Deep Learning

Week 15

Intro to Reinforcement Learning

Why you should care

Supervised learning

Given:

- objects and answers $\left(x, y\right)$
- algorithm family $a_\theta\left(x\right) \rightarrow y$
- loss function $L\left(y, a_\theta\left(x\right)\right)$

Find:

$$\theta' \leftarrow argmin_\theta L\left(y, a_\theta\left(x\right)\right)$$

Supervised learning

Given:

- objects and answers $\left(x, y\right)$

  [banner,page], ctr

- algorithm family $a_{\theta}\left(x\right) \rightarrow y$

  linear / tree / NN

- loss function $L\left(y, a_{\theta}\left(x\right)\right)$

  MSE, crossentropy

Find:

$$\theta' \leftarrow argmin_{\theta} L\left(y, a_{\theta}\left(x\right)\right)$$

Supervised learning

Great... except if we have no reference answers

Online Ads

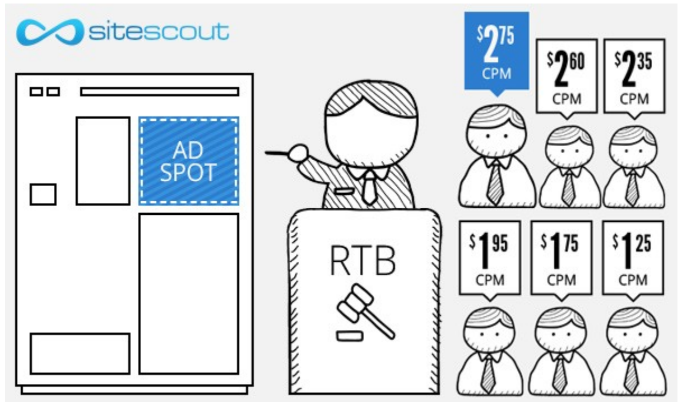
Great... except if we have no reference answers

We have:

- YouTube at your disposal
- Live data stream
  (banner & video features, #clicked)
- (insert your favorite ML toolkit)

We want:

- Learn to pick relevant ads

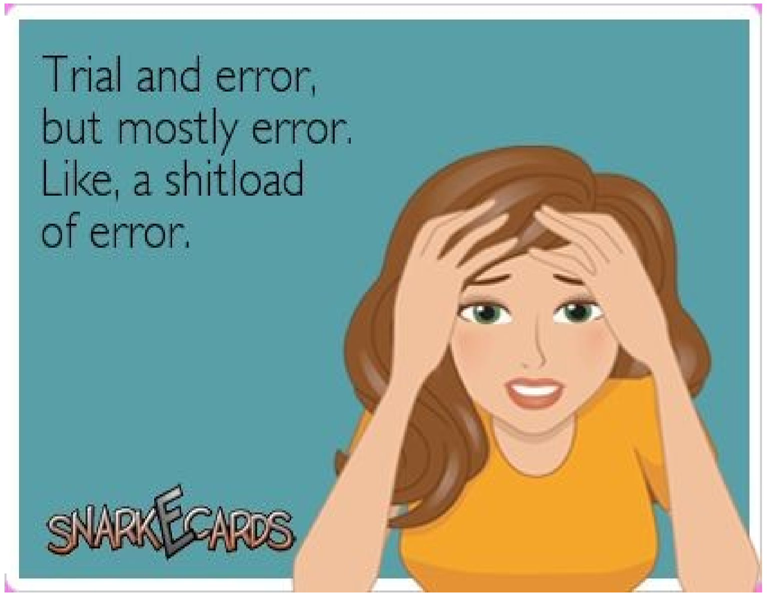


Ideas?

Duct tape approach

Common idea:

- Initialize with naïve solution
- Get data by trial and error and error and error and error
- Learn (situation) → (optimal action)
- Repeat



Giant Death Robot (GDR)

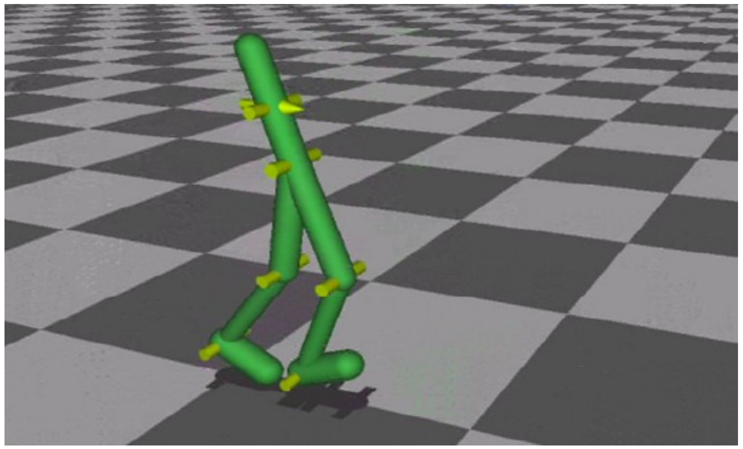
Great... except if we have no reference answers

We have:

- Evil humanoid robot
- A lot of spare parts to repair it :)

We want:

- ~~Enslave humanity~~
- Learn to walk forward

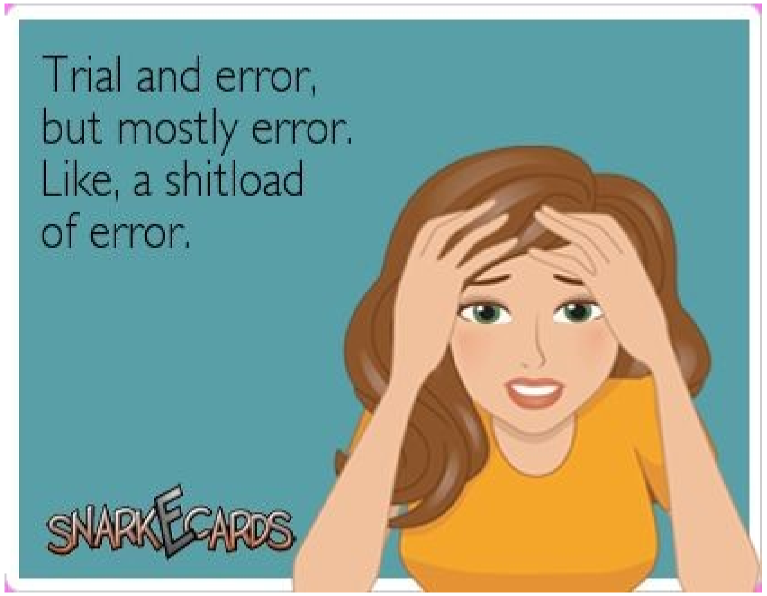


Ideas?

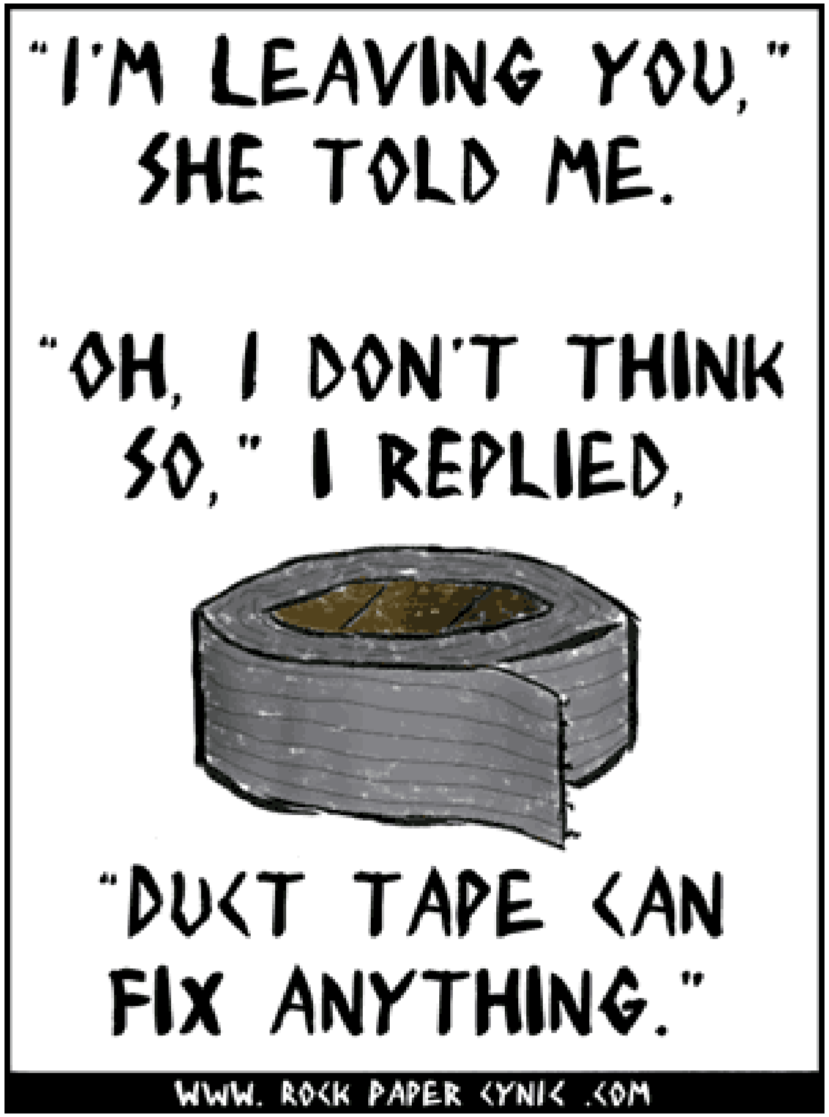
Duct tape approach (again)

Common idea:

- Initialize with naïve solution
- Get data by trial and error and error and error and error
- Learn (situation) → (optimal action)
- Repeat


Trial and error, but mostly error. Like, a shitload of error.

Duct tape approach


"I'M LEAVING YOU," SHE TOLD ME.
"OH, I DON'T THINK SO," I REPLIED,
"DUCT TAPE CAN FIX ANYTHING."
WWW. ROCK PAPER CYNIC .COM

Problems

Problem 1:

- What exactly does the “optimal action” mean?

Extract as much money as you can right now

VS

Make user happy so that he would visit you again

Problems

Problem 2:

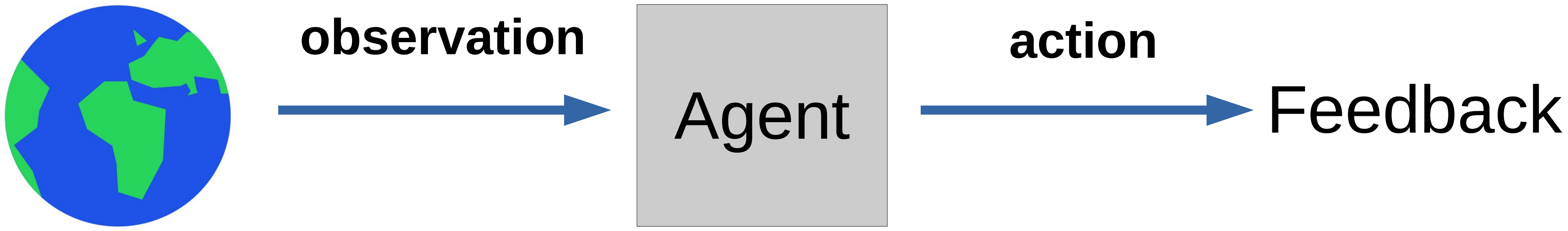
- If you always follow the “current optimal” strategy, you may never discover something better.
- If you show the same banner to 100% users, you will never learn how other ads affect them.

Ideas?

Duct tape approach

Reinforcement learning

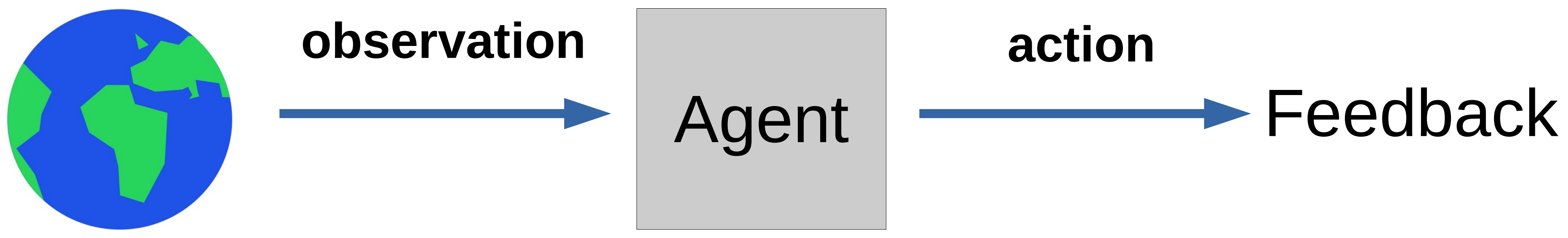
What is: bandit



Examples:

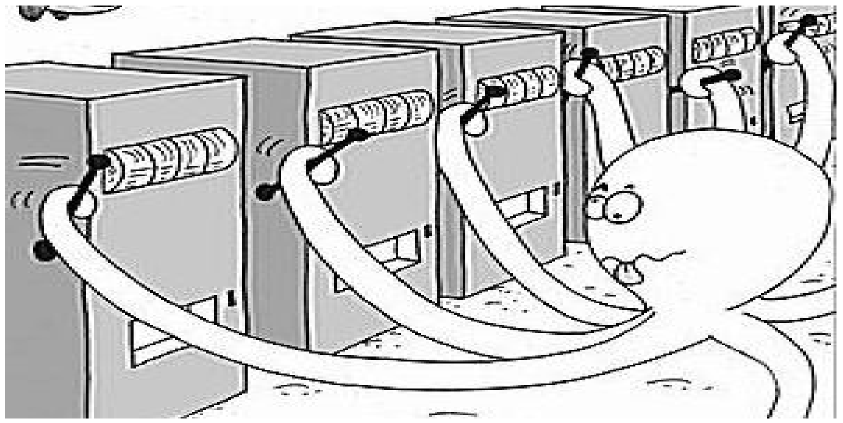
- banner ads (RTB)
- recommendations
- medical treatment

What is: bandit

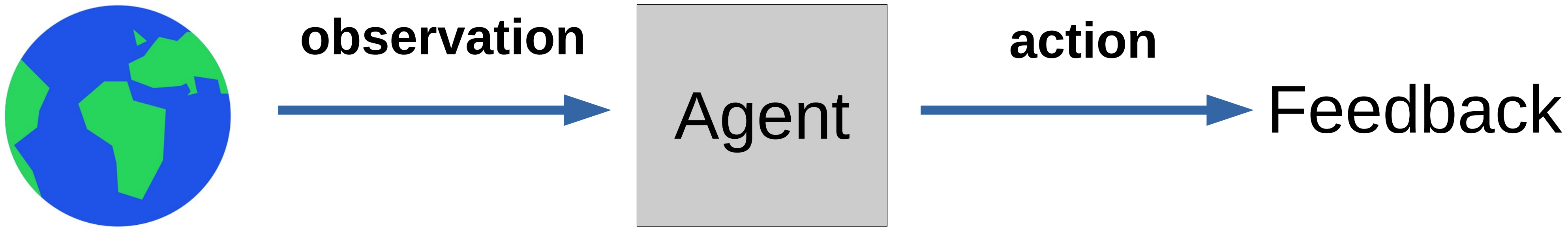


Examples:

- banner ads (RTB)
- recommendations
- medical treatment



What is: bandit

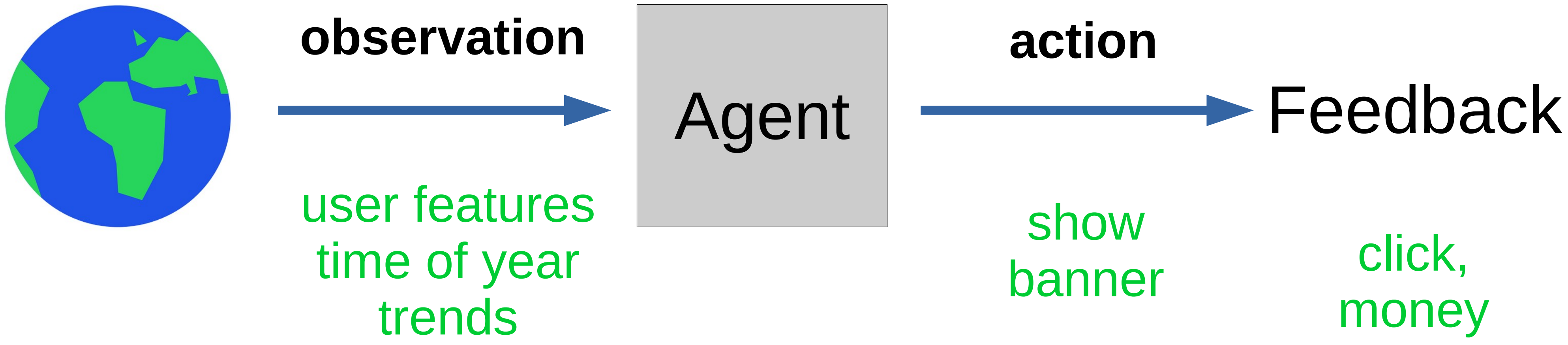


Examples:

- banner ads (RTB)
- recommendations
- medical treatment

Q: what's observation, action and feedback in the banner ads problem?

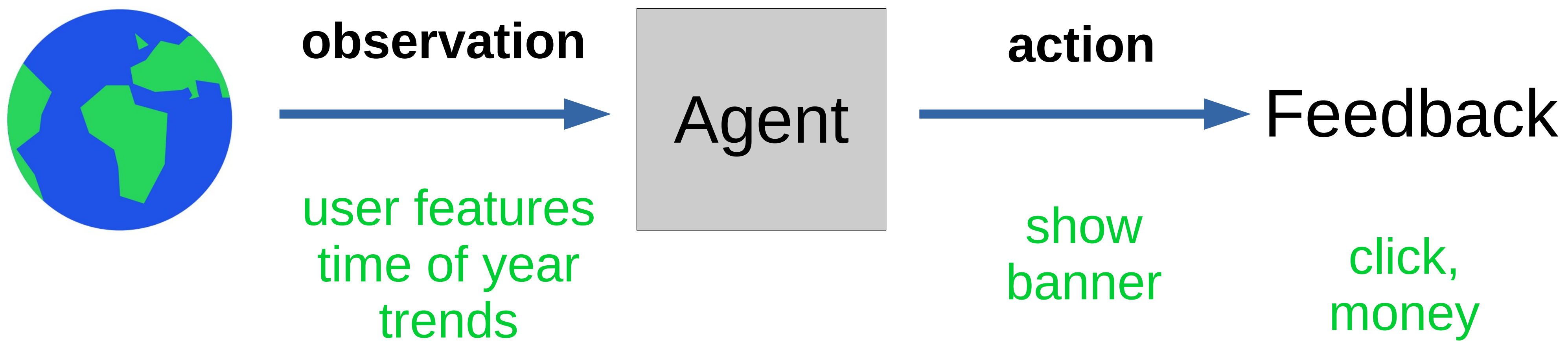
What is: bandit



Examples:

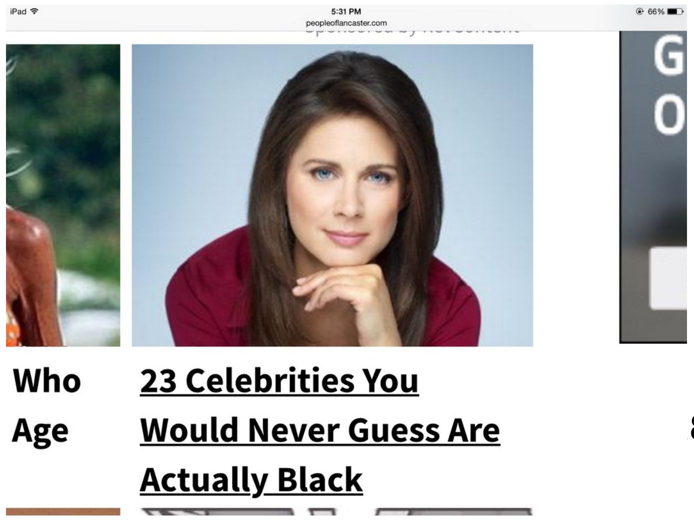
- banner ads (RTB)
- recommendations
- medical treatment

What is: bandit

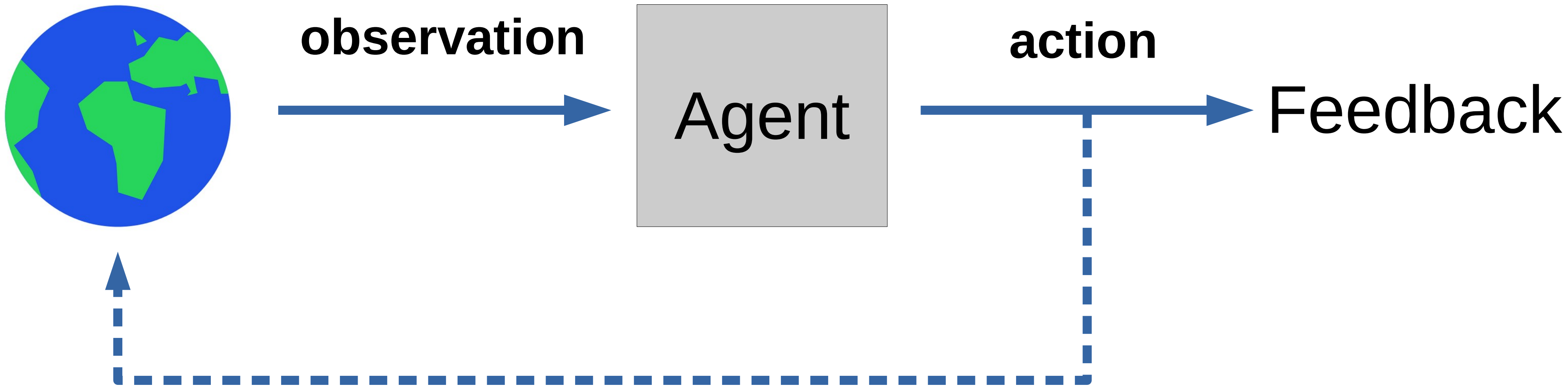


Q: You're Yandex/Google/Youtube.
There's a kind of banners that would have great click rates: the “clickbait”.

Is it a good idea to show clickbait?

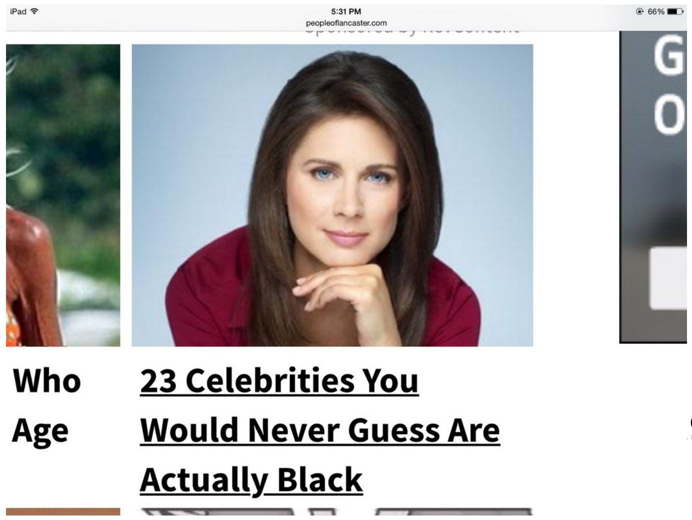


What is: bandit

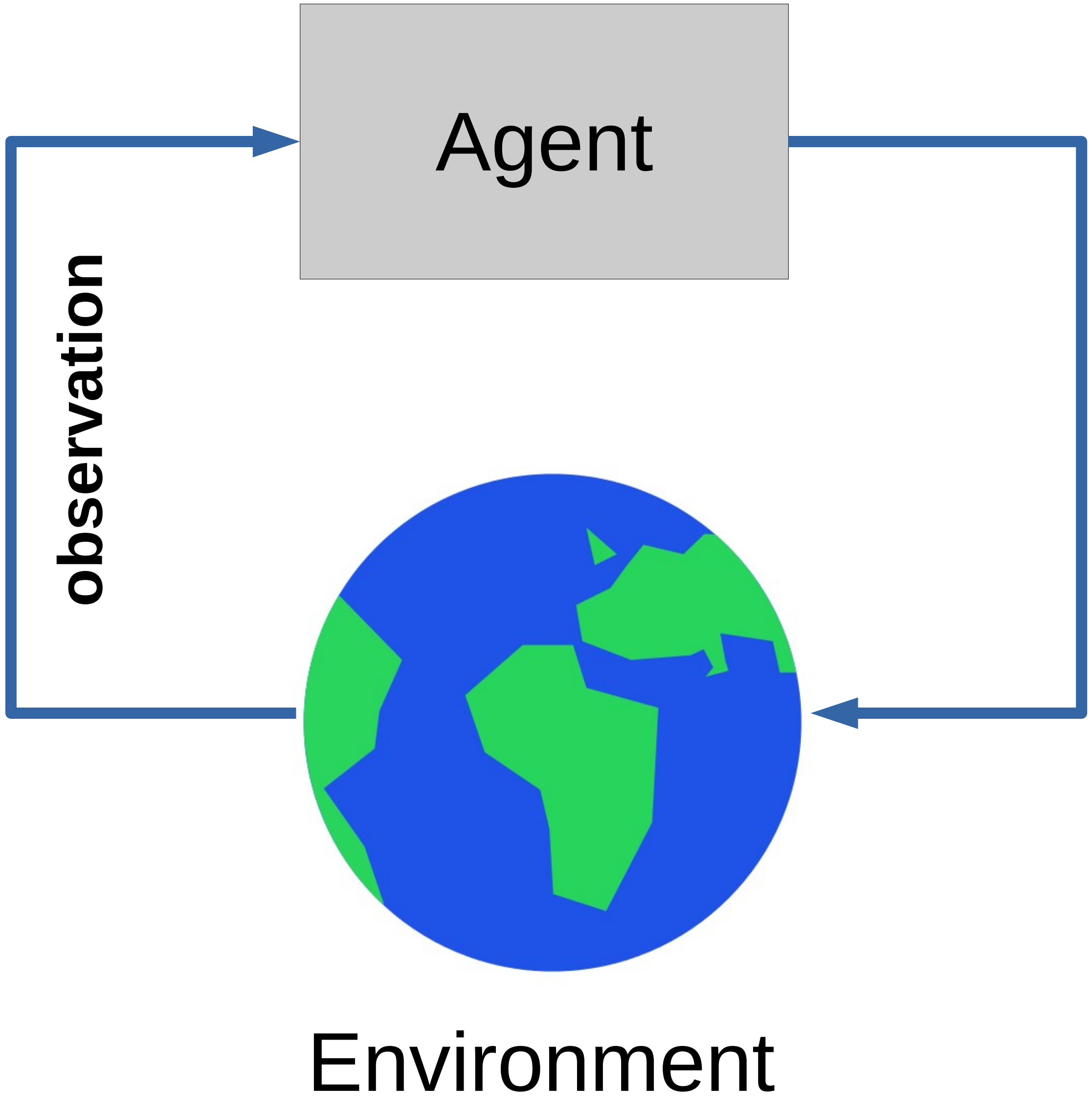


Q: You're Yandex/Google/Youtube. There's a kind of banners that would have great click rates: the “clickbait”.

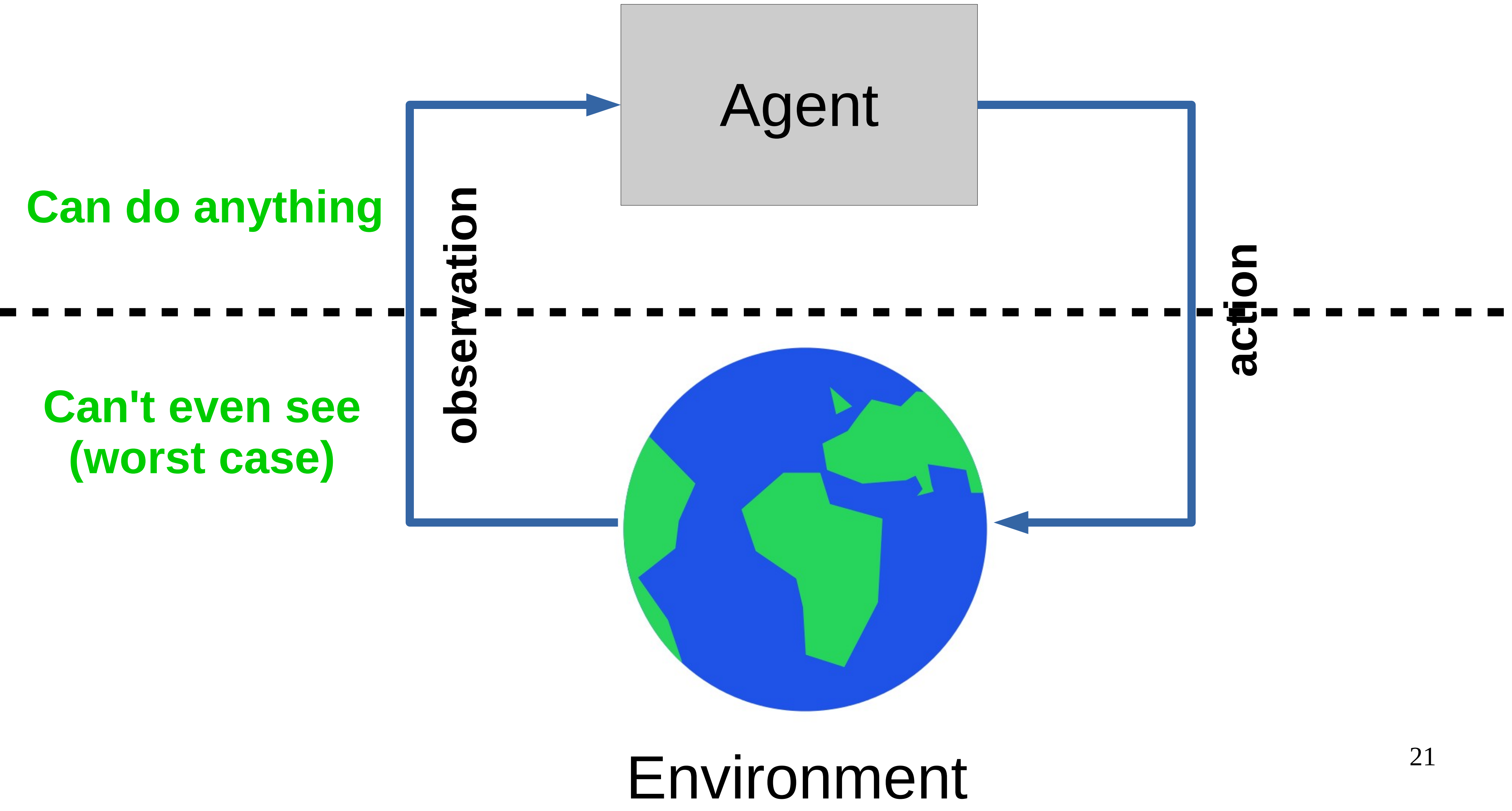
Is it a good idea to show clickbait?
No, no one will trust you after that!



What is: decision process



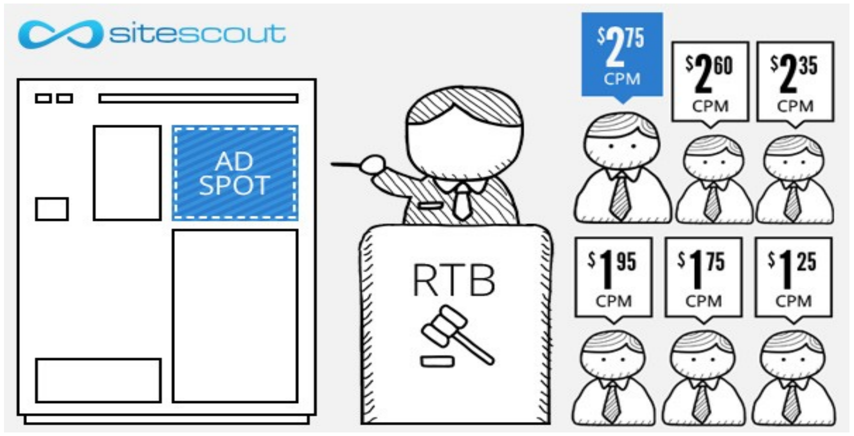
What is: decision process

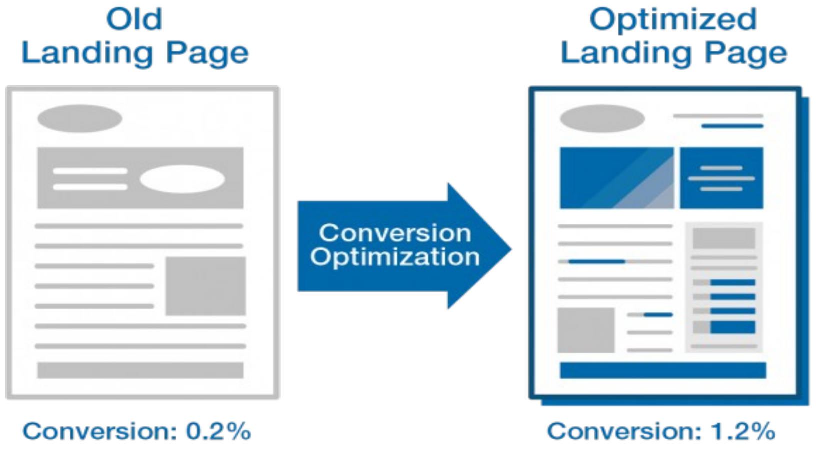


Reality check: web

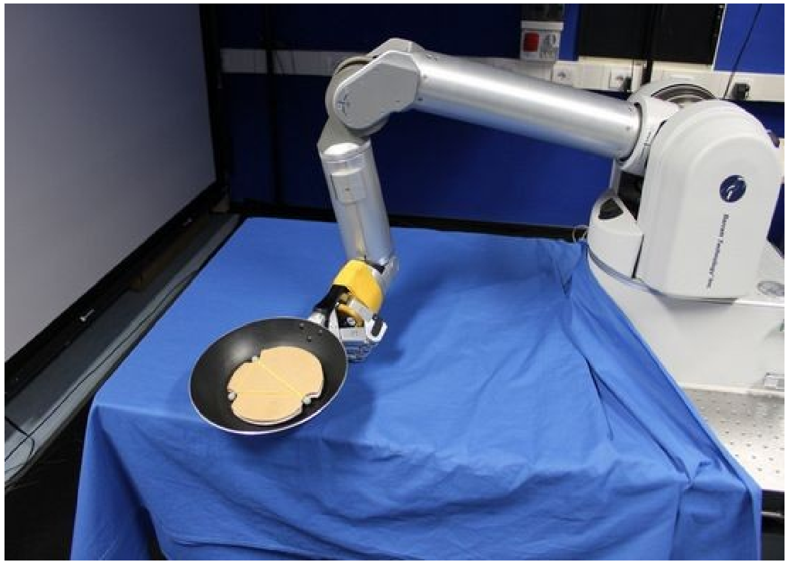
- **Cases:**
  - Pick ads to maximize profit
  - Design landing page to maximize user retention
  - Recommend movies to users
  - Find pages relevant to queries

- **Example**
  - Observation – user features
  - Action – show banner #i
  - Feedback – did user click?





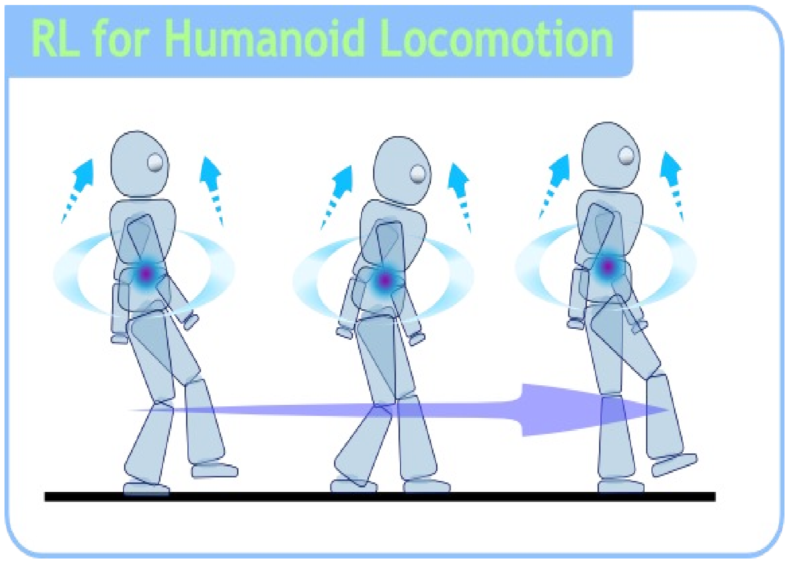
Reality check: dynamic systems



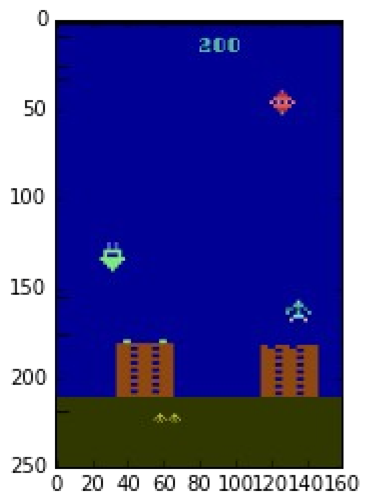
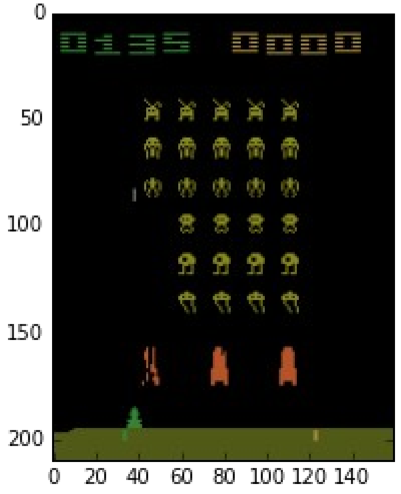
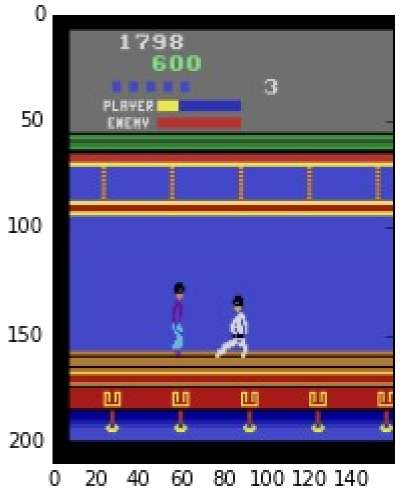
Reality check: dynamic systems

- **Cases:**
  - Robots
  - Self-driving vehicles
  - Pilot assistant
  - More robots!

- **Example**
  - Observation: sensor feed
  - Action: voltage sent to motors
  - Feedback: how far did it move forward before falling



~~Reality~~ check: videogames

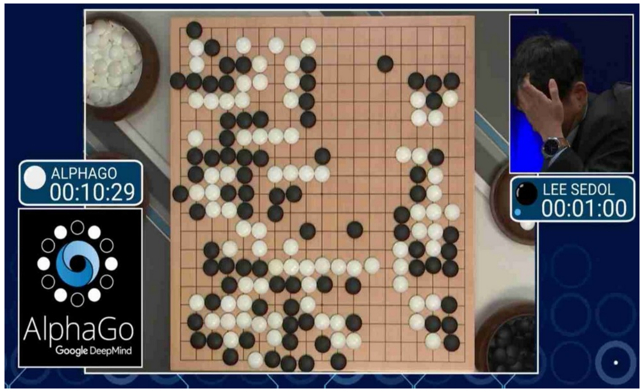




- **Q:** What are observations, actions and feedback?

Other use cases

- Personalized medical treatment

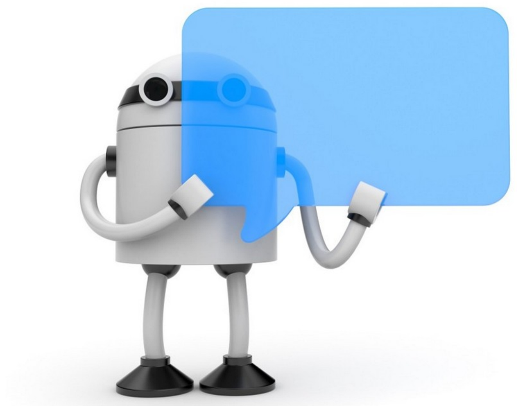
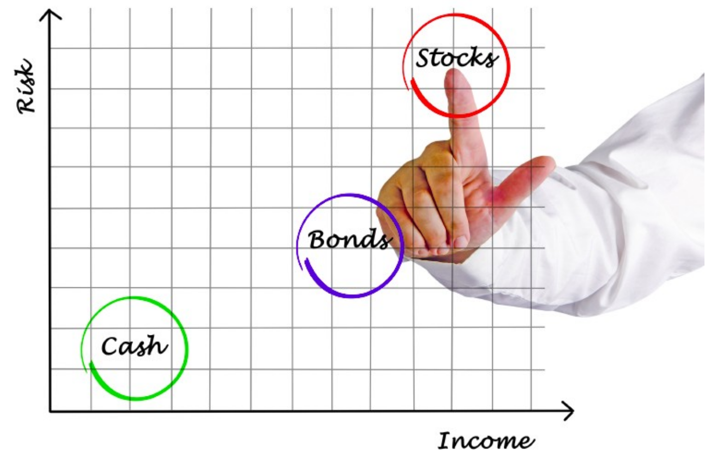
- Even more games (Go, chess, etc)

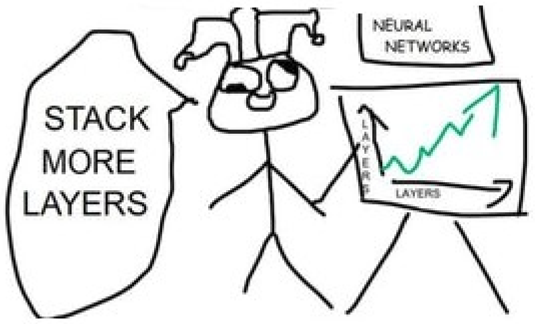


- **Q:** What are observations, actions and feedback?

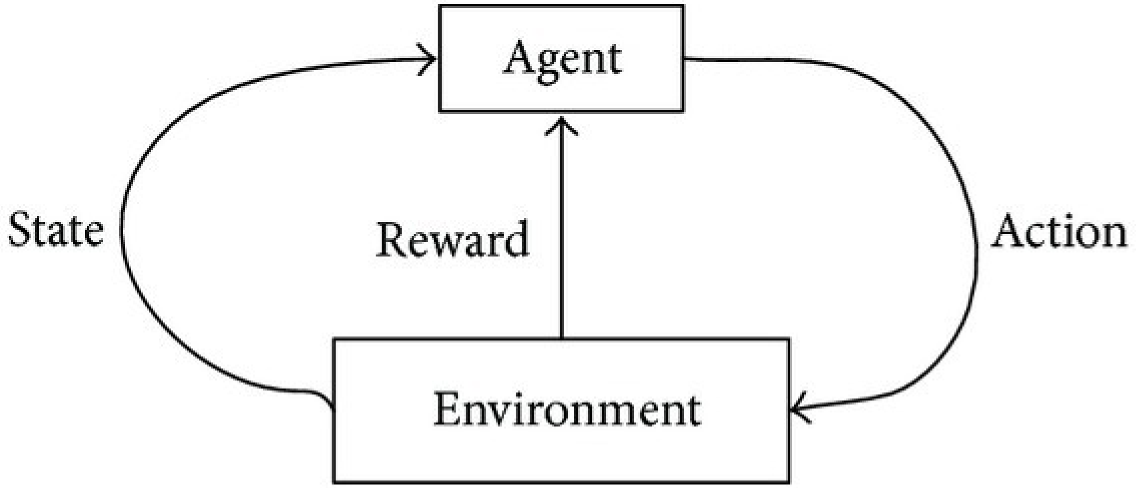
Other use cases

- Conversation systems
  - learning to make user happy
- Quantitative finance
  - portfolio management
- Deep learning
  - optimizing non-differentiable loss
  - finding optimal architecture



The MDP formalism

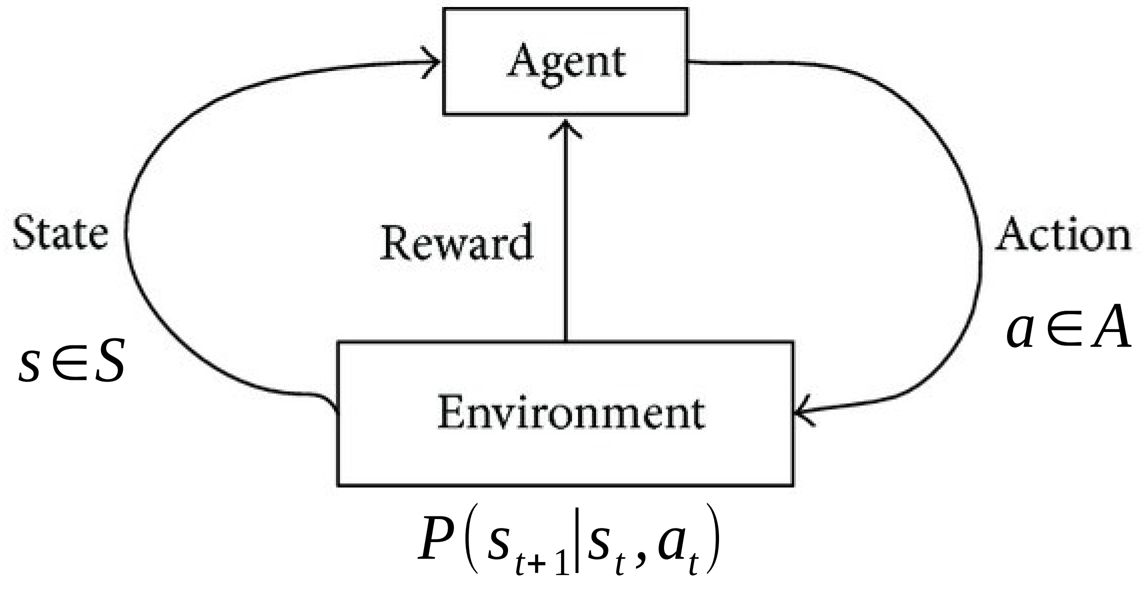


Markov Decision Process

- Environment states: $s \in S$
- Agent actions: $a \in A$
- Rewards $r \in \mathbb{R}$

- Dynamics: $P(s_{t+1} | s_t, a_t)$

The MDP formalism

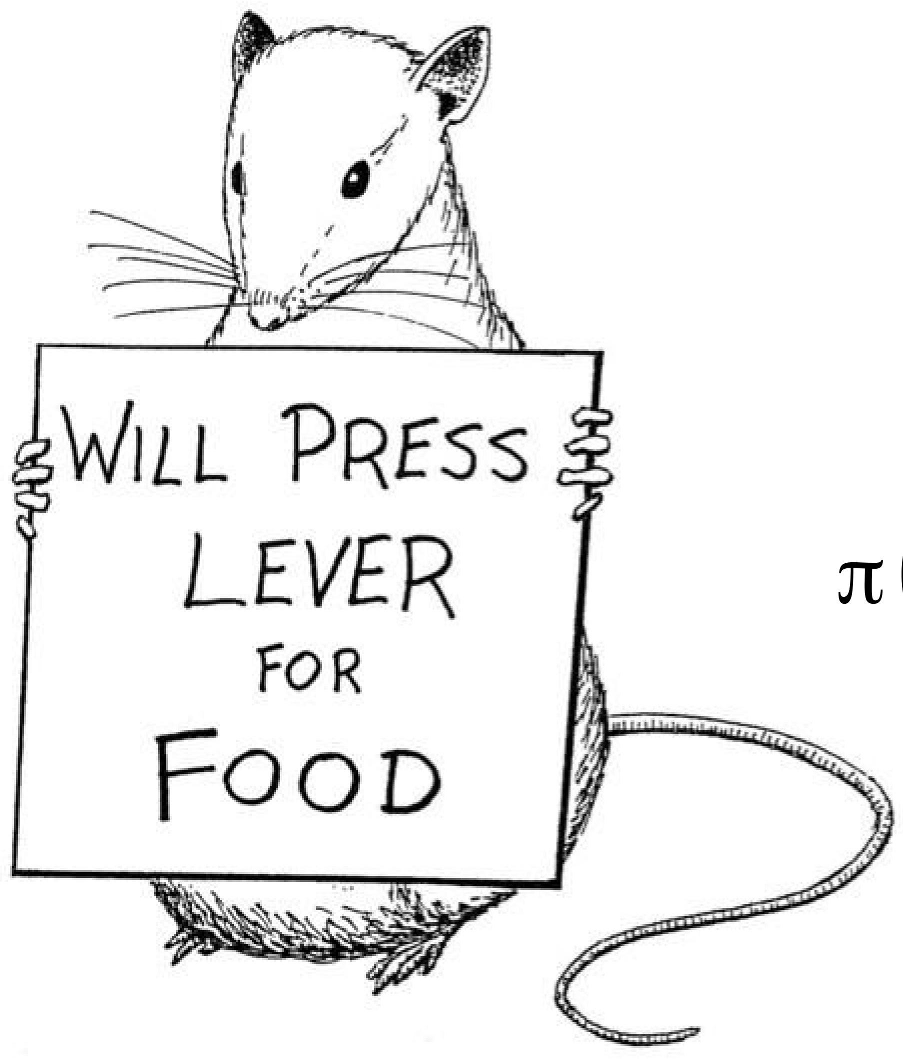


Markov Decision Process

Markov assumption

$$P(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}) = P(s_{t+1}|s_t, a_t)$$

Total reward



Total reward for session:

$$R=\sum_t r_t$$

Agent's policy:

$$\pi(a|s)=P(\text{take action } a|\text{in state } s)$$

Problem: find policy with highest reward:

$$\pi(a|s): E_\pi[R] \rightarrow max$$

Objective

The easy way:

$Q = E_\pi R$ is an expected sum of rewards that agent with policy π earns per session

Objective

The easy way:

$Q = E_{\pi} R$ is an expected sum of rewards that agent with policy π earns per session

The hard way:

$$\underset{s_0 \sim p(s_0),\, a_0 \sim \pi(a|s_0),}{E} \quad \underset{s_1, r_0 \sim P(s',r|s,a)}{E} \quad \underset{}{E} \quad \dots \quad \underset{s_T, r_T \sim P(s',r|s_{T-1},a_{t-1})}{E} \left[r_0 + r_1 + r_2 + \dots + r_T \right]$$

How do we solve it?

General idea:

Play a few sessions

Update your policy

Repeat

Objective

$$J = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(s|a)}}{E} Q(s,a) = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a) \, da \, ds$$

Objective

Expected reward:

$$J = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(s|a)}}{E} Q(s,a)$$

We need a gradient!

Objective

Agent's policy

$$J = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(s|a)}}{E} Q(s,a) = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a) \, da \, ds$$

state visitation frequency
(may depend on policy)

True action value

Q: how do we compute that?

Objective

$$J = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(s|a)}}{E} Q(s,a) = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a) \, da \, ds$$

True action value
a.k.a. E[R(s,a)]

$$J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} Q(s,a)$$

sample N sessions

Objective

$$J = \mathop{E}_{\substack{s \sim p(s) \\ a \sim \pi_\theta(s|a)}} Q(s,a) = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a)\, da\, ds$$

True action value

a.k.a. E[R(s,a)]

$$J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} Q(s,a)$$

sample N sessions

Can we optimize policy now?

Objective

$$J = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(s|a)}}{E} Q(s,a) = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a)\, da\, ds$$

parameters "sit" here

True action value
a.k.a. E[R(s,a)]

$$J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} Q(s,a)$$

We don't know how to compute dJ/dtheta

Objective

$$J = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(s|a)}}{E} Q(s,a) = \int_s p(s) \int_a \pi_\theta(a|s) Q(s,a)\, da\, ds$$

Wish list:

– Analytical gradient

– Easy/stable approximations

Log-derivative trick

Simple math

$$\nabla \log \pi(z) = ???$$

(try chain rule)

Log-derivative trick

Simple math

$$\nabla \log \pi(z) = \frac{1}{\pi(z)} \cdot \nabla \pi(z)$$

$$\pi \cdot \nabla \log \pi(z) = \nabla \pi(z)$$

Policy gradient

Analytical inference

$$\nabla J = \int_s p(s) \int_a \nabla \pi_\theta(a|s) Q(s,a) \, da \, ds$$

$$\pi \cdot \nabla \log \pi(z) = \nabla \pi(z)$$

Policy gradient

Analytical inference

$$\nabla J = \int_s p(s) \int_a \nabla \pi_\theta(a|s) Q(s,a)\, da\, ds$$

$$\pi \cdot \nabla \log \pi(z) = \nabla \pi(z)$$

$$\nabla J = \int_s p(s) \int_a \pi_\theta(a|s) \nabla \log \pi_\theta(a|s) Q(s,a)\, da\, ds$$

Trivia: anything curious about that formula?

Policy gradient

Analytical inference

$$\nabla J = \int_s p(s) \int_a \nabla \pi_\theta(a|s) Q(s,a)\, da\, ds$$

$$\pi \cdot \nabla \log \pi(z) = \nabla \pi(z)$$

$$\nabla J = \underbrace{\int_s p(s) \int_a \pi_\theta(a|s)}_{\text{that's expectation :)}} \nabla \log \pi_\theta(a|s) Q(s,a)\, da\, ds$$

Policy gradient

Analytical inference

$$\nabla J = \int_s p(s) \int_a \nabla \pi_\theta(a|s) Q(s,a)\, da\, ds$$

$$\pi \cdot \nabla \log \pi(z) = \nabla \pi(z)$$

$$\nabla J = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(s|a)}}{E} \nabla \log \pi_\theta(a|s) \cdot Q(s,a)$$

Policy gradient (REINFORCE)

- Policy gradient

$$\nabla J = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(s|a)}}{E} \nabla \log \pi_\theta(a|s) \cdot Q(s,a)$$

- Approximate with sampling

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \nabla \log \pi_\theta(a|s) \cdot Q(s,a)$$

REINFORCE algorithm

- Initialize NN weights $\theta_0 \leftarrow random$

- Loop:
  - Sample N sessions **z** under current $\pi_\theta(a|s)$
  - Evaluate policy gradient

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \nabla \log \pi_\theta(a|s) \cdot Q(s,a)$$

  - Ascend $\theta_{i+1} \leftarrow \theta_i + \alpha \cdot \nabla J$

REINFORCE baselines

- Initialize NN weights $\theta_0 \leftarrow random$
- Loop:
  - Sample N sessions **z** under current $\pi_\theta(a|s)$
  - Evaluate policy gradient

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \nabla \log \pi_\theta(a|s) \cdot Q(s,a)$$

What is better for learning:
random action in good state
or
great action in bad state?

REINFORCE baselines

We can subtract arbitrary baseline b(s)

$$\nabla J = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s) \left(Q(s,a) - b(s)\right) = \dots$$

$$\dots = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s) Q(s,a) - \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s) b(s) = \dots$$

REINFORCE baselines

We can subtract arbitrary baseline b(s)

$$\nabla J = \underset{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s)(Q(s,a)-b(s)) = \dots$$

$$\dots = \underset{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s) Q(s,a) - \underset{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s) b(s) = \dots$$

Q: Can you simplify the second term?

Note that b(s) does not depend on a

REINFORCE baselines

A: How to simplify the second term

$$\underset{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s) b(s) = \underset{s\sim p(s)}{E} b(s) \underset{a\sim\pi_\theta(a|s)}{E} \nabla \log \pi_\theta(a|s) =$$

$$\underset{s\sim p(s)}{E} b(s) \int_a \pi_\theta(a|s) \frac{\nabla \pi_\theta(a|s)}{\pi_\theta(a|s)} da = \underset{s\sim p(s)}{E} b(s) \int_a \nabla \pi_\theta(a|s) da =$$

$$\underset{s\sim p(s)}{E} b(s) \nabla \int_a \pi_\theta(a|s) da = \underset{s\sim p(s)}{E} b(s) \nabla 1 = 0$$

REINFORCE baselines

We can subtract arbitrary baseline b(s)

$$\nabla J = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s)(Q(s,a) - b(s)) = \ldots$$

$$\ldots = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s) Q(s,a) - \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s) b(s) = \ldots$$

Gradient direction doesn't change!

$$\ldots = \underset{\substack{s \sim p(s) \\ a \sim \pi_\theta(a|s)}}{E} \nabla \log \pi_\theta(a|s) Q(s,a)$$

REINFORCE baselines

- Gradient direction ∇J stays the same
- Variance may change

Gradient variance: $Var[Q(s,a)-b(s)]$

as a random variable over (s, a)

$$Var[Q(s,a)]-2\cdot Cov[Q(s,a),b(s)]+Var[b(s)]$$

REINFORCE baselines

- Gradient direction ∇J stays the same
- Variance may change

Gradient variance: $Var[Q(s,a)-b(s)]$

as a random variable over (s, a)

$$Var[Q(s,a)]-2\cdot \underset{\uparrow}{Cov}[Q(s,a),b(s)]+Var[b(s)]$$

If b(s) correlates with Q(s,a), variance decreases

REINFORCE baselines

- Gradient direction ∇J stays the same
- Variance may change

Gradient variance: $Var[Q(s,a)-b(s)]$

as a random variable over (s, a)

$$Var[Q(s,a)]-2\cdot Cov[Q(s,a),b(s)]+Var[b(s)]$$

Q: can you suggest any such b(s)?

REINFORCE baselines

- Gradient direction ∇J stays the same
- Variance may change

Gradient variance: $Var[Q(s,a)-b(s)]$

as a random variable over (s, a)

$$Var[Q(s,a)]-2\cdot Cov[Q(s,a),b(s)]+Var[b(s)]$$

Naive baseline: b = moving average Q

over all (s, a), Var[b(s)] = 0, Cov[Q, b] > 0

Duct tape zone

- Superior algorithms exist
  - For harder environments google A3C, PPO, TRPO
- Regularize with entropy
  - to prevent premature convergence
- Learn on parallel sessions
  - Or super-small experience replay
- Use logsoftmax for numerical stability

Q: How is RL different from **supervised learning**?

What-what learning?

Supervised learning

- Learning to approximate reference answers
- Needs correct answers
- Model does not affect the input data

Reinforcement learning

- Learning optimal strategy by trial and error
- Needs feedback on agent's own actions
- Agent can affect it's own observations

What-what learning?

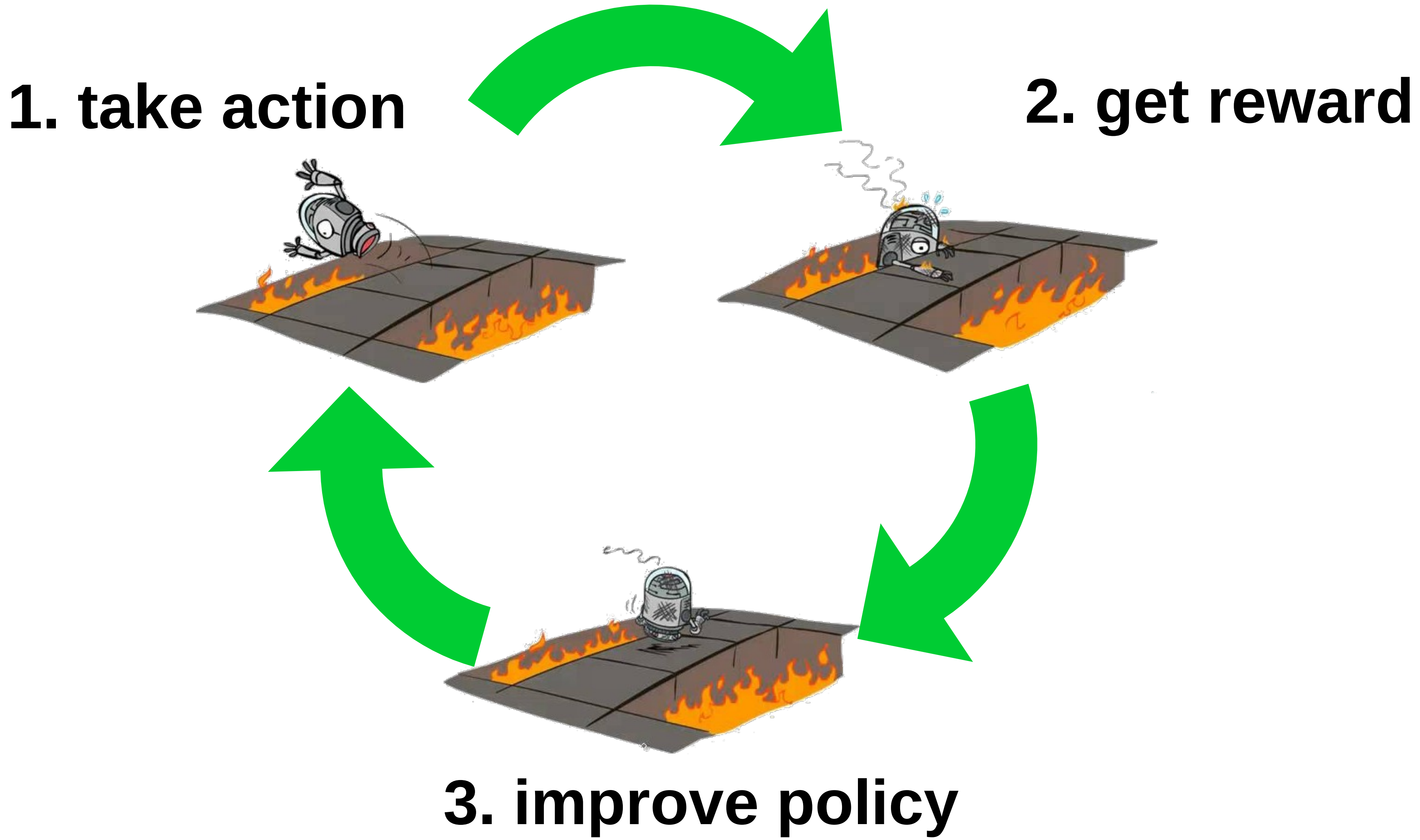
Unsupervised learning

- Learning underlying data structure
- No feedback required
- Model does not affect the input data

Reinforcement learning

- Learning optimal strategy by trial and error
- Needs feedback on agent's own actions
- Agent can affect its own observations

Reinforcement learning is easy!



Reinforcement learning is challenging!

How to explore?

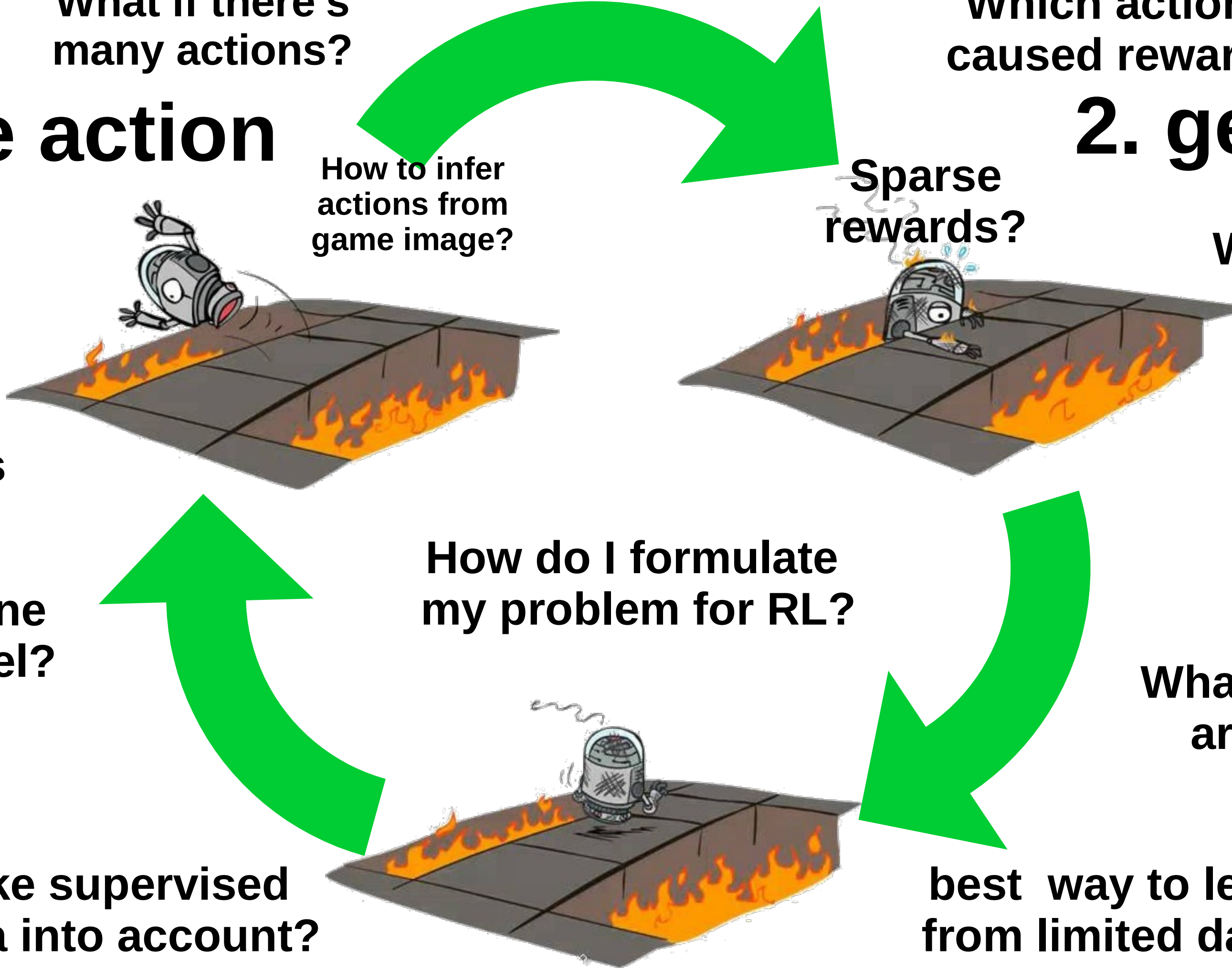
What if there's many actions?

Which actions caused reward?

Can agent cheat reward

1. take action

2. get reward



How to infer actions from game image?

Sparse rewards?

How not to break anyting when acting?

Which rewards are easier to learn?

Continuous actions?

What if there are multiple agents?

How do I formulate my problem for RL?

How to define policy model?

What if observations are incomplete?

When not to use RL?

take supervised data into account?

best way to learn from limited data?

did my algorithm overfit to simulation?

3. improve policy

How do I know if it converged or not?

Now go and implement that :)