Unit 5: Machine Learning & AI

Dave Abel

March 4th, 2016

Reinforcement Learning

Demo time!

Overfitting

- **Generalization is the goal of learning.**

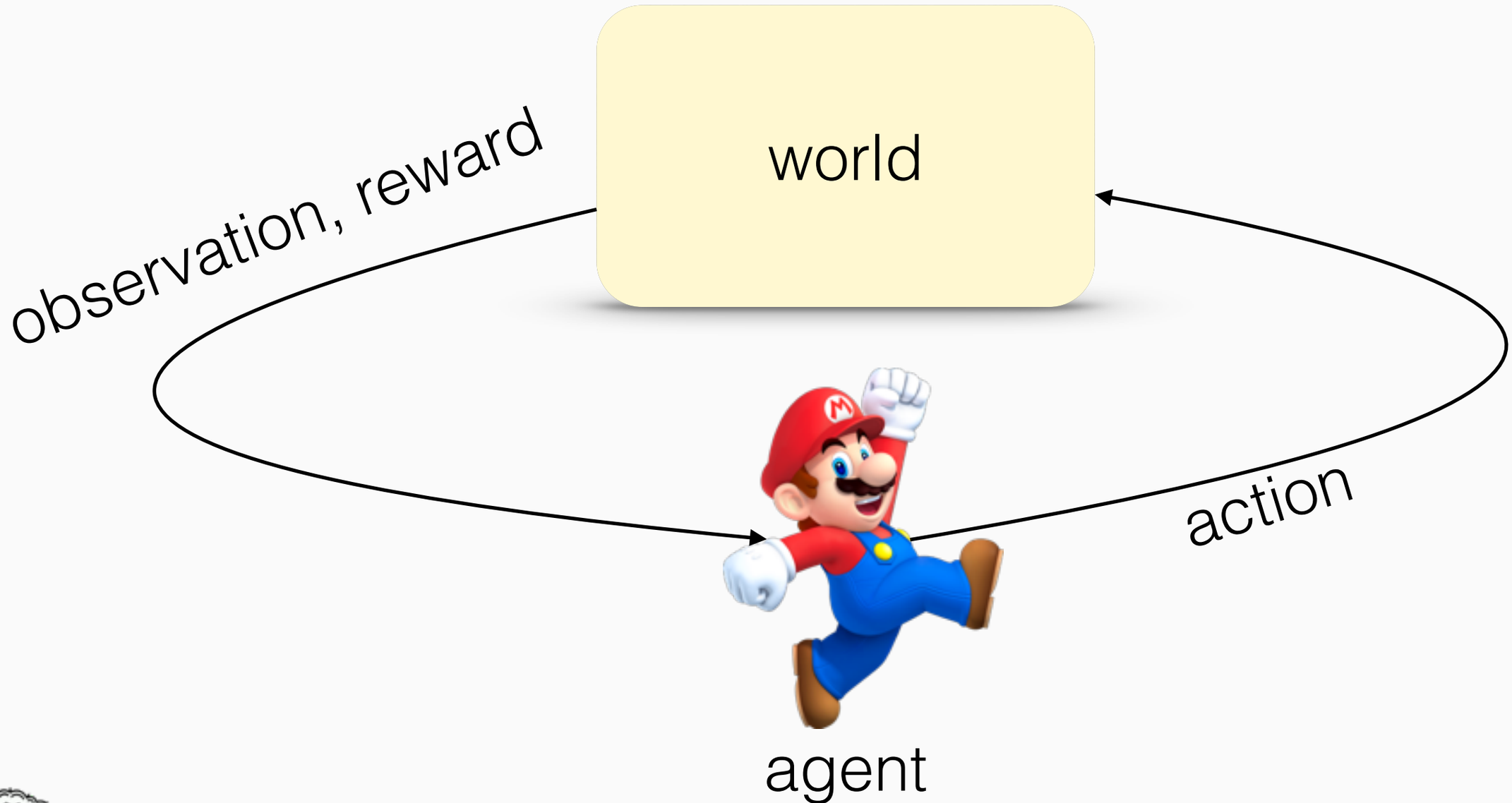
- Overfitting: when our classifier pays too much attention to noisy details of our data instead of the underlying trends.

- (When we do well on classifying training data, but poorly on testing data)

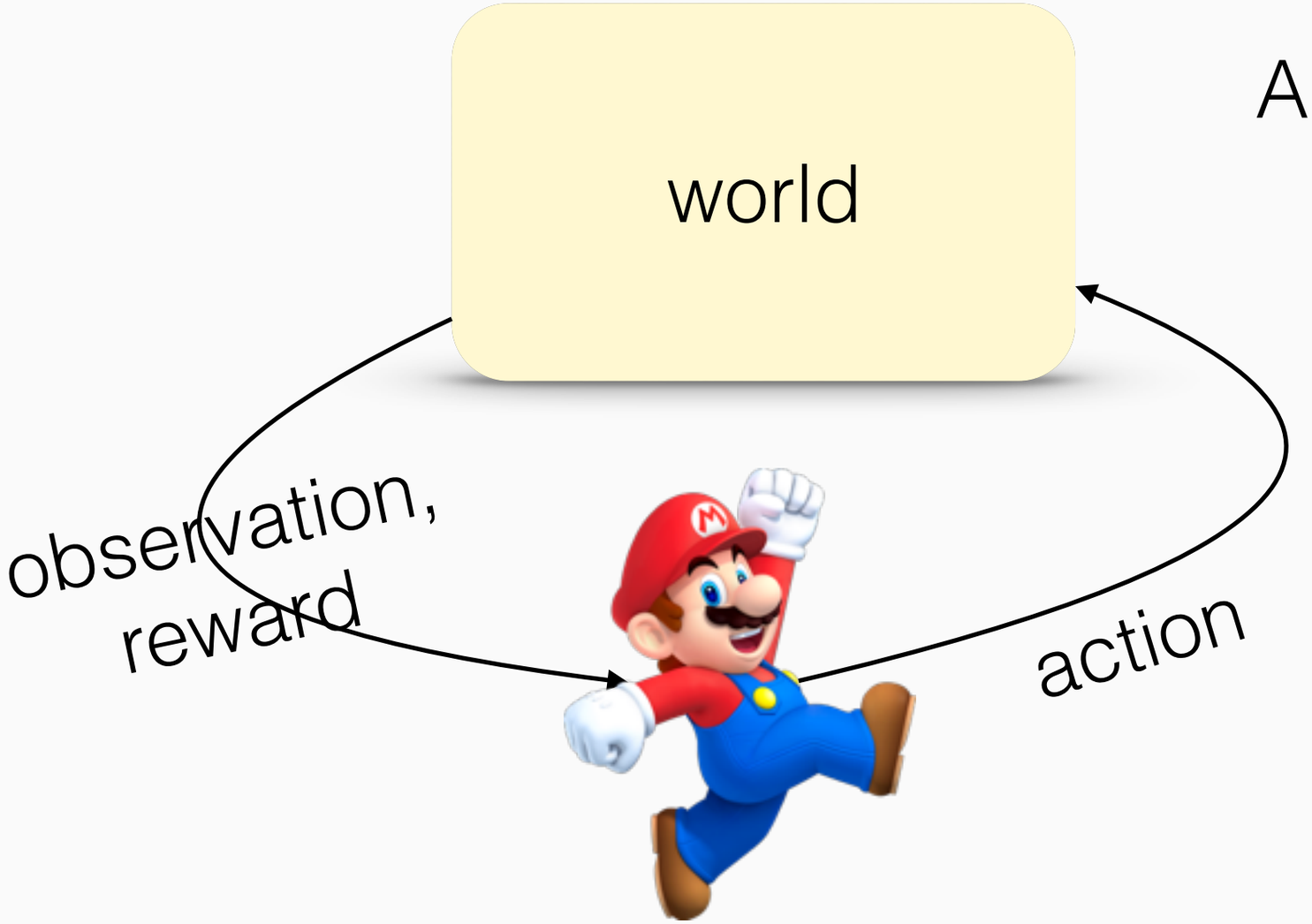
Overfitting

Reinforcement Learning

Reinforcement Learning



Reinforcement Learning



Any problem with a goal…

Goal: Maximize reward!

Or repeated behavior!

Reinforcement Learning

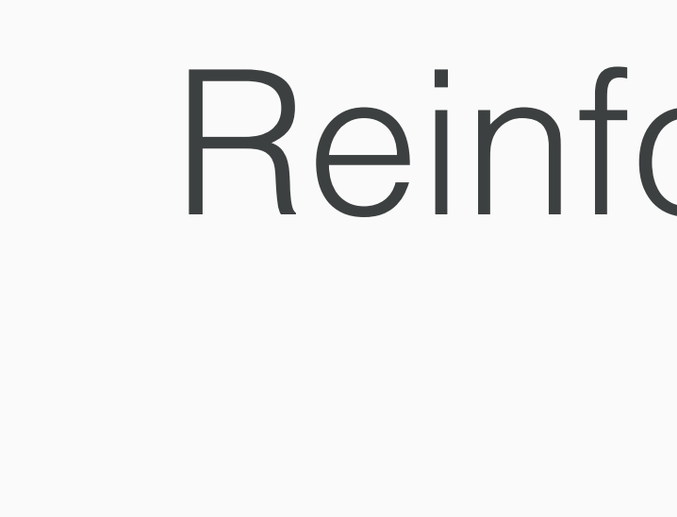
observation, reward

action

Memory

Reinforcement Learning

observation, reward



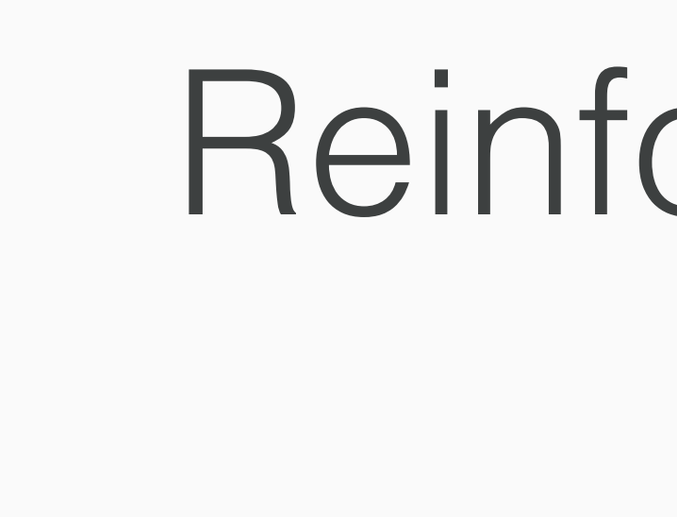
0

action

Memory

Reinforcement Learning

observation, reward



0

action

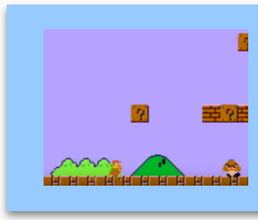
JUMP

Memory

Reinforcement Learning

observation, reward

action

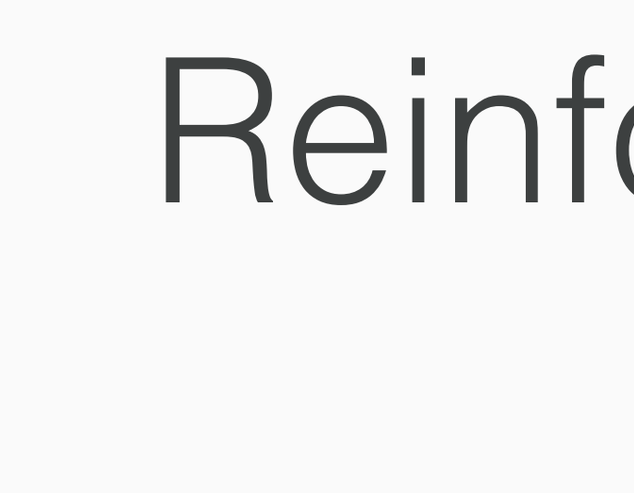


0 *JUMP*

Memory

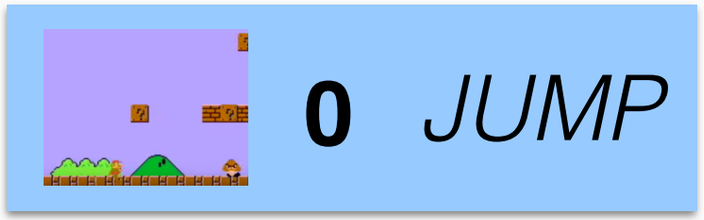
Reinforcement Learning

observation, reward



20

action



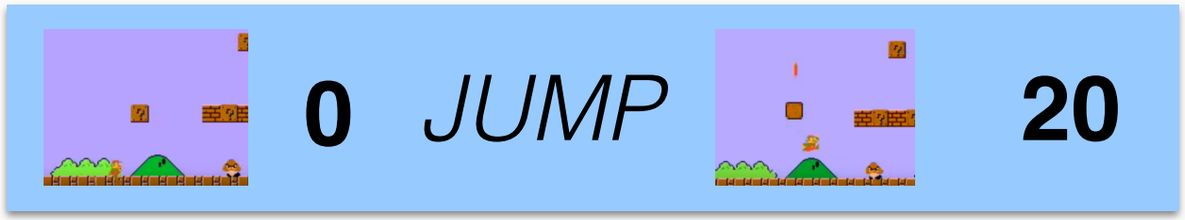
0 *JUMP*

Memory

Reinforcement Learning

observation, reward

action



Memory

Reinforcement Learning

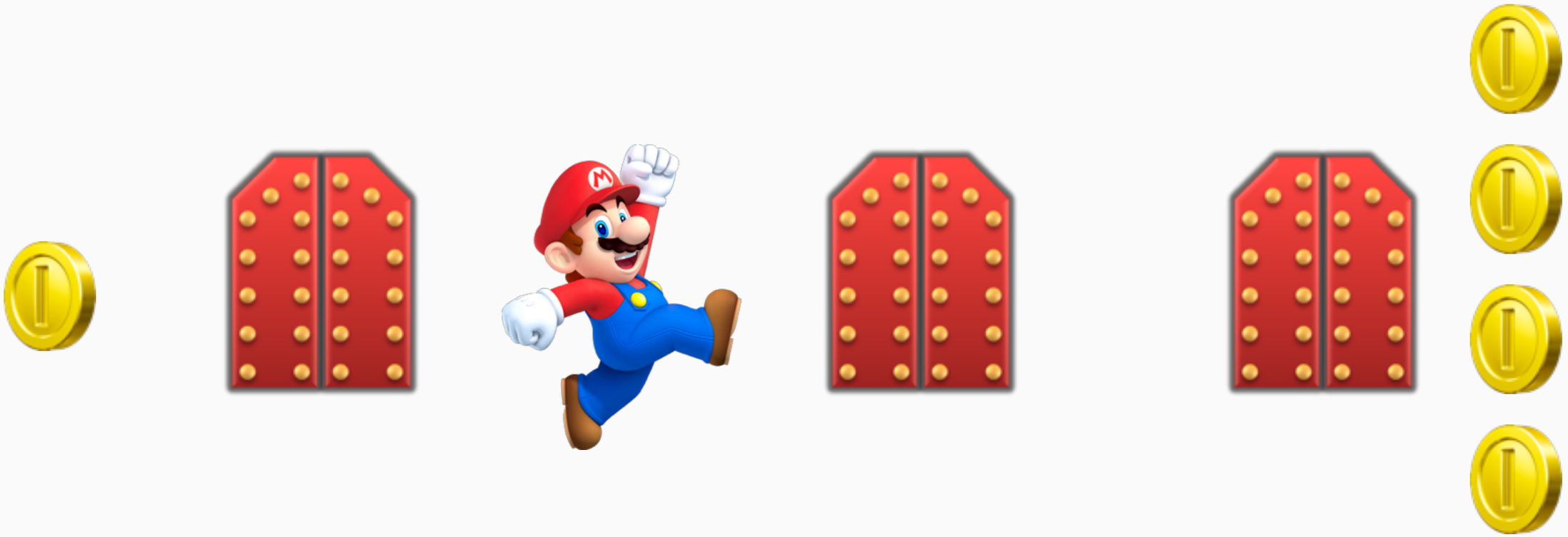
observation, reward

action

Goal: Maximize reward!

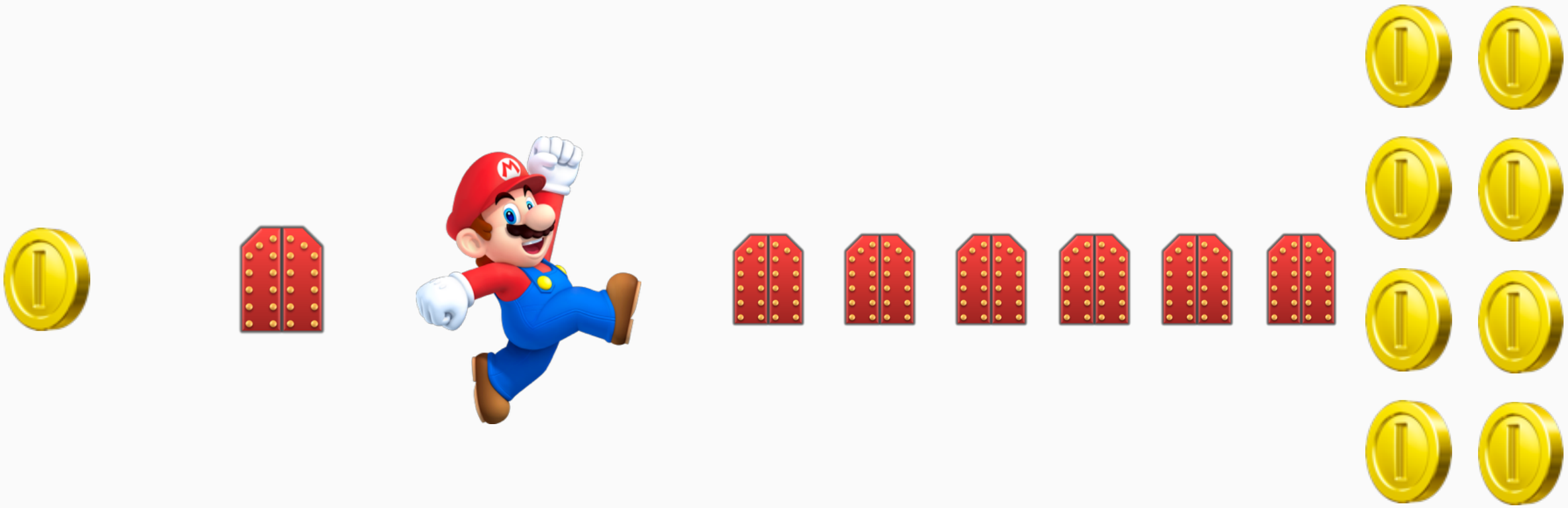
Memory

Reinforcement Learning



Goal: Maximize reward!

Reinforcement Learning



Goal: Maximize reward!

Reinforcement Learning

Reinforcement Learning

Q: How can we provide reward in Mario so that an RL agent would be incentivized to win the game?

Reinforcement Learning

Q: How can we provide reward in Mario so that an RL agent would be incentivized to win the game?

Some ideas:

- Give Reward when Mario beats the level
- Give Reward when Mario's score goes up
- Punish when Mario dies
- Punish when Mario loses a mushroom
- Give Reward when Mario kills a Goomba
- etc.

Clicker Question!

Q: What method of providing reward would incentivize an agent to do well in chess?

Clicker Question!

[A] Every time the agent moves a pawn, get punished

[B] Every time the agent moves a knight, get punished

[C] Every time the agent captures a piece, get reward

[D] If the agent wins, get reward.

Q: What method of providing reward would incentivize an agent to do well in chess?

Clicker Answer!

[A] Every time the agent moves a pawn, get punished

[B] Every time the agent moves a knight, get punished

[C] Every time the agent captures a piece, get reward

[D] If the agent wins, get reward.

Q: What method of providing reward would incentivize an agent to do well in chess?

Clicker Answer!

Could do more reward for better pieces.

[C] Every time the agent captures a piece, get reward

vs.

[D] If the agent wins, get reward.

Q: What method of providing reward would incentivize an agent to do well in chess?

Clicker Answer!

In general: a trail of breadcrumbs is much easier for an agent than just getting reward at the end!

If we only give reward at the end, all of its behavior leading up to the end is basically arbitrary!

Q: What method of providing reward would incentivize an agent to do well in chess?

Problem: Maximize Reward

INPUT: world & a set of actions

OUTPUT: Strategy for maximizing reward

Problem: Maximize Reward

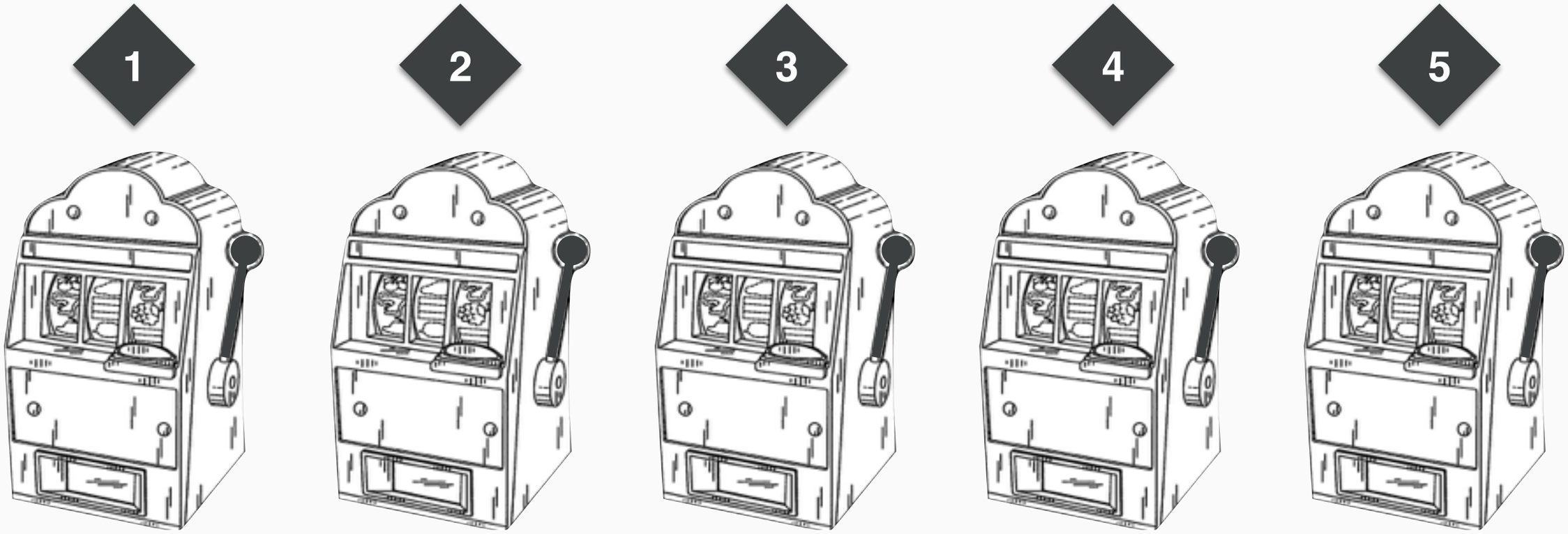
INPUT: world & a set of actions

OUTPUT: observation → Strategy for maximizing reward → action

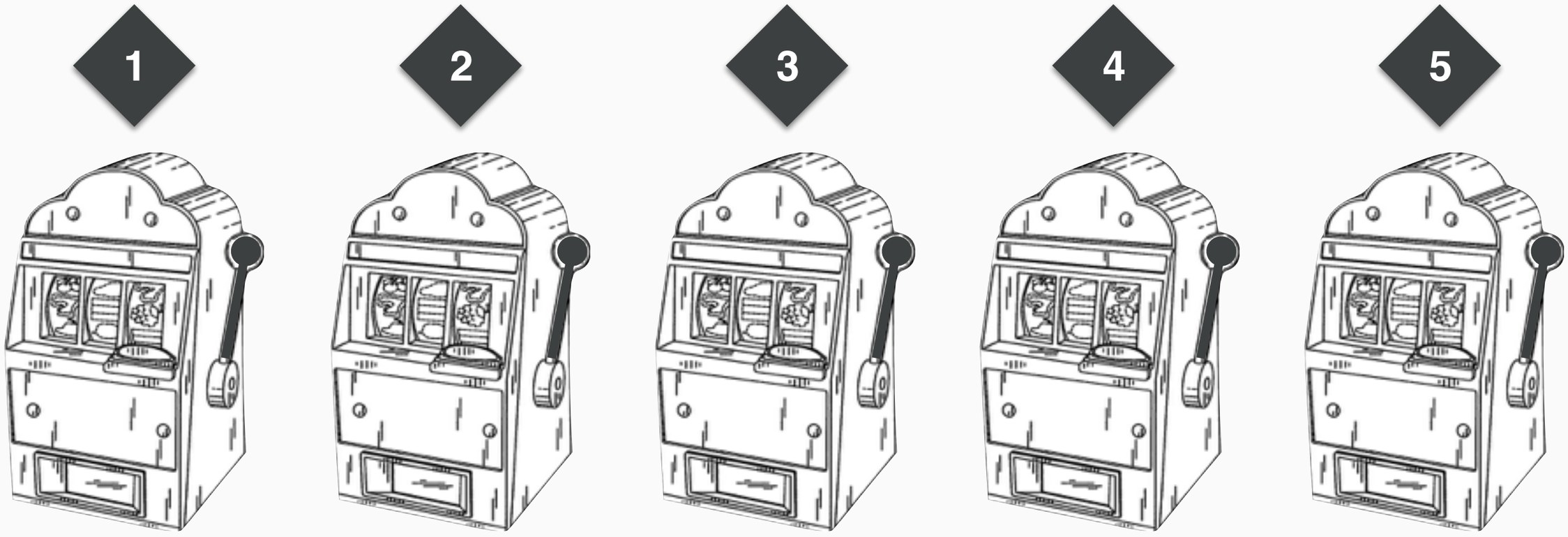
Reinforcement Learning

- Two things the agent can *learn*:
  - (A) The dynamics of the world!
    - *What will my next observation be, if I act in this way?*
  - (B) How do observations relate to reward?
    - *Where's the positive reward located?*

Slot Machines



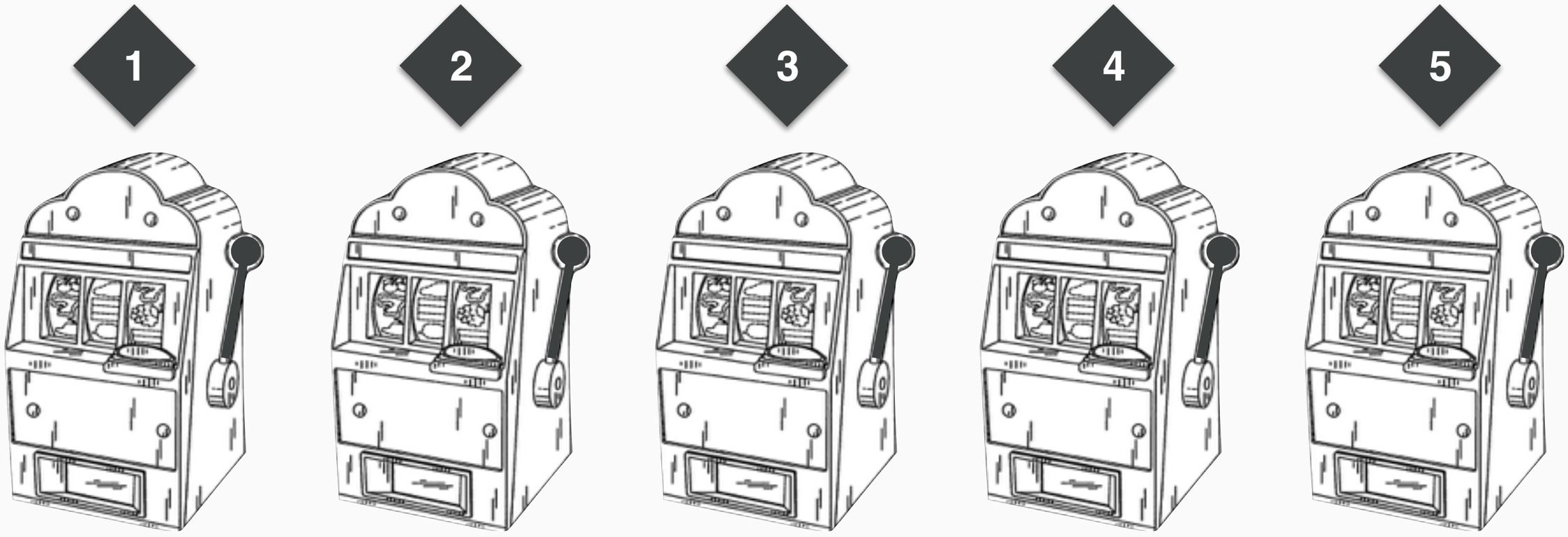
Slot Machines



Rules:

- Costs one token to pull an arm. Given *T* tokens.
- Each slot machine will pay some amount of money on average.
- Maximize money earned with fewest arm pulls.
- **NOTE**: Tokens are *not* money. Tokens represent # pulls.

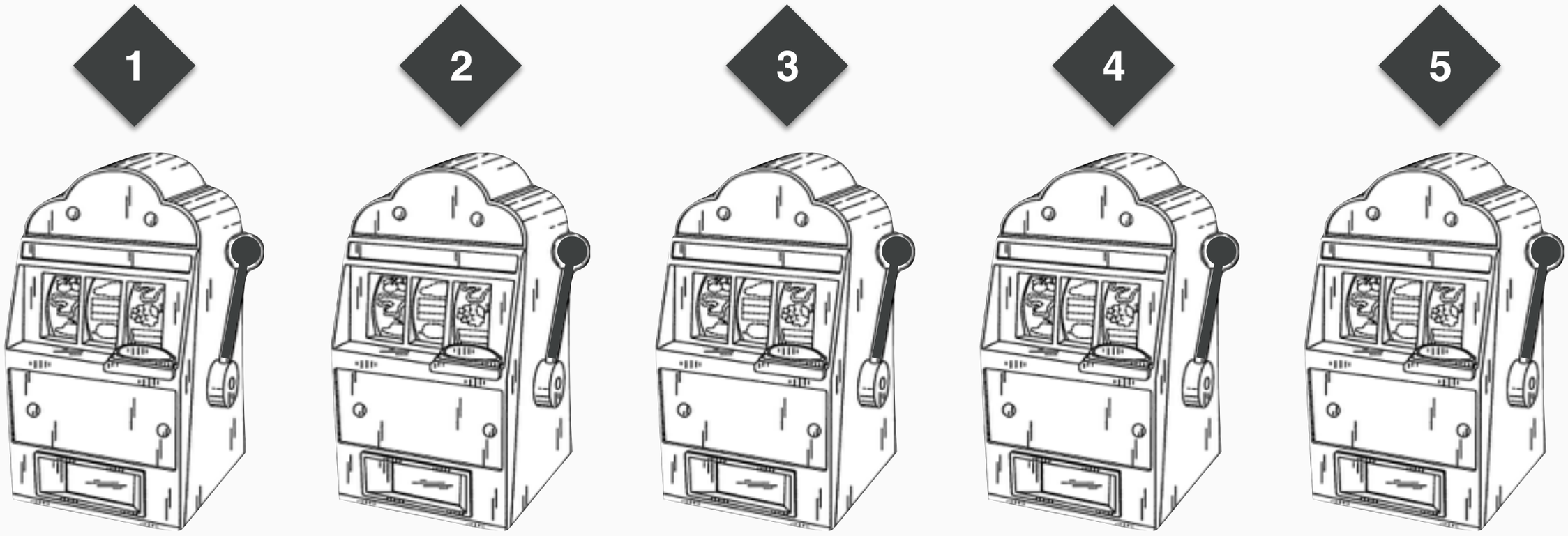
Slot Machines



Rules:

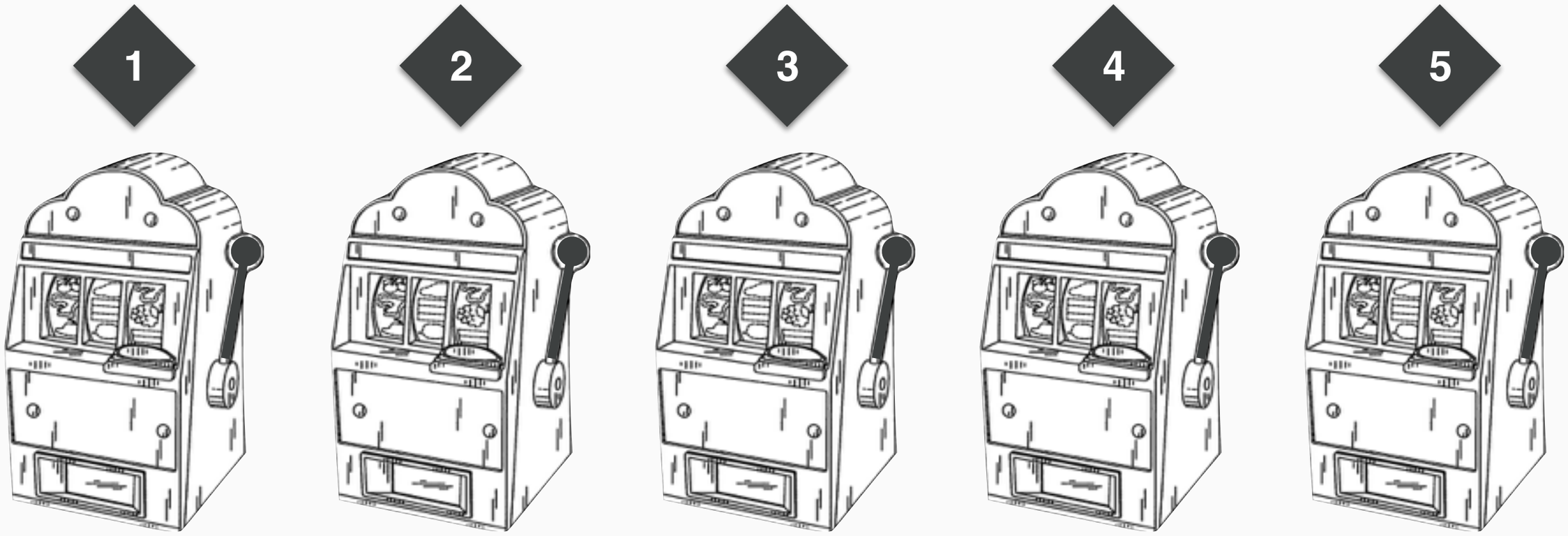
- Costs one token to pull an arm. Given *T* tokens.
- **Each slot machine pays some money *on average.* Could pay $5, then $15, then $10 if it pays $10 avg.**
- Maximize money earned with fewest arm pulls.
- NOTE: Tokens are *not* money. Tokens represent # pulls.

Slot Machines



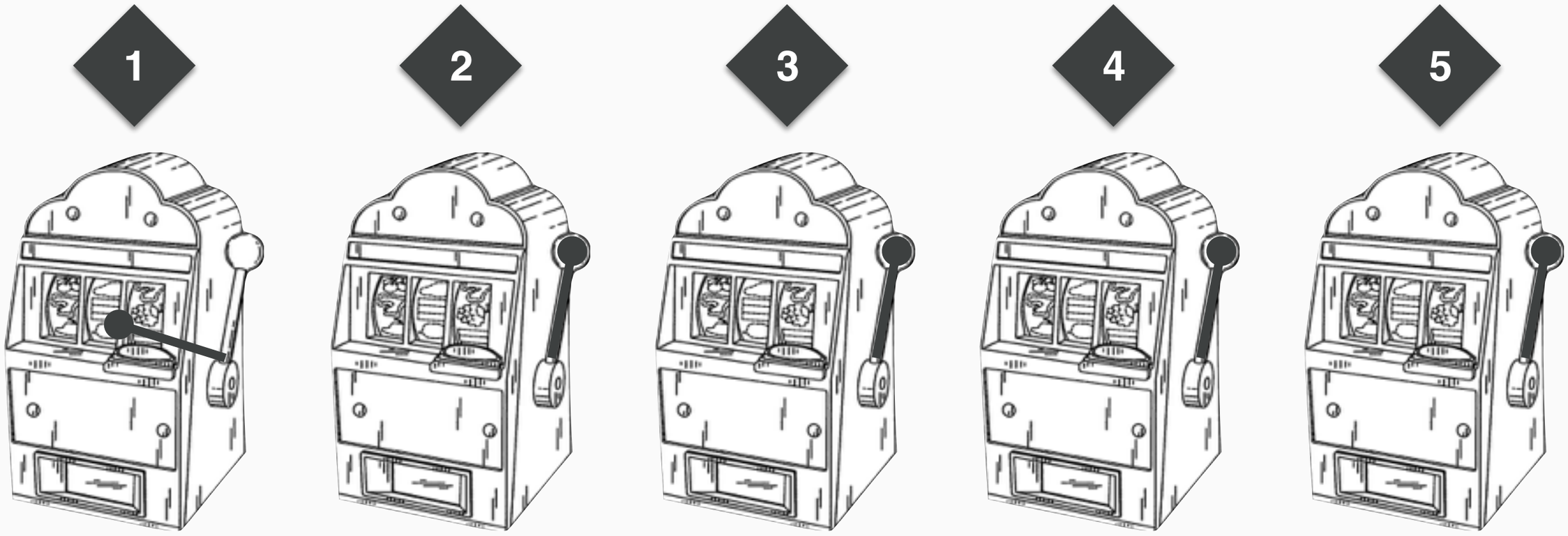
The Problem: How do I (or the agent) know when I should *explore* (to gain more information), as opposed to *exploiting* the information I have?

Slot Machines



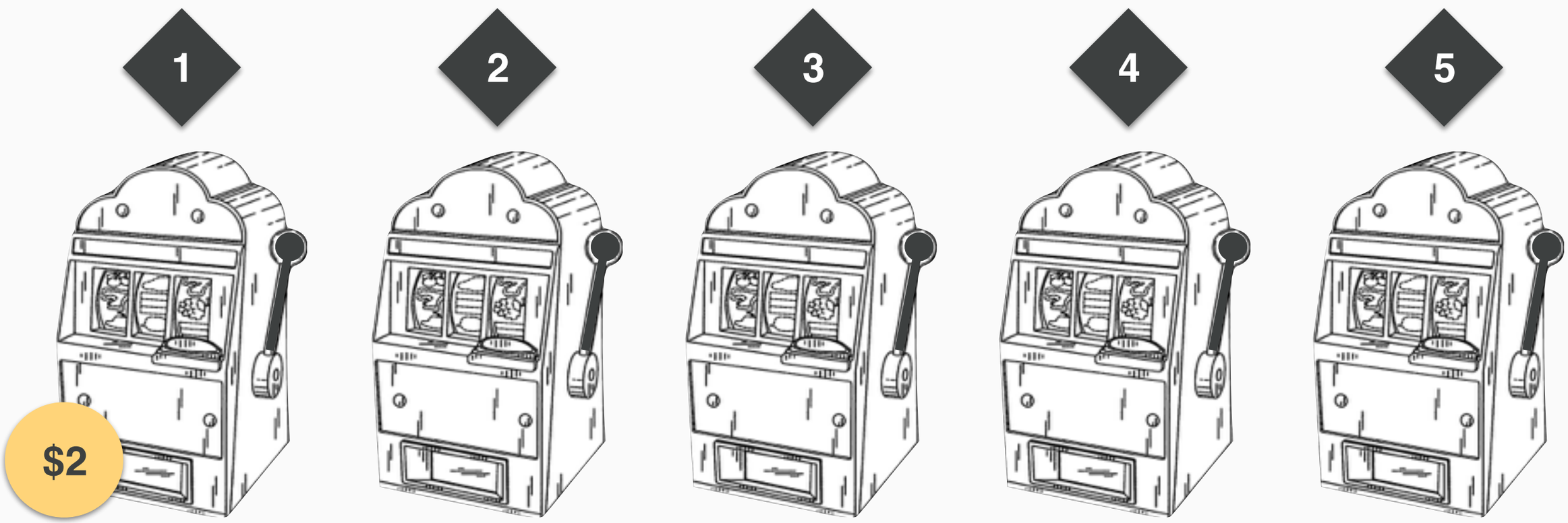
???	???	???	???	???

Slot Machines

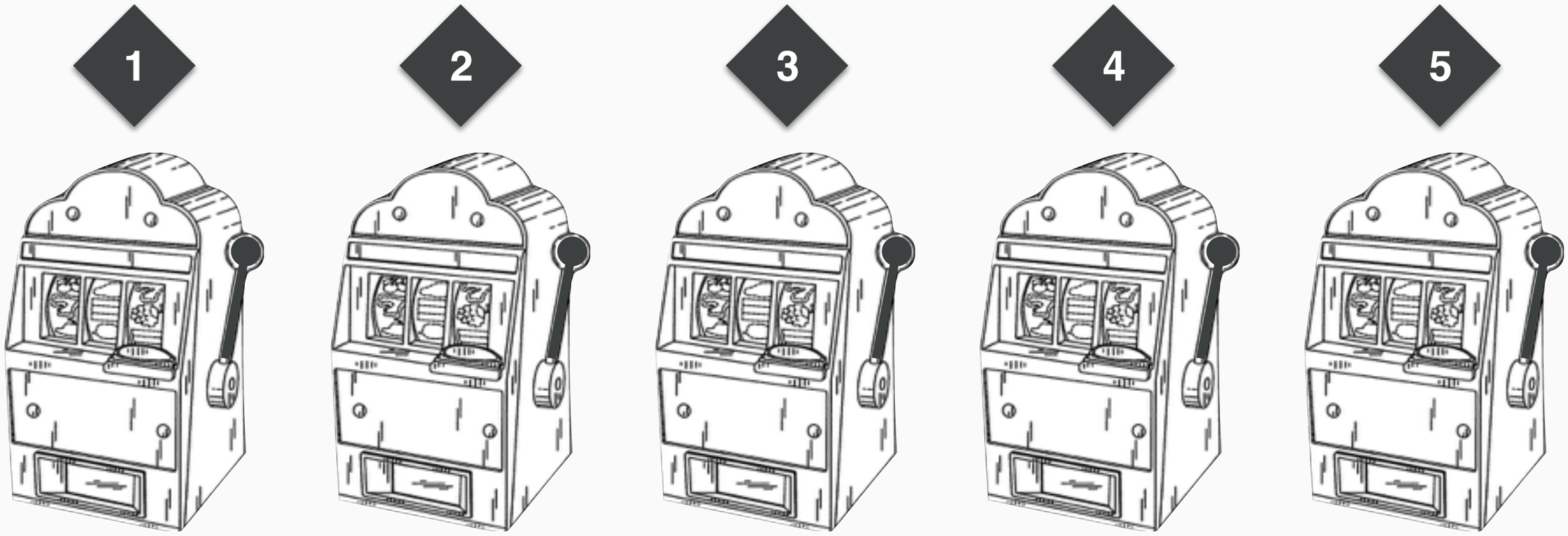


???	???	???	???	???

Slot Machines

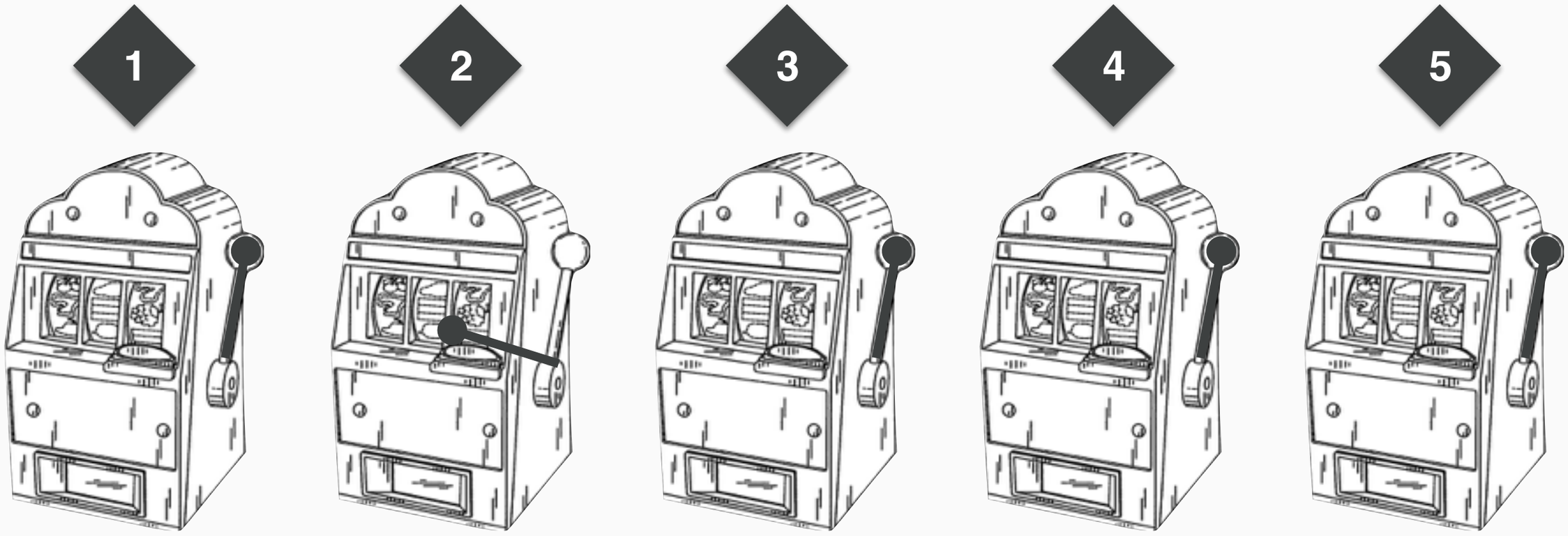


Slot Machines



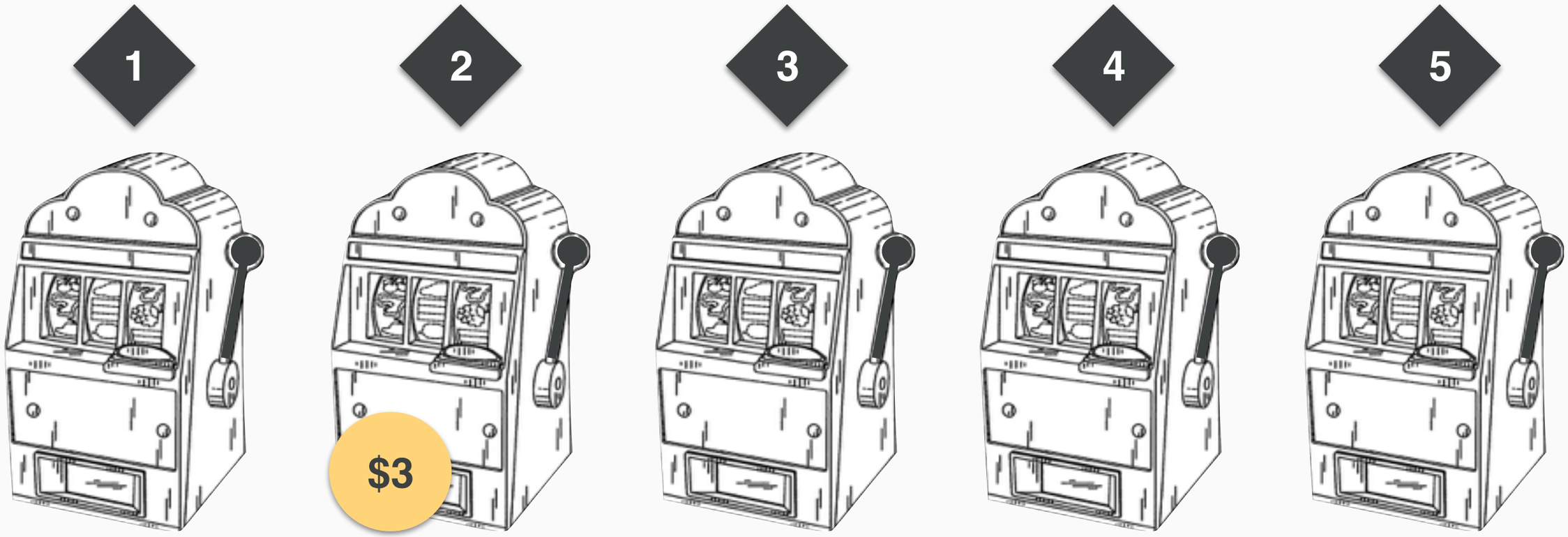
$2?	???	???	???	???

Slot Machines



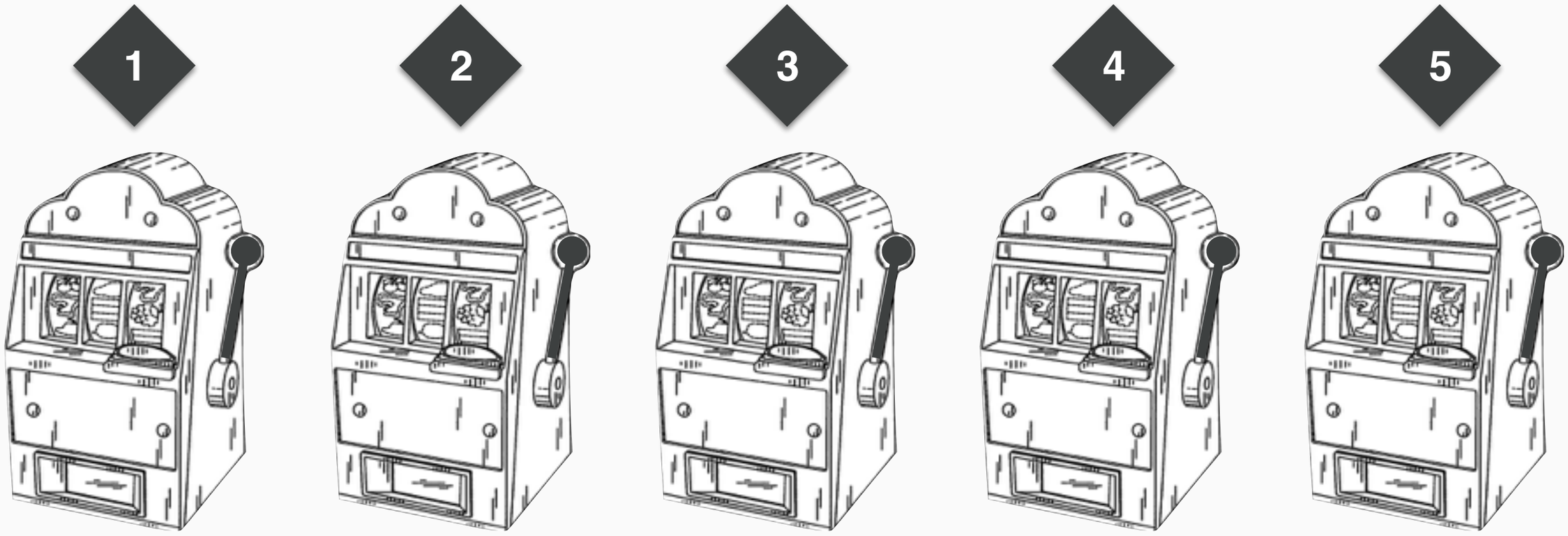
$2?	???	???	???	???

Slot Machines



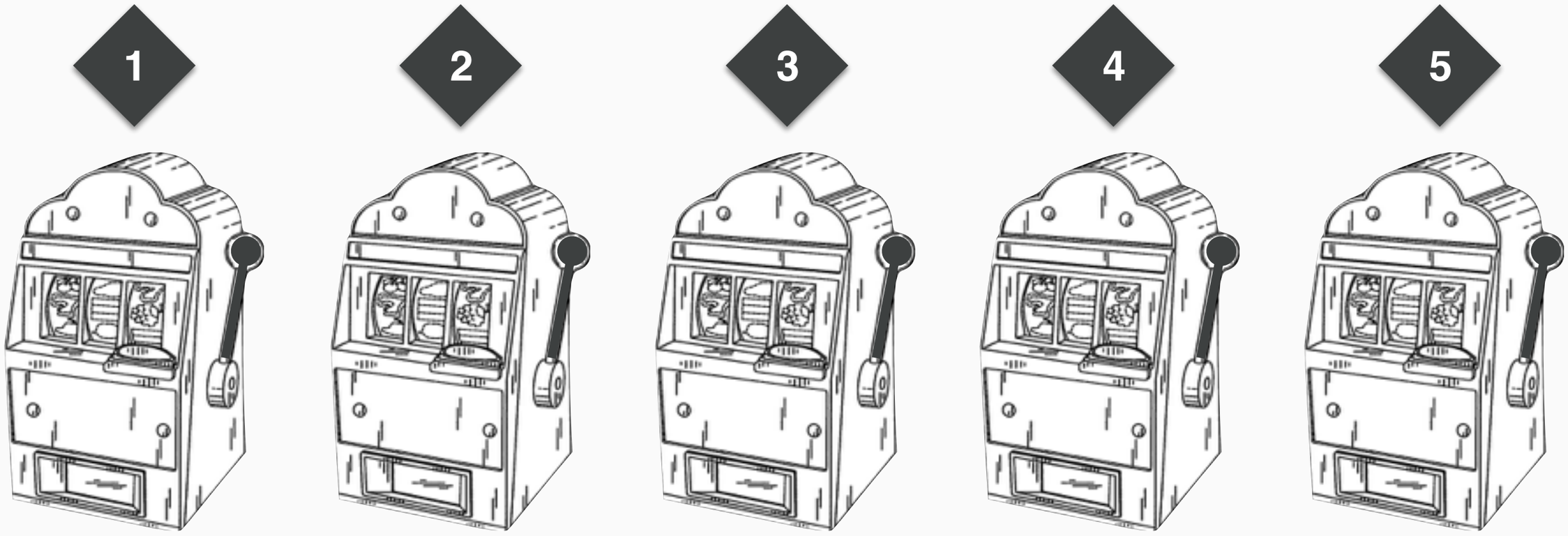
$2?	???	???	???	???

Slot Machines



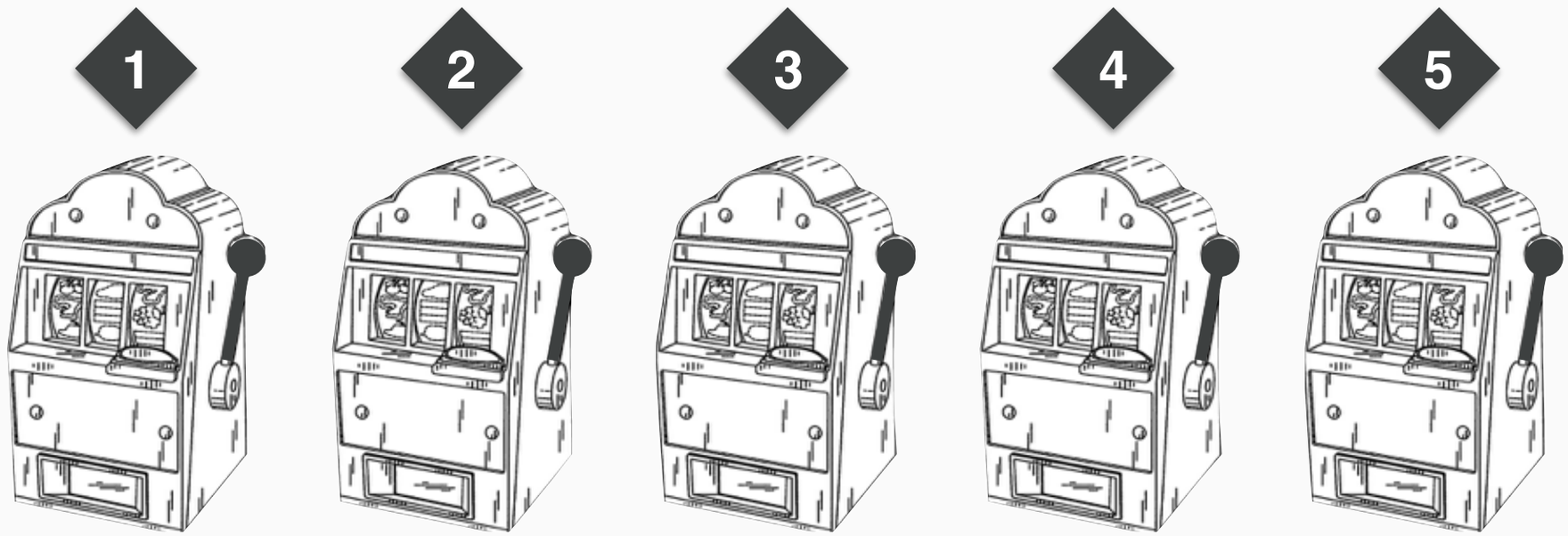
$2?	$3?	???	???	???

Slot Machines



Issue: How do we trade off *exploring* new strategies with *exploiting* the information we have now?

Slot Machines



Issue: How do we trade off *exploring* new strategies with *exploiting* the information we have now?

Cool note: general model of decision making, in life!

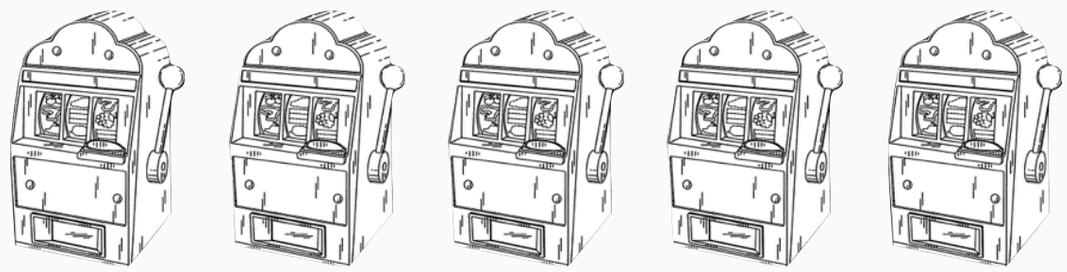
Slot Machines

- What city to live in?
- What apartment to move in to?
- Who to date/marry?
- What field to concentrate in?
- What food to order at a restaurant?
- What instrument to learn?

Issue: How do we trade off *exploring* new strategies with *exploiting* the information we have now?

Problem: Slot Machines

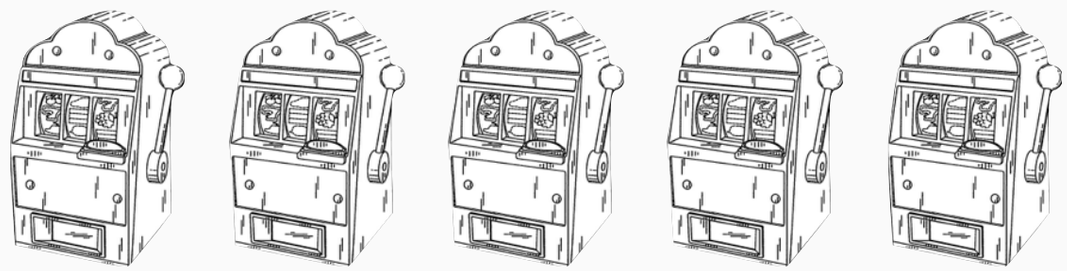
- INPUT: some number of slot machines, K, and T tokens for playing (so you get to pull T times).
- OUTPUT: Some amount of money made (want to be as high as possible!)



Issue: How do we trade off *exploring* new strategies with *exploiting* the information we have now?

Algorithm 1: Dora The Explora!

- Pull every arm equally often so you know which arm is probably the best!

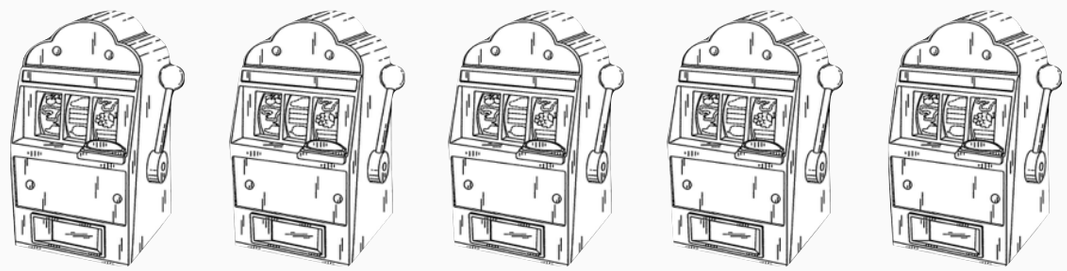


Issue: How do we trade off *exploring* new strategies with *exploiting* the information we have now?

Algorithm 1: Dora The Explora!

- Pull every arm equally often so you know which arm is probably the best!

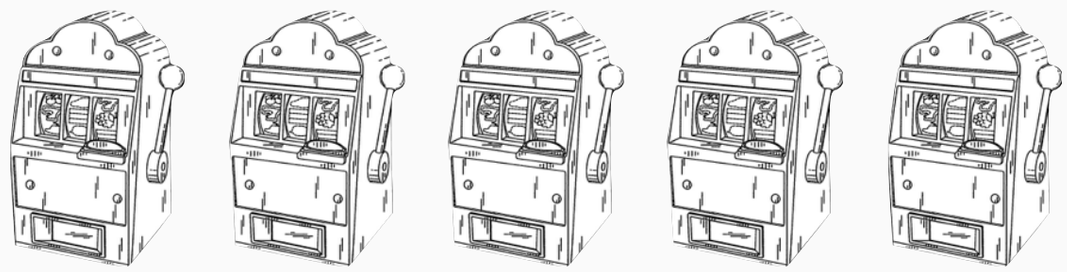
- Explores *way too much*. (We *never* exploit!)



Issue: How do we trade off *exploring* new strategies with *exploiting* the information we have now?

Algorithm 2: Hoyt The Exployter!

- Pull each arm *once.*
- Then pull the arm that paid the most until you run out of tokens.



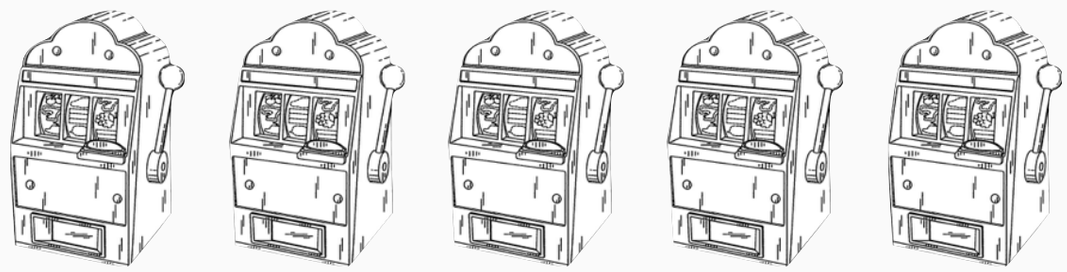
Issue: How do we trade off *exploring* new strategies with *exploiting* the information we have now?

Algorithm 2: Hoyt The Exployter!

- Pull each arm *once.*

- Then pull the arm that paid the most until you run out of tokens.

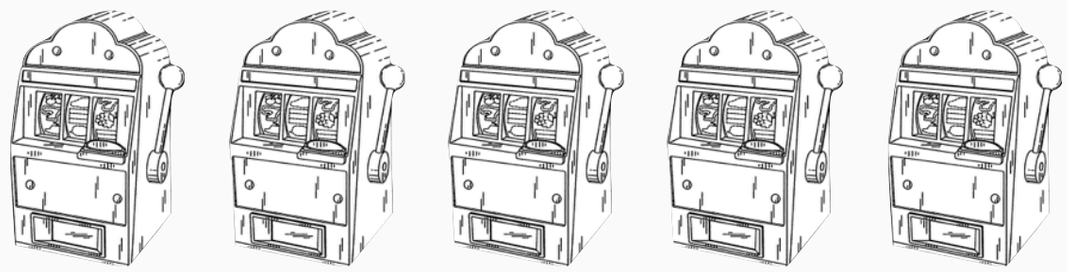
- Exploits way too much! (We barely explore..)



Issue: How do we trade off *exploring* new strategies with *exploiting* the information we have now?

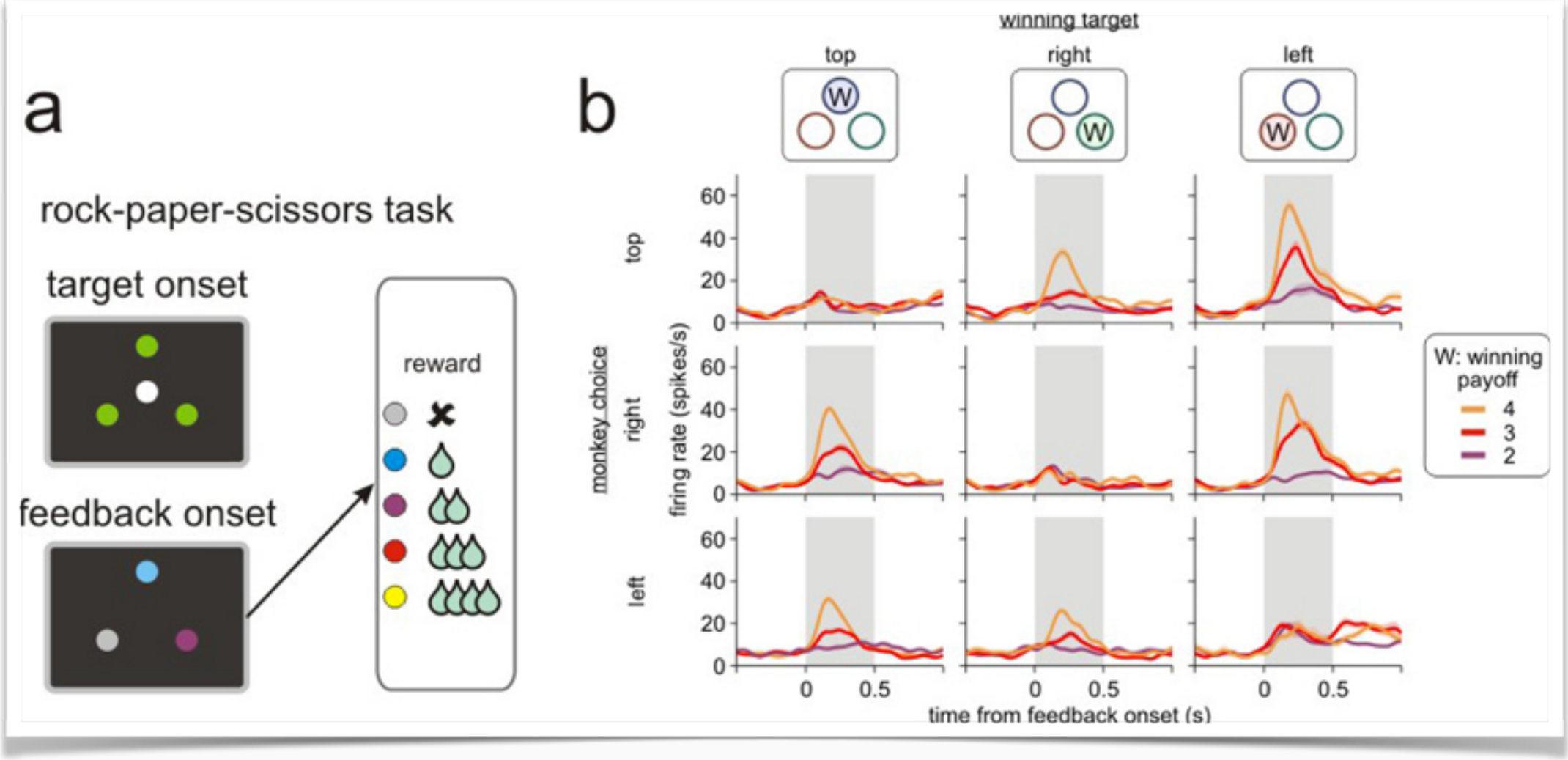
The Goal

- Q: How do we *perfectly* balance exploration and exploitation?



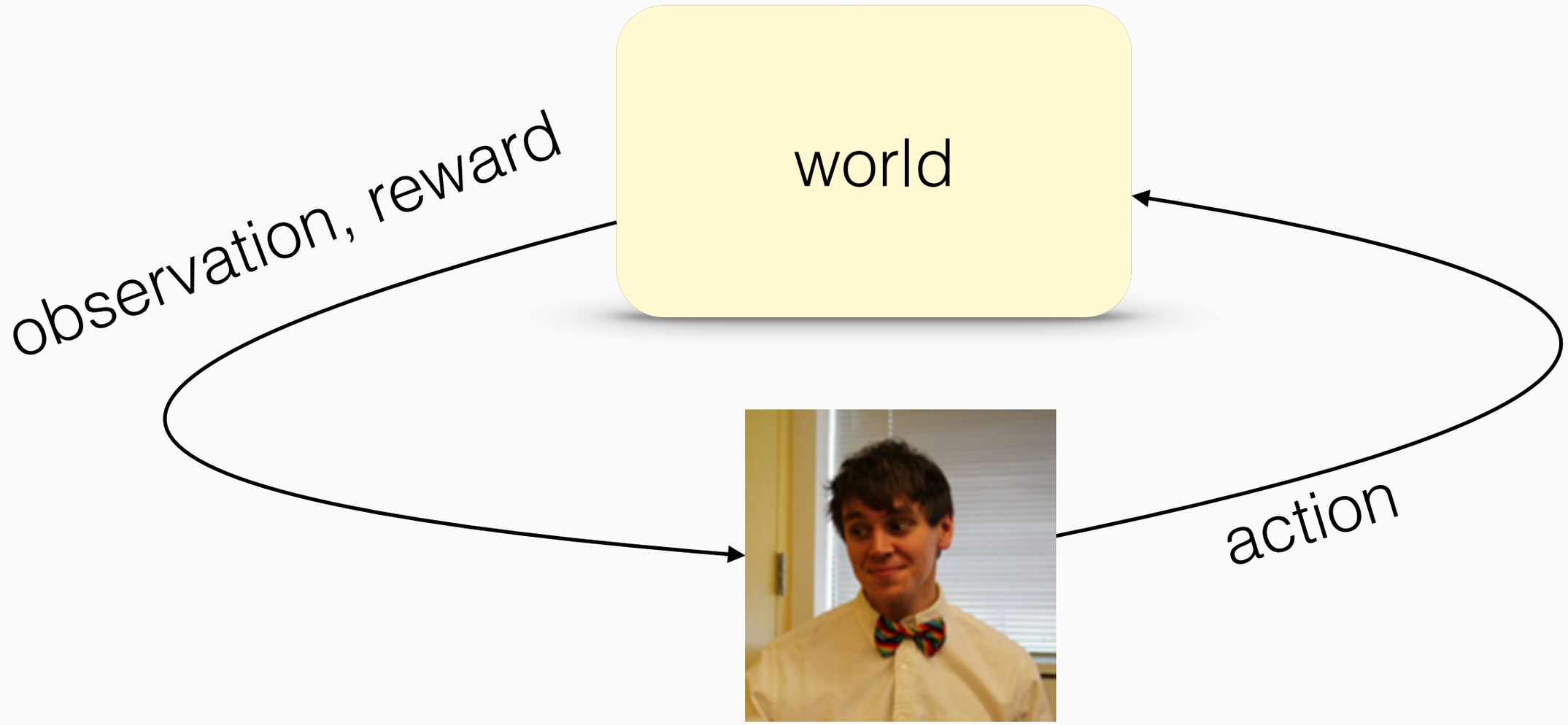
Issue: How do we trade off *exploring* new strategies with *exploiting* the information we have now?

Biology and Reinforcement Learning



Lee, D., Seo, H., & Jung, M. W. (2012). Neural basis of reinforcement learning and decision making. Annual review of neuroscience, 35, 287.

Reinforcement Learning



One Scratch Note…

Let’s Take a Look!

AI Reflection

- Learning can be formalized as an algorithm
- Classification is learning rules to determine concepts, based on labeled examples.
- Reinforcement Learning is a (biologically inspired) *super general* model of learning based on rewards and punishment.
- An exciting road ahead!