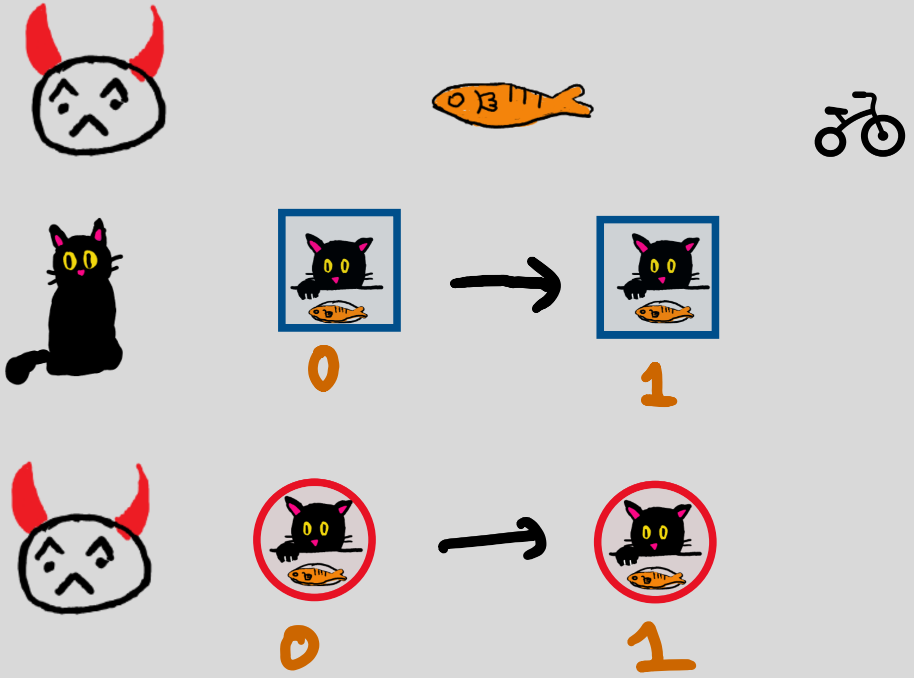


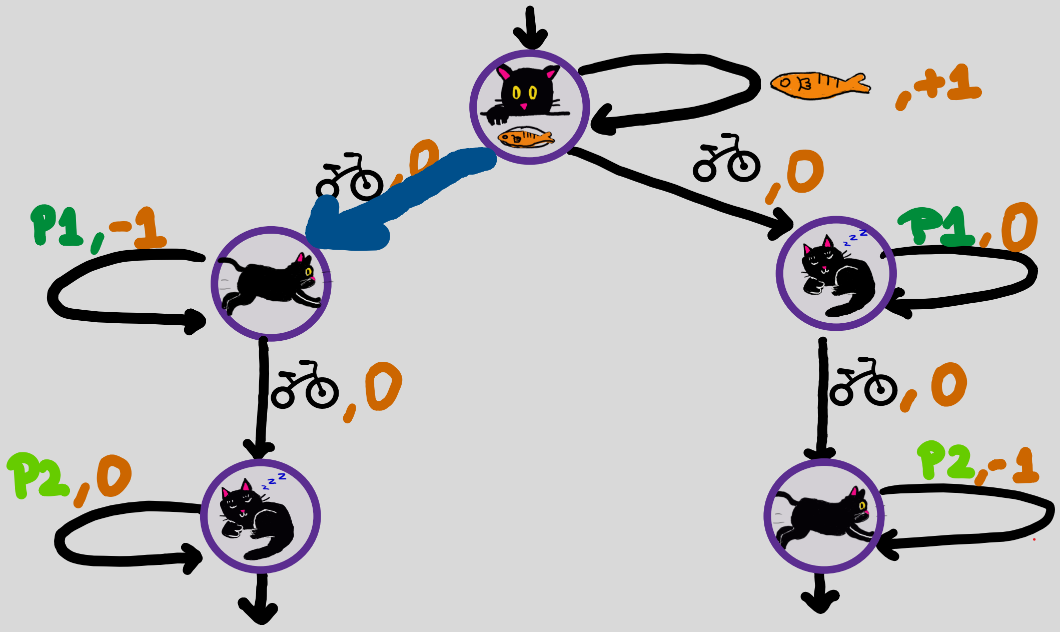


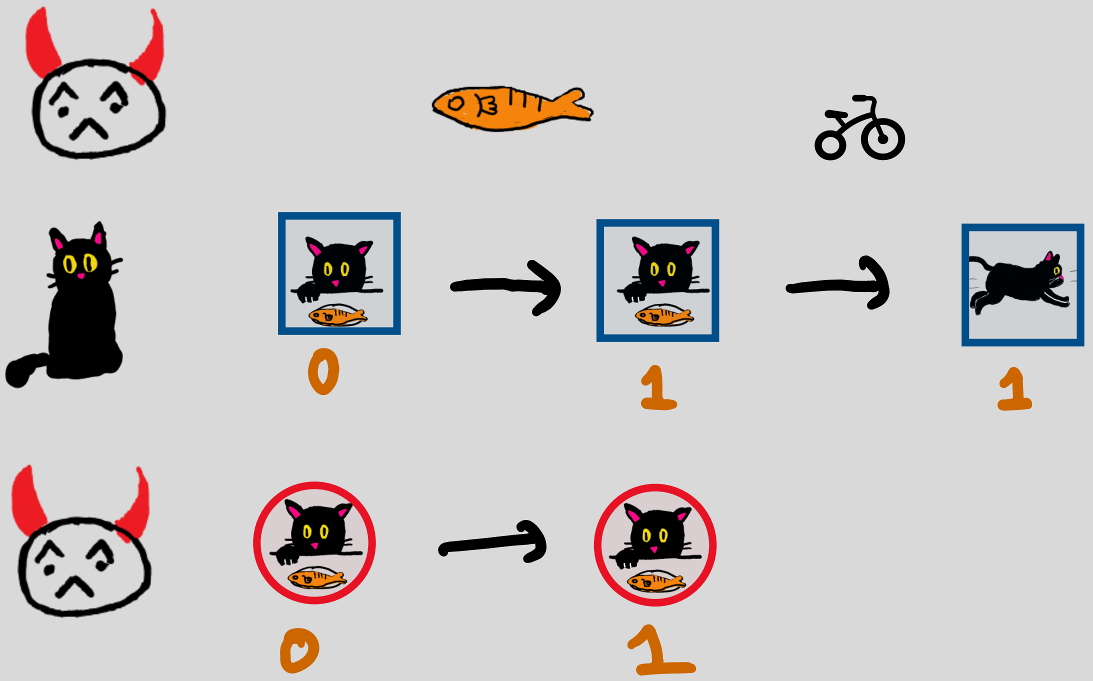
L = {🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$}



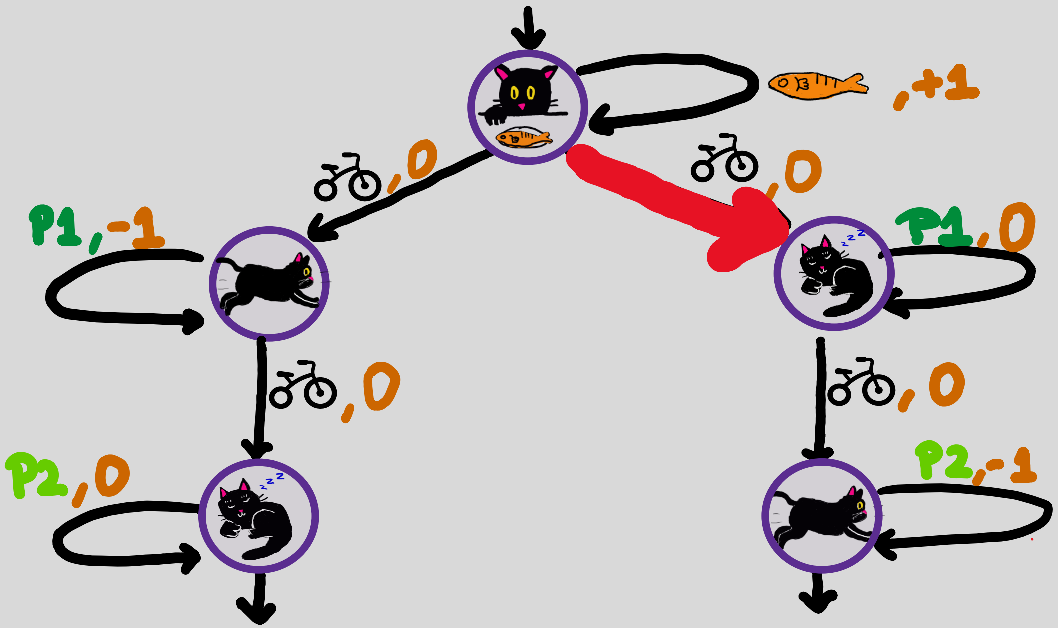
L = { 🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$ }



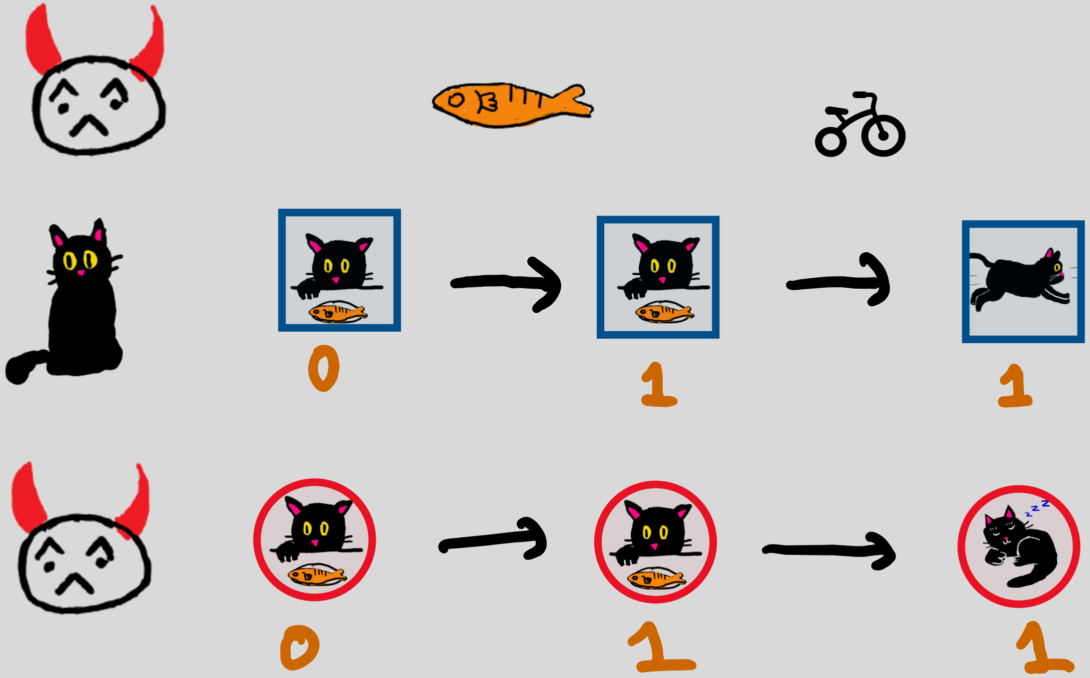


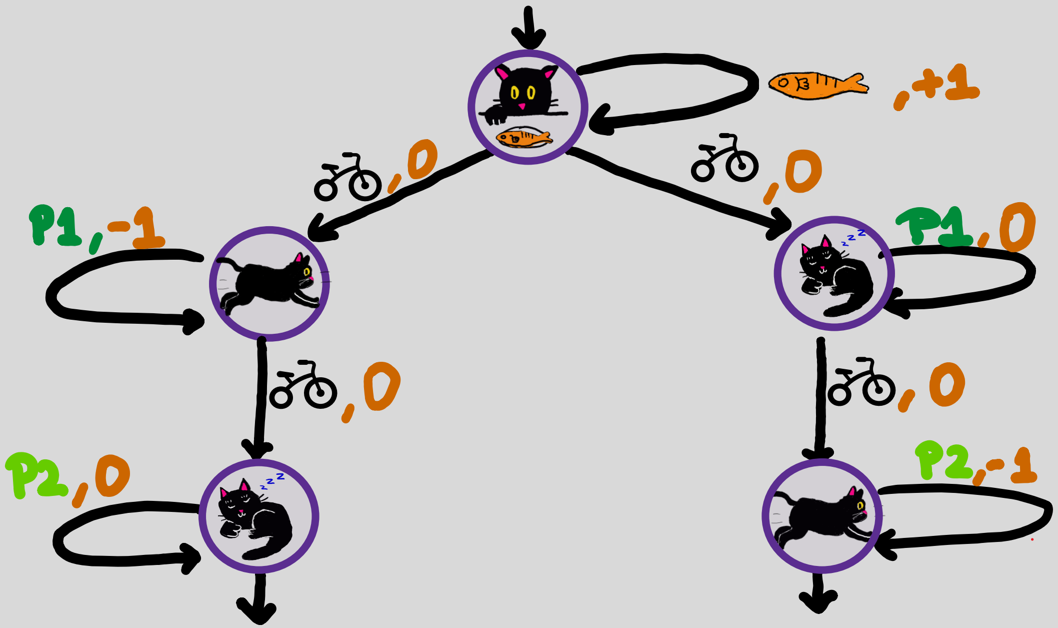


L = {🐟^{i} 🚲 P1^j 🚲 P2^k | i ≥ j or i ≥ k}

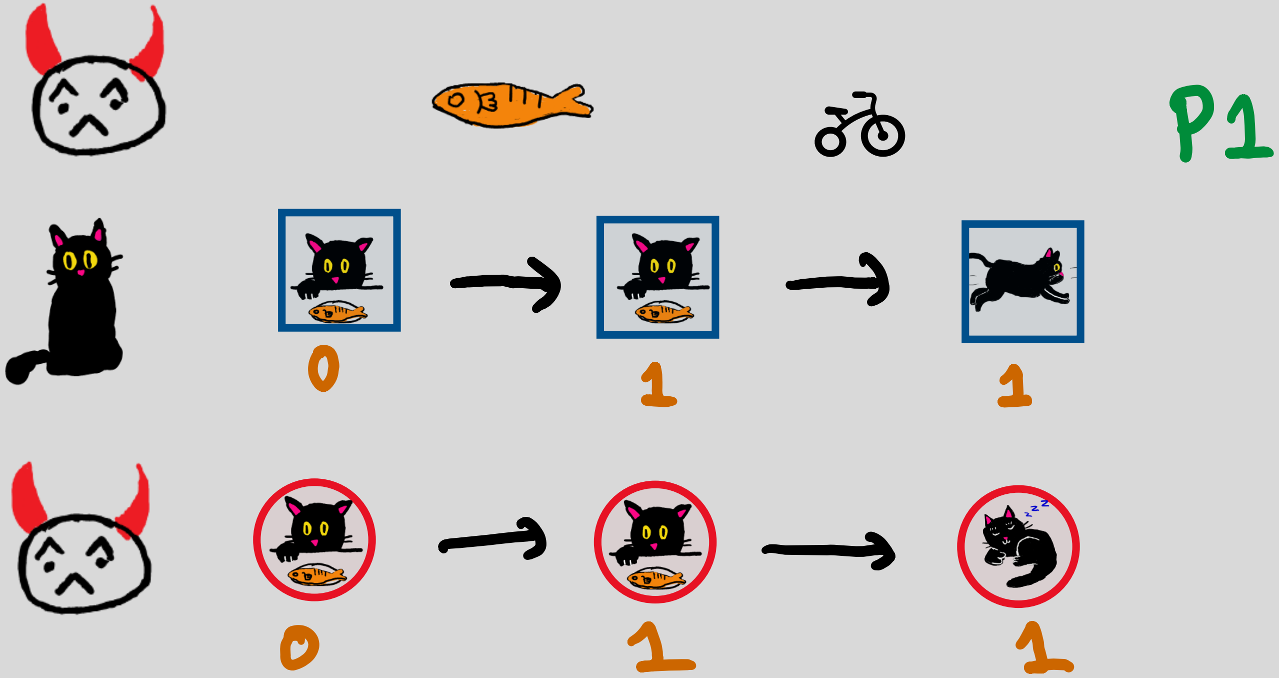


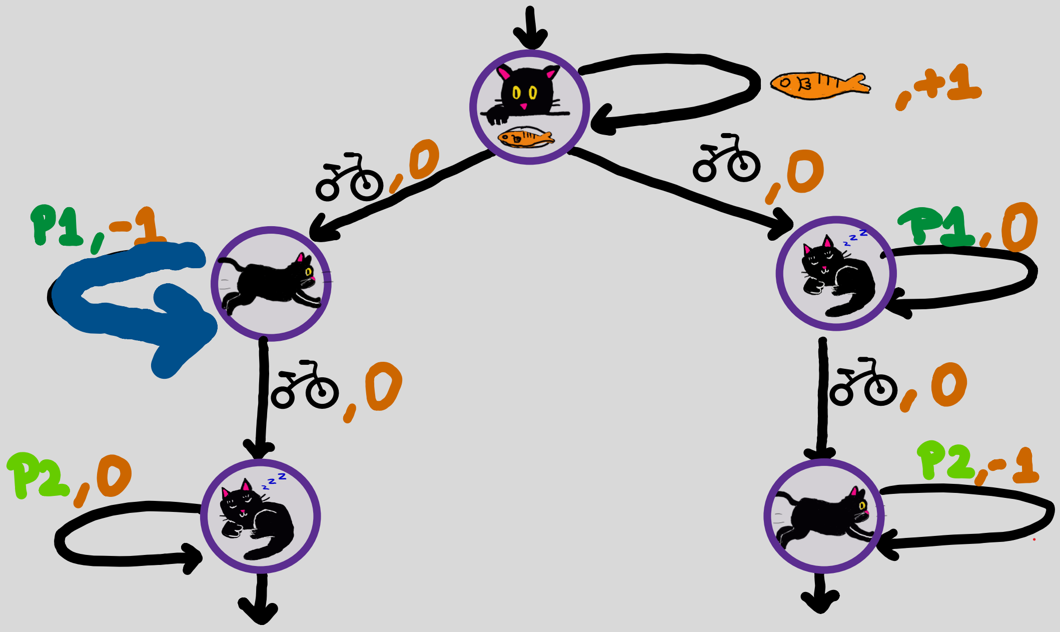
L = {🐟^{i} 🚲 P1^j 🚲 P2^k | i ≥ j or i ≥ k}



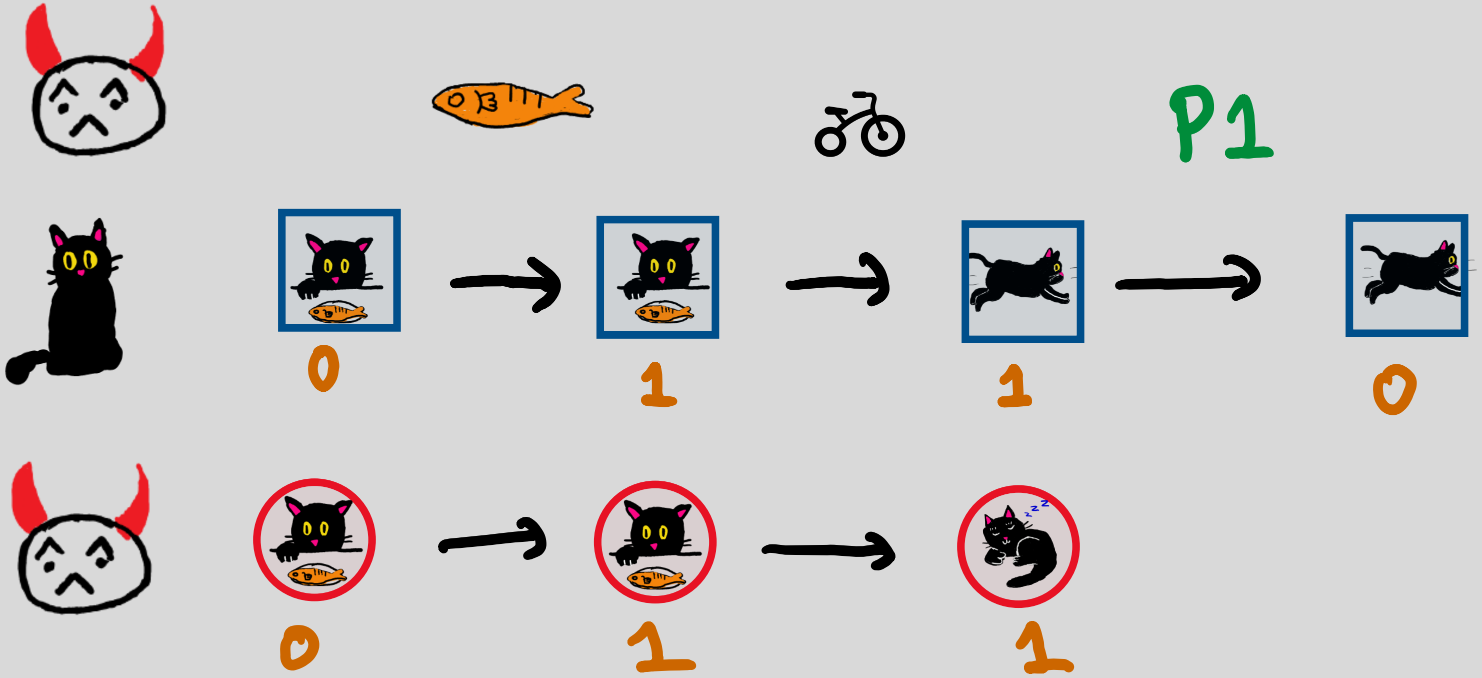


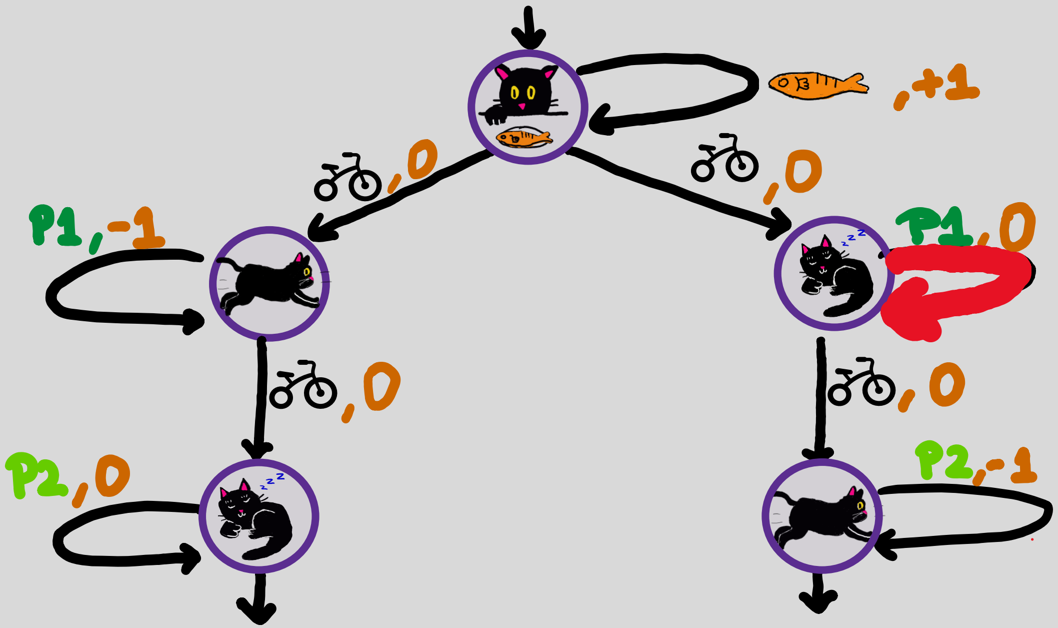
L = {🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$}



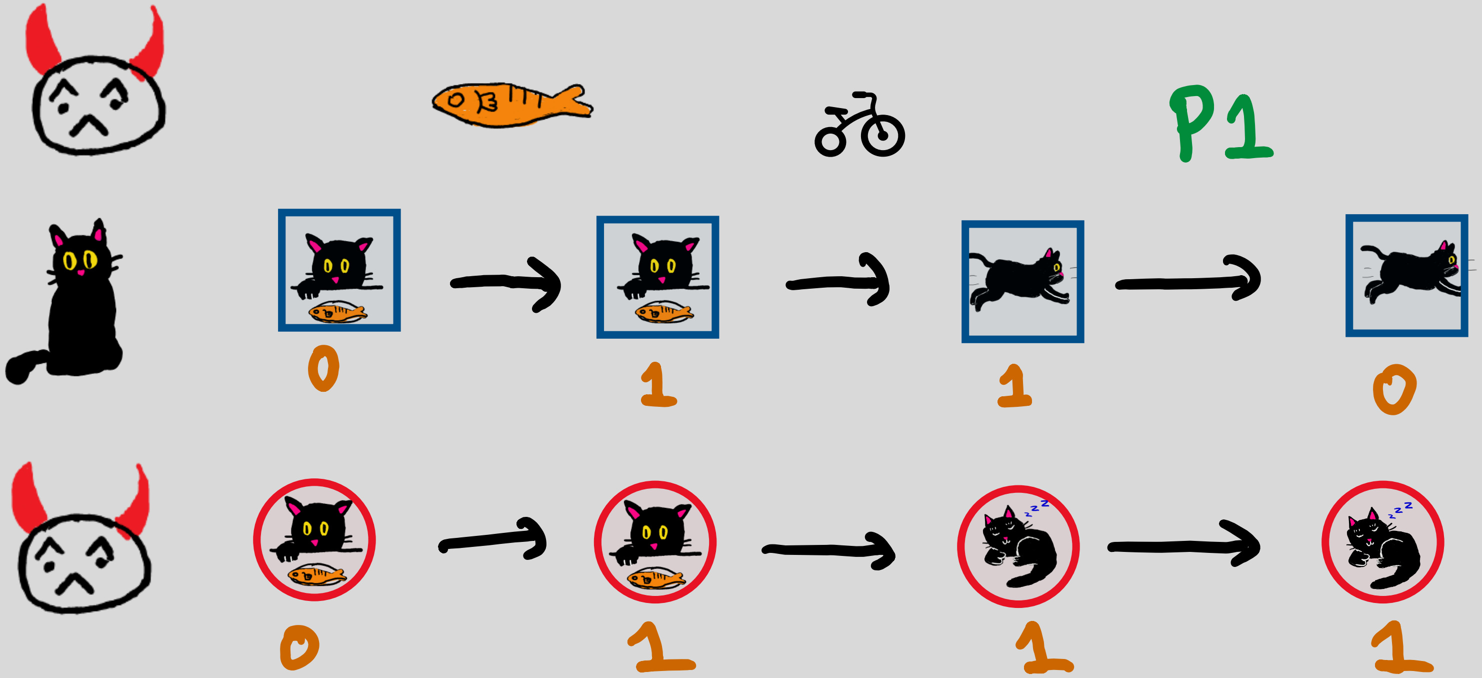


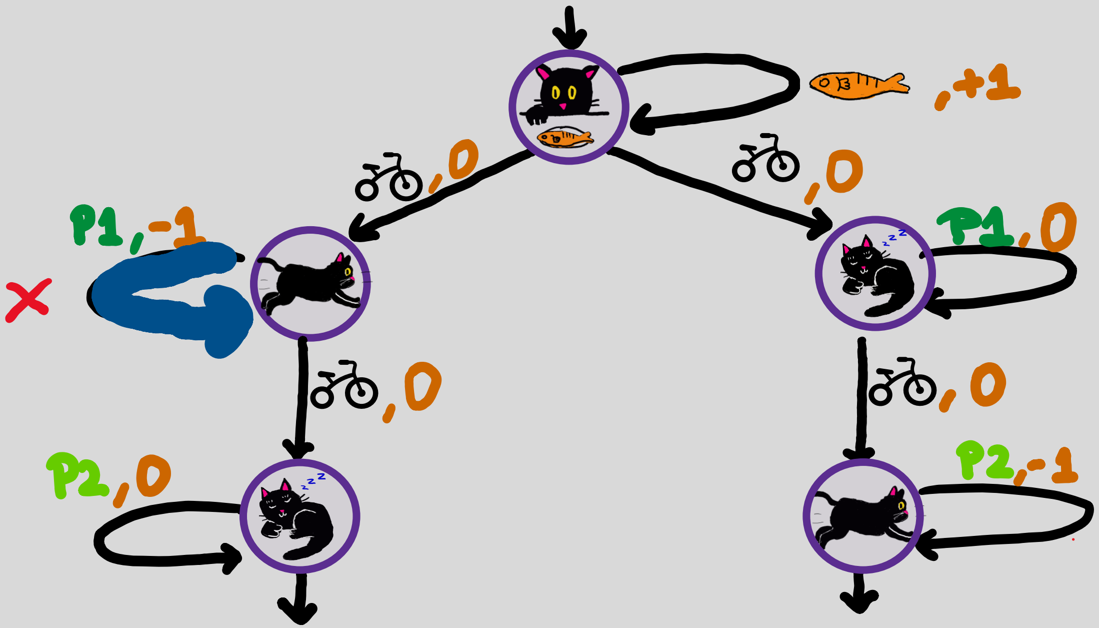
L = {🐟^{i} 🚲 P1^j 🚲 P2^k | i ≥ j or i ≥ k}



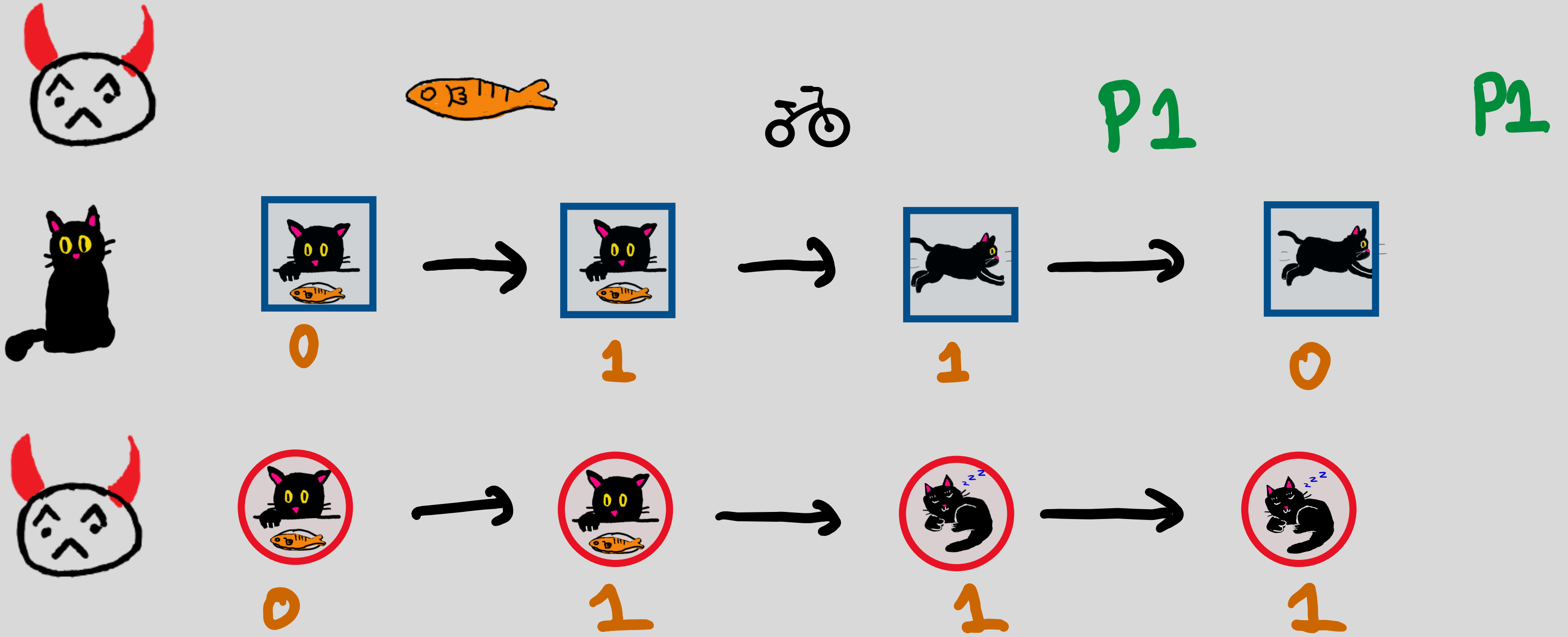


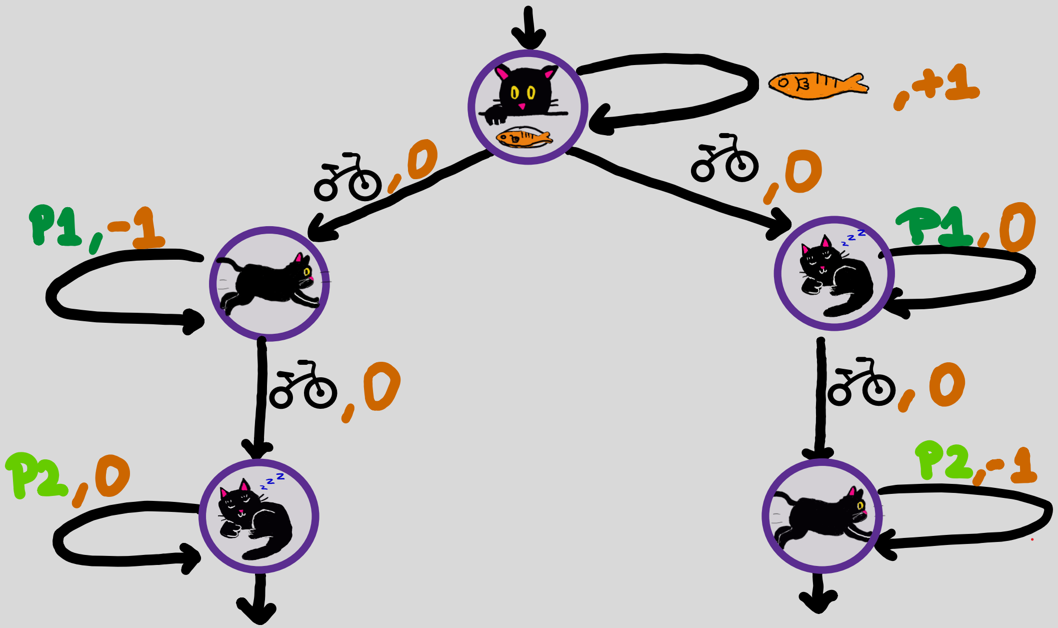
L = {🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$}



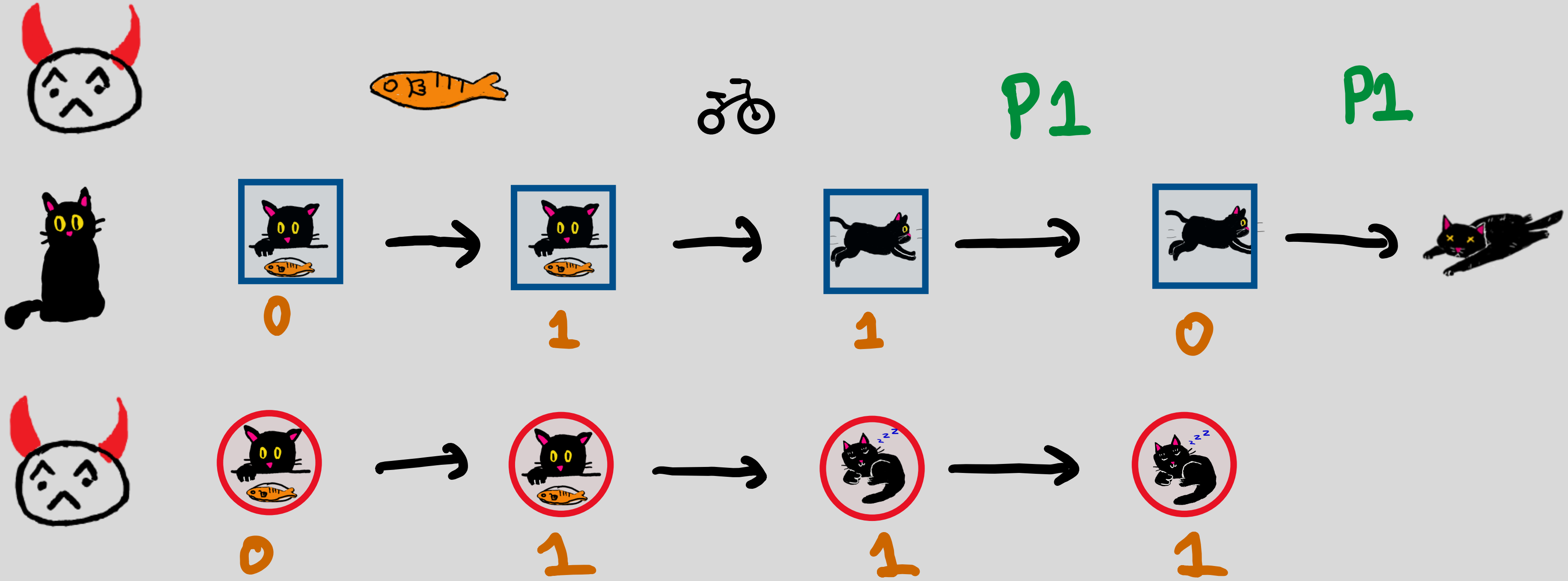


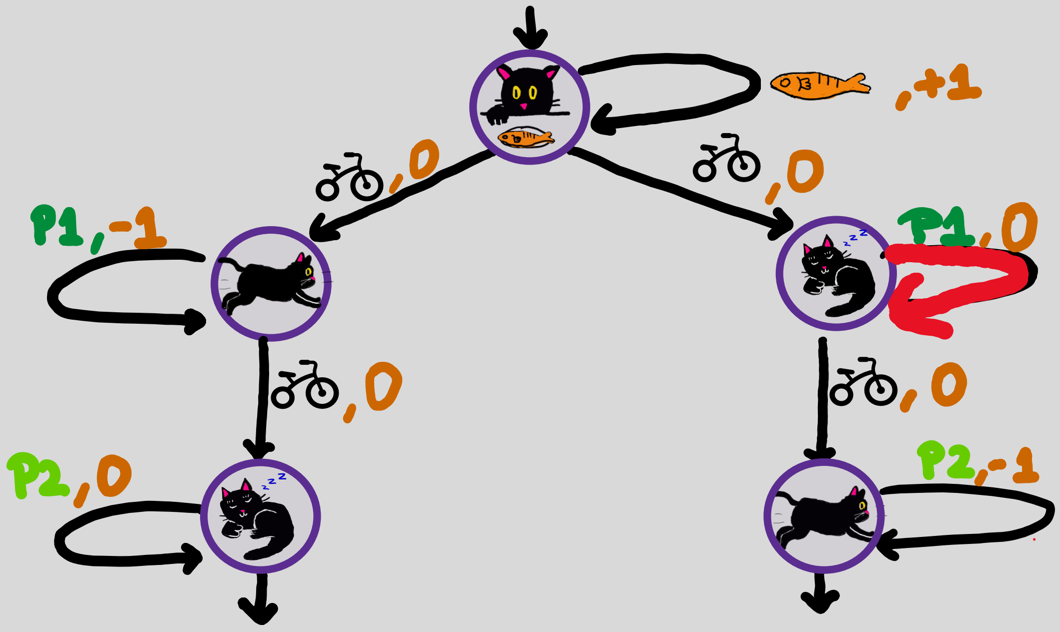
$L = \{ 🐟^{i} \; 🚲 \; P1^{j} \; 🚲 \; P2^{k} \mid i \geq j \text{ or } i \geq k \}$



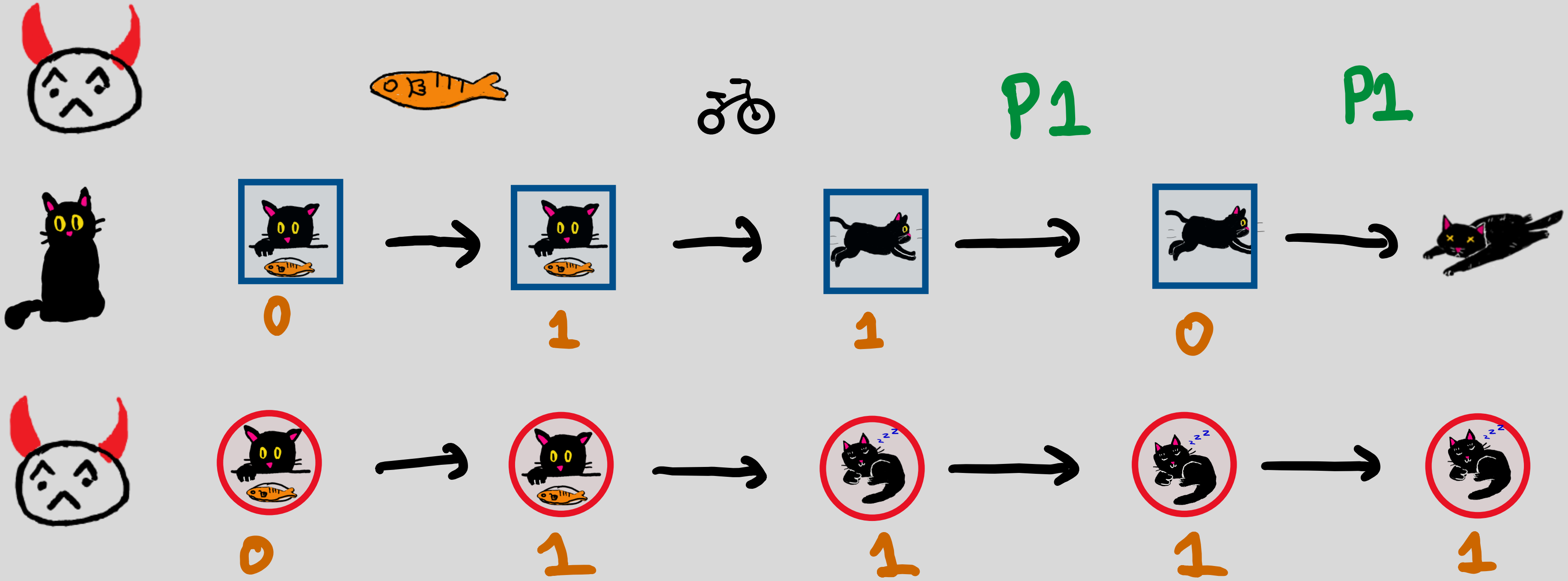


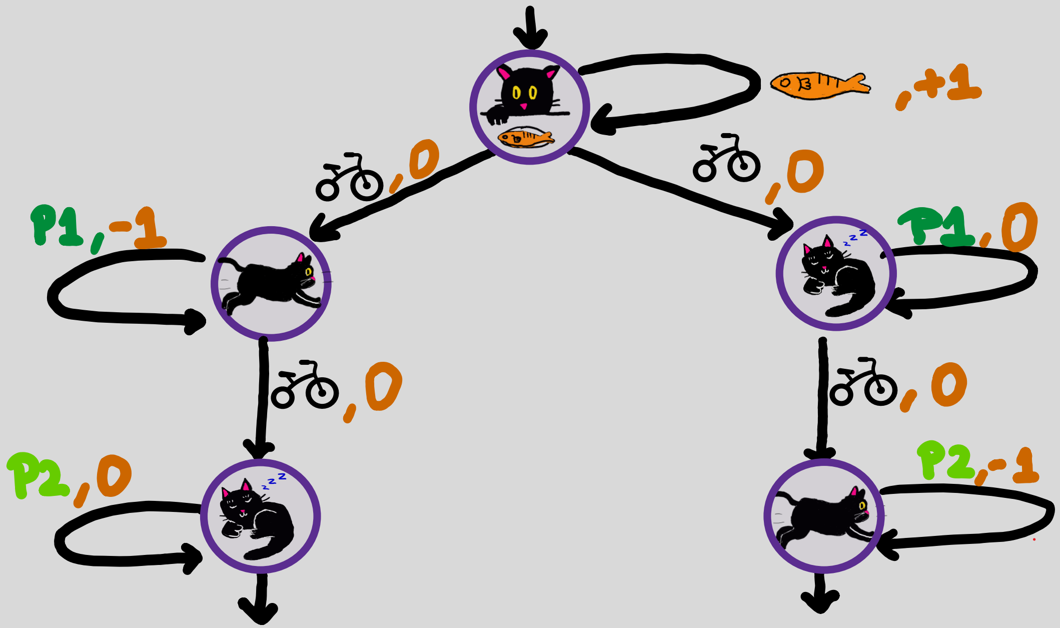
L = {🐟^{i} 🚲 P1^{j} 🚲 P2^{k} | $i \geq j$ or $i \geq k$}



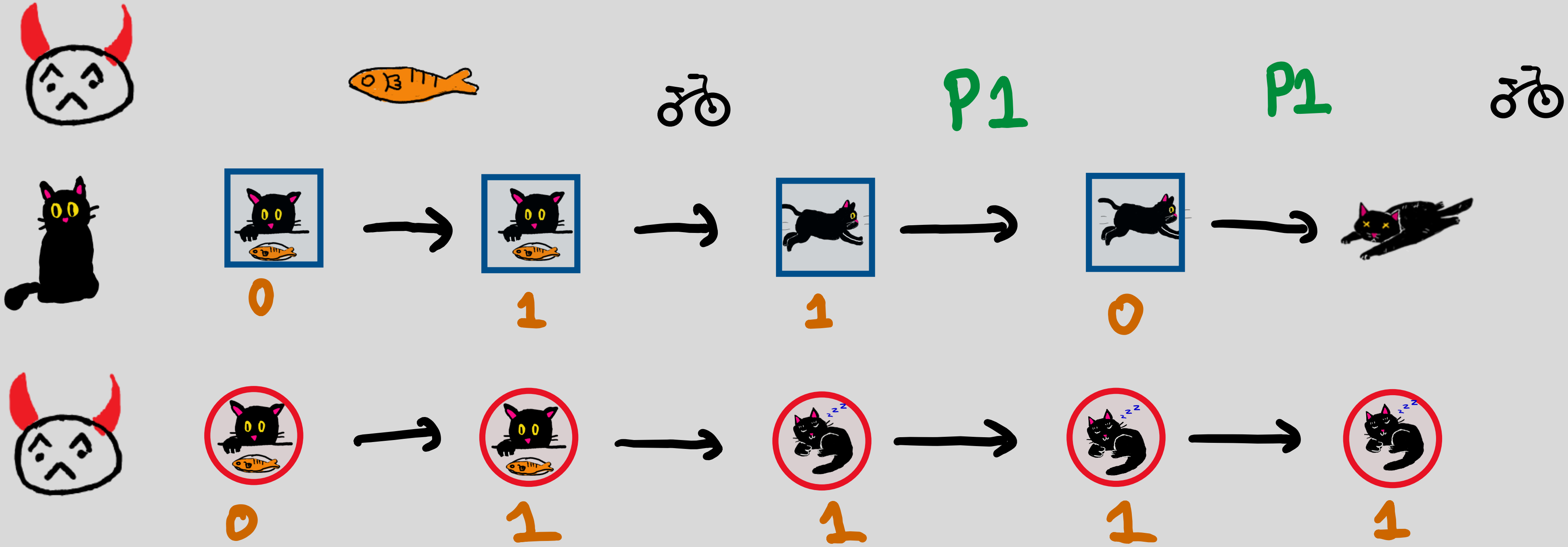


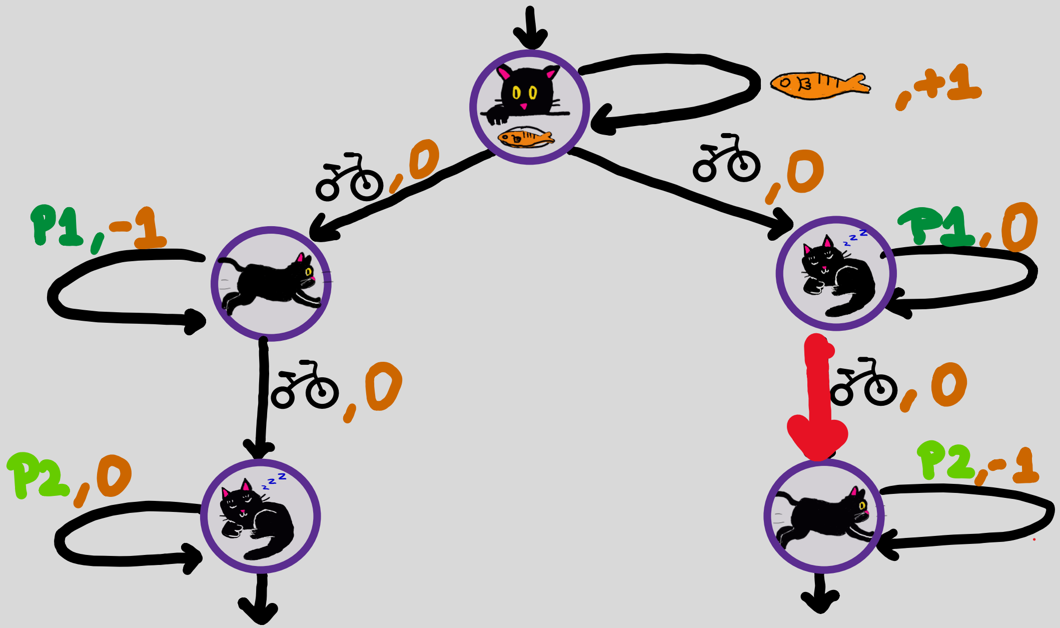
L = { 🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$ }



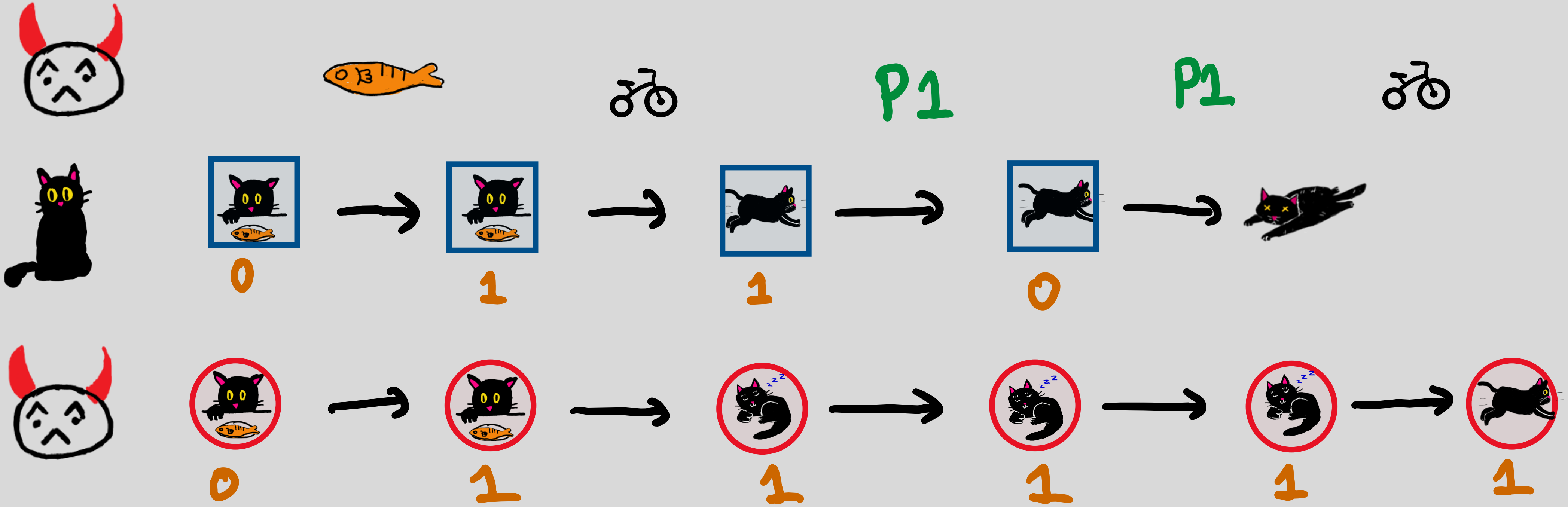


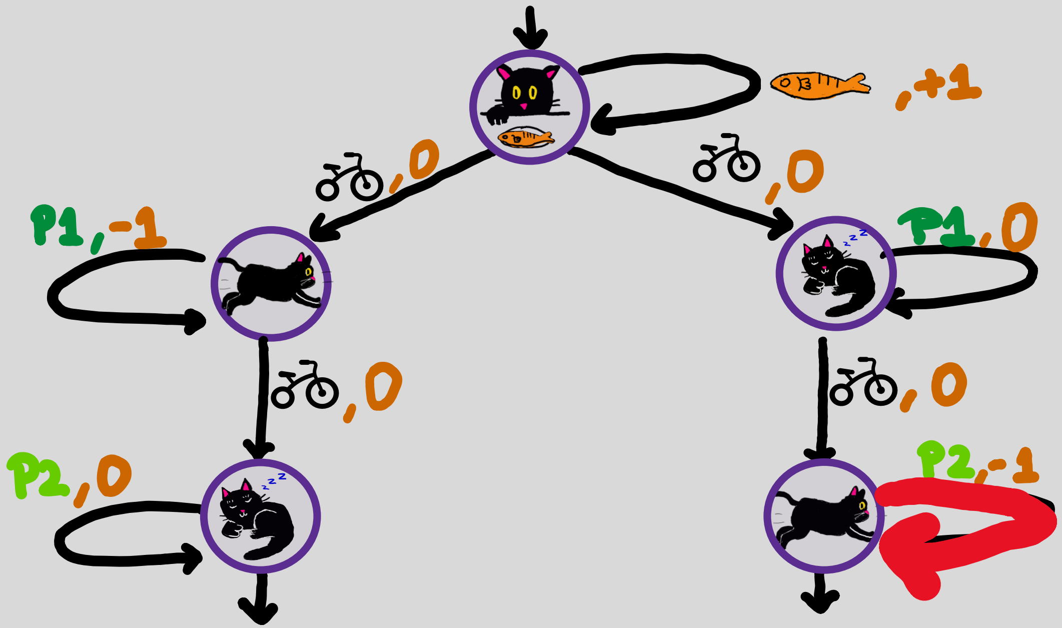
L = { 🐟^{i} 🚲 P1^{j} 🚲 P2^{k} | $i \geq j$ or $i \geq k$ }



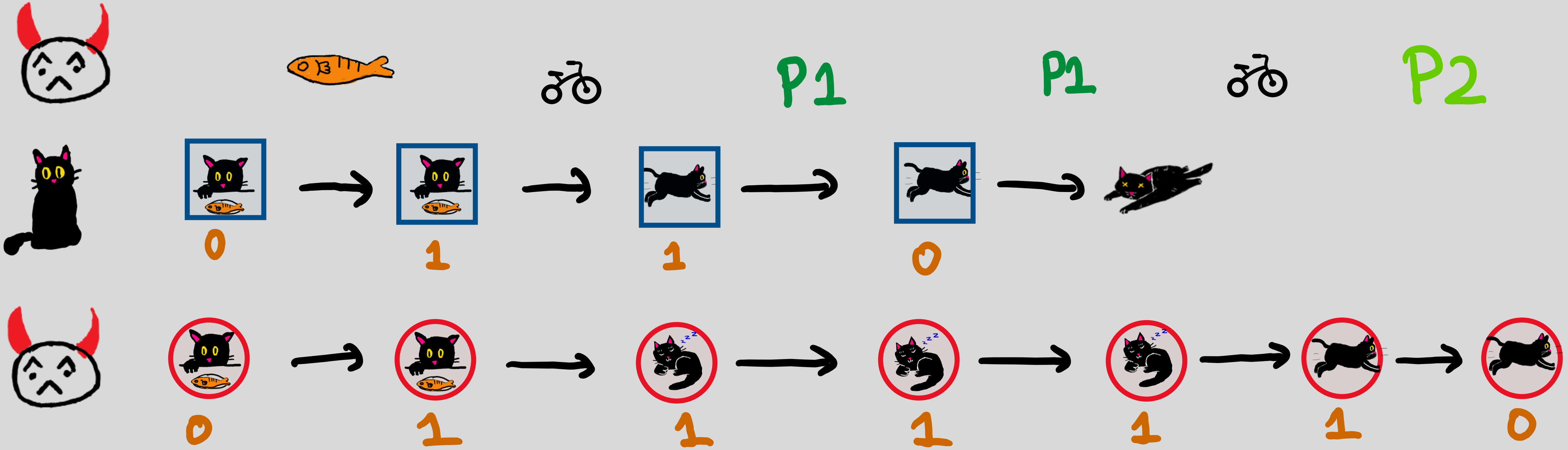


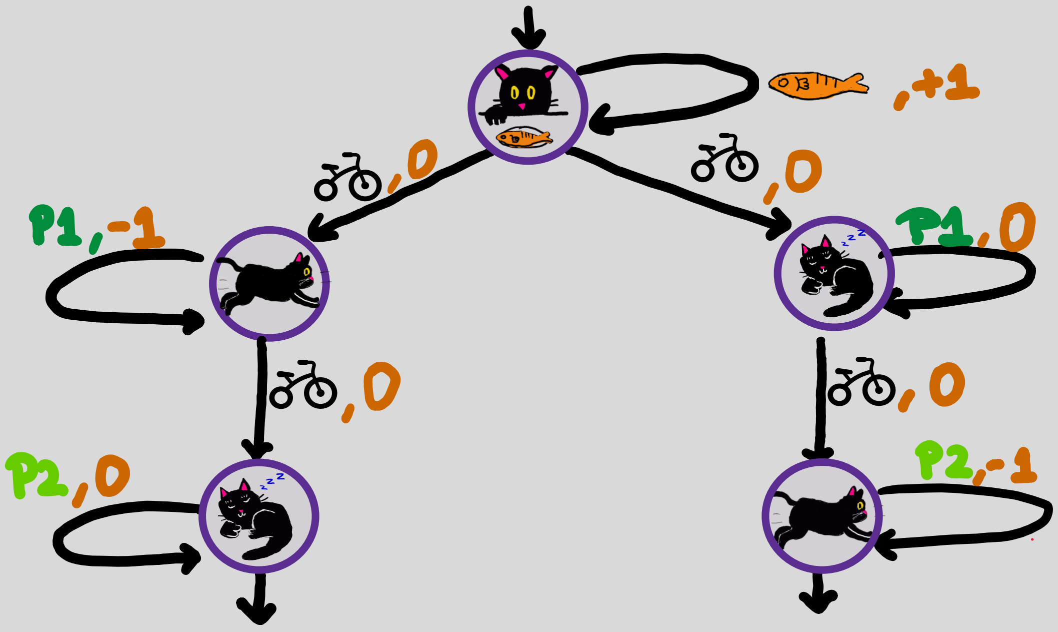
L = {🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$}



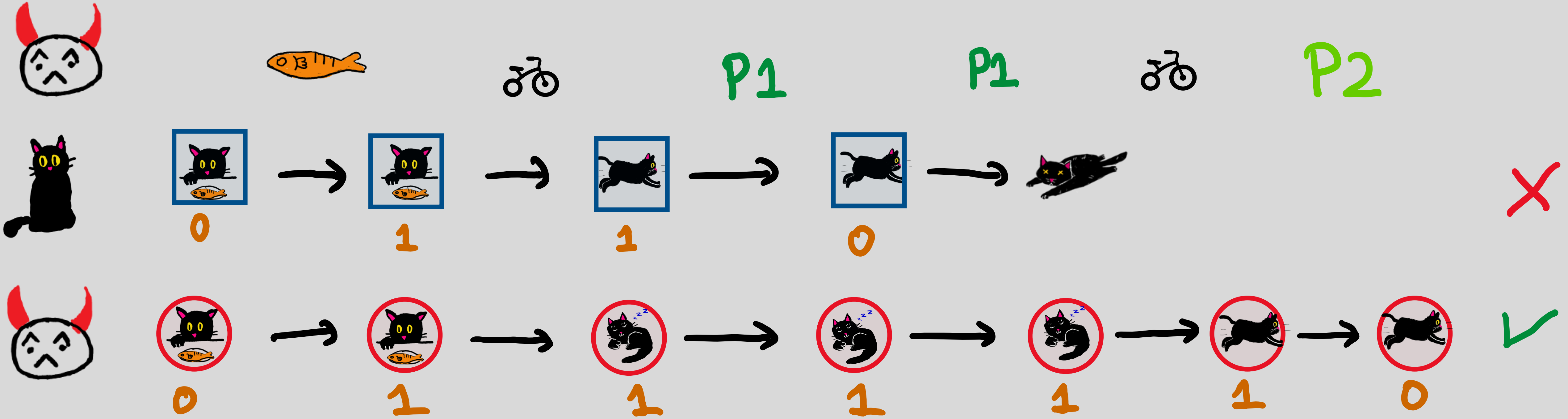


L = {🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$}





L = { 🐟^{i} 🚲 P1^j 🚲 P2^k | $i \geq j$ or $i \geq k$ }



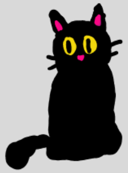
HD Game:

Winning Condition for 😈 : If 😈's word is accepting, and 🐈‍⬛'s run is rejecting.

Token Game:

Winning condition of 🐈‍⬛ : If 😈's run on 🔴 is accepting and 😈's run on 🟦 is rejecting.

Theorem [Boker, Lehtinen'22]

 wins 1-token game $\Leftrightarrow$  wins HD-game

Theorem: Given an history-deterministic OCN, there is a semilinear strategy for  that is effectively computable.

Proof:

HD Game ⟺ Token Game ⟹ Simulation Game

[Boker, Lehtinen'22] (HD Game ⟺ Token Game)

[Our work!!] (Token Game ⟹ Simulation Game)

Theorem [Hofman, Lasota, Mayr, Totzke '16]

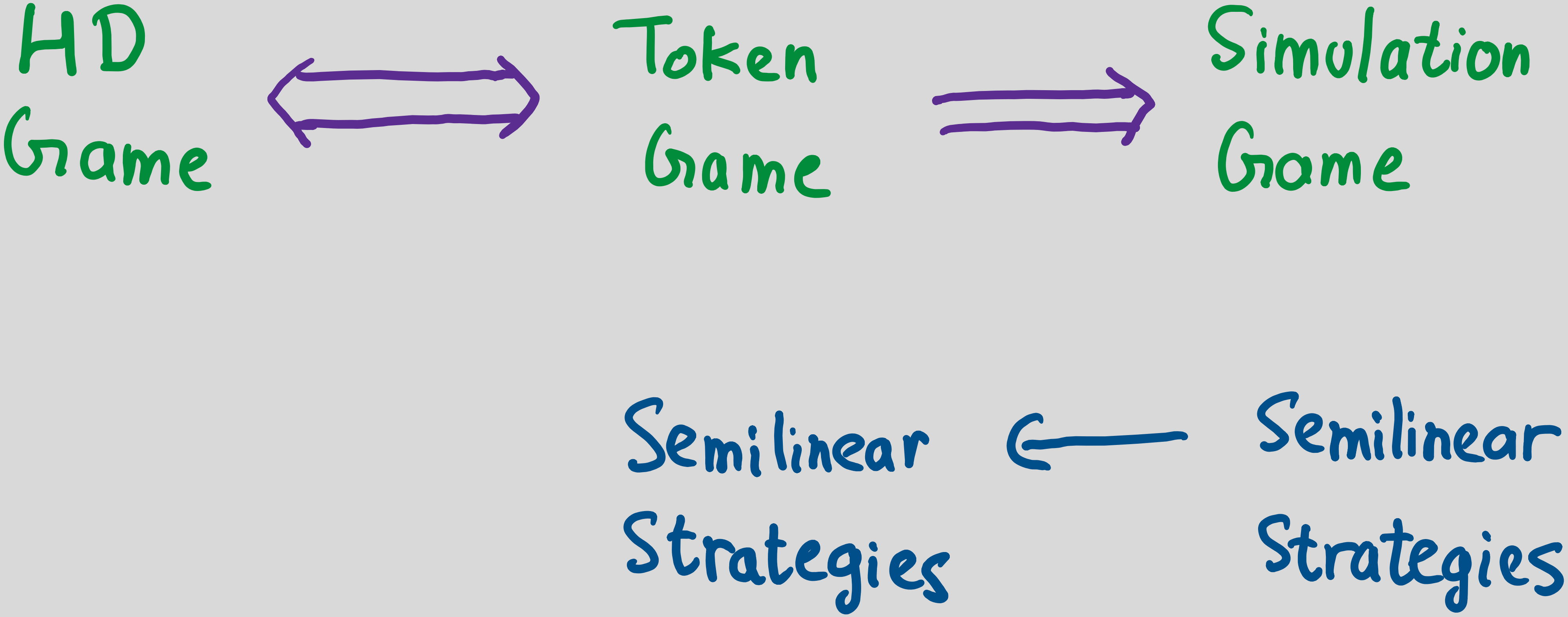
Deciding the winner in the simulation game between two OCNs can be done in PSPACE, and the winning player has semilinear stategies.

Proof:

HD Game $\Longleftrightarrow$ Token Game $\Longrightarrow$ Simulation Game

Semilinear Strategies

Proof:



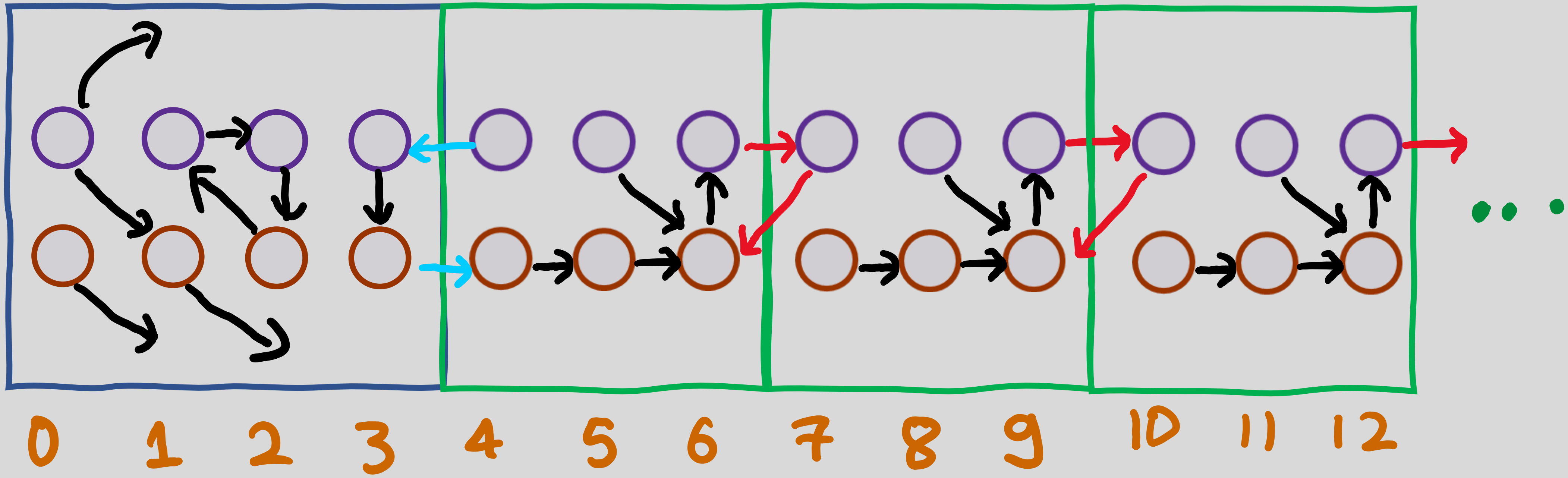
HD Game ⟺ Token Game ⟹ Simulation Game

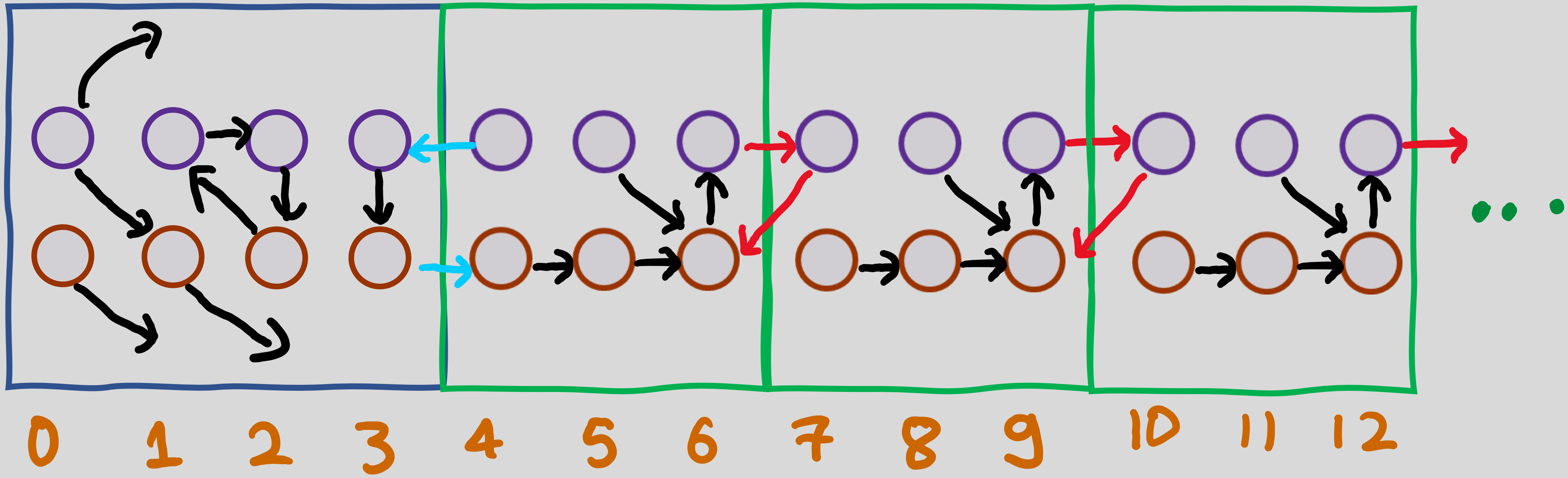
Semilinear Strategies ⟸ Semilinear Strategies

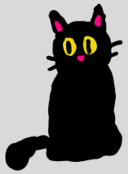
Proof:

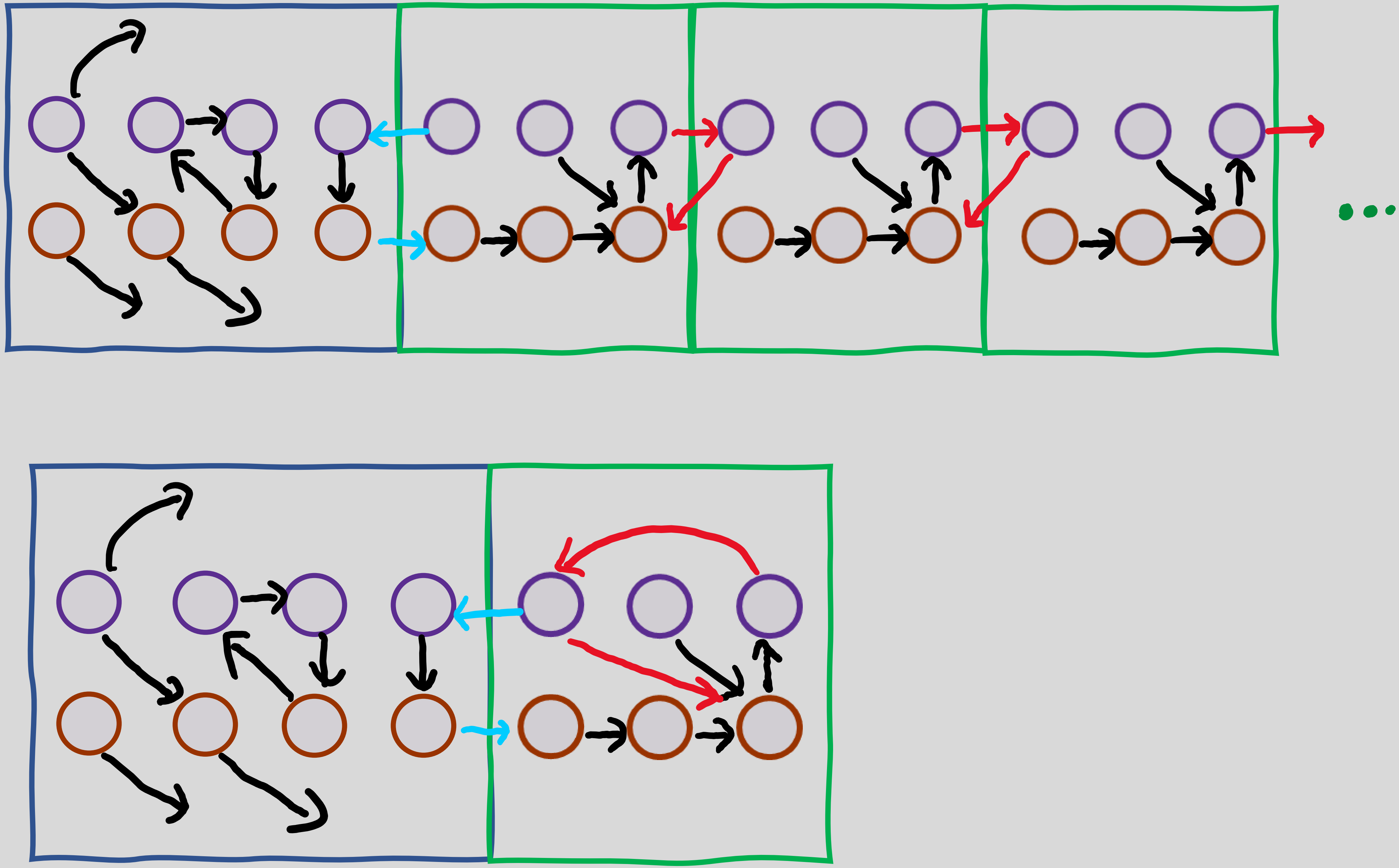
$$\begin{tikzcd}
\text{HD Game} \arrow[r, Leftrightarrow] & \text{Token Game} \arrow[r, Rightarrow] & \text{Simulation Game} \\
\text{Semilinear Strategies} & \text{Semilinear Strategies} \arrow[l] & \text{Semilinear Strategies} \arrow[l]
\end{tikzcd}$$

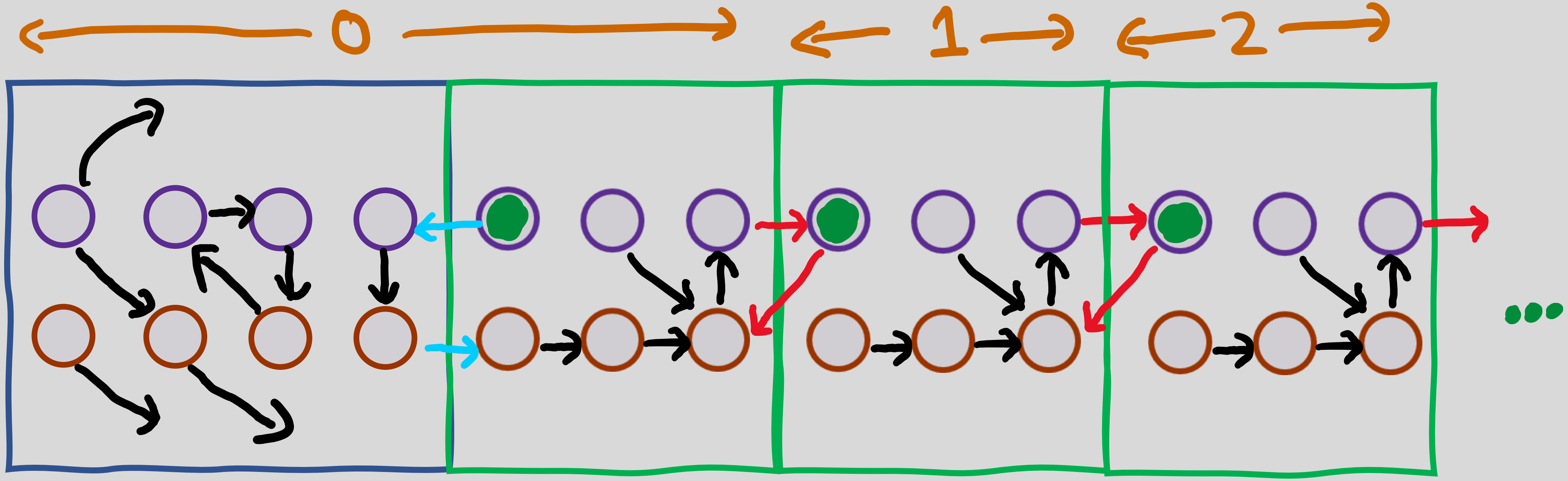
Theorem: Given an history-deterministic OCN, there is a semilinear strategy for  that is effectively computable.

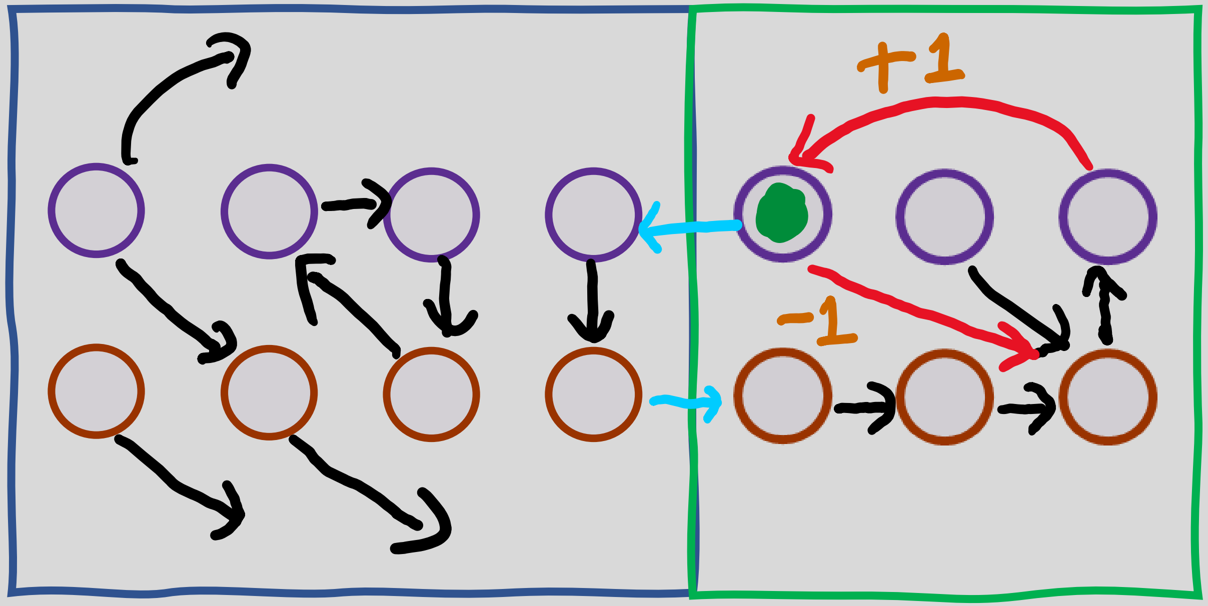

0
1
2
3
4
5
6
7
8
9
10
11
12
...

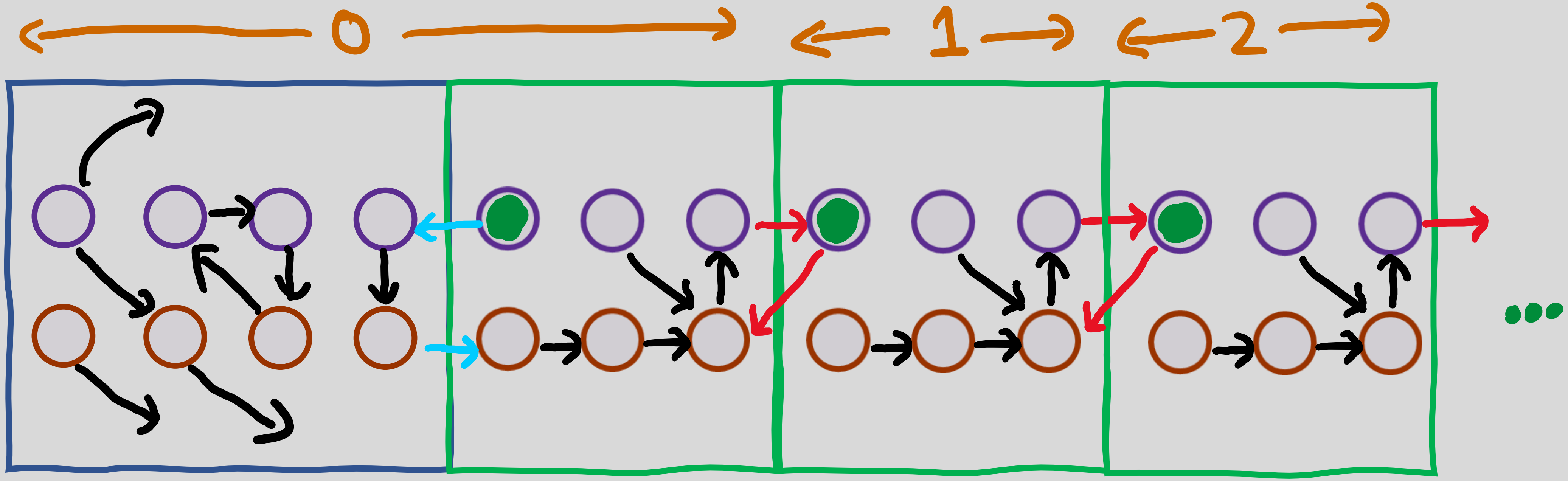


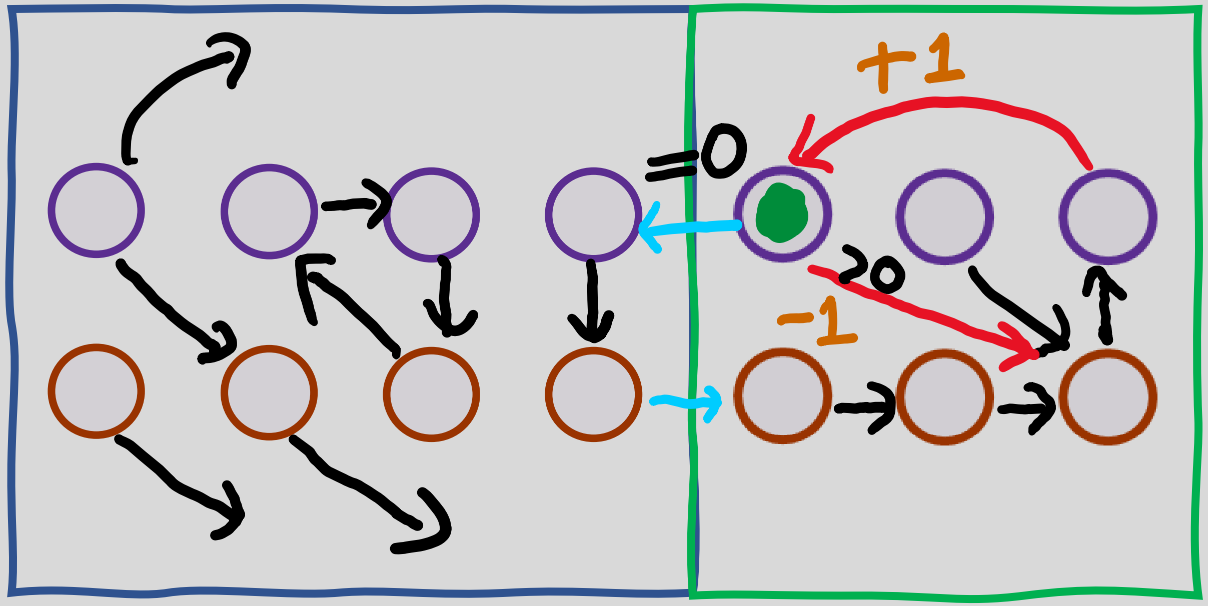
Note that any run that follows 's winning strategy is accepting on all accepting words.




0
1
2
...


+1
-1





Deterministic

one-counter automaton

Theorem: Given an history-deterministic OCN, there is a semilinear strategy for  that is effectively computable.

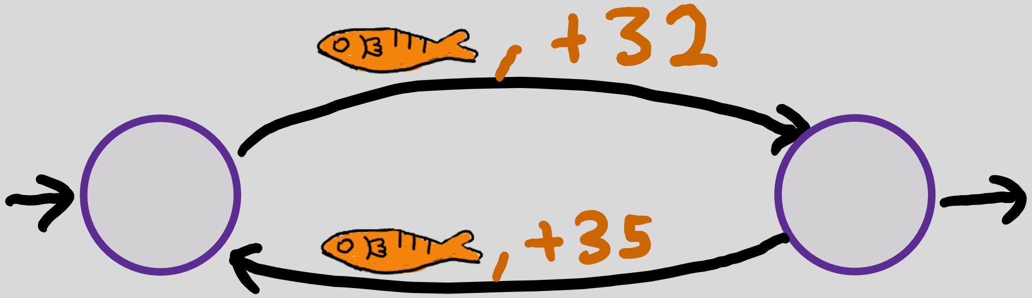
Corollary: Every history-deterministic OCN can be converted to a language equivalent deterministic OCA.

Theorem [Hofman, Lasota, Mayr, Totzke '16]

Deciding the winner in the simulation game between two OCNs can be done in PSPACE.

Theorem: Deciding whether a given OCN is history-deterministic is PSPACE-complete, and EXPSPACE-complete when the counters are encoded in binary.

Theorem: Deciding whether a given OCN is history-deterministic is PSPACE-complete, and EXPSPACE-complete when the counters are encoded in binary.



Theorem: Deciding whether a given OCN is history-deterministic is PSPACE-complete, and EXPSPACE-complete when the counters are encoded in binary.

Proof: Upper bound - Reduction to Simulation game

Theorem: Deciding whether a given OCN is history-deterministic is PSPACE-complete, and EXPSPACE-complete when the counters are encoded in binary.

Proof: Upper bound - Reduction to Simulation game

Lower bound - Reduction from Reachability game on OCN

History-Deterministic One-Counter Nets

- The resolvers in an HD-OCN are semilinear

  Corollary: Every HD-OCN can be converted to a deterministic one-counter automaton.

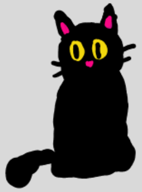
- Complexity of checking history-determinism:

  1. PSPACE-complete for unary encoding
  2. EXPSPACE-complete for binary encoding

3. History-Deterministic One-Counter Automata

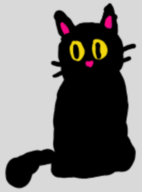
What about One-Counter Automata?

What about One-Counter Automata?



Checking for history-determinism:

What about One-Counter Automata?



Checking for history-determinism: Undecidable

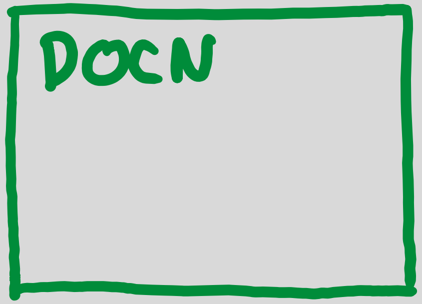
What about One-Counter Automata?

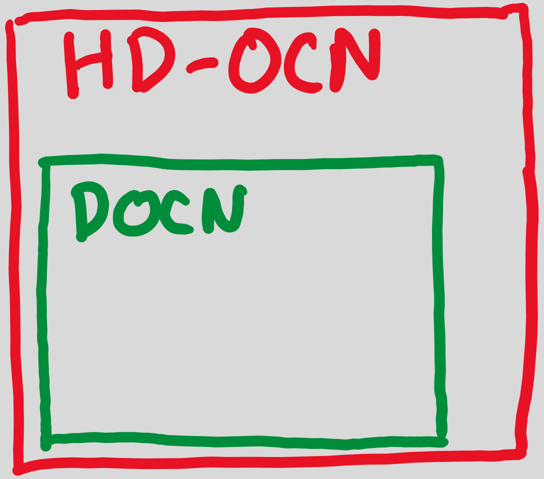
- Checking for history-determinism: Undecidable
- Do all HD-OCA admit language-equivalent DOCA?

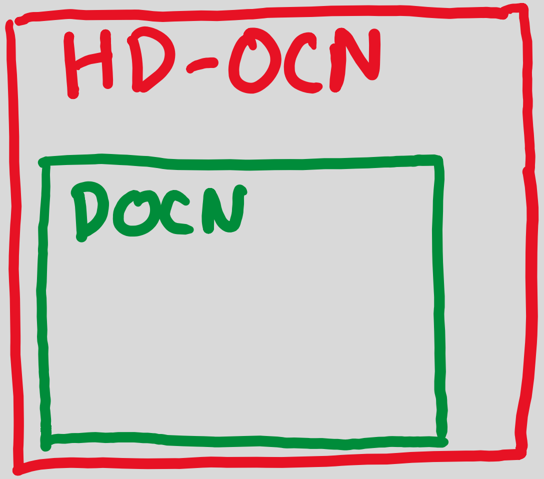
What about One-Counter Automata?

- Checking for history-determinism: Undecidable

- Do all HD-OCA admit language-equivalent DOCA?
  Open. Semilinearity of strategies suffice.

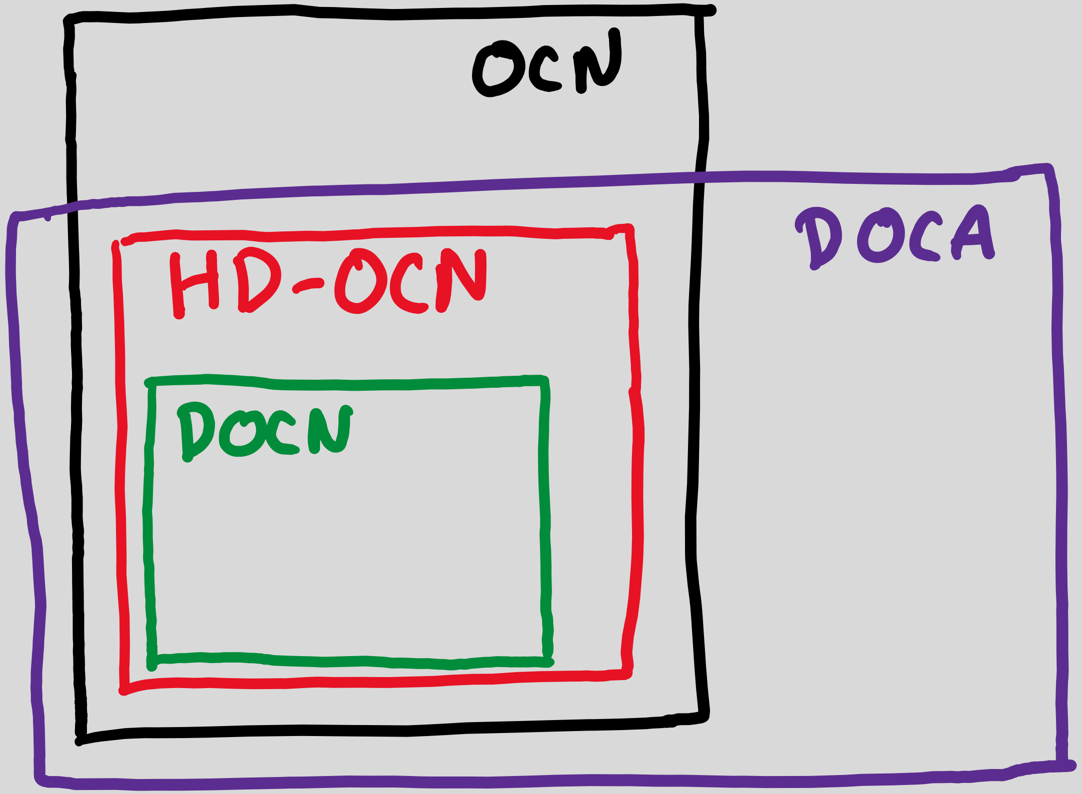

DOCN


HD-OCN
DOCN

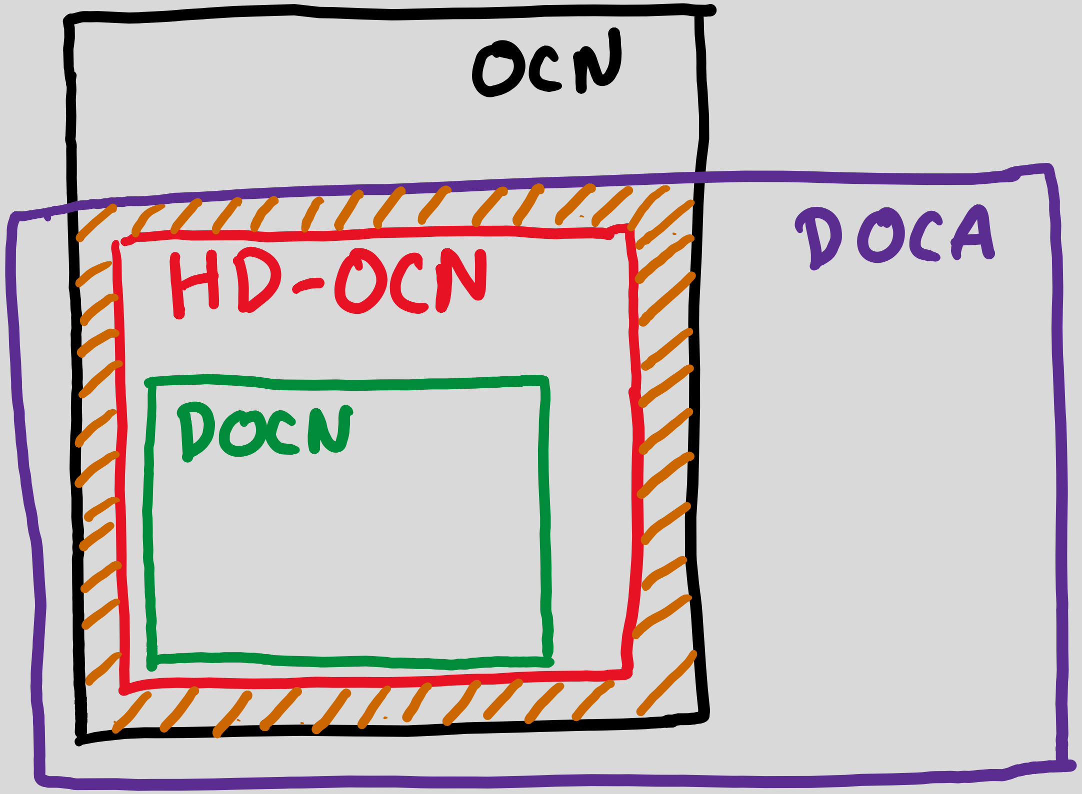


HD-OCNs are more expressive than DOCNs.

[S. Almagor, A. Yeshurun]
Private Communication



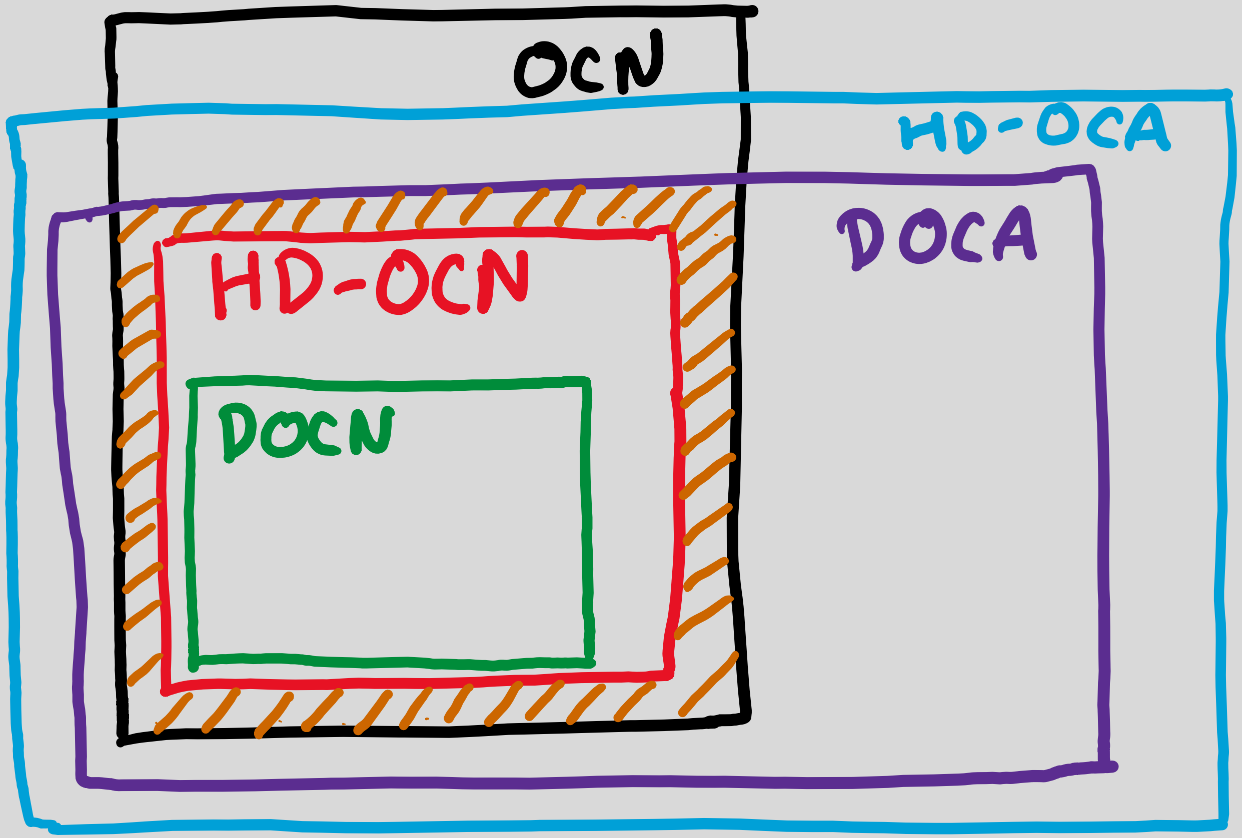
$$\text{HD-OCN} \subseteq \text{OCN} \cap \text{DOCA}$$



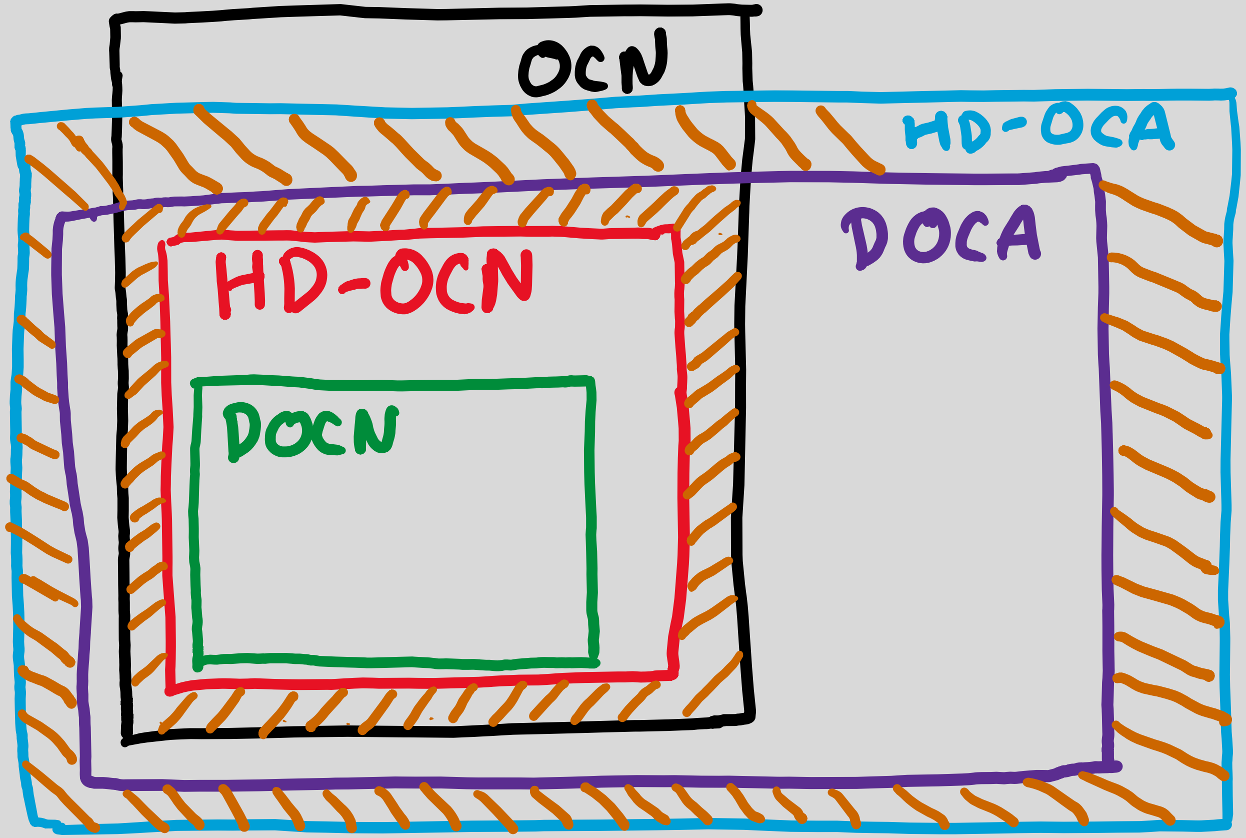
$\text{HD-OCN} \subseteq \text{OCN} \cap \text{DOCA}$

Is $\text{HD-OCN} = \text{OCN} \cap \text{DOCA}$?

Open.



Is HD-OCA = DOCA ?



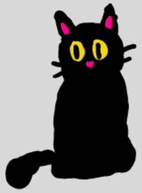
Is HD-OCA = DOCA ?

Open

Conclusion

- Every history-deterministic OCN has a language-equivalent DOCA.
- Checking history-determinism for OCNs is PSPACE-complete.
- Checking history-determinism for OCAs is undecidable.

Open Problems



Is HD-OCA = DOCA?

Is HD-OCN
= OCN ∩ DOCA?