# 第１章 強化学習の基礎的理論

- 「これからの強化学習」（森北出版）(amazon) を自分用メモとして要約してみます。いろいろ省略しています。
- 青字は私のコメントであり、書籍に書かれているものではありません。

## 1.1 強化学習とは

### 探索と利用のトレードオフ

強化学習の問題では、エージェントは環境に関しての事前知識を持っていないことが多いため、探索と利用のトレードオフの問題がある。探索と利用のトレードオフの典型的な問題として多腕バンディット問題がある。

### 多腕バンディット問題に対するアルゴリズム

多腕バンディット問題に対するアルゴリズムとしては、以下がある。

- Argorithm 1.1.1 greedy アルゴリズム
  - まだ n 回選んだことのない腕がある場合、その腕を選ぶ
  - それ以外の場合、これまでの報酬の平均 $\mu_i$ が最大の腕を選ぶ
- Argorithm 1.1.2 $\epsilon$-greedy アルゴリズム
  - まだ n 回選んだことのない腕がある場合、その腕を選ぶ
  - 確率 $\epsilon$ で、すべての腕からランダムに選ぶ
  - 確率 $1-\epsilon$ で、これまでの報酬の平均 $\mu_i$ が最大の腕を選ぶ
- Argorithm 1.1.3 楽観的初期値法

  報酬の上界を $r_{sup}$ とする。
  腕 $i$ を選んだ回数を $n_i$、これまでの報酬の平均を $\mu_i$ とする。
  $\mu_i^{'} = \dfrac{n_i\mu_i + Kr_{sup}}{n_i + K}$ が最大のものを選ぶ
- Argorithm 1.1.4 UCB1 アルゴリズム

  払戻額の最大値と最小値の差を $R$ とする。
  - まだ n 回選んだことのない腕がある場合、その腕を選ぶ
  - そうでない場合、各々の腕から得られる報酬の信頼区間の半幅 $U_i = R\sqrt{\dfrac{2\ln(\sum n_i)}{n_i}}$ を求め、$\mu_i + U_i$ が最大の腕を選ぶ

**memo**

UCB1 の信頼区間の計算方法は良くわからなかったが、分散を保守的に見積もっているということだろうか？

## 1.2 強化学習の構成要素

### 強化学習の枠組み

強化学習の枠組みはエージェント・環境・それらの相互作用からなる。環境は原則として所与で、エージェントの内部構造を設計する。

- エージェント：意思決定の主体
- 環境：エージェントが相互作用を行う対象
- 相互作用：情報の受け取りと引き渡しを行うこと

### マルコフ決定過程 (MDP)

基本的な数理モデルとしてマルコフ決定過程がある。マルコフ決定過程 (MDP) においては、時間ステップごとに以下の情報をやり取りする。

- 状態：エージェントが置かれている状況
- 行動：エージェントが環境に対して行う働きかけの種類
- 報酬：その行動の即時的な良さ

マルコフ決定過程は、状態空間 $S$, 行動空間 $A(s)$, 初期状態分布 $P_0$, 状態遷移確率 $P(s'|s,a)$, 報酬関数 $(s,a,s')$ により記述される確率過程である。
なお、

- 次の状態は現在の状態および行動によって確率的に決定される
- 報酬関数は決定的な場合と確率的に求まる場合がある

以下のように用語を定義する。

- 方策 $\pi$：エージェントが行動を決定するためのルール。良い方策とはより多くの収益を得られる方策である。方策には状態 s のもとで常に同じ行動である決定論的方策と確率的に行動を決定する確率論的方策がある。
- 収益：累積の報酬であり、割引報酬和が良く用いられる。
- 状態価値 $V^{\pi}(s)$:ある状態、ある方策のもとでの収益の期待値

- 最適方策 $\pi^*$:どの状態でも他の方策以上の状態価値を持つ方策。最適方策は少なくとも一つ存在する。
- 最適状態価値関数 $V^{\pi^*}(s)$:最適方策の下での状態価値
- 行動価値 $Q^{\pi}(s,a)$:ある方策で状態 a, 行動 s を取った場合の報酬の期待値
- 最適行動価値関数 $Q^*(s,a)$:最適方策の下での行動価値

#### 方策の定め方

以下のように方策の定め方がある。他にもいくつかのパラメータを用いて直接方策を定める方法がある。(1.4 節参照)

- greedy 方策：最も行動価値の大きな行動を選ぶ
- $\epsilon$-greedy 方策：確率 $\epsilon$ でランダムに行動し、確率 $1-\epsilon$ で最も行動価値の大きな行動を選ぶ
- ボルツマン方策:$\pi(a|s) = \dfrac{\exp(Q(s,a)/T)}{\sum_{b\in A} exp(Q(s,b)/T)}$、T は温度パラメータ

## 1.3 価値反復に基づくアルゴリズム

#### ベルマン方程式

ある方策 $\pi$ のもとでの価値関数 $V^{\pi}(s)$、行動価値関数 $Q^{\pi}(s,a)$ は、報酬が割引報酬和のとき、ベルマン方程式により表すことができる。なお、ベルマン方程式から直接価値を求めるには、状態遷移確率が予め分かっている必要があるが、一般には状態遷移確率は未知である。
ベルマン方程式：

$$V^{\pi}(s) = \sum_{a\in A(s)} \pi(a|s) \sum_{s'\in S} P(s'|s,a)(r(s,a,s') + \gamma V^{\pi}(s'))$$

$$Q^{\pi}(s,a) = \sum_{s'\in S} P(s'|s,a)(r(s,a,s') + \sum_{a\in A(s')} \gamma P(a'|s')Q^{\pi}(s',a'))$$

#### Sarsa

ベルマン方程式を試行錯誤で解くアルゴリズムの 1 つが Sarsa である。以下の式でアップデートする。

$$Q\left(S_t,A_t\right) = Q\left(S_t,A_t\right) + \alpha(R_{t+1} + \gamma Q(S_{t+1},A_{t+1}) - Q(S_t,A_t))$$

ここで、$R_{t+1} + \gamma Q(S_{t+1},A_{t+1}) - Q(S_t,A_t)$ を TD 誤差と呼ぶ。

#### ベルマン最適方程式

最適状態価値関数や、最適行動価値関数には、ベルマン最適方程式がある。

$$V^*(s) = \max_{a \in A} \sum_{s' \in S} P(s'|s,a)(r(s,a,s') + \gamma V^*(s'))$$

$$Q^*(s,a) = \sum_{s' \in S} P(s'|s,a)(r(s,a,s') + \gamma \max_{a' \in A} Q^*(s',a'))$$

状態遷移確率が既知であれば DP で計算可能だが、状態遷移確率が未知の場合は試行錯誤で計算するため、Q-learning などを使用することになる。

#### Q-learning

Q-learning では、Sarsa とは異なり、遷移後の行動選択についての確率は含まれず、常に最大の行動価値を目標値として更新する

$$Q\left(S_t, A_t\right) = Q\left(S_t, A_t\right) + \alpha(R_{t+1} + \gamma \max_{a' \in A(s')} Q(S_{t+1}, a') - Q(S_t, A_t))$$

#### 方策のアップデート

Sarsa や Q-learning は方策に対する価値関数を学習するものであり、より良い方策を求めるためには、方策をアップデートする必要がある。以下のアプローチに分けられる。

- 価値反復法: 方策を greedy 方策などの価値関数から簡単に計算する方策に限定する。方策を表現する必要がないため、実装が簡単。
- 方策反復法: 学習状態として方策を表現し、それを利用して価値関数を計算する。多様な方策を表現できる。

#### memo

ベルマン方程式やベルマン最適方程式は、単に状態価値や行動価値の式の時点を 1 時点分展開しただけだと思われる。

### 1.4 方策勾配に基づくアルゴリズム

#### 確率的方策の表現

期待収益を目的関数 $J(\theta)$ として、これを最大化する確率的方策 $\pi_\theta$ のパラメータ $\theta$ を求めることを考える。$\pi_\theta$ は $\theta$ に関して微分可能な関数であるとする。

例えば、softmax 関数を使うと、状態空間と行動空間がともに離散の場合には以下のように表せる。

$$\pi_\theta(a|s) = \frac{\exp(\theta_{sa})}{\sum_{b \in A} \exp(\theta_{sb})}$$

状態空間が連続、行動空間が離散の場合には以下のように表せる。($\phi(s,a)$ は任意の関数。)

$$\pi_\theta(a|s) = \frac{\exp(\theta^T \phi(s,a))}{\sum_{b \in A} \exp(\theta^T \phi(s,b))}$$

### 勾配法

勾配法では、パラメータ $\theta^t$ を勾配方向へ以下のように更新していく。

$$\theta^{t+1} = \theta^t + \eta \nabla_\theta J(\theta)$$

方策勾配定理によると（割引報酬の期待値の場合)、勾配は行動価値関数を用いて以下のように表される。

$$\nabla_\theta J(\theta) = E_{\pi_\theta}[\frac{\partial \pi(a|s)}{\partial \theta} \frac{1}{\pi(a|s)} Q^\pi(s,a)] = E_{\pi_\theta}[\frac{\partial \log \pi_\theta(a|s)}{\partial \theta} Q^\pi(s,a)]$$

勾配は解析的に求まらないため、行動を行ったサンプルから近似する必要がある。また行動価値関数についても未知なので、推定する必要がある。

### 即時報酬での近似

最も単純には行動価値関数を以下のように即時報酬で近似する方法で、REINFORCE アルゴリズムに用いられる。

$$Q^\pi(s_t, a_t) \approx R_t$$

### Actor-Critic モデル

Actor-Critic モデルという、方策のモデルと行動価値関数のモデルを別々にモデル化する方法がある。状態空間や行動空間が連続の場合、何らかの関数により行動価値関数を近似する必要がある。以下のモデルが考えられる。

$$Q^w(s,a) = w^T \phi(s,a)$$

$w$ はパラメータベクトル。$\phi(s,a)$ は任意の関数であるが、$\nabla_\theta \log \pi_\theta(a|s)$ を採用すると都合が良い。

#### 自然勾配法

パラメータ $\theta$ が確率分布を定めていることを勘案し、確率分布間の距離を KL ダイバージェンスで定めると、自然勾配とよばれる勾配方向が導出される。

$$\tilde{\nabla}_\theta J(\theta) = F^{-1}(\theta)\nabla_\theta J(\theta)$$

$$F(\theta) = E[(\nabla_\theta \log \pi_\theta(a|s)(\nabla_\theta \log \pi_\theta(a|s)^T]$$

#### 具体的なアルゴリズム例

（省略）

**memo**

- 行動価値関数の近似に即時報酬しか使わないのでは、さすがに一定程度複雑なタスクには適用できなさそうである。
- 勾配方策定理の $\frac{1}{\pi(a|s)}$ が出てくる理由は、期待値を取っているためだと思われる。
- 自然勾配法は、どういった流れで”自然勾配”を求めたのかはわからなかった。ただ、1.4.3 の自然勾配方策法で示されているように、このように勾配を採用すると自然勾配は $w$ となりシンプルな計算となるようである。

### 1.5 部分観測マルコフ決定過程と強化学習

#### 部分観測マルコフ決定過程 (POMDP)

部分観測マルコフ決定過程は、マルコフ決定過程に以下を加えたもの。

- 観測集合 $\Omega$：エージェントの観測を要素に持つ有限な集合
- 観測関数 $O(s', a, o)$:エージェントの観測を記述する関数

なお、エージェントは観測を知ることはできるが、状態は知ることができない。

### 解法の分類

- 環境に対するモデルの事前知識
  - モデルベースド：事前知識として環境の知識を陽に用いる
  - モデルフリー：環境の知識を陽に用いない。Q-learning は直接環境の知識を用いるわけではないので、モデルフリーである。
- 価値や方策を求めるタイミングの視点
  - オフライン：価値計算や方策を完全に求めてから、得られた方策を実行する
  - オンライン：価値計算や方策を求めながら、その時点で得られている方策を実行する
- 価値や方策を求める理論的な視点
  - 厳密解法：理論通り正確に解を求める方法
  - 近似解法：求解が難しい場合に近似的に解を求める方法
  - ヒューリスティクス：理論的な裏づけはないが実証実験で確かめられている方法

### 信念状態

どの状態にいるかの確率を並べてできる状態。状態の要素数が 2 つであれば、$b(s_1, s_2) = (p_1, p_2)$ と表される。ここで、$0 \leq p_1, p_2 \leq 1$, $p_1 + p_2 = 1$ である。

### $\alpha$-ベクトル

ある観測に対してどのような行動を取るかを定めた場合の価値関数 (状態数の次元を持つベクトル)

ある信念状態に対する価値関数を、$\Gamma$ を $\alpha$-ベクトルの集合として以下のように表せる。

$$V(b) = \max_{\alpha \in \Gamma} \sum_{s \in S} b(s)\alpha(s)$$

### exact value iteration

価値関数をステップで展開した式を用いて、価値関数を更新する。しかしながら、更新とともに $\alpha$-ベクトルの数がどんどん増えていくため、枝狩りを行っても計算量が厳しい。

### Point-Based Value Iteration

（省略）

### 関連するモデル、モデルフリーな手法

（省略）

### memo

- POMDP では状態遷移確率や観測関数は既知という前提なのかどうかは良くわからなかった。
- この節の中盤の議論は、信念状態を状態だとみなせば、POMDP は MDP の枠組みで考えられるということだと思った。（計算は複雑になるが）
- Point-Based Value Iteration は、ある信念状態に限定して $\alpha$-ベクトルを更新していくということだと思うのだが、部分集合 B の選び方など、ロジックは良くわからなかった。