

# NICO2AI #11

## 強化学習 I

妹尾 卓磨

- **強化学習の問題設定**

そもそも強化学習とは何だろうか

- **価値に基づく学習と探索**

実際にどのようにして学習を行うのか

- **基礎演習: OpenAI Gym 入門**

OpenAI Gym で遊ぶ

- **生物における強化学習**

生物は実際に強化学習をやっているのか

# 強化学習の問題設定

- **強化学習の問題設定**

そもそも強化学習とは何だろうか

- **価値に基づく学習と探索**

実際にどのようにして学習を行うのか

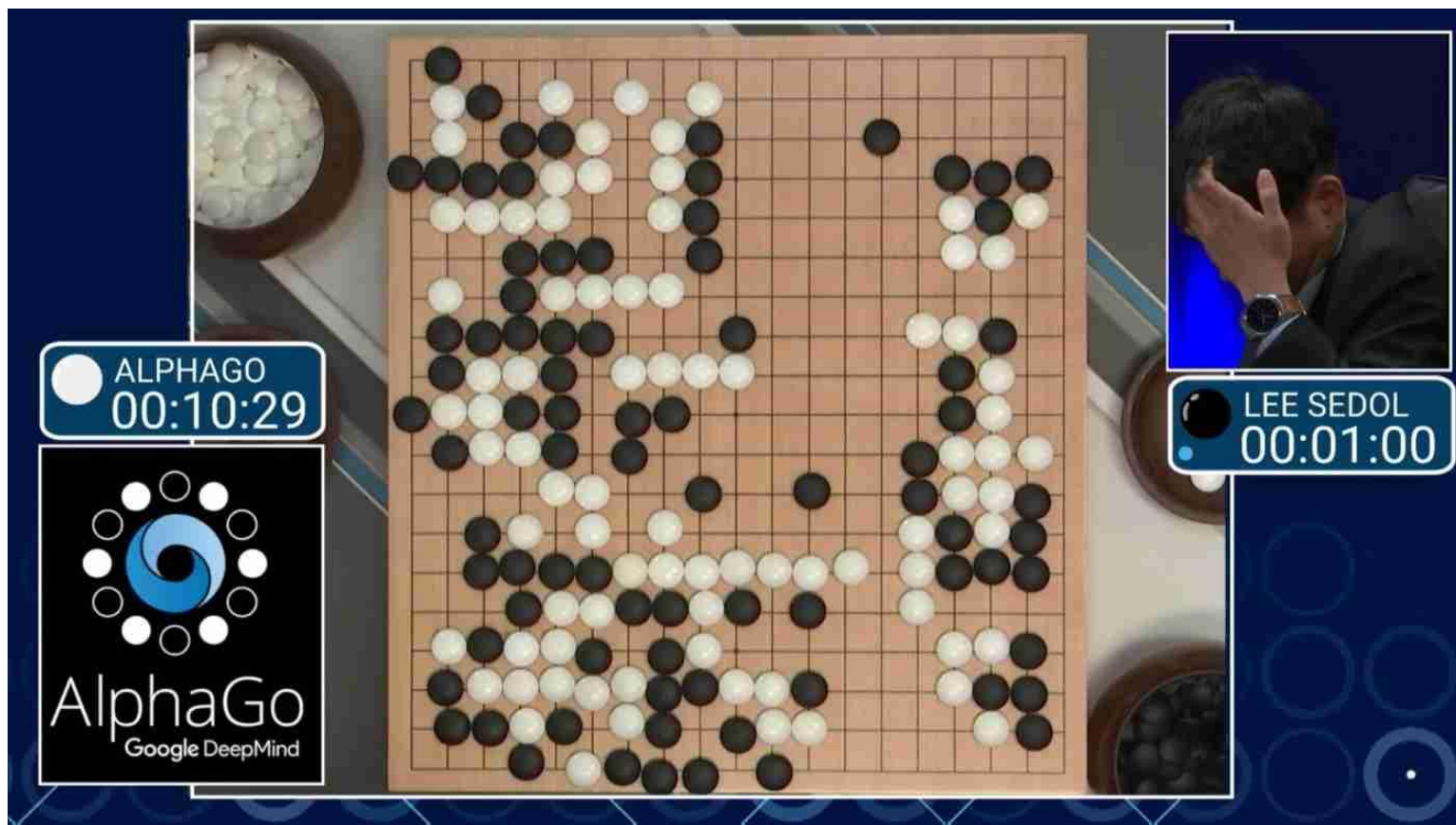
- **基礎演習: OpenAI Gym 入門**

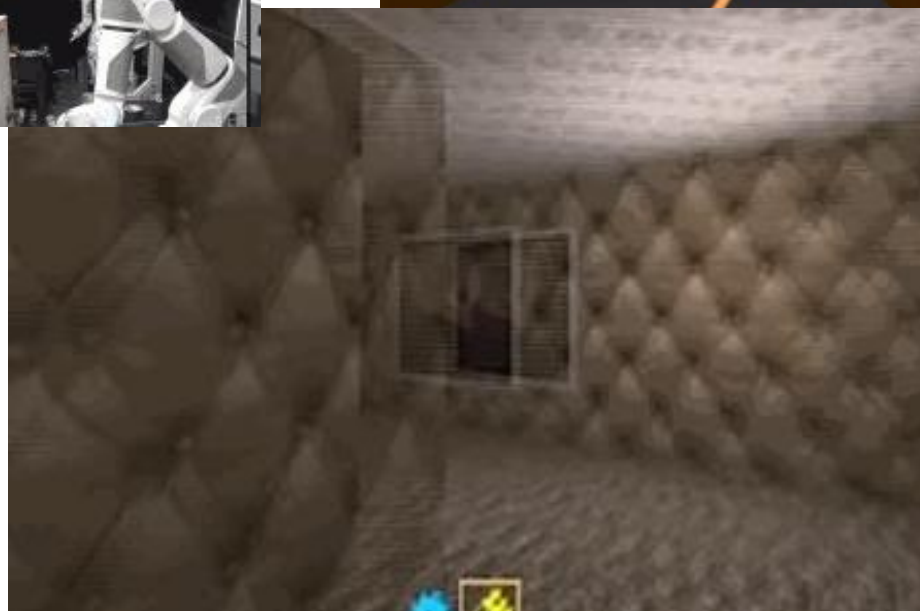
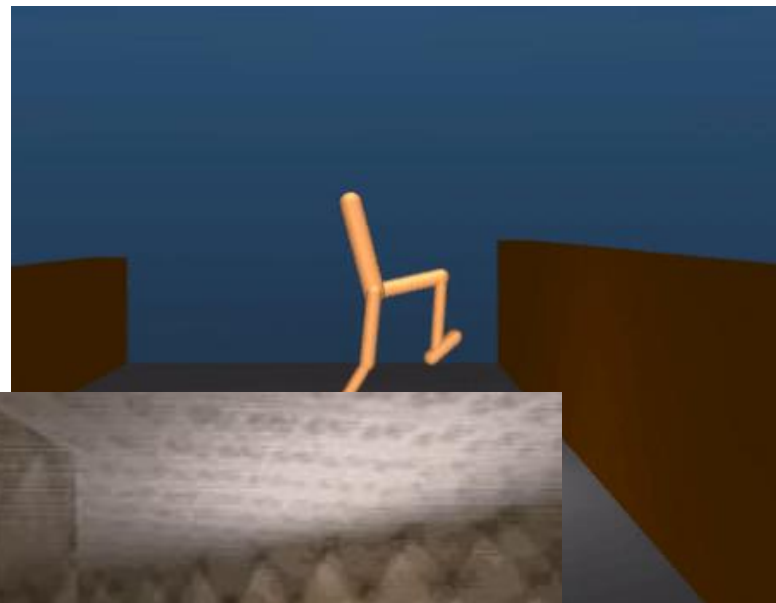
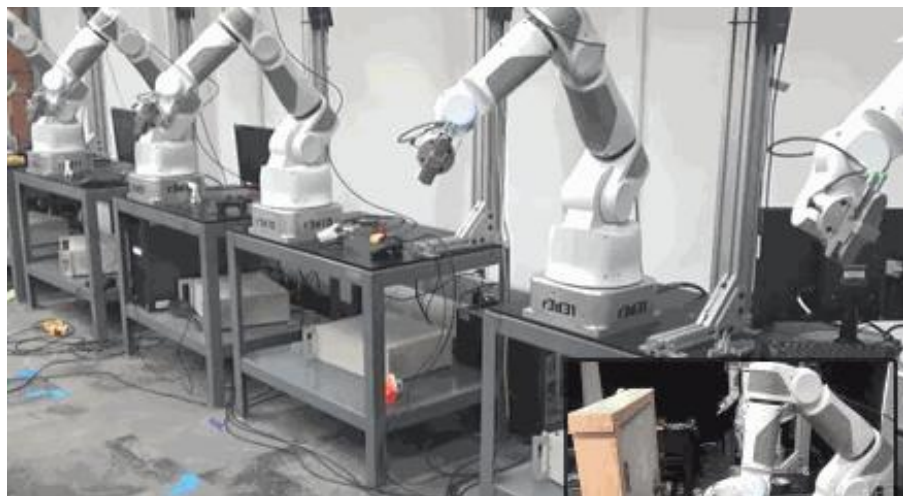
OpenAI Gym で遊ぶ

- **生物における強化学習**

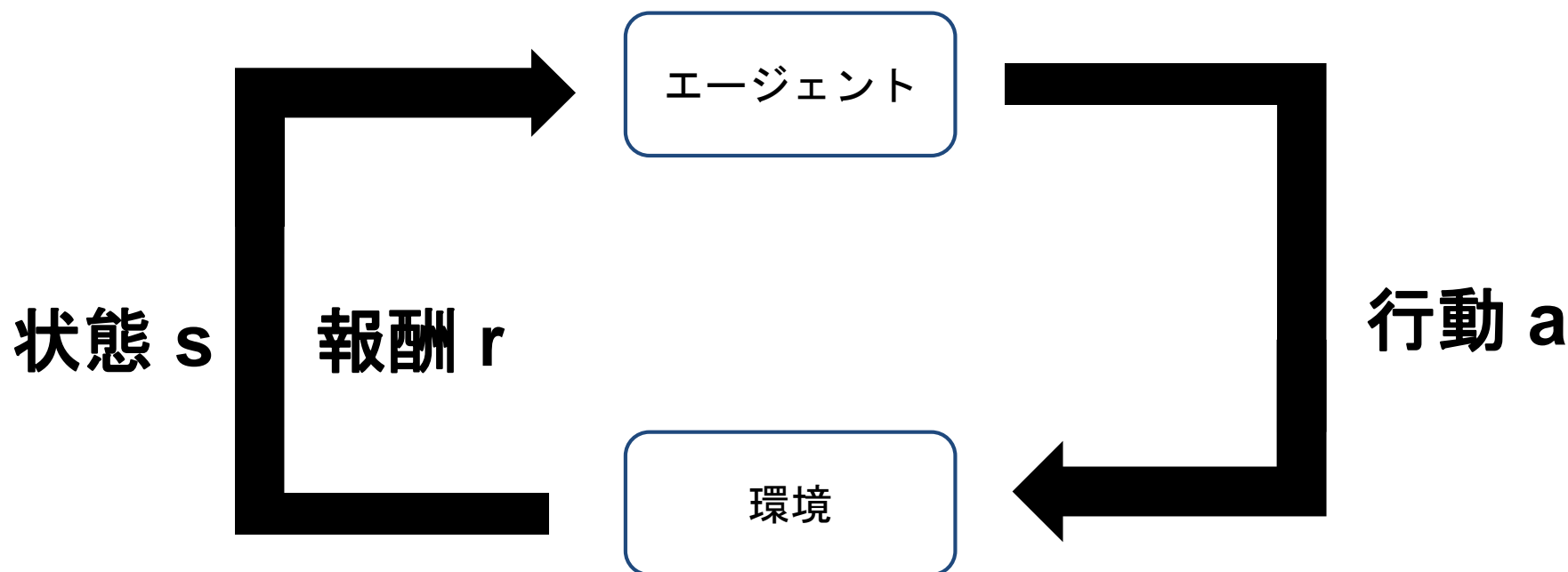
生物は実際に強化学習をやっているのか

## DeepMind の AlphaGo





環境からの**報酬の和を最大化**するような  
行動を学習する手法



- 教師あり学習 (supervised learning)

教師データを基にデータの規則を見つける

- 教師なし学習 (unsupervised learning)

与えられたデータから規則性を見つける

- **強化学習 (reinforcement learning)**

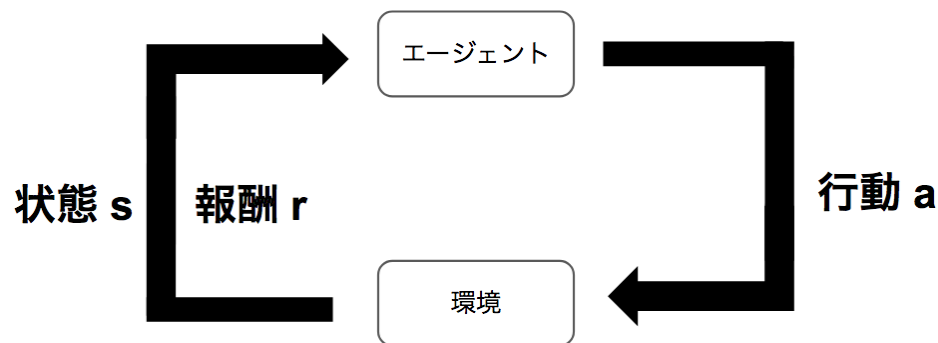
環境からの報酬を最大化する規則を見つける



# 無人島に漂着してしまいました



[http://highfive-aloha.com/wp-content/uploads/2016/06/6294b8755e69-98d1-45b0-a167-0c74fb44fb11\\_m.jpg](http://highfive-aloha.com/wp-content/uploads/2016/06/6294b8755e69-98d1-45b0-a167-0c74fb44fb11_m.jpg)



**エージェント:** 人間

**環境:** 無人島

**状態(s):** 人間の観測する情報

**報酬(r):** + 食べ物や飲み水を発見する

— 疲労や食べられないものを食べる

**行動(a):** 人間の行動

目標：できるだけ長く生存する



**報酬の総和を最大化すること**

試行錯誤で**探索**と**知識の利用**を行って  
目標を達成する方法を求める

## 探索 (exploration)

- 食べ物を探しに歩き回る
- 見たことないものを食べてみる
- 魚を取りに潜ってみる

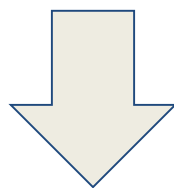


## 知識の利用 (exploitation)

- 知っている食べ物を食べる
- 一度発見した川に行く
- 簡単に取れる魚を取りに行く



探索して情報を増やしつつ  
知識を利用して生き延びる



**探索と知識の利用のバランスをとる**

無人島で浜辺にいるときにどの方角に行くかの選択を学習

## ● 教師あり学習

エージェント：西に向かうのを選択



直ちに正解を教える

教師データ：ブッブー、東に行くべきです

## ● 強化学習

エージェント：西に向かうのを選択

しばらくしてから



フィードバックを返す

環境：食べ物（報酬）をエージェントが発見

## ● 教師あり学習

学習データは最初から与えられている

学習が進んでもデータに変化はない

## ● 強化学習

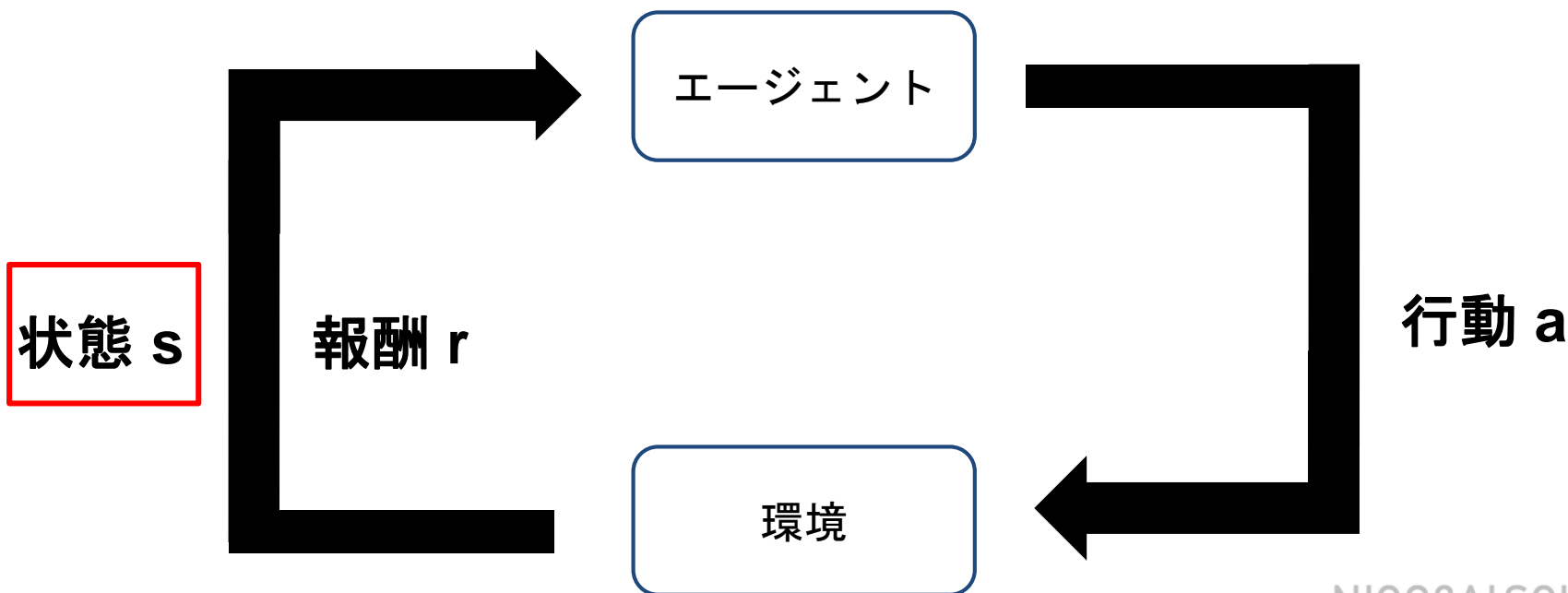
**探索**によりエージェントが学習データを集める必要がある

学習が進むと学習に使う**データが変化する**

## 状態 (state)

エージェントが観測する環境の様子

$s_t$ :  $t$ での環境の様子

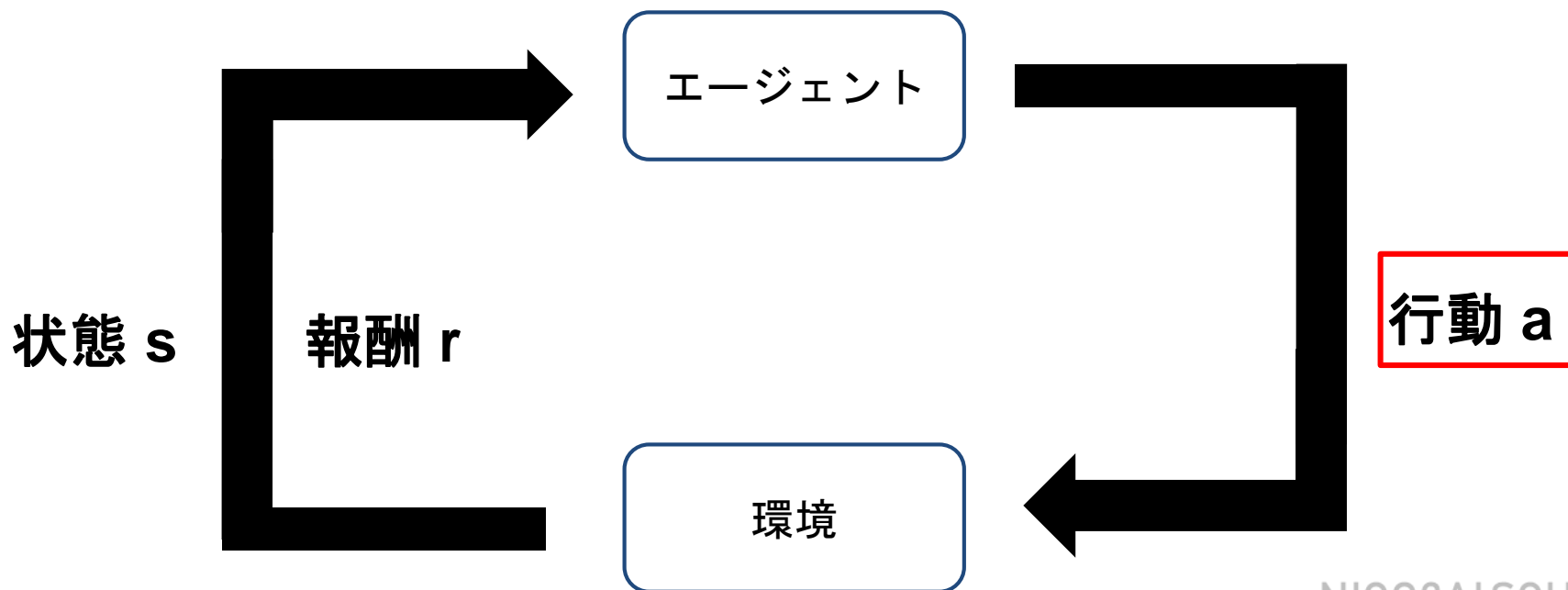




## 行動 (action)

エージェントが環境に働きかける行動

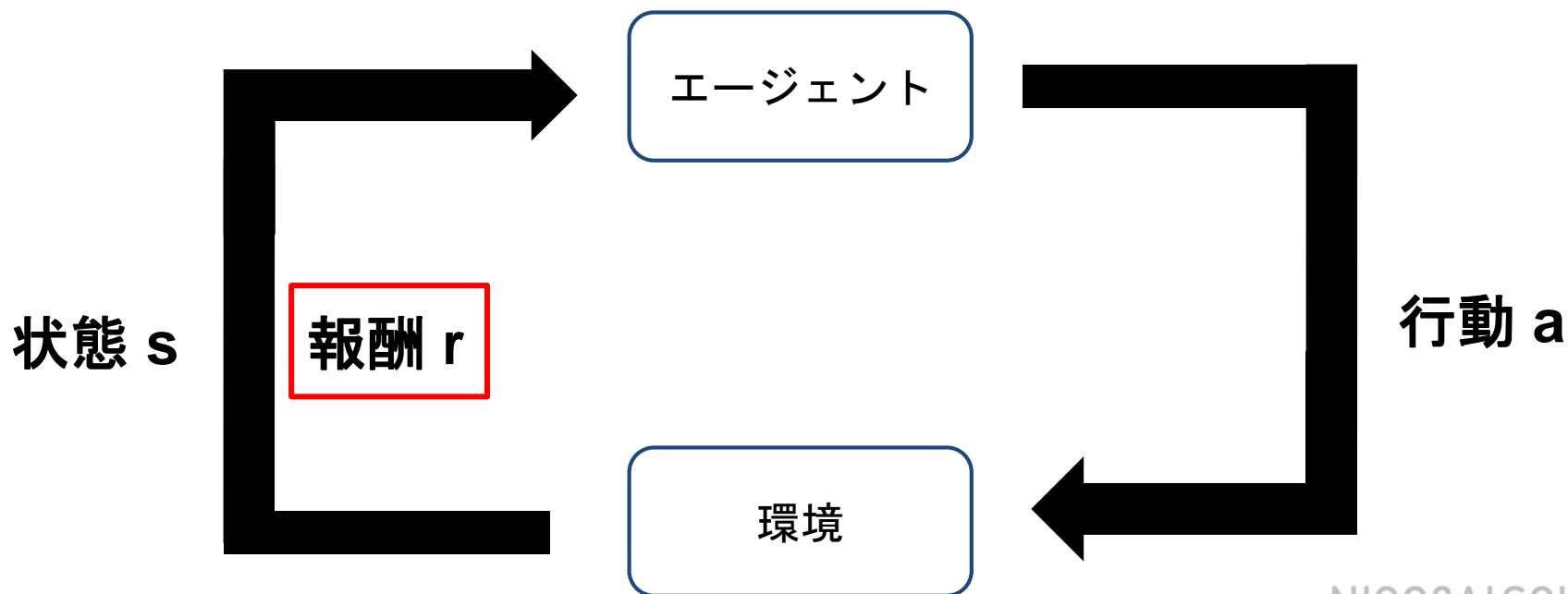
$a_t$ :  $t$ での行動



## 報酬 (reward)

環境からエージェントへ与えられる評価

$r_t$ :  $t$ での環境からの評価



次の状態が**今の状態と行動にのみ依存**することを  
マルコフ性をもつという

$$P(s_{t+1}, r_{t+1} \mid \underbrace{s_t, a_t, r_t, \dots, s_0, a_0, r_0}_{\text{今までの遷移全て}}) = P(s_{t+1}, r_{t+1} \mid \underbrace{s_t, a_t, r_t}_{\text{今の状態}})$$

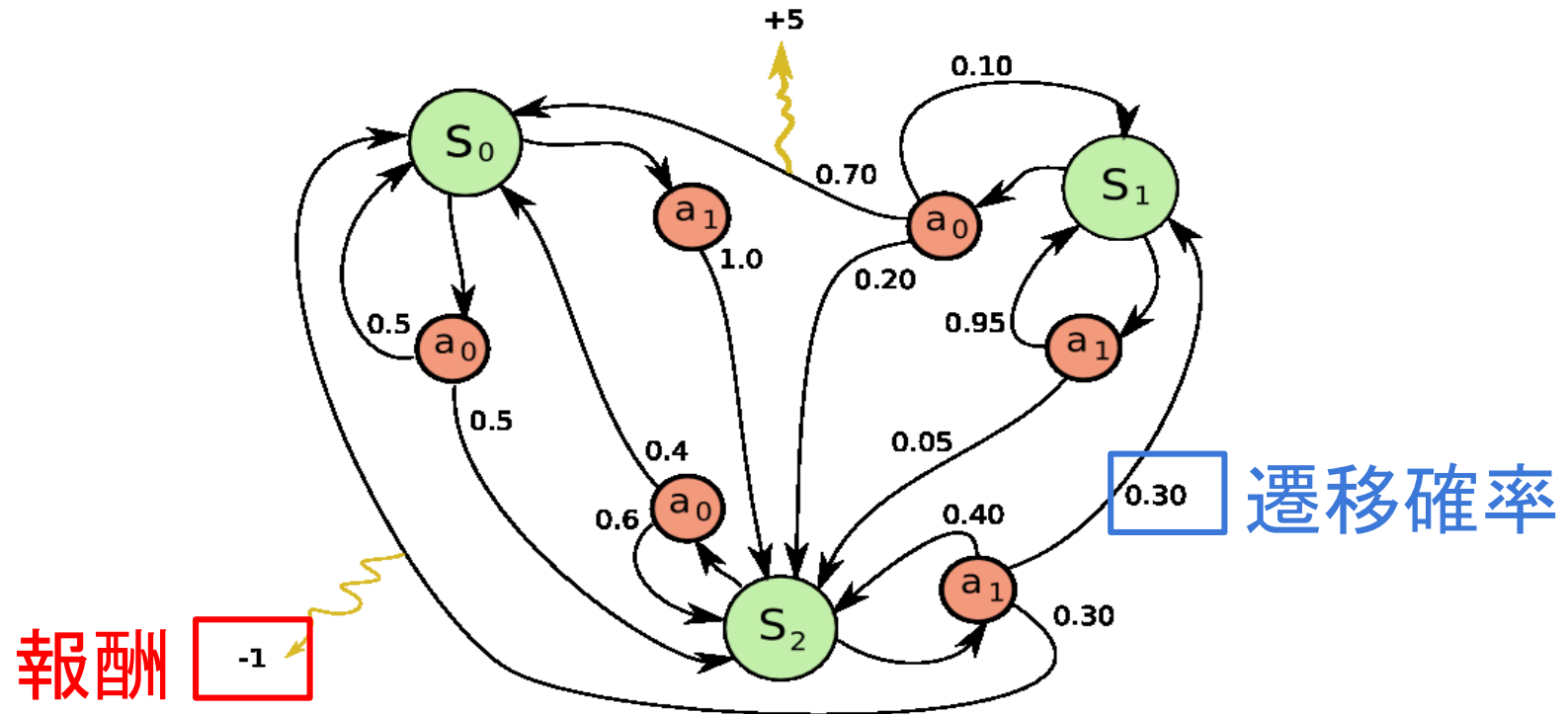
例：オセロの次の盤面は現在の盤面と次に打つ手によって決まる

# Markov Decision Process (MDP)

20

MDPとはマルコフ性をもつ環境を対象にした強化学習

現在のほとんどのタスクは環境をMDPとして扱っている



[https://upload.wikimedia.org/wikipedia/commons/2/21/Markov\\_Decision\\_Process\\_example.png](https://upload.wikimedia.org/wikipedia/commons/2/21/Markov_Decision_Process_example.png)

## ● 強化学習のコンセプト

エージェントが環境との**インタラクション**を通じて、**探索と利用**によって環境から得られる**報酬を最大化**する方法を学習する

## ● 用語

- **状態**: エージェントが観測する情報
- **行動**: エージェントが行う行動
- **報酬**: エージェントが環境から受け取るフィードバック

## ● マルコフ性

環境はマルコフ性をもち、今の状態から決定論的に次の状態へ遷移する

ある環境を考えたときに

- **状態**: エージェントが観測する情報
- **行動**: エージェントが行う行動
- **報酬**: エージェントが環境から受け取るフィードバック

を当てはめて、報酬の和を最大化する方策も考えてみよう

例: オセロ

**状態**: 盤面

**行動**: 次に置くマス目

**報酬**: 勝利したら1、敗北したら-1

# 価値に基づく学習と探索

- **強化学習の問題設定**

強化学習とは何だろうか

- **価値に基づく学習と探索**

実際にどのようにして学習を行うのか

- **基礎演習: OpenAI Gym 入門**

OpenAI Gym で遊ぶ

- **生物における強化学習**

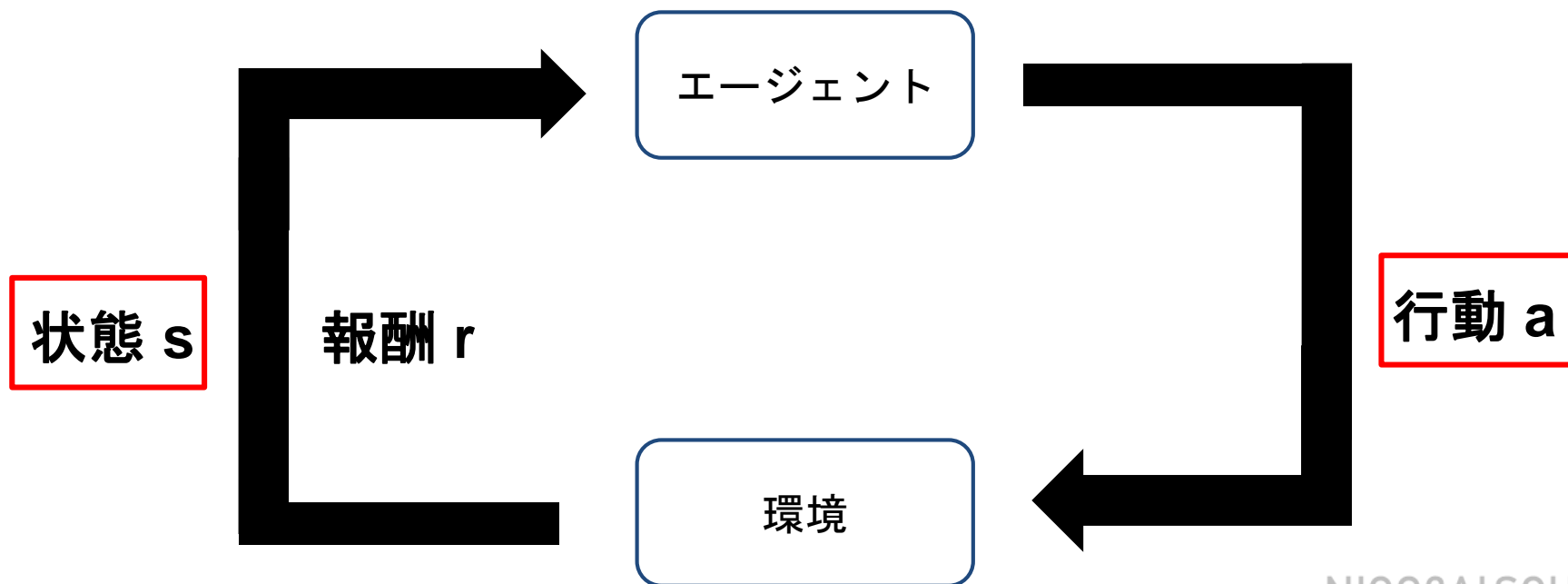
生物は実際に強化学習をやっているのか



## 方策（policy）

エージェントの行動方針。行動の選択確率として表す。

$\pi(s_t, a)$ :  $s_t$  で行動  $a$  を行う確率



## 収益 (return)

t以降の報酬の和, **価値 (value)** ともいう

**割引率 (discounted factor)**  $\gamma$  で後続の報酬の大きさを調整

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots$$

感覚的には、テスト前に

**割引率大** : 楽しいから、遊んで今すぐ報酬をもらう

**割引率小** : 楽しくないけど勉強頑張って多くの報酬をもらう

## 収益 (return)

$$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots$$

- $\gamma=1$  (今の報酬と未来の報酬が同じ大事さ)

$$R_t = r_t + r_{t+1} + r_{t+2} + r_{t+3} + \dots$$

- $\gamma=0.9$  (遠い未来は考えない)

$$R_t = r_t + 0.9r_{t+1} + 0.81r_{t+2} + 0.729r_{t+3} + \dots$$

- $\gamma=0$  (今を生きる)

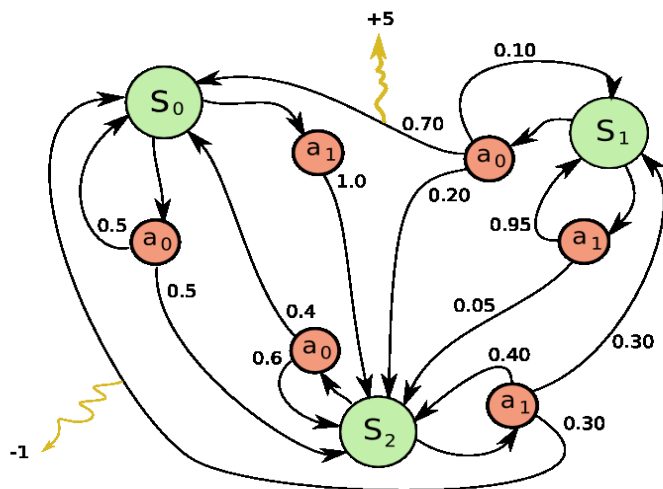
$$R_t = r_t$$

- モデルフリー (model-free) ← 講義ではここを扱う

環境のダイナミクスのモデルなしで学習を行う方法

- モデルベース (model-base)

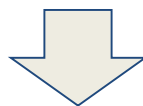
環境のダイナミクスのモデルを基に学習を行う方法



← これが分かっているかどうか

- 方策ベース (policy based)

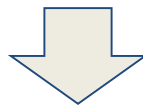
現在の方策での収益を計算して、収益が高くなる方策を学習する



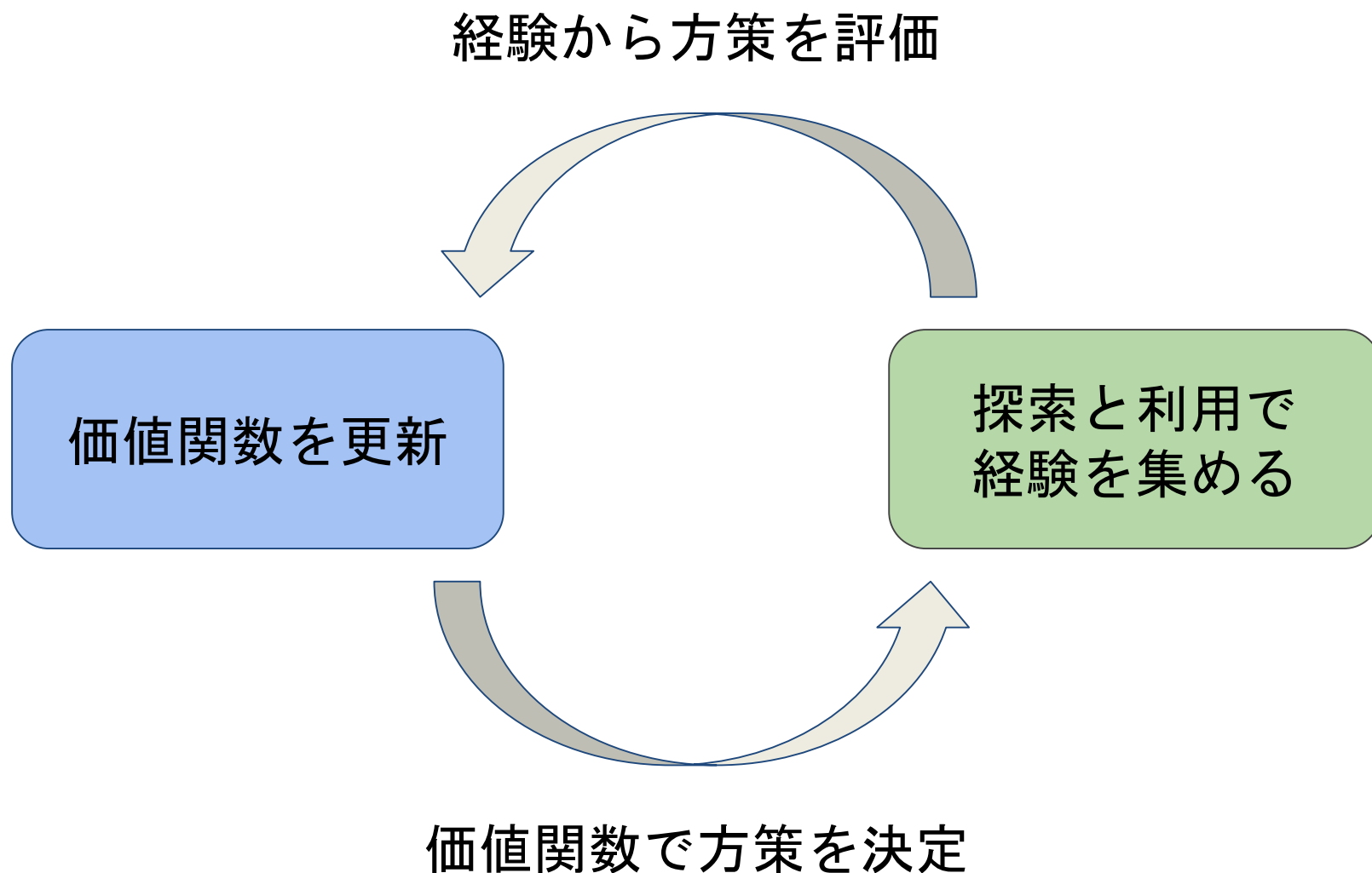
方策自体を求める

- 価値ベース (value based) ← 今日はこの話

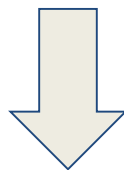
状態や行動に価値を設定して、その価値を元に方策を決定する



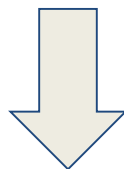
方策自体は求めない



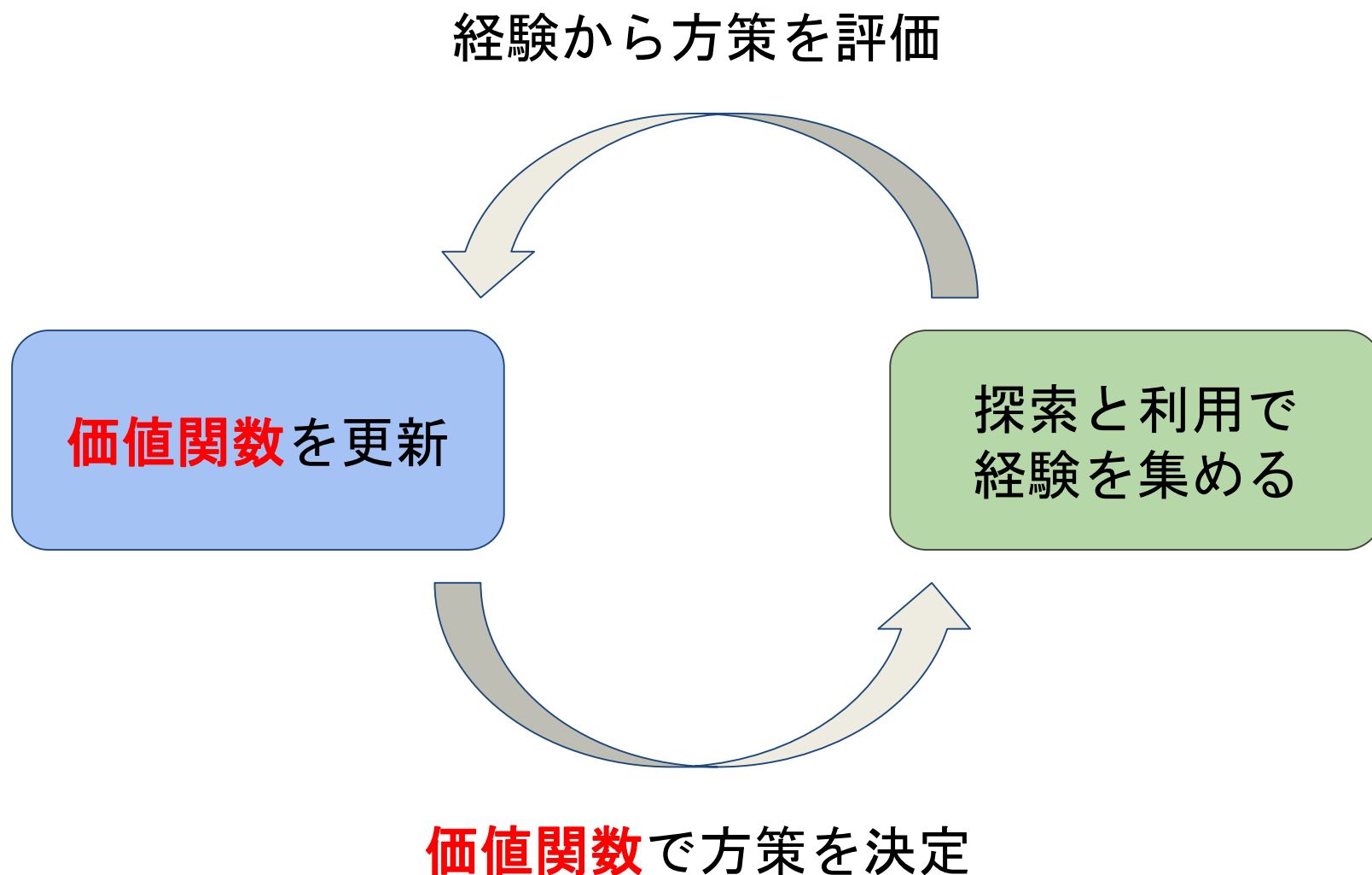
環境から受け取る報酬の和を最大化する



状態や行動に対する収益 (価値) がわかれば  
常に収益が大きくなる行動選択を行って報酬の和を最大化できる



最適な状態価値関数または行動価値関数を求める





## 状態価値関数 (state-value function)

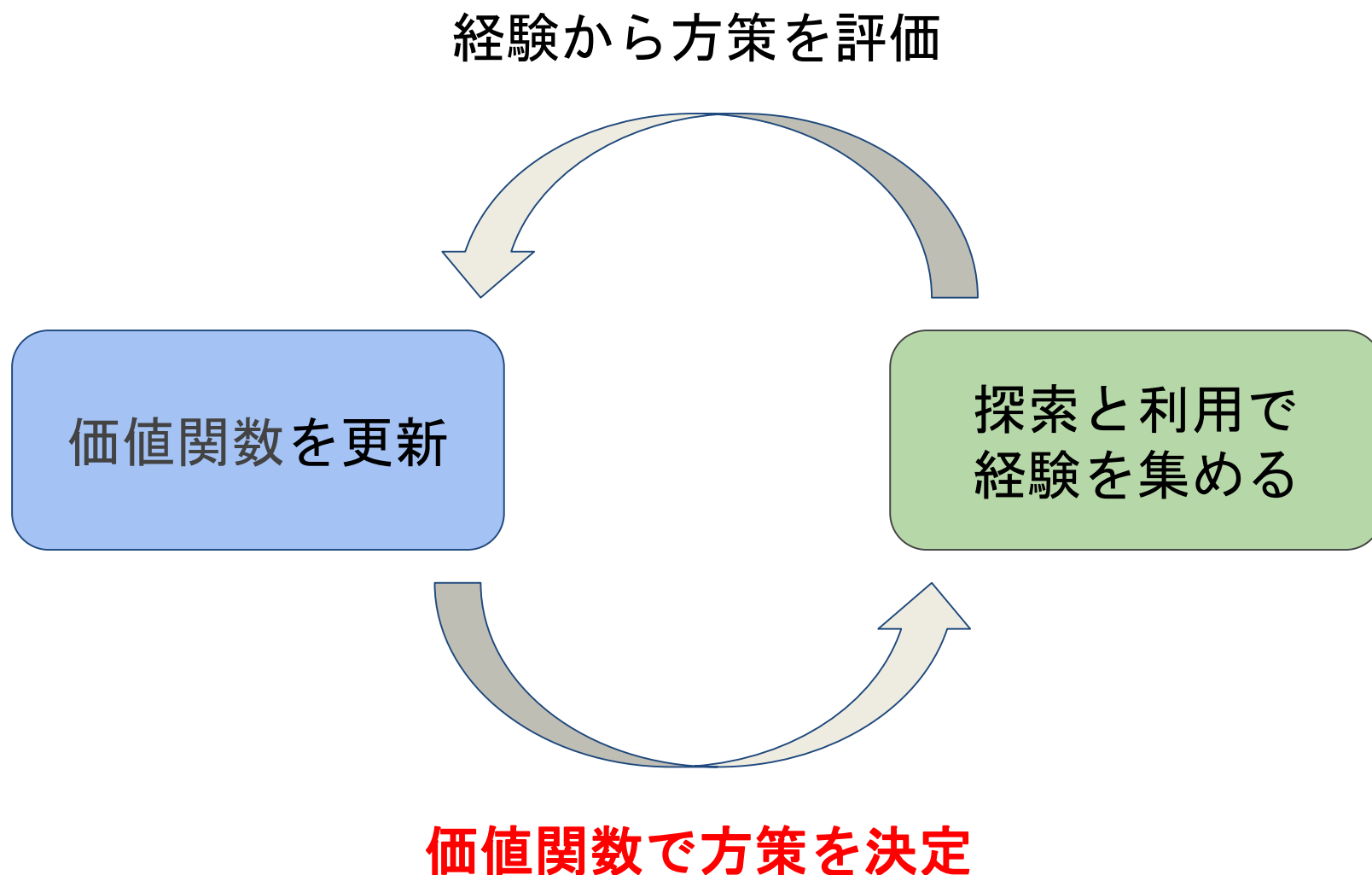
状態sから最後の状態までの収益の関数

$$V^{\pi}(s) = E_{\pi} \{R_t | s_t = s\} = E_{\pi} \left\{ \underbrace{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}}_{\text{収益}} | s_t = s \right\}$$

## 行動価値関数 (action-value function)

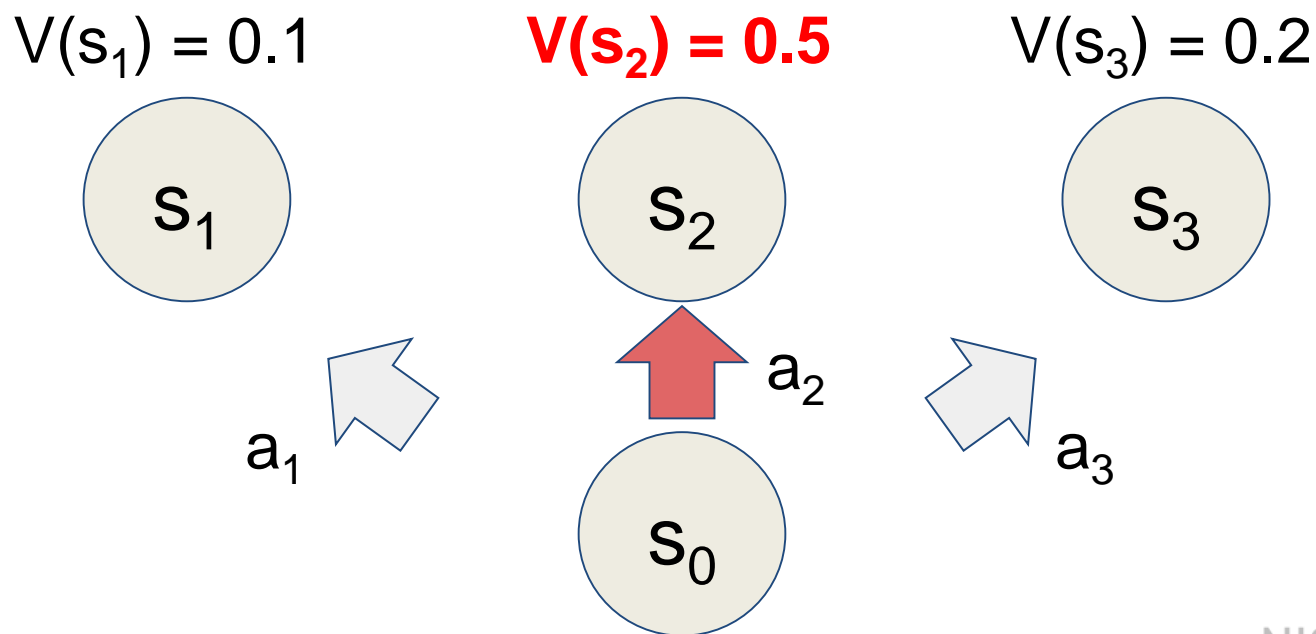
状態sで行動aを選択してから最後の状態までの収益の関数

$$Q^{\pi}(s, a) = E_{\pi} \{R_t | s_t = s, a_t = a\} = E_{\pi} \left\{ \underbrace{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}}_{\text{収益}} | s_t = s, a_t = a \right\}$$



常に一番大きな価値を持つ状態に移動

$$s_{t+1} = \operatorname{argmax}_s V(s)$$



常に一番大きな価値を持つ行動を選択する

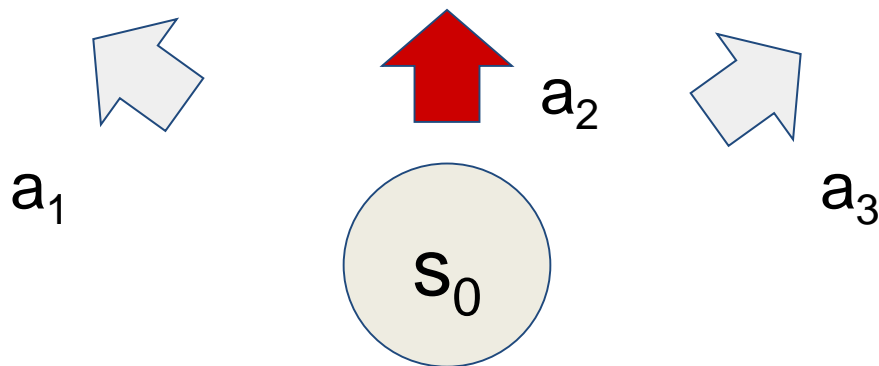
$$a = \operatorname{argmax}_a Q(s, a)$$

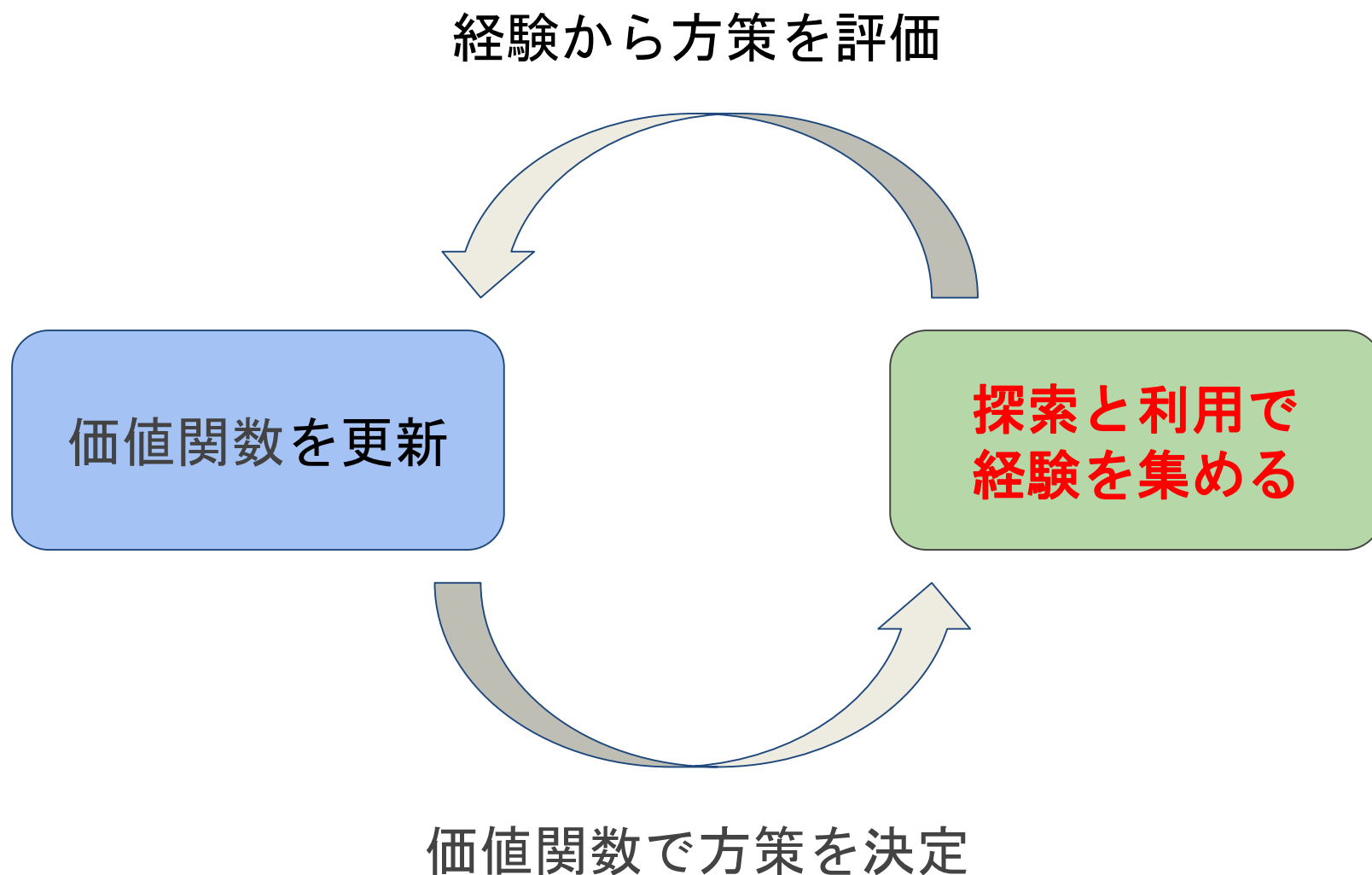
次の状態が何であるかは考えなくていい

$$Q(s_0, a_1) = 0.1$$

$$Q(s_0, a_2) = 0.5$$

$$Q(s_0, a_3) = 0.2$$





探索することでまだ見ぬ報酬を見つけることができる

- **epsilon-greedy**

確率 $\epsilon$ でランダムに行動選択を行う

- **Boltzman行動選択 (softmax行動選択)**

選択できる価値をsoftmaxした確率分布で行動を選択する

- **UCB1 (Upper Confidence Bounds)**

知らない状態（行動）を優先して選択する

確率 $\epsilon$ でランダムに行動して、それ以外は最良の行動を選択する

- いいところ

シンプルに実装できる

- わるいところ

完全にランダムなので明らかに悪い行動も選択してしまう

選択できる価値をsoftmaxした確率分布で行動を選択する

$$P_t(a) = \frac{\exp(q_t(a)/\tau)}{\sum_{i=1}^n \exp(q_t(i)/\tau)},$$

↑  
価値

↑  
差の大きさを調整

## ● いいところ

悪いと分かっている状態へ遷移する確率を下げられる

## ● わるいところ

常にランダムに行動を選択する



# UCB1 (Upper Confidence Bounds)<sup>41</sup>

知らない状態（行動）を優先して選択する

## 全体の回数

$$B_{UCB}(i) = \hat{\mu}_i + \sqrt{\frac{2\log(T)}{N_i(t)}}$$

## Bが最大の選択を行う

# 價值

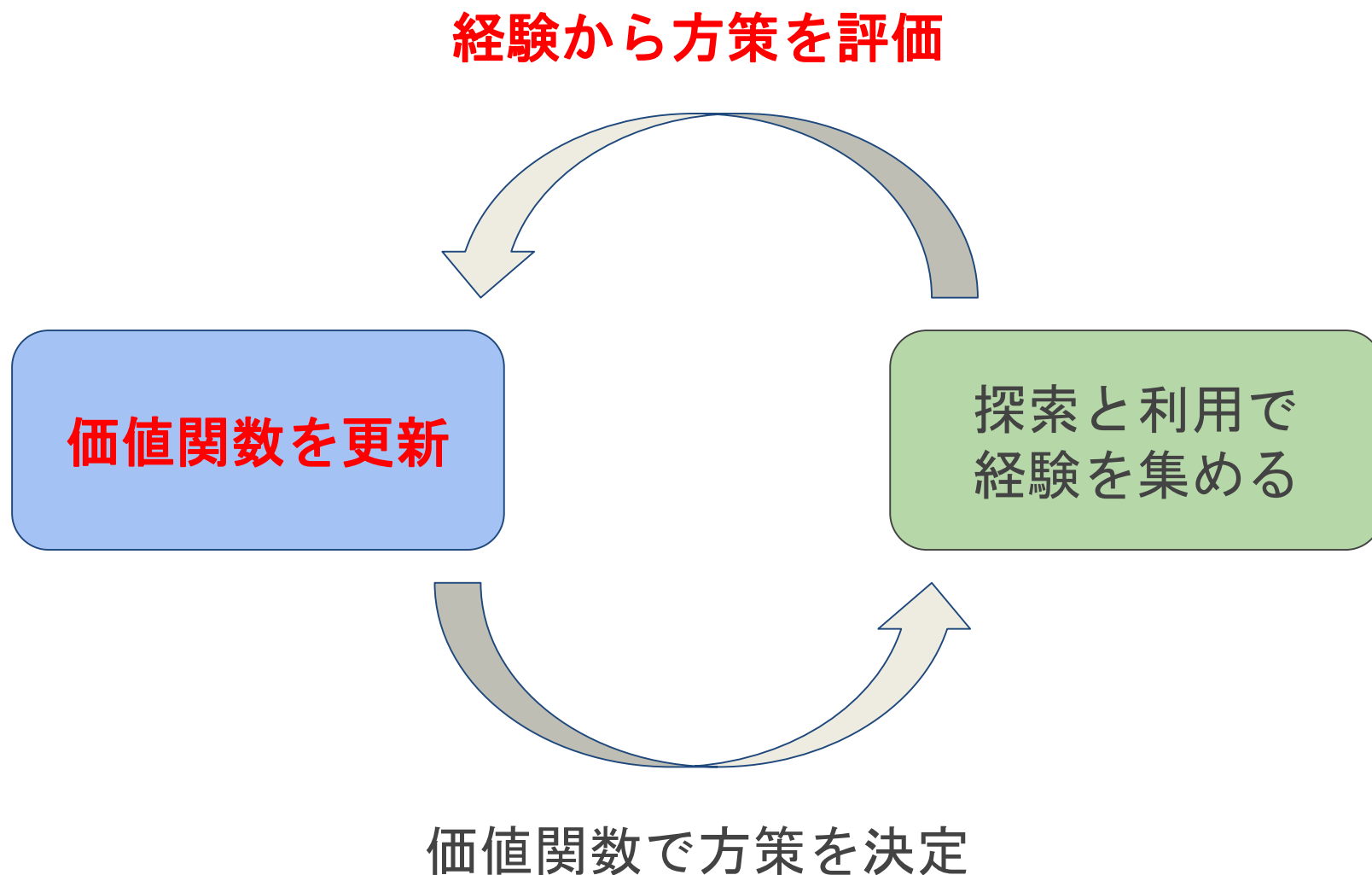
## 選択した回数

## ● いいところ

おとづれたことがない状態に優先して探索する

## ● わるいところ

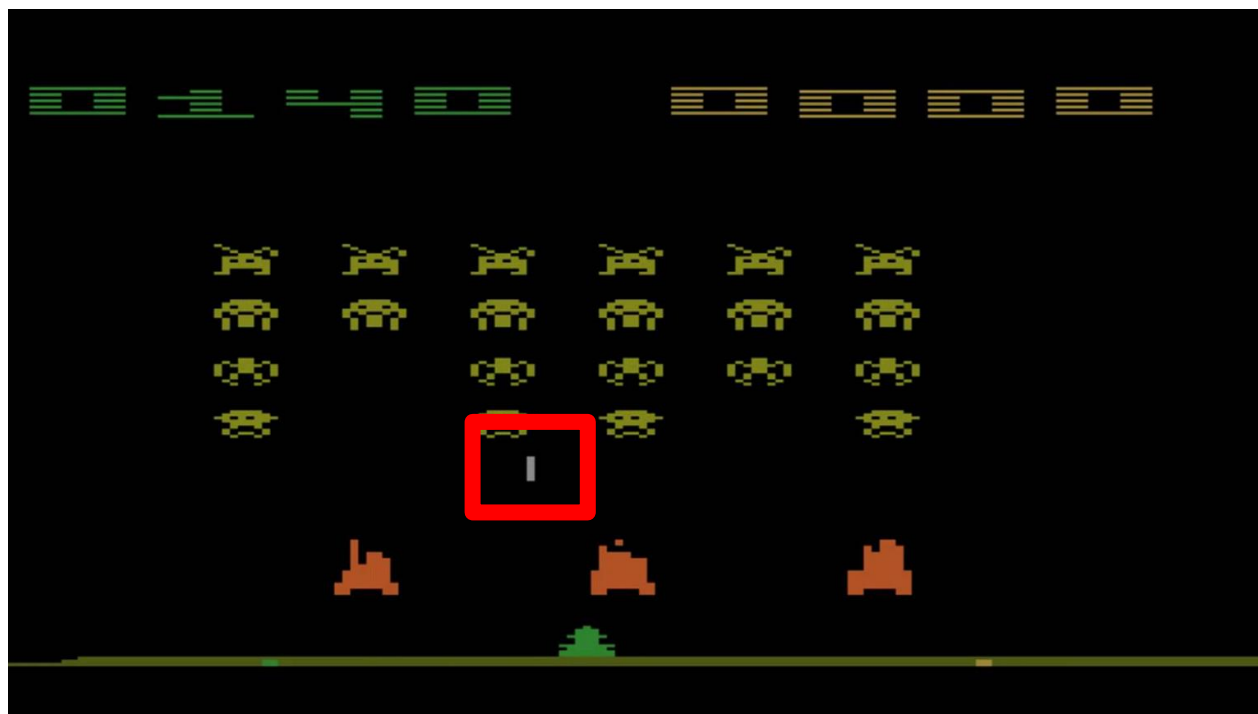
## 価値が低くても知らないところに積極的に向かう



インベーダーゲームだと撃ってからしばらくしてスコアが貰える



打った瞬間の状態にも価値を与える必要がある



ある状態の報酬と後続の状態群の報酬の関係を表した式

$$\begin{aligned} V^\pi(s) &= E_\pi \{R_t | s_t = s\} \\ &= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s \right\} \\ &= E_\pi \left\{ \boxed{r_{t+1}} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_t = s \right\} \end{aligned}$$

即時報酬

1step先での収益

次の状態の価値との関係を表せる

$$\begin{aligned} V^\pi(s) &= E_\pi \{ R_t | s_t = s \} \\ &= E_\pi \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s \right\} \\ &= E_\pi \left\{ r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_t = s \right\} \\ &= E_\pi \{ \boxed{r_{t+1}} + \gamma \boxed{V^\pi(s_{t+1})} | s_t = s \} \end{aligned}$$

即時報酬

1step先の状態価値

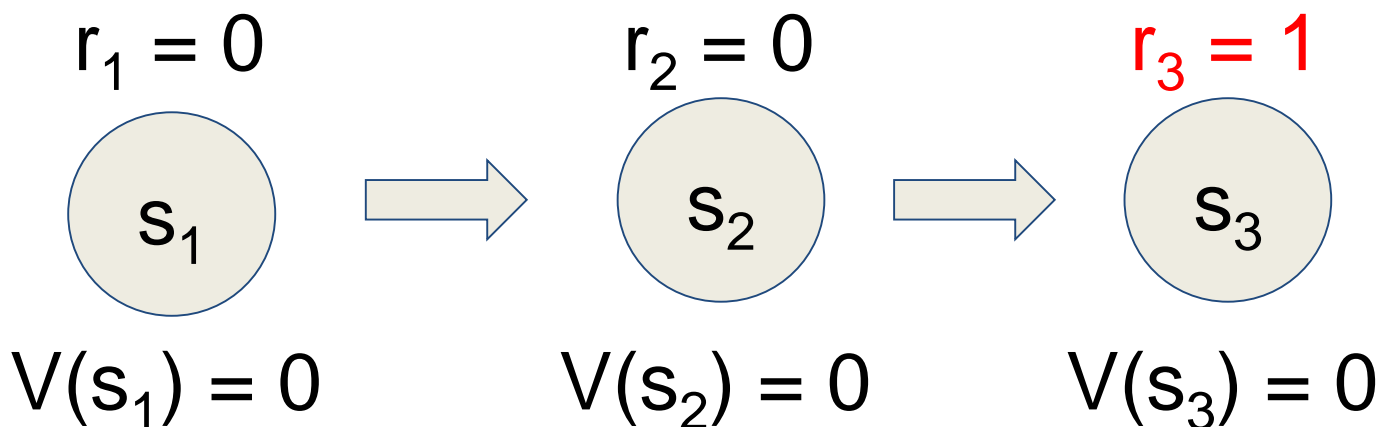
状態価値関数をオンラインで更新するモデルフリーの手法

$$V(s_t) \leftarrow V(s_t) + \underbrace{\alpha}_{\text{学習率}} \underbrace{[r_{t+1} + \underbrace{\gamma V(s_{t+1})}_{\text{近づける目標}} - \underbrace{V(s_t)}_{\text{今の値}}]}_{\text{TD誤差 (TD error)}}$$

各ステップで上を計算することで

**価値を前の状態に伝播させる**

$$V(s_t) \leftarrow V(s_t) + \alpha [r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$$

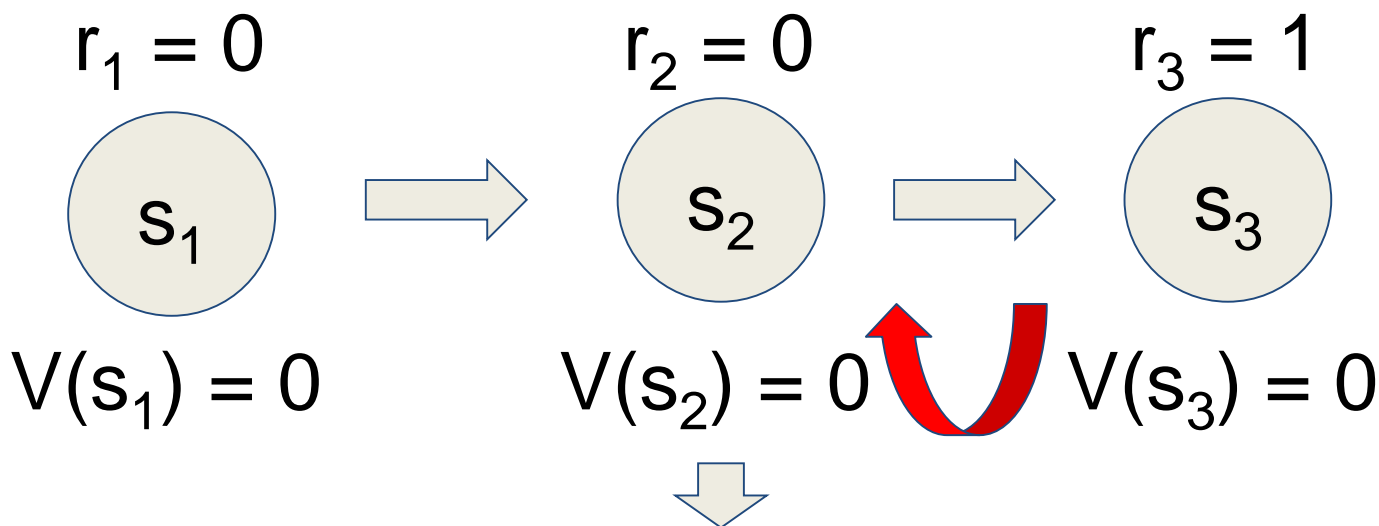


例として必ず  $s_1 \rightarrow s_2 \rightarrow s_3$  と遷移する環境があるとする

# TD学習の様子 ( 2 ): iteration1

48

$$\alpha = 1, \gamma = 0.9$$



$$V(s_2) = 0 + 1 \times [1 + 0.9 \times 0 - 0] = 1$$

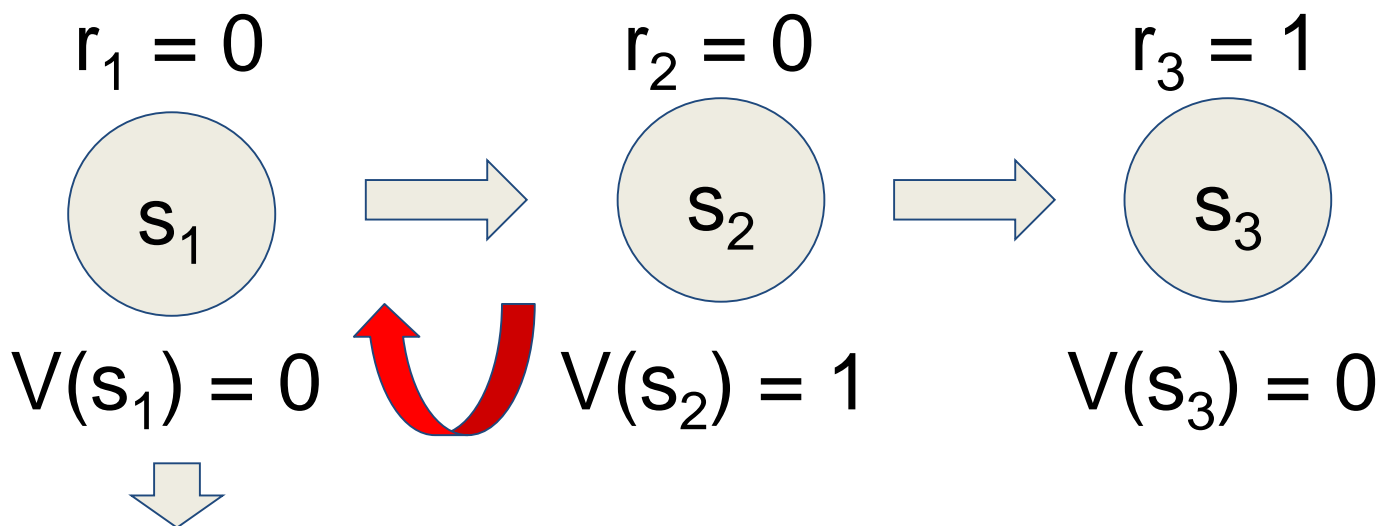
$$V(s_t) \leftarrow V(s_t) + \alpha [r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$$



# TD学習の様子 (3): iteration2

49

$$\alpha = 1, \gamma = 0.9$$



$$V(s_1) = 0 + 1 \times [0 + 0.9 \times 1 - 0] = 0.9$$

$$V(s_t) \leftarrow V(s_t) + \alpha [r_{t+1} + \gamma V(s_{t+1}) - V(s_t)]$$

割り引かれた価値が前の状態に伝播した！

TD学習と同じ要領で行動価値関数も学習ができる

- Sarsa (State-Action-Reward-State-Action)

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \underbrace{[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]}_{\text{TD誤差}}$$

- Q学習

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \underbrace{\left[ r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right]}_{\text{TD誤差}}$$

状態×行動の配列を用いて単純にQ関数を表すことができる

状態/行動	$a_1$	$a_2$	$a_3$
$s_1$	0.1	0.2	0.3
$s_2$	0.2	0.4	0.5
$s_3$	0.3	0.5	0.6

評価に使う方策と行動に使う方策が同じかどうかで異なる性質

- Sarsa (方策オン型)

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [r_{t+1} + \underbrace{\gamma Q(s_{t+1}, a_{t+1})}_{\text{現在の行動方策で選んだ行動の価値}} - Q(s_t, a_t)]$$

現在の行動方策で選んだ行動の価値

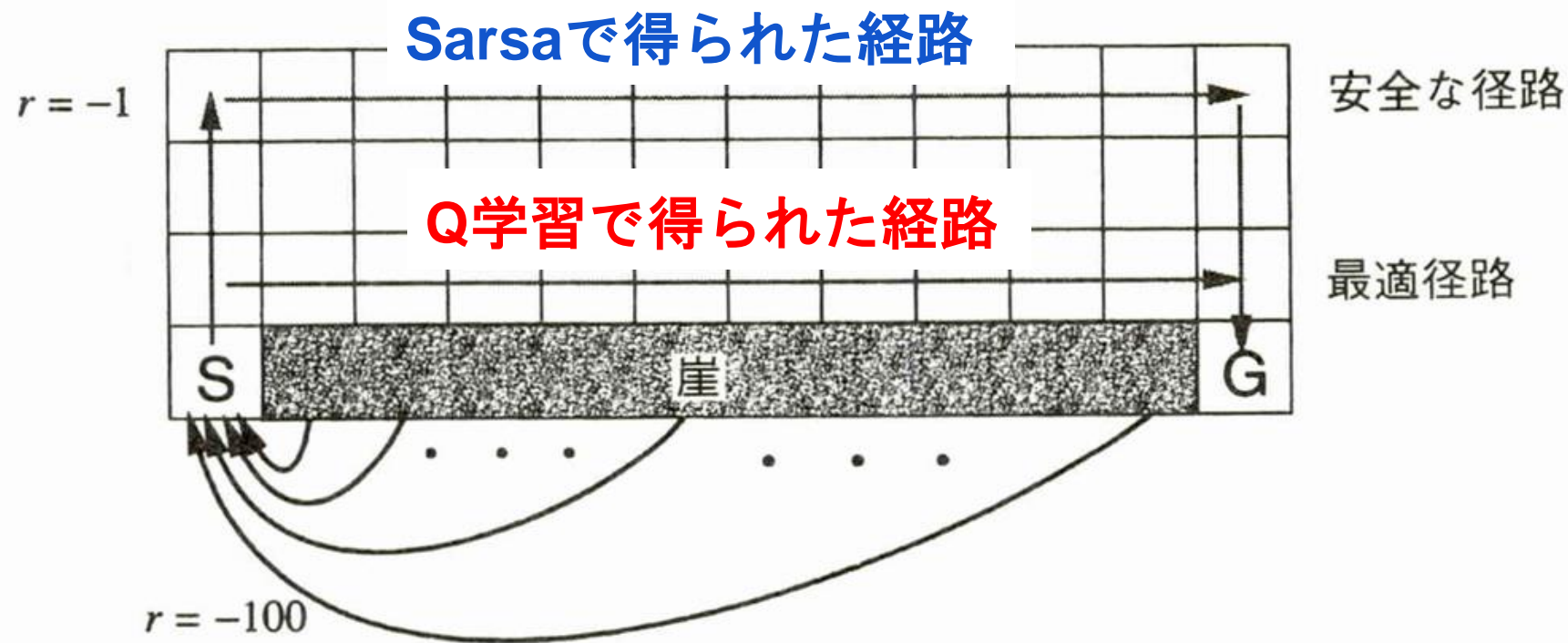
- Q学習 (方策オフ型)

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[ r_{t+1} + \underbrace{\gamma \max_a Q(s_{t+1}, a)}_{\text{行動方策とは関係ない次状態の最大価値}} - Q(s_t, a_t) \right]$$

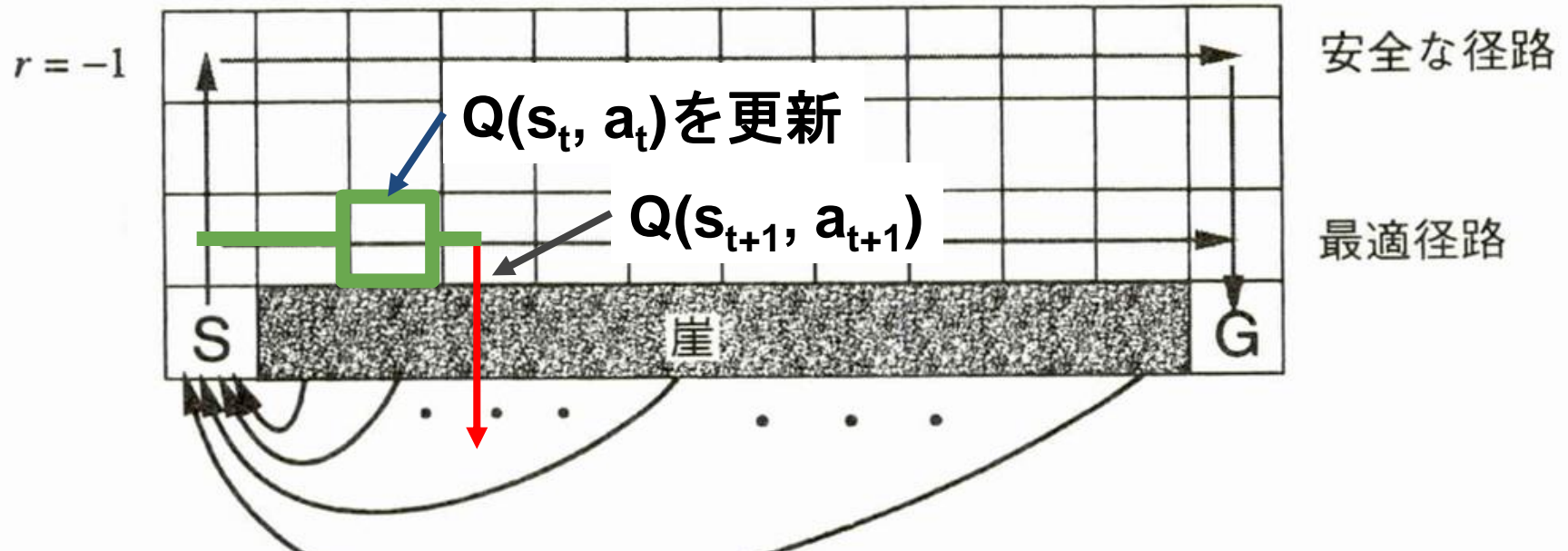
行動方策とは関係ない次状態の最大価値

崖のそばを歩くタスク

1step毎に-1の報酬で、崖に落ちると-100の報酬が与えられる



# 崖から落ちた時の学習: Sarsa (方策オン型) 54

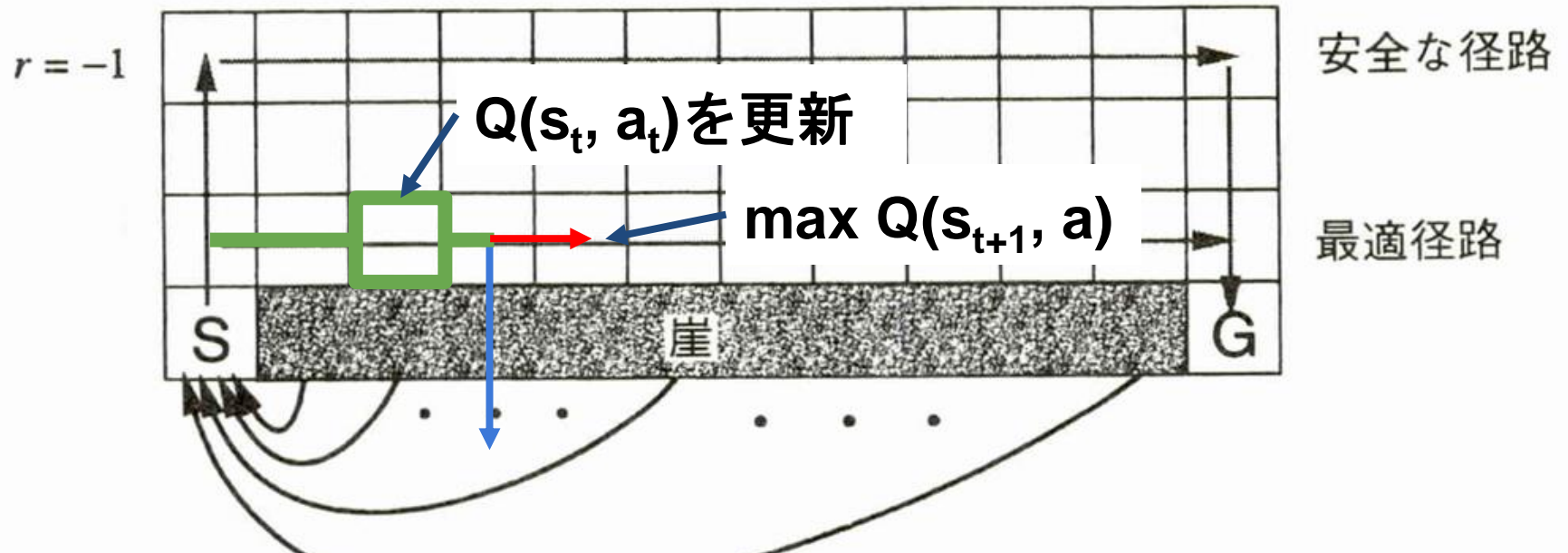


$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[-1 + \underbrace{\gamma Q(s_{t+1}, a_{t+1})}_{\text{崖に落ちているので-100}} - Q(s_t, a_t)]$$

崖に落ちているので-100

$Q(s_t, a_t)$  が  $Q(s_{t+1}, a_{t+1})$  に引きずられて低くなる

# 崖から落ちた時の学習: Q学習 (方策オフ型) 55



$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[-1 + \underbrace{\gamma \max Q(s_{t+1}, a)}_{\text{最適な経路の価値}} - Q(s_t, a_t)]$$

最適な経路の価値

$Q(s_t, a_t)$  は最適な経路の価値になる

## ● 用語

- **収益 (価値)**: ある状態からの割引率を考慮した報酬の和
- **方策**: ある状態において、行動aを選択する確率
- **状態価値関数**: 状態sにおける価値を表す関数
- **行動価値関数**: 状態sで行動aを行う価値を表す関数

## ● TD学習、Q学習、Sarsa

**Bellman方程式**に基づいて、次のステップでの価値と報酬にしたがって現在の価値を更新する

## ● 探索

- **epsilon-greedy**: 確率epsilonでランダムに探索を行う
- **Boltzman行動選択**: softmaxした確率分布で行動選択を行う
- **UCB1**: 知らない状態へ積極的に訪れる



# 基礎演習：OPENAI GYM 入門

- **強化学習の問題設定**

強化学習とは何だろうか

- **価値に基づく学習と探索**

実際にどのようにして学習を行うのか

- **基礎演習: OpenAI Gym 入門**

OpenAI Gym で遊ぶ

- **生物における強化学習**

生物は実際に強化学習をやっているのか

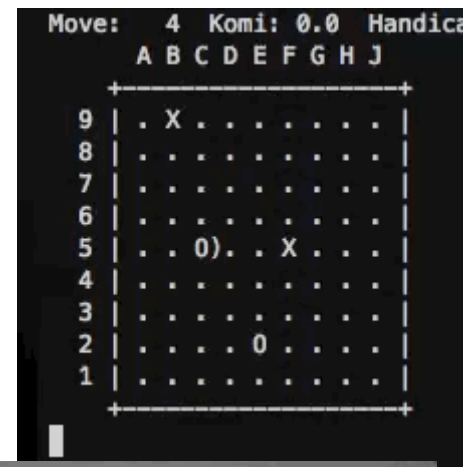
イーロンマスクが率いるOpenAIという団体が提供している  
**AI用の学習環境**

様々な環境を提供している

- 倒立振子
- Atari
- 囲碁
- マインクラフト
- Doom (FPS)

などなど

**さらに有志の作った環境も使える**

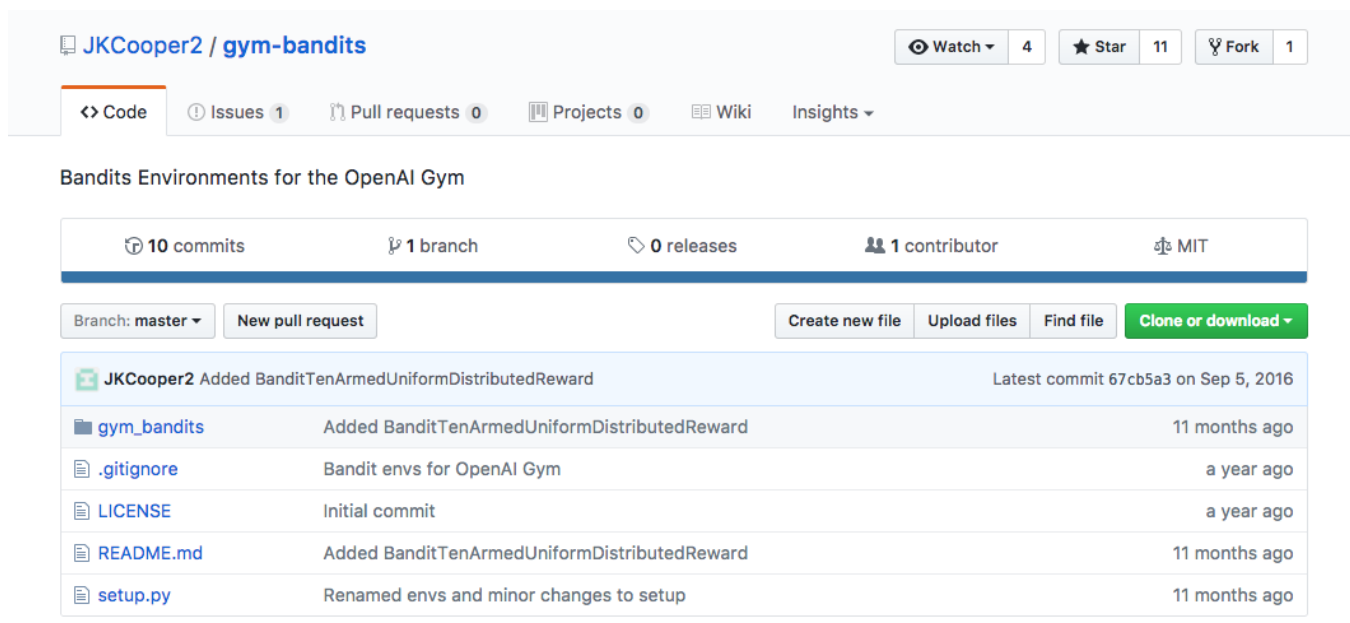


## 多腕バンディット (multi-armed bandit) で遊ぶ

以下の有志の作った環境を使って学習してみる

BanditTenArmedRandomFixed-v0

<https://github.com/JKCooper2/gym-bandits>



JKCooper2 / gym-bandits

Watch 4 Star 11 Fork 1

Code Issues 1 Pull requests 0 Projects 0 Wiki Insights

Bandits Environments for the OpenAI Gym

10 commits 1 branch 0 releases 1 contributor MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

Commit	Message	Time
JKCooper2	Added BanditTenArmedUniformDistributedReward	Latest commit 67cb5a3 on Sep 5, 2016
gym_bandits	Added BanditTenArmedUniformDistributedReward	11 months ago
.gitignore	Bandit envs for OpenAI Gym	a year ago
LICENSE	Initial commit	a year ago
README.md	Added BanditTenArmedUniformDistributedReward	11 months ago
setup.py	Renamed envs and minor changes to setup	11 months ago

当たる確率の異なるスロットマシンからの  
一番当たる確率の高いスロットを見つけるタスク

状態遷移のない単純な強化学習として考えることができる

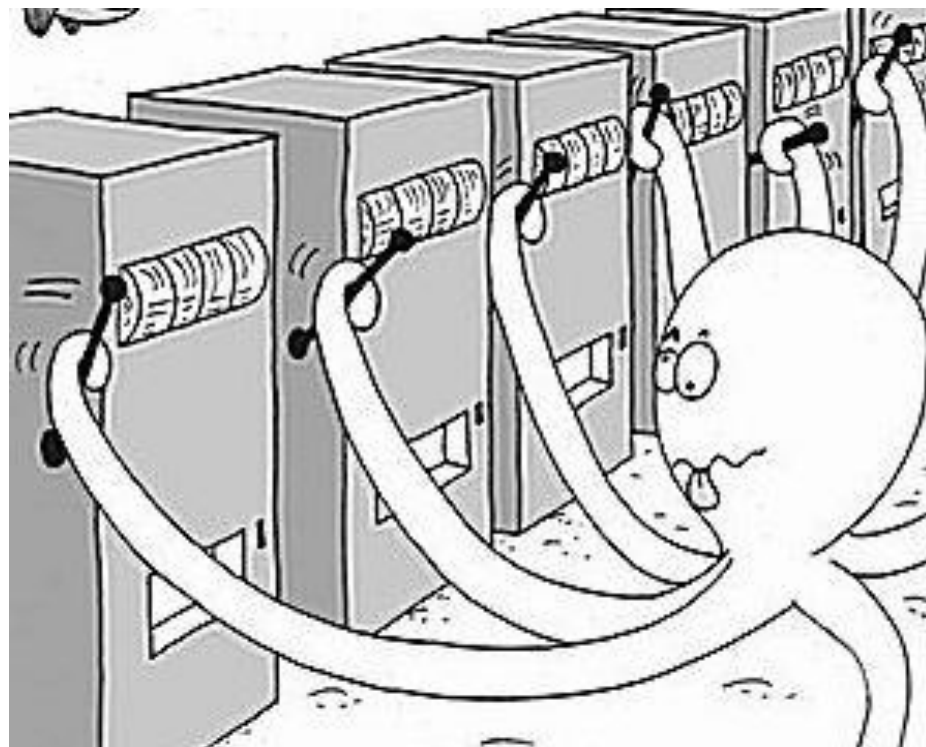
今回の強化学習設定

状態: なし

行動: どの腕を引くか

報酬: 当たり

価値: 当たる確率



# 生物における強化学習

- **強化学習の問題設定**

強化学習とは何だろうか

- **価値に基づく学習と探索**

実際にどのようにして学習を行うのか

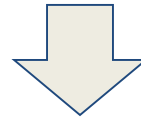
- **基礎演習: OpenAI Gym 入門**

OpenAI Gym で遊ぶ

- **生物における強化学習**

生物は実際に強化学習をやっているのか

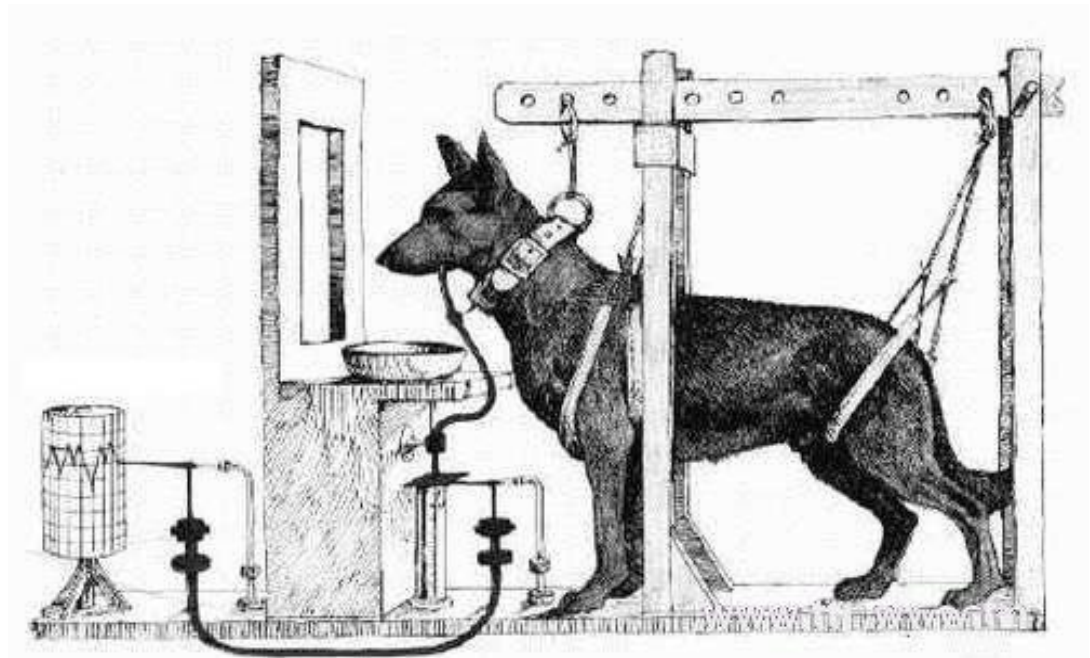
犬に餌を与える直前に**必ずベルを鳴らす**というのを繰り返す



**ベルを鳴らした**だけで唾液が分泌されるようになる

最初は無反応だった刺激が別の刺激と心理的に関連づけられる

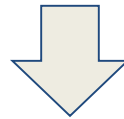
→関連づけが**強化**された





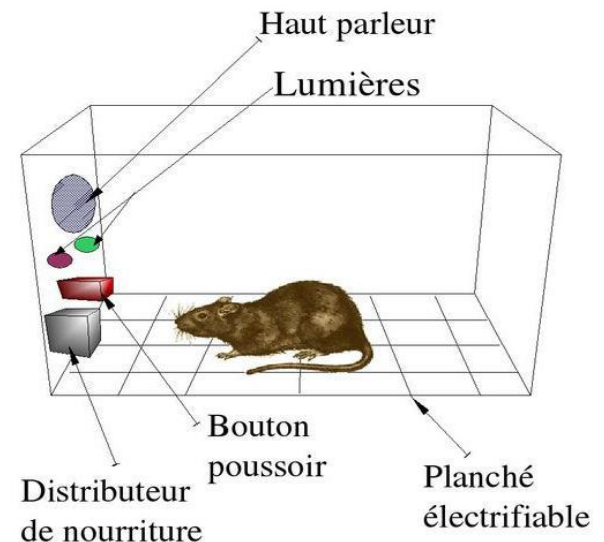
ブザー音の後にレバーを押すと餌が出てくる箱

ラットは最初**ランダム**に行動しているが...

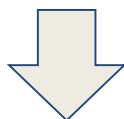


偶然レバーを押した時に餌が手に入ると  
**徐々にレバーを押すようになる**

この時、生物に好ましい効果（報酬）を  
**強化信号**と呼ぶ



スキナーの箱で餌の代わりに脳に**電気刺激**を与えてみた



餌の時と同じようにレバーを押し続けるようになり  
**腹側被蓋野**と**線条体**を結ぶ**内側前脳束の刺激**が最も効果があった

腹側被蓋野は**ドーパミン**を  
神経伝達物質として放出する神経細胞を豊富に含んでいる  
(ドーパミン作動性ニューロン)

→ **ドーパミンが報酬と関係している？**

Schultzらは、サルに対して光刺激を与えてからレバーを押すとジュースがもらえる実験を行なった

この時の腹側被蓋野 (VTA) と黒質緻密部 (SNc) の  
**ドーパミン作動性ニューロン**の活動を観察すると...

- 学習前

**報酬が与えられた直後**に活発に活動する

- 学習後

**光刺激提示後**に活性化し、報酬直後の活性化がなくなった  
さらに、しばらく報酬を与えないと活動が抑制される

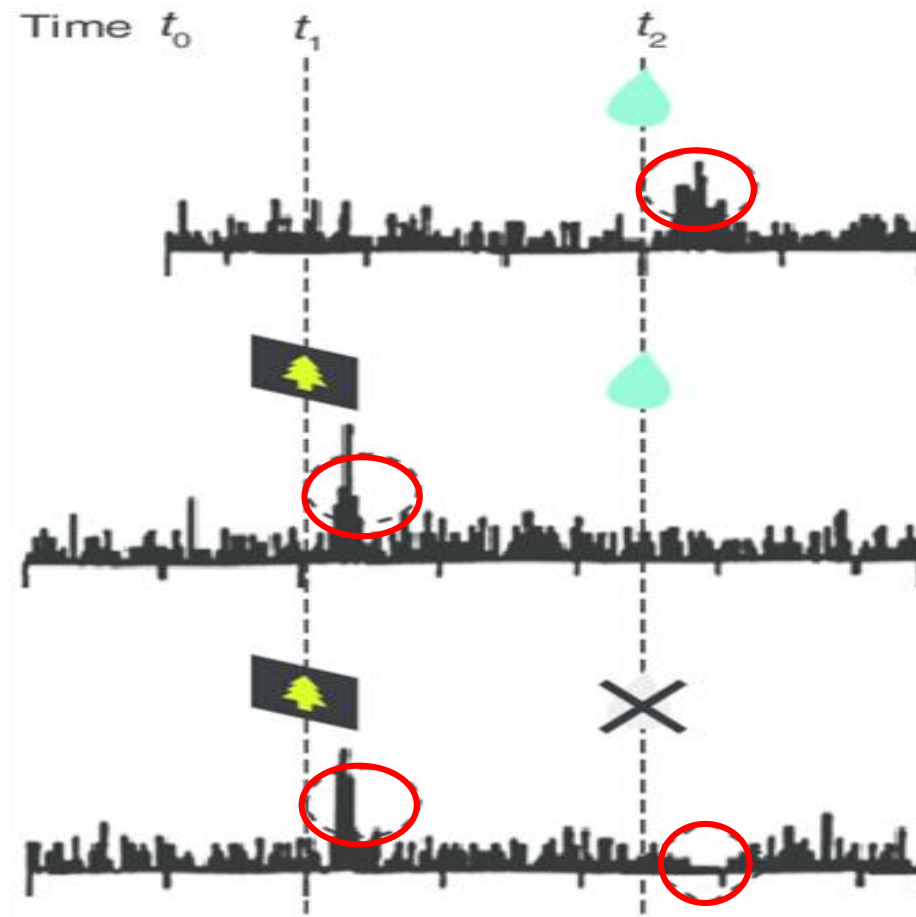
# ドーパミン作動性ニューロンとTD誤差 68

ドーパミンは報酬ではなく、**TD誤差**を伝搬している

光刺激なし

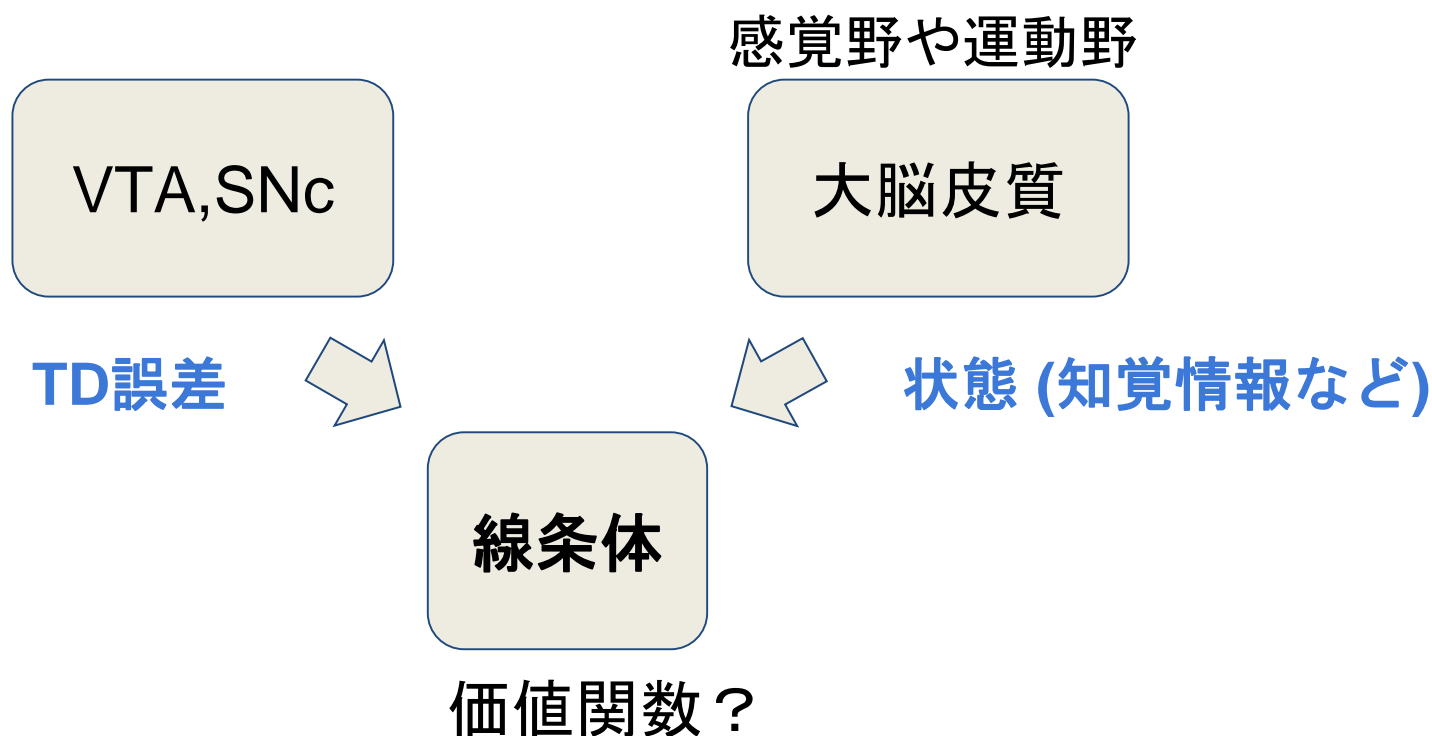
光刺激あり

報酬なし



[https://www.researchgate.net/profile/Mieko\\_Morishima/publication/225094122/figure/fig1/AS:393609133215754@1470855138803/Figure-1-Dopamine-neurons-encode-reward-prediction-error-a-It-has-been-suggested-that.png](https://www.researchgate.net/profile/Mieko_Morishima/publication/225094122/figure/fig1/AS:393609133215754@1470855138803/Figure-1-Dopamine-neurons-encode-reward-prediction-error-a-It-has-been-suggested-that.png)

大脳皮質からも投射を受けている線条体が  
状態価値関数や行動価値関数を推定している可能性がある

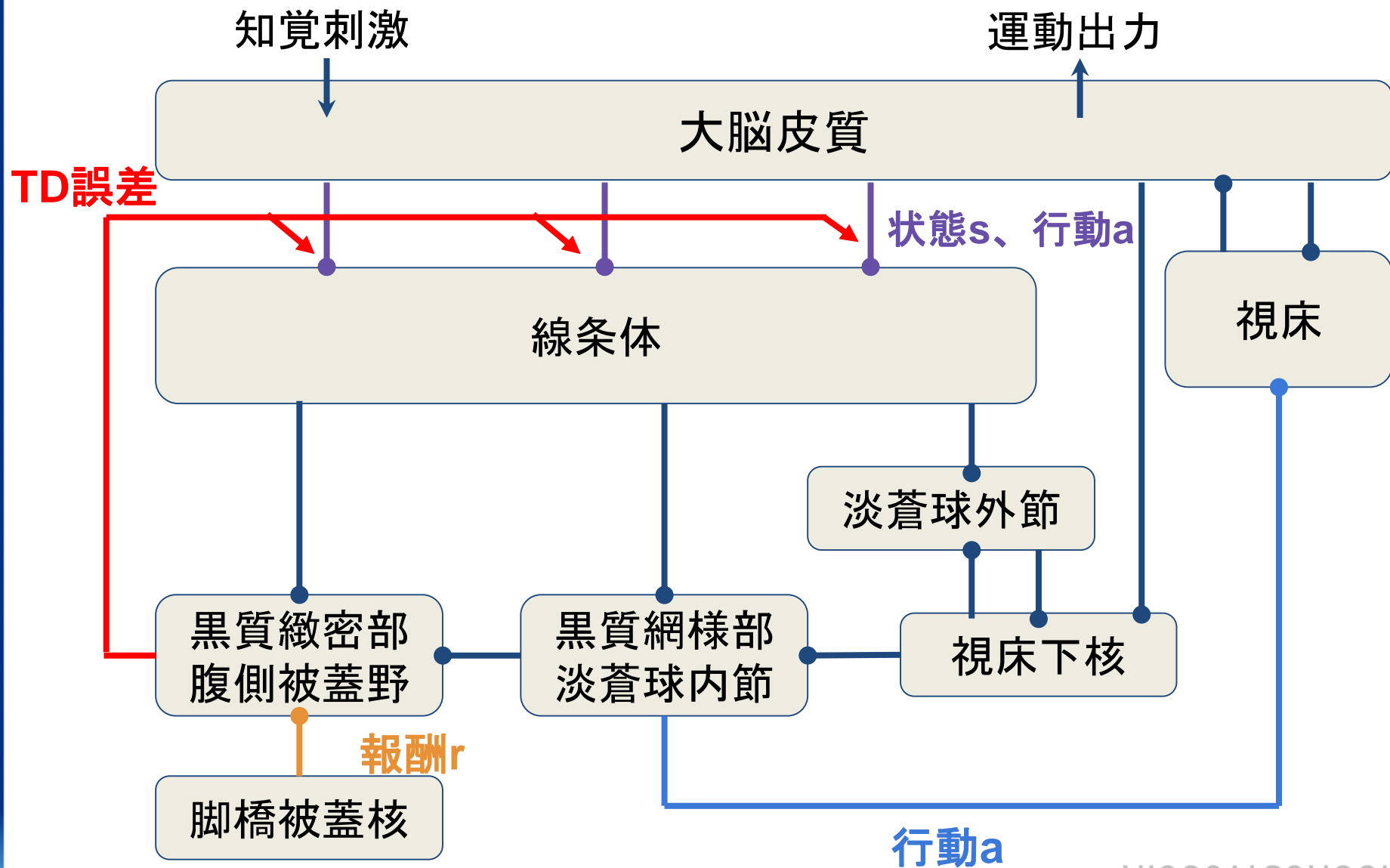


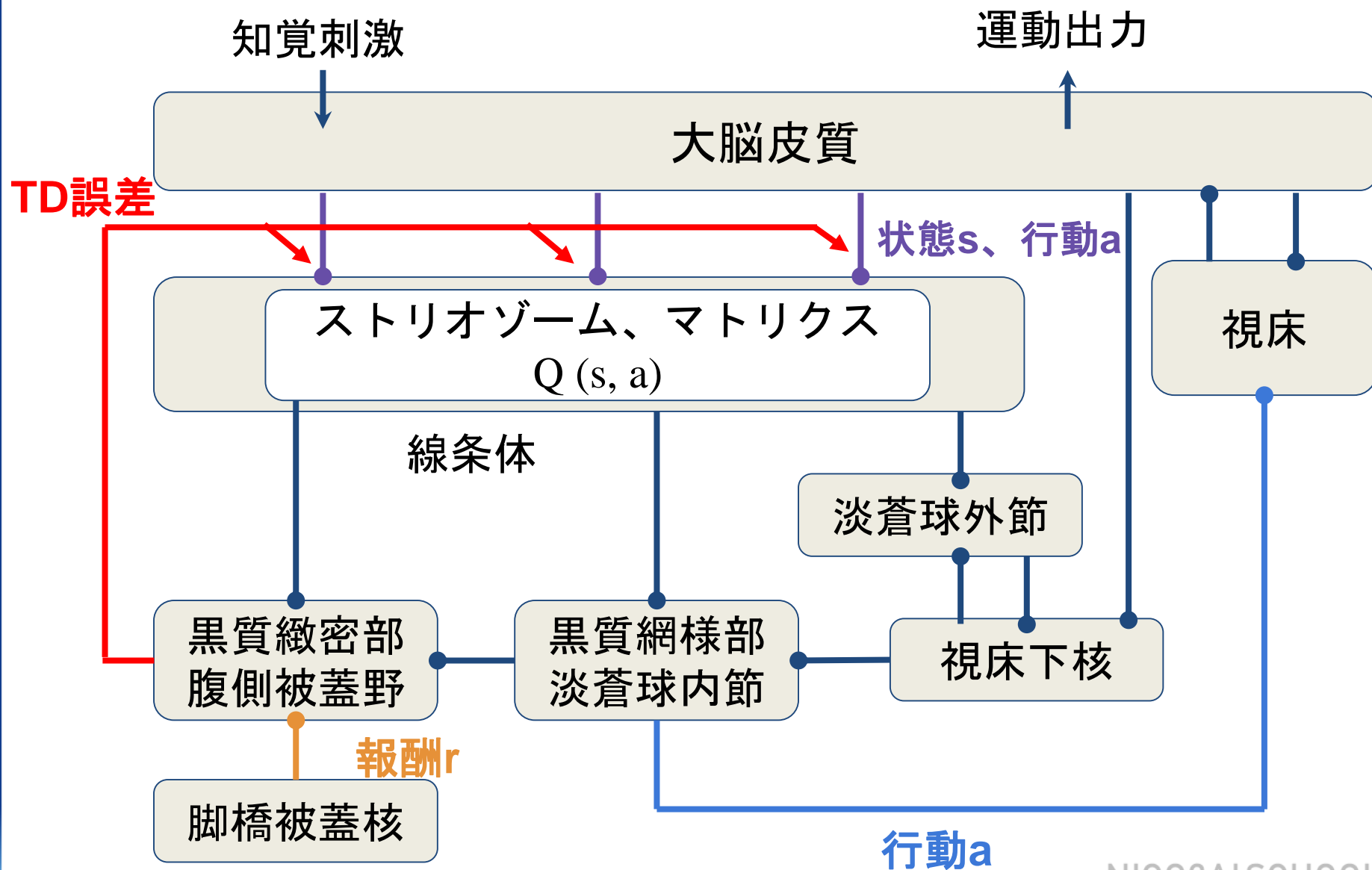
- 順番に3段階で光刺激を変えて一番明るい時に報酬を与えた  
段階に応じて**相関のある**ニューロンの活動が記録された  
→**状態価値関数**の推定

- レバーの左右で報酬発生確率が異なる条件でのタスク  
同じ行動でも確率が異なると違う活動が記録された  
**行動ごとに違う**活動も記録された  
→**行動価値関数**の推定

# 大脳基底核の強化学習モデル

71

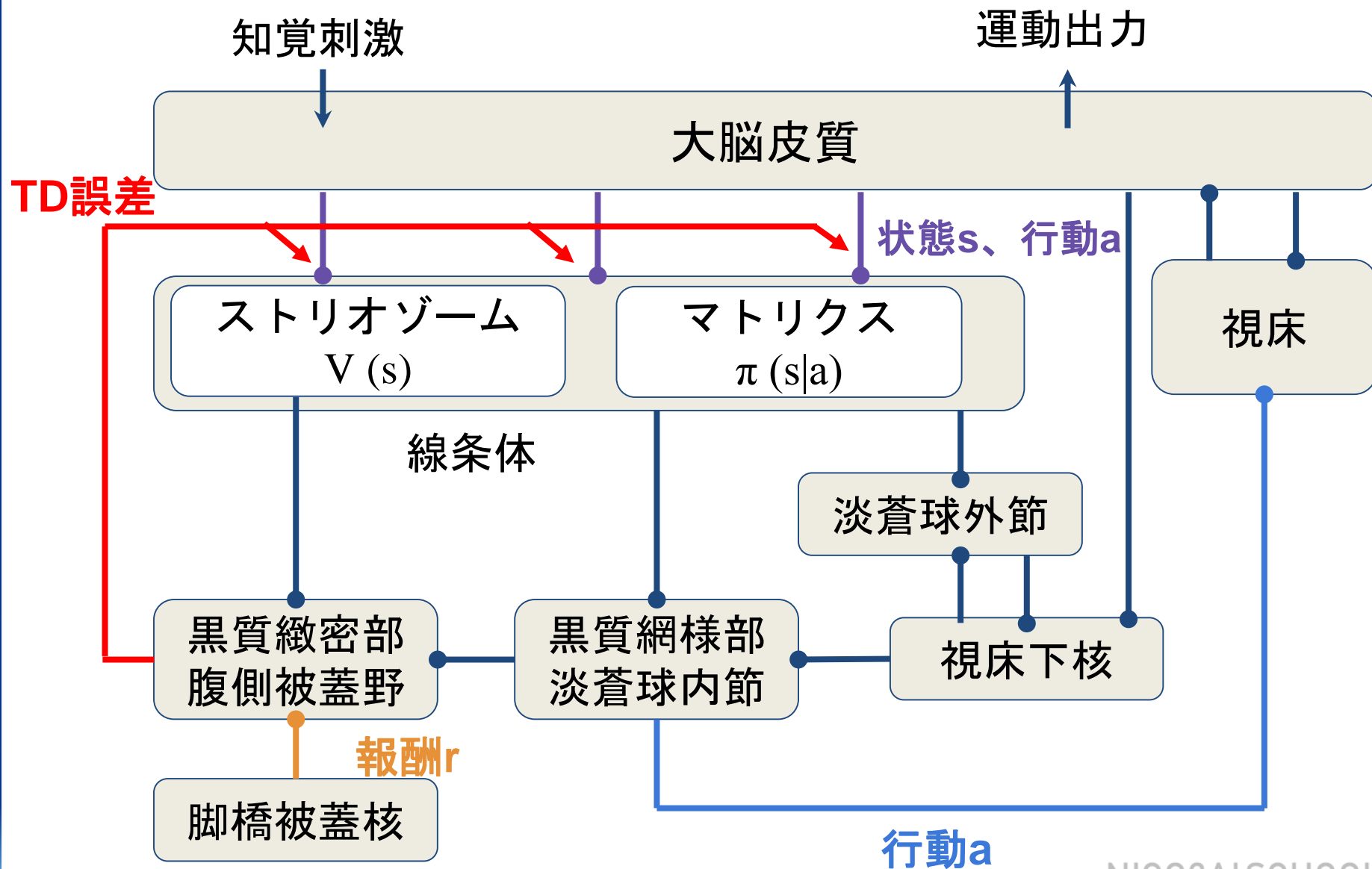






# BartoによるActor-Criticモデル

73



- 強化

生物は**強化信号**を元に学習を行なっている

- ドーパミンとTD誤差

腹側被蓋野 (VTA) と黒質緻密部 (SNc) のドーパミン作動性ニューロンは**TD誤差を線条体へ送っている**

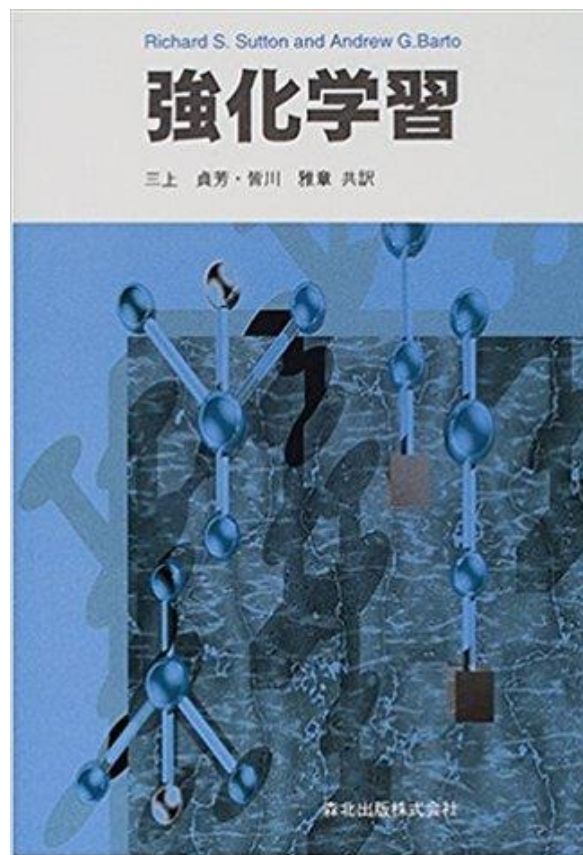
- 線条体と価値関数

腹側被蓋野 (VTA) と黒質緻密部 (SNc) からのTD誤差と大脳皮質からの情報を使って**価値関数を推定している**

## 強化学習

Richard S. Sutton and Andrew G. Barto

- 強化学習のバイブル的存在
- 基礎からわかりやすく書いてある
- 古いが未だに必須の一冊



これからの強化学習

牧野貴樹、澁谷長史、白川真一

- 基礎的な理論から最新までカバー
- 幅広い分野を参照することができる
- 入門書としては難しすぎる
- 今回の資料作成でお世話になった

