2020 年度　卒業論文

『粘菌を活用した新たな経路探索アルゴリズムに関する研究』

~ダイクストラ法・粘菌アルゴリズム・強化学習の比較検証~

早稲田大学高等学院

3 年 A 組 43 番　山本翔太

目次

『粘菌を活用した新たな経路探索アルゴリズムに関する研究』

~ダイクストラ法・粘菌アルゴリズム・強化学習の比較検証~

早稲田大学高等学院
3 年 A 組 43 番　山本翔太

要約

本研究では、粘菌アルゴリズムと強化学習の生物らしさという共通点に着目した新たなアルゴリズムの開発を目指した。本論文で紹介する内容は、その第一歩としての位置づけである。粘菌アルゴリズムと強化学習、及びダイクストラ法を python で実装し迷路を解かせた場合の比較実験を行った。対象とする迷路は、粘菌が迷路を解くことを発見した中垣氏の実験から参考にした。その結果、ダイクストラ法、粘菌アルゴリズム、強化学習の順番で処理時間が短いことが分かった。また、強化学習は安定して解を求めることができる一方で、粘菌アルゴリズムは安定しなかった。一方で、アルゴリズム開発に向けて、扱うデータの形式の違い、粘菌アルゴリズムの不安定性、大規模実験の必要性などの課題が明らかとなった。

1. 緒言

1.1　研究背景

COVID-19 による”Stay Home”の動きが世界中で広まることにより、フードデリバリーやオンラインショッピングによる配達サービスの需要の増加が一層顕著になっている。そのため、より効率的な配達を行うことで配達員の負担軽減やサービス向上を実現することが課題となっている。配達時には企業独自の経路探索アルゴリズムを使うが、そこには改善の余地が多くある。複数の集荷地と複数の配達地の経路導出、渋滞状況や信号の数、車種によるリアルタイムの情報更新、配達する料理や荷物の種類によって優先順位を変えるなどがある。これらの多くの要素、膨大な数の情報を処理するには、強力な経路探索アルゴリズムが必要となってくる。

最短経路を求める手法として、従来からダイクストラ法が最も基本である。これは現在カーナビやマップなどに利用されているものである。だが、分岐が増えるほど計算量が爆発的に増えてしまうというデメリットがある。生物からヒントを得た手法では、遺伝的アルゴリズムや蟻コロニー最適化、粘菌アルゴリズムなどが提案されている。近年活発に研究されている機械学習の一種である強化学習も、最短経路問題に応用可能であると考えられる。

そこで、本研究では生物由来の手法であり、最短経路問題を解くことができる粘菌アルゴリズム及び強化学習に着目した。粘菌という生物は構造が単細胞生物並みに単純だが、学習、記憶、予測など、神経回路があるような動きをする場合がある。粘菌から着想を得た粘菌アルゴリズムは迷路を最短経路で解くだけでなく、時間変化する渋滞を考慮した高速道路の経路導出や各都市間をつなぐ鉄道網の生成など様々な問題に対応できる。粘菌アルゴリズムの特徴として、比較的に短時間で最適ではないが実用に値する解を求めることができる。また、強化学習は脳の報酬予測の仕組みを取り入れており、深層学習を組合せた深層強化学習は膨大なデータに対応することができる。両者の共通点として、餌または報酬を求めるという点がある。その生物らしい点にこそ、より効率的な経路探索アルゴリズムを開発する糸口があると考えた。

1.2　研究目的

本研究の目的は、粘菌アルゴリズムと強化学習の生物らしさという共通点に着目した新たなアルゴリズムの開発することである。本論文では、その第一歩として両アルゴリズムの数理モデルを調べ、実装を通してその共通点や相違点を考える。また、従来の手法であるダイクストラ法も比較対象として扱う。特に、アルゴリズム開発に向けてどのような課題があるか、生物らしさはどれほど共通しているかに着目し考察する。

2.　各種アルゴリズム

2.1　粘菌とは

まず初めに粘菌に関して説明する。粘菌は変形菌(真性粘菌)に分類され、一種の単細胞生物である。粘菌は子実体と変形体の 2 種類の形態を持つ。粘菌の変形体はスライムのような形をしている多核体である。図 1[i]と図 2 にその一例を示す。つまり、粘菌の変形体は核を複数持つが全体で一つの細胞であり、核分裂をするが細胞分裂をしないというユニークな

特徴を持っている。子実体に関しては本論文では割愛する。粘菌のからだは原形質でできており、管がネットワーク状に広がっている。また、小さな塊として切り分けてもそれぞれが個体として生存することができ、個体が再び集まって融合することもできる。通常の単細胞生物は神経細胞を持たないが、化学物質の伝達によって情報伝達を行える。一方、粘菌の変形体は数センチ、数十センチの大きさでシート状に広がるため、前述の方法では間に合わない。しかし、粘菌は迷路解き、記憶や学習、個性や逡巡など様々な面で高い情報処理能力を持っている。

図 1：野外の倒木の隙間を這う粘菌

図 2：真性粘菌の変形体。シート状のからだに多くの管が伸びている。

北海道大学の中垣氏はここに注目し、粘菌の一種である和名モジホコリ(学名 Physarum Polycephalum)が迷路を解くことを実験で証明した[ii]。まず、モジホコリを小さな塊に切り分け、迷路に敷き詰める。次に、2 か所に餌であるオートミールを置き、スタートとゴールに見たて、一定時間観察を行う。最初に、行き止まり経路にある管が徐々に細くなり消失する。次に、ゴールへと続く 4 通りの組み合わせが残る。最後に、最短経路を作る 1 通りの組み合わせのみが残る。図 3[iii]は実際の粘菌による解法を示す。

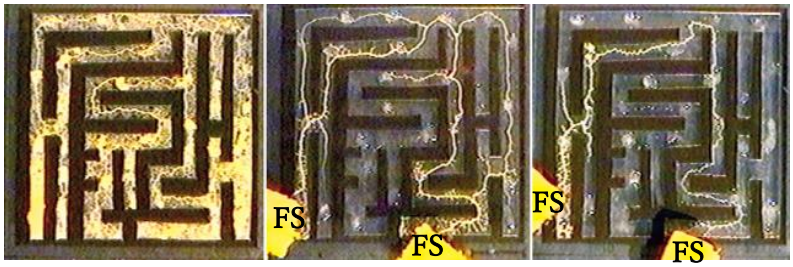


図 3：実際の粘菌による迷路の解法。1 辺ほぼ 4 ㎝の四角形。FS は餌を意味する。

この原理を説明しよう。まず前提として、粘菌のからだは外側が高粘度で流動性がないゲル状、内側が低粘度で流動性があるゾル状となっている。また、管の中には周期的に前進と後退を繰り返す粘り気のある原形質が流れている。その中には栄養や情報伝達に用いる化学信号が含まれ、それらを効率よく伝達するには管の物理的抵抗は小さい方が良い。そのため、粘菌は餌同士を連結する管が最も短くかつ太くなるようにからだを変化させる。また、

管の中を原形質が流れる際、管と原形質間にずり応力と言う力が発生し、これが管を成長させる。一方、原形質が流れない管は徐々に細くなり消える。これらの結果が迷路上の最短経路として表れている。

2.2　グラフ理論

ダイクストラ法の説明を始める前に、グラフ理論に関して少し解説する。

グラフ理論とはグラフ、ネットワークに関する数理的理論である。グラフとは、節点(ノード=Node)と辺(エッジ=Edge)による集合である。ネットワークとは、グラフに物理的な意味を持たせたものである。グラフは隣接しているかどうか、つまり節点同士が直接つながっているかどうかを 1 と 0 で表すことで隣接行列に変換することができる。

また、グラフには有向グラフと無向グラフがあり、有向グラフはエッジに矢印が付き、ノードに順序を持つ。グラフのエッジには重み(weight)という属性を持つ。経路問題では距離、時間などが重みとして扱われる。重み付きグラフの隣接行列には 1 の代わりに重みを代入する。図 4 は重み付きグラフの一例を示す。図 5 は図 4 のグラフを隣接行列に変換したものである。

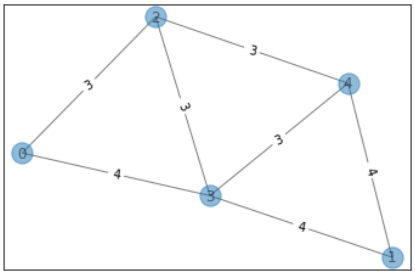


図 4 ：重み付きグラフの例

$$\begin{pmatrix} 0 & 0 & 3 & 4 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 3 & 0 & 0 & 3 & 3 \\ 4 & 4 & 3 & 0 & 3 \\ 0 & 4 & 3 & 3 & 0 \end{pmatrix}$$

図 5 ：隣接行列の例

2.3　ダイクストラ法

ダイクストラ法は最短経路問題を解くアルゴリズムの一種であり、グラフの形式を取り扱う。カーナビやマップなどで広く使われているが、分岐が増えると計算量が爆発的に増える欠点がある。現実では、主要道路と分岐道路分けること分岐の数を減らしている。ダイクストラ法の更新式は以下の通りである。

1. 全ての節点に∞を代入する
2. 始点に 0 を代入し、**確定**とマークする
3. 始点(**確定**)から直接つながる全ての節点までの距離を計算し、元の値(∞)より小さけれ

ば、新しい値に更新する

4-1.　それらの節点の中から距離が最も短いものを選び、**確定**とマークする

4-2.　**確定**とマークされた節点から直接つながる全ての**未確定**節点までの距離を計算し、節点に代入されている値より小さければ、新しい値に更新する

4-3.　ステップ 4 を繰り返し、全て**確定**なら終了する。

2.4　粘菌アルゴリズム

手老氏、小林氏、中垣氏らは粘菌から得られた経路探索方法をフィザルムソルバ[iv](Physarum Solver)と名付けた。これが前述した粘菌アルゴリズムの正体であり、本論文では便宜上粘菌アルゴリズムと呼ぶ。図 6[v]は粘菌アルゴリズムによる迷路の解法を示す。現実の粘菌を用いたものと同じような結果が得られる。

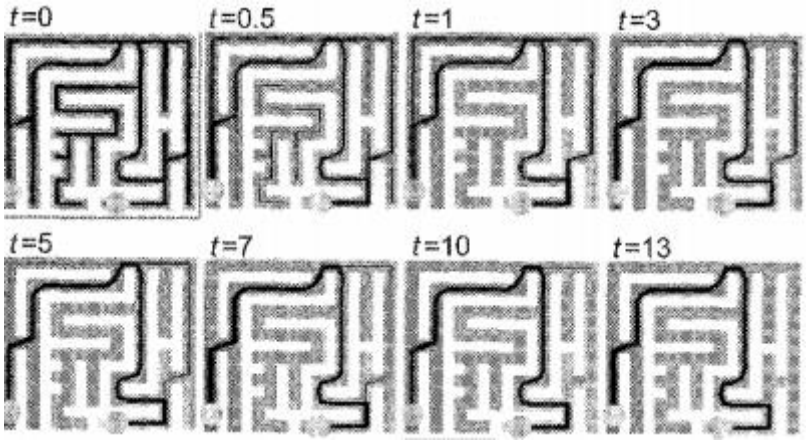


図 6 ：粘菌アルゴリズムによる迷路の解法を示す。適当に取った時間単位 t=0~13 の経路探索を表す。

以下では粘菌アルゴリズムの数理モデルを解説する。図 7[vi]は迷路の辺と節点を文字で表したものである。

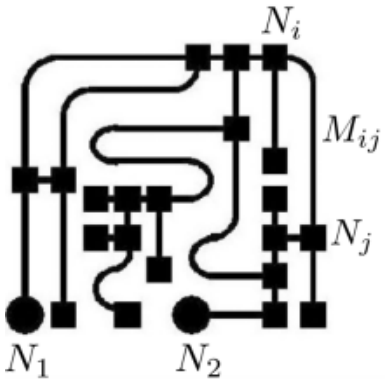


図 7 ：迷路の辺と節を文字で表したもの

各文字の意味：

・餌を置いた地点をそれぞれスタート、ゴールとみなしN_1, N_2とする。

・その他の節点をN_3, N_4 ... と番号順に表す。

・節点N_i, N_jを結ぶ辺をM_{ij}とする。節点の組が複数の辺で繋がるときM_{ij}^1, M_{ij}^2 ...と表す。

・節点N_i, N_jにおいて、管の壁が原形質を押す圧力をそれぞれp_i, p_jとする。

・辺M_{ij}の長さはL_{ij}とする。

・辺M_{ij}における単位時間あたりの流量をQ_{ij}とする。

・辺M_{ij}の伝導率(Conductivity)をD_{ij}とする。伝導率は管の中を原形質がどれだけ通りやすいかを表すことができる。

上記の各文字を用いて、原形質の方程式(2)と管の成長方程式(3)を表現することができる。粘菌アルゴリズムの数式は次の論文[vii]を参考にした。本論文の冗長性を避けるためその導出を付録 A にまとめた。

粘菌アルゴリズムを数学的に解くことで、最短経路問題を解くことができる。時刻t_0 から時刻t_nまで n 回ループさせるため、下記の式(1)(2)(4)を用いて実装する。式(3)を整理すると、時刻t_{n+1}の伝導率${D_{ij}}^{n+1}$の更新式(4)を得ることができる。

$$ {Q_{ij}}^n = \frac{D_{ij}^n}{L_{ij}}\left(p_i^n - p_j^n\right) \quad (1) $$

$$ \sum_i \frac{D_{ij}^n}{L_{ij}}\left(p_i^n - p_j^n\right) = \begin{cases} -\mathrm{I}_0(\mathrm{j}=1)\ \text{スタート} \\ \mathrm{I}_0(\mathrm{j}=2)\text{ゴール} \\ 0(\mathrm{j}\neq 1,2)\text{その他} \end{cases} \quad (2) $$

$$ \frac{D_{ij}^{n+1} - D_{ij}^n}{\delta t} = f\left(\left|Q_{ij}^n\right|\right) - D_{ij}^n \quad (3) $$

$$ D_{ij}^{n+1} = D_{ij}^n + \delta t\left(f\left(\left|Q_{ij}^n\right|\right) - D_{ij}^n\right) \quad (4) $$

最短経路問題を解く手順は以下の通りである。

時刻t_nにおいて

(a) それぞれの辺の数値、長さL_{ij}、伝導率D_{ij}^n、圧力p^nを式(2)に代入し、連立一次方程式を作る。

(b) ガウスの消去法(Gaussian Elimination Method)を用いて連立一次方程式を解き、それぞれのp^nを求める。

(c) 長さL_{ij}、伝導率D_{ij}^{n}、及び(b)で求めた圧力p^{n}の配列を式(1)に代入し、流量Q_{ij}^{n}を得ることができる。

(d) 時刻t_nの流量Q_{ij}^{n}と伝導率D_{ij}^{n}を式(4)に代入して、時刻t_{n+1}の伝導率D_{ij}^{n+1}を更新する。式(4)は微分方程式を解くための手法であるオイラー法を用いて解く。

時刻t_{n+1}において

(e) ステップ(a)~(d)を一定の条件を満たすまで繰り返す。

(f) 流量が多い経路ほど粘菌が最短とみなしているため、最終時刻においてのN_i,N_j間のQ_{ij}が相対的に大きい経路を始点から終点までつなげたものが最短経路であると言える。

上記の手順において実装のため以下の工夫がなされている。

・グラフを隣接行列の形に変換する。

・それぞれの辺の長さL_{ij}を入力する。

・それぞれの辺の伝導率D_{ij}^{n}の初期値D_{ij}^{0}(時刻)を[0.5, 1.0]の範囲でランダムにとる。

・圧力p^{n}の初期値p^{0}は 0 とする。pを変数として扱い連立一次方程式を解くからである。

・式(1)を$Q = \frac{D}{L}(p^{T} - p)$ の形で実装する。$p^{T} - p$で $p_i - p_j$を表現することができる。

・式(2)の連立 1 次方程式を作る際、行列形式で解を求めるため、pを使わずに方程式を表現する必要がある。そのため、以下の法則に従い$\frac{D}{L}$の行列を作成する。

対角成分がある場合は$\sum_j \frac{D_{ij}}{L_{ij}}$、

非対角成分で隣接成分があれば$-\frac{D_{ij}}{L_{ij}}$、

非対角成分で隣接成分がなければ 0 を隣接行列に代入する。

・方程式の右辺では、スタートに$-\mathrm{I}_0$、ゴールにI_0、その他に 0 を代入した行列を作成する。

2.5　強化学習

強化学習(Reinforcement Learning)とは、近年盛んに研究がされている機械学習の一種である。機械学習(Machine Learning)は文字通り「機械」を「学習」させるアルゴリズムである。機械学習の発展の背景には、機械学習が人工知能(Artificial Intelligence)を実現するために必要な技術であることがある。

強化学習の応用例として、迷路を解かせることやブロック崩しなどのゲームがある。全くプレイ方法を知らない前提で、学習により最終的に高いスコアを出すことができるようになる。また、AI が人間の囲碁のチャンピオンを打ち負かした例として有名な AlphaGo も強

化学習によるものである。実世界では、ロボットの自立歩行制御や自動運転技術にも応用されている。

では、強化学習とは一体どういうものだろうか。あるシステムを動かし、設定された目標を達成できれば報酬を与える。この仕組みによって、目標に達成できるように試行錯誤しながら、最適な動きを見つけることができる。これが強化学習の原理の簡単なイメージである。

・定義

強化学習に関する厳密な数学的な定義はないが、参考資料で比較的に分かりやすい下記の定義[viii]を引用する。

> 定義：強化学習
> 強化学習とは、ある環境におけるエージェントが現在の状態を観測して行動を選択し、それに応じて得られた報酬をもとに最適な行動を学習するアルゴリズムである。

・用語解説

次に、いくつかの重要な用語を解説する。

エージェント(agent)：強化学習における意思決定者である。迷路解き問題ならその迷路を解く主体、ゲームならそのゲームをプレイするプレイヤーのことを指す。また、エージェントは環境に対し相互作用を行う。

環境(environment)：強化学習において、エージェントを除く全ての部分のことであり、エージェントが相互作用を行う対象である。迷路解き問題における迷路そのもの、ゲームの枠組みを指す。

相互作用：相互作用は時間ステップ(t = 1, 2, 3, …, T)に渡って環境とエージェントとの間で行われている。なお、時間は取り扱い易さのため離散的な値を使う。エージェントは時刻tに状態s_tをもとに行動a_tを行い、それに対し環境が次ステップ(時刻$t+1$)の報酬r_{t+1}と状態s_{t+1}を返す。そして、またエージェントが状態s_{t+1}をもとに行動a_{t+1}を行う。図 8[ix]にエージェントと環境の相互作用を示す。

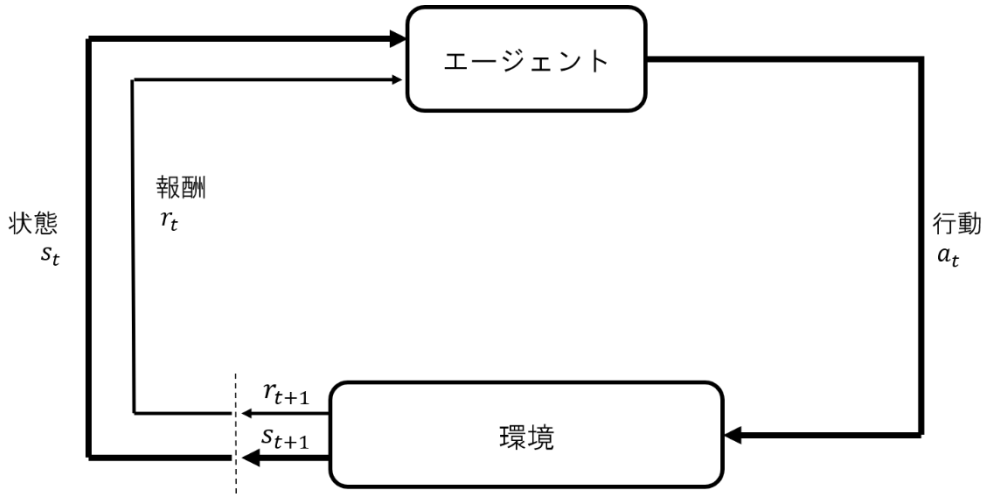


図 8：強化学習におけるエージェントと環境間の相互作用

エピソード(episode) ：環境の開始から終了までを 1 エピソードと言う。強化学習においてエージェントは 1 エピソードの報酬の総和が最大になるように学習する。

報酬(reward)：強化学習においてエージェントは報酬の総和が最大になるように学習する。そのため、報酬は学習させたいものに合わせて設定する。例えば、ボードゲームなどの環境において以下のように報酬を設定することが多い。(勝ち：+1、負け：-1、引き分け： 0）この報酬という仕組みによって、駒を捨てる代わりに有利な局面に立つなど長期的な視点をもつことができる。1 エピソードにおける報酬の総和は報酬関数R_tで表される。

下記の行動と状態に関しては迷路の例を用いて解説する。8×8 の迷路を想定する。

状態(state)：例えば、迷路におけるエージェントの位置、ロボット歩行制御における各脚の関節の角度や動作速度など。状態の集合を状態空間$\mathcal{S}$と言う。例：8×8 迷路の状態空間$\mathcal{S} = \{m \in \mathbb{Z}^{64} \mid m = 0, 1, \ldots, 13\}$　0~13 は壁がどの方向にあるかによって 14 通りがあることを表している。

行動(action)：例えば、迷路における上下左右どの方向に進むか、ロボット歩行制御における前進するか後退するかなど。行動の集合を行動空間$\mathcal{A}$と言う。例：基本的な迷路の行動空間$\mathcal{A} = \{0, 1, 2, 3\}$

方策(policy)：ある状態における選択可能な行動を選ぶ確率。一部の資料では戦略または政策と呼ぶ場合もある。時刻tおいて状態s_tならば行動a_tになる確率を$\pi_t(a_t|s_t)$と表す。また、$\pi(s, a)$と表す場合もある。

価値(value)：ある状態においてある行動を取ることがどのくらい良いのかに対する評価。状態価値V_πと行動価値Q_πの 2 種類がある。

・アルゴリズム実装に関して

論文が冗長になることを避けるため、強化学習に関する詳しい理論や式の導出は付録 B にまとめる。まだ説明されてない用語や概念は付録を参照して頂きたい。

以下では実験で用いる 2 種類の強化学習アルゴリズム及び実装上の工夫である ε -greedy 法を紹介する。両者ともに 1 ステップごとに方策を改善できる TD 学習の手法である。

・Q 学習

Q 学習における Q は行動価値である。Q 学習は価値が最大化するような行動のみを選択する Value ベースの手法である。Q 値を収めた表である Q-table(Q[s][a])を利用して価値を更新する。下記のように Q 学習の更新式(5)表す。ηは学習率を表す。

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta * \left(R_{t+1} + \gamma \max_a Q(s_{t+1}, a) - (s_t, a_t) \right) \quad (5)$$

・SARSA

SARSA(State-Action-Reward-State-Action)は Q 学習と同じく TD 学習の手法である。SARSA の実装や更新式は Q 学習に非常に似ており、違う点は方策に基づき行動を選択することである。そのため、SARSA は Policy ベースの手法である。また、SARSA は方策の改善と価値の更新を交互に繰り返す。下記のように SARSA の更新式(6)表す。

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta * \left(R_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - (s_t, a_t) \right) \quad (6)$$

・ε -greedy 法

モデルフリーの手法において、エージェントは自ら行動することで状態遷移関数や収益の推定値を求める。そのためにはランダムで行動することが良いが、報酬の総和(または価値)を最大化することはできない。どのくらい探索目的の行動を取り、どのくらい報酬目的の行動を取るか、これを探索と活用のトレードオフ(Exploration-exploitation trade-off)と呼ぶ。限られた行動回数の中で、一定の確率で探索を行い、それ以外は活用するようバランスを取る手法を ε -greedy 法と呼ぶ。 ε は探索目的の行動を行う確率であり、0.1 前後に設定することが多い。一般的に方策πは式(7)のように表す。

$$\pi(a_t|s_t) = \begin{cases} 1 - \varepsilon & (a_t = arg \max_{a \in \mathcal{A}} Q\left(s_t, a\right)) \\ \varepsilon & (otherwise) \end{cases} \quad (7)$$

3. 実験準備

3.1 実験方法・環境

実験内容は python で各アルゴリズムを実装し、簡単な迷路を解かせるという非常にシンプルなものである。この実験の目的は、処理時間や解の安定性の違いを比較し、その原因となるアルゴリズム間の違いを考察することにある。考察において、入出力形式、データ形式や移動ログに着目し分析する。

実験はアルゴリズムごとに 1〜3 に分けられる。

実験 1：ダイクストラ法

実験 2：粘菌アルゴリズム

実験 3：強化学習(Q 学習と SARSA)

また、実験で用いる環境及びライブラリなどのバージョンは下記の通りである。

Google Colaboratory

Python 3.6.9

Numpy 1.18.5

Pandas 1.1.2

Matplotlib 3.2.2

Networkx 2.5

draw.io

対象とする迷路は、中垣氏が粘菌の迷路解き実験に用いたものである。その迷路(図 9)をもとに復元したものが本実験で使った図 10 の迷路である。15×17 迷路でマス目の数は 255 である。図 10 では各マス目には 0~254 の数字が振り分けられている。白色は通路を示し、黒色は壁を示す。図 10 では餌の部分を始点と終点にしているため、図 11 では黄色でスタート、赤色でゴールを表している。また、迷路の復元に関して、通路と壁の関係といった迷路の形を重点に置いた。図 9 では各通路の横幅が一様でないが故に正方形であったが、図 10 では横の長さが長くなってしまった。そのため、図 10 における黄色で示した最短経路は図 9 と違うこと先に述べておく。

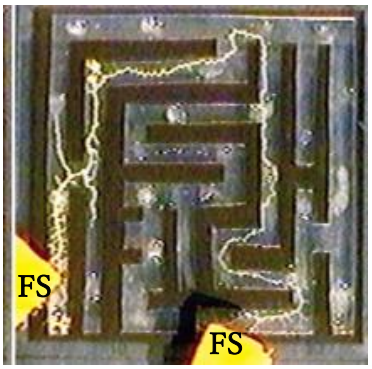


図 9；中垣氏が実験に使った迷路

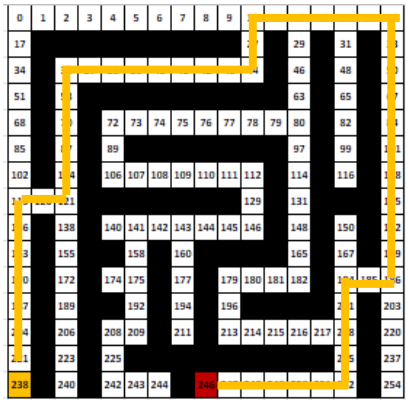


図 10；本実験で使った迷路

ダイクストラ法と粘菌アルゴリズムではグラフを扱うため、図 10 の迷路を下記の図 11 のグラフに変換した。B1 は B2 より 2 マス長く、最短経路は A1 or A2 +B2 の 53 マスである。

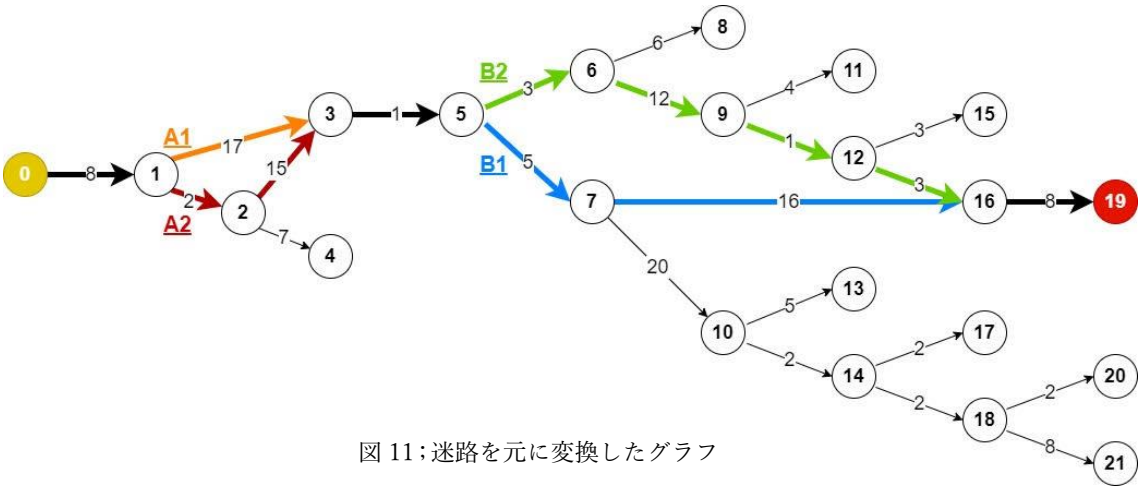


図 11；迷路を元に変換したグラフ

3.2　実装時の工夫

実験 1：ダイクストラ法

コードは次の書籍[x]のものを参考にした。通常の方法であればキューにある全ての頂点をループさせるが、優先度付きキュー(priority queue)を用いることで距離が短いものから取り出せるキューを作成できる。優先度付きキューの実装方法としてヒープ(heap)を使う。Python で提供されている heapq というモジュールを用いる。

実験 2：粘菌アルゴリズム

コードは Qiita の記事[xi]と次の論文[xii] [xiii]を参考にした。このアルゴリズムではグラフを隣接行列の形に変換して入力する必要がある。また連立一次方程式を解く際 numpy.linalg のライブラリを使うことでは計算できず、 LinAlgError: Singular matrix というエラーが出た。これを解決するため、次のサイト[xiv]を参考しガウス消去法を実装した。更新式はオイラー法を用いて実装した。更新回数$t = 100$、単位時間の変化量$\delta t = 1$とする。なお、$f(|Q_{ij}^n|)$は指数関数$|Q_{ij}^n|^{\mu}$とし、$\mu = 1$とする。

実験 3： 強化学習

実験 2 では強化学習の中の SARSA と Q-leaning という 2 つの手法を実装した。コードは次の書籍[xv]のものを参考した。学習率$\eta = 0.1$、割引率$\gamma = 0.9$、$\varepsilon = 0.5$、エピソード＝1000 とする。なお、εはエピソードごとに半減するようにした。学習が進むにつれ ε を小さくすることで前半を探索優先、後半を活用優先にすることができる。実験 3 では以下の 3 つに分けて行う。

(a) Random choice

ランダムに確率を選んだ場合。

(b) Q-learning

Q 学習のアルゴリズムで実装した場合。

(c) SARSA

SARSA のアルゴリズムで実装した場合。

4. 結果

4.1 各アルゴリズムの処理時間

実験 1；ダイクストラ法

ダイクストラ法のアルゴリズムを 10 回施行し、処理時間を表 1 にまとめた。処理時間は 0 .0001~0.0002 秒と安定しており、いずれも最短経路長 53 マスで解くことができた。平均時間は 0.0001 秒とかなり速い。

処理時間	0.0002	0.0001	0.0001	0.0001	0.0001	0.0002	0.0001	0.0001	0.0002	0.0001
平均時間	0.0001									

表 1：ダイクストラ法の処理時間

実験 2： 粘菌アルゴリズム

粘菌アルゴリズムのアルゴリズムを 10 回施行し、処理時間を表 2 にまとめた。処理時間はある程度安定しているが、毎回最短経路長で解くことはできなかった。平均時間は 0.1750 秒とダイクストラ法の 1000 倍以上である。

処理時間	0.1757	0.1746	0.1617	0.1555	0.1580	0.1935	0.1550	0.1647	0.1754	0.2361
平均時間	0.1750									

表 2：粘菌アルゴリズムの処理時間

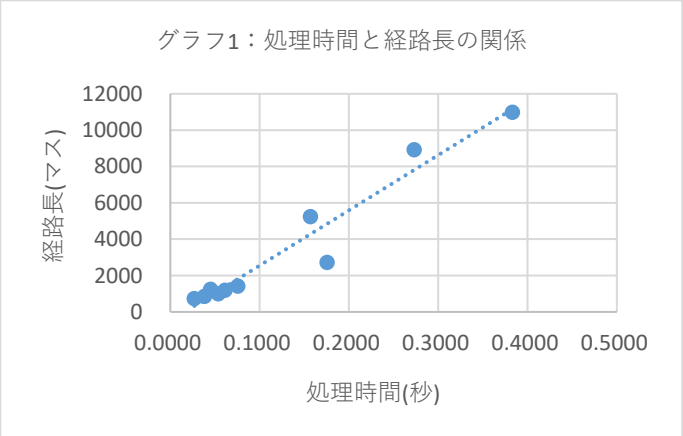
実験 3(a)：Random choice

エージェントがランダムに方向を選択するように実装し 10 回施行した。ゴールに至るまでの経路長と処理時間を表 4 にまとめた。処理時間と経路長は安定せず、10 回の施行中でも最長 8927 と最短 733 の差は大きい。

処理時間	0.0383	0.0270	0.0537	0.0452	0.1570	0.2730	0.1756	0.3833	0.0757	0.0613
経路長	849	733	1003	1241	5247	8927	2727	10975	1427	1185

表 3：ランダムに方向を選択する際の処理時間と経路長

表 4 を元に散布図を作成した。グラフ 1 から分かるように、またコンピューターの性質上自明であるが、処理時間と経路長に比例関係が見られた。



実験 3(b)：Q 学習

Q 学習のアルゴリズムを 10 回施行し、処理時間を表 5 にまとめた。ダイクストラ法ほど安定ではないが、概ね偏りは少ないことが見られる。また、粘菌アルゴリズムの平均時間の約 25 倍を使っている。

処理時間	4.9200	4.5943	3.9496	3.5222	6.1816	3.4918	3.7475	3.8981	4.3500	4.4539
平均時間	4.3109									

表 4：Q 学習の処理時間

実験 3(c)：SARSA

SARSA のアルゴリズムを 10 回施行し、処理時間を表 6 にまとめた。ダイクストラ法ほど安定ではないが、概ね偏りは少ないことが見られる。また、粘菌アルゴリズムの平均時間の約２０倍を使っている。平均時間は Q 学習よりやや短いが、両者に実用上大きな差はない。

処理時間	3.0413	2.6241	3.0303	2.4851	3.3793	3.2831	2.9913	3.7811	3.5158	2.6713
平均時間	3.0803									

表 5：SARSA の処理時間

4.2 粘菌アルゴリズムの不安定性

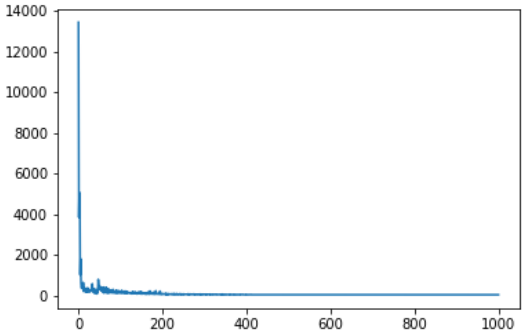
実験 2 では粘菌アルゴリズムを 10 回施行した。その一回の最終的な流量Q_{ij}^{99}を以下に示す。比較的に大きい流量は粘菌が良く使う管であり、粘菌が解いた最短経路だと解釈できる。したがって、最短経路は 0→1→2→3→5→6→9→12→16→19 となる。これは図 11 で見られる A2+B2 のルートと一致している。しかし、実験２での実装方法では毎回必ずこのルートを求めることができない。行き止まりに入ってしまう場合や、Q_{ij}が NaN になってしまう場合があった。また、粘菌アルゴリズムではグラフの通ったルートを戻ることが想定されておらず、スタートからゴールまで一方通行である。そのため、一旦行き止まりに入ったら改善することは難しい。また、更新回数$t = 300$以上からはQ_{ij}が NaN になってしまうため、回数を重ねることで改善することには限界がある。これらから粘菌アルゴリズムで解を安定して求めることは難しいと分かった。これは粘菌アルゴリズムの実用化を考える際の課題の一つである。

$Q_{01}^{99} = \mathbf{0.014}$,
$Q_{12}^{99} = \mathbf{0.783}$, $Q_{13}^{99} = 0.052$,
$Q_{23}^{99} = \mathbf{0.406}$, $Q_{24}^{99} = 0.236$,
$Q_{35}^{99} = \mathbf{9.26E + 16}$,
$Q_{56}^{99} = \mathbf{5774758}$, $Q_{57}^{99} = 0.035$,
$Q_{68}^{99} = 0.053$, $Q_{69}^{99} = \mathbf{35.826}$,
$Q_{710}^{99} = 6.30E - 05$, $Q_{716}^{99} = 9.97E - 04$,
$Q_{911}^{99} = 0.851$, $Q_{912}^{99} = \mathbf{55.129}$,
$Q_{1013}^{99} = 0.014$, $Q_{1014}^{99} = 0.120$,
$Q_{1215}^{99} = 0.051$, $Q_{1216}^{99} = \mathbf{1.67E + 21}$,
$Q_{1417}^{99} = 1.23E + 15$, $Q_{1418}^{99} = 0.099$,
$Q_{1619}^{99} = \mathbf{0.002}$,
$Q_{1820}^{99} = 4361.197$, $Q_{1821}^{99} = 0.041$

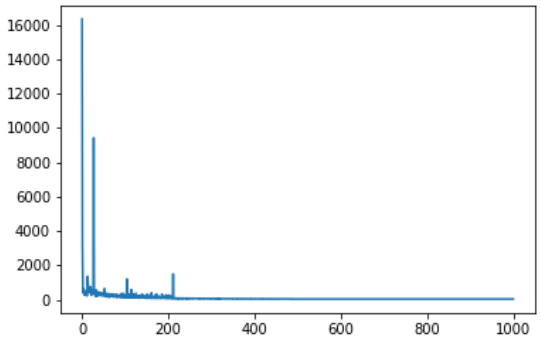
各辺における最終的な流量Q_{ij}^{99}

4.3　強化学習の安定性

実験 3 では Q 学習と SARSA をそれぞれ 10 回施行した。実装では更新する回数＝エピソードを 1000 と設定し、どれもエピソードが 1000 に達する前に最短経路長 53 マスで解くことができた。実験 3(b)(c)の任意の一回の施行のエピソードと経路長の関係を下記の折れ線グラフ 2, 3 で表す。横軸がエピソード数、縦軸が経路長を表す。



グラフ 2：Q 学習でのエピソードと経路長の関係



グラフ 3：SARSA でのエピソードと経路長の関係

グラフでは見づらいが、グラフの元となった資料では、Q 学習は 549 エピソード以降、SARSA は 483 エピソード以降最短経路長 53 マスを保っていることが分かる。これは学習が終了したことを意味する。なお、これらより前に 53 を求めたものもあったが、その後に 55、57 や 53 と近い値を繰り返していた。ここからエージェントが一旦通ったルートを往復していることが分かる。また、初期の段階では分岐点において前進と後退を繰り返す傾向があり、エピソード数が増えてもそれは残ることがエージェントの移動ログから分かった。グラフ 2, 3 から、約 200 エピソードあたりから安定し大きな起伏がなくなっている。これらから、強化学習は時間がかかるも、最終的には安定し必ず解を求めることができるアルゴリズムだと分かった。

5. 考察

5.1　処理時間の違いから見る特徴

4 種類のアルゴリズムの平均処理時間を比較したところ、ダイクストラ法が最も速く、次に粘菌アルゴリズム、最後に強化学習の SARSA、 Q 学習と続いた。本実験は簡単な迷路を使用しているため、処理速度が本当に試される分岐の数が多い最短経路問題では違う

結果が現れるかもしれない。また、実装方法の違いによって速度が変わることも考慮に入れるべきである。しかし、処理時間はいずれもある程度安定しており、平均時間の比較はアルゴリズムの性能の比較として参考できるものであると考える。なお、強化学習アルゴリズムにおいて、エピソード数を最短経路が求められるだろう最小の 600 にした場合、エピソードが 1000 のときと比べ大きな差はなかった。

この処理時間の違いはなぜ起こるのか、それは入力形式と扱うデータ形式から考察できる。ダイクストラ法はグラフを扱うため、節点、辺や重みを設定し、始点と終点を決めることで準備は完了する。要素の代入、比較、選択や入れ替えなどの処理を行うことで成り立つ。粘菌アルゴリズムでは、本質的に行列形式で連立一次方程式を解き、値を更新式に代入し計算することで成り立つ。強化学習では迷路の形をそのまま扱う。各マスにおいて、進める方向を 1 と 0 で表した配列を準備する必要がある。この配列を元に方策を作り、方策を元に行動を選択する。そして、移動先の報酬などから価値を更新し、価値高くなるように繰り返し方策を改善する。価値は Q-table と呼ばれる、各状態で各行動を取る価値を示す表の形式で扱われる。このように、強化学習、粘菌アルゴリズム、強化学習の順に処理に必要な情報量が多いことが分かる。これが強化学習のアルゴリズムの時間がかかる原因である。

5.2　新たなアルゴリズム開発の課題

前述したように、本研究の目的は粘菌アルゴリズムと強化学習の生物らしさという共通点に着目し、新たなアルゴリズムを開発することである。本研究を通して実際に数理モデルを理解し実装することで、以下の課題を発見することができた。

一つは、粘菌アルゴリズムはグラフを用いるのに対し、強化学習が迷路の形を元に計算している。つまり、強化学習の実装では、前述した各マスにおいて、進める方向を 1 と 0 で表した配列が必要である。両者を組み合わせるためには、まず扱うデータの形を統一する必要があると考える。そこで、強化学習をグラフに適応させる手法として、深層学習をグラフ理論に応用したアルゴリズムとして GNN[xvi]がある。そのなかでも強化学習の要素をさらに取り入れた GCPN (Graph Convolutional Policy Network)などが有力であると考えられる。

二つは、粘菌アルゴリズムで解を求める際の不安定性である。これは計算時のオーバーフローが原因だと考えられる。より複雑な迷路ではより多くの計算量及び更新回数を必要とする。また、実験 2 では粘菌が通らない経路の流量が大きくなる場合もあったが、それらは消滅し最短経路のみ残るほうが望ましい。そのため、対策として更新式にオイラー法ではなく、指数オイラー法やその他の常微分方程式の解法を用いる。また、単純な指数関数$f(|Q|) = Q^{\mu}$をより複雑な増加関数$f(|Q|) = \frac{Q^{\mu}}{1+Q^{\mu}}$に変更する、更新式のパラメータ減衰率$r$や指数$\mu$を

調節するなどのアプローチを行っていくつもりである。

三つは、本実験では小規模な迷路に適応したのみであり、実際の道路ではより大規模で複雑である。配達サービスへの運用を考えたとき、巡回セールスマン問題(Traveling Salesman Problem：TSP)でシュミレーションすることも考えられる。巡回セールスマン問題とは、各都市と都市間の距離が与えられ、各都市を一度だけ巡るとき最小の総経路長になるものを求める最適化問題である。この問題において、全ての経路を計算することで最適解を得る手法では計算量が非常に大きくなってしまい、都市数が 20 以上になると現実的に解を求めることが不可能になる。これに対し、粘菌アルゴリズムは比較的に短い時間で、実用可能な精度の解を求めることができる。巡回セールスマン問題には強化学習を応用した研究も見られる。そのため、巡回セールスマン問題に対する両アルゴリズムの解法を比較することで何か新しい発見があると考える。また、現実では距離だけでなく時間や渋滞の変動に対する応答性なども考慮する必要がある。これらの課題を試行錯誤しながら解決することに、アルゴリズム開発への糸口があると考える。

5.3　生物らしさと数理モデル

本研究で粘菌アルゴリズムと強化学習を取り扱ったきっかけは、両者に生物らしさという共通点を見つけたからである。粘菌アルゴリズムは餌を求めて最短経路をつなぎ、強化学習は報酬を最大化するように学習する。餌や報酬とはいかにも生物らしいキーワードである。また、粘菌アルゴリズムは単細胞ながらも高い情報処理能力を持つ粘菌の仕組みを取り入れており、強化学習は大脳基底核のドーパミンによる報酬予測の仕組みを取り入れている。そのため、これらのアルゴリズムには知能らしきものが存在することを期待していた。

本研究を通して実際に数理モデルを理解し実装することで、前述した生物らしさは知能ではなく、数式のみによって成り立っていたことが分かった。粘菌アルゴリズムにおける餌は単にスタートとゴールを決めるものであり、強化学習における報酬は更新式のパラメータの一つでしかない。もちろんアルゴリズムの挙動は実物と似ているが、その仕組みを考察しようとすればいずれも後付けのようなものになってしまう。例えば、粘菌が最短経路を求める理由は、できるだけ短く、太い管で体をつなぎ、効率良く栄養や化学信号を伝達したいという生存本能が働いていると言える。粘菌にはそのようなことを考えているわけではなく、私たちが推測したものである。

この事実には失望したが、同様に生物の情報処理で観察できる現象を数理的モデルで単純化した例[xvii]が多くあることに気づいた。遺伝アルゴリズム(Genetic Algorithm：GA)、進化的戦略(Evolution Strategies：ES)、その中の群知能(Swarm Intelligence)という分類では、蟻コロニー最適化(Ant Colony Optimization：ACO)や人工蜂コロニーアルゴリズム

(Artificial Bee Colony Algorithm：ABC)などがある。また、深層学習及び人工知能研究においても、先述した強化学習だけでなく、ニューロンの仕組みを取り入れたニューラルネットや、視覚細胞の仕組みを取り入れた CNN(畳み込みニューラルネットワーク)などがある。

こうすることのメリットは、数学を使って解析できる汎用性を持ち、工学上の製品としても応用できる可能性を持つことを知った。そのため、今後はアルゴリズム開発を粘菌アルゴリズムと強化学習の数理モデルの共通点から模索していく。元の粘菌アルゴリズムの更新式では、餌を食べることによって流量が増えることは考慮されてなかったが、それに強化学習の報酬の考えを組みあわせることを検討している。

6. 結言

本研究では、粘菌アルゴリズムと強化学習の生物らしさという共通点に着目した新たなアルゴリズムを開発することを目的とした。その第一歩として、数理モデルの理解及び実装を通して、共通点や相違点を考えた。そのため、ダイクストラ法、粘菌アルゴリズム、強化学習の Q 学習と SARSA を実装し、簡単な迷路を解く実験を行った。入出力形式、データ形式や移動ログの分析を通し、処理時間や解の安定性の違いを考察した。各アルゴリズムの計算の特徴を理解することができた。

また、アルゴリズムの開発への実現に向けて、粘菌アルゴリズムと強化学習の扱うデータ形式の違い、粘菌アルゴリズムの不安定性、大規模実験の必要性などの課題を発見できた。生物らしさに関して、そこに知能はなく、数式のみによって実現されていたことが分かった。そのため、数理的モデルの共通点を模索する方向へ切り替える。今後の研究としては、強化学習をグラフに取り入れるため GNN の導入を検討する。粘菌アルゴリズムにおいて、更新式や増加関数、パラメータの変更を検討する。また、実用利用可能なより大規模で複雑な問題に対する応用を段階的に行う。まず、深層強化学習などを利用しより大規模の迷路で比較実験を行う。次に、時間や距離、渋滞などの要素を考慮した巡回セールスマン問題に適応する。より実用に近づけるため、マップを利用し実際の道路データで検証する。粘菌アルゴリズムに関して、常に同じような結果が得られるようパラメータを調節する、計算方法を見直すなどの工夫を行う。上記の課題をクリアすることで、配達サービスへ応用可能な新たなアルゴリズムの開発が実現することを期待する。

7. 付録 A：粘菌アルゴリズム詳説

粘菌アルゴリズムの重要な 2 つの方程式の導出を解説する。数式は次の論文[xviii]を参考にした。

１．原形質流動の方程式(管のゾル流動の方程式)

辺M_{ij}を長さL_{ij}、半径r_{ij}、入口と出口の圧力がそれぞれp_i, p_jの円筒形の管とすると、管の中を流れる流量Q_{ij}はハーゲン・ポアズイユ流れ(Hagen–Poiseuille flow)に従って下記の式(1)で表すことができる。κは液体粘性係数を表す。

$$Q_{ij} = \frac{\pi a_{ij}^4}{8\kappa} \frac{p_i - p_j}{L_{ij}} \quad (1)$$

また、前述した伝導率(Conductivity)D_{ij}は下記の式(2)で表すことができる。

$$D_{ij} = \frac{\pi a_{ij}^4}{8\kappa} \quad (2)$$

従って、式(2)を利用して式(1)を下記の式(3)で書き換えることができる。

$$Q_{ij} = \frac{D_{ij}}{L_{ij}} \left(p_i - p_j\right) \quad (3)$$

I_0を流量と置き、粘菌の管を水道管に見立てると、キルヒホッフの第一法則に従って下記の式(4)が成立する。

$$\sum_{\mathrm{i}} Q_{ij} = \begin{cases} -\mathrm{I}_0(\mathrm{j}=1)\ \text{スタート} \\ \mathrm{I}_0(\mathrm{j}=2)\text{ゴール} \\ 0(\mathrm{j}\neq 1,2)\text{その他} \end{cases} \quad (4)$$

式(3)を利用して、式(4)を下記の式(5)に書き換えることができる。これが原形質流動を表す公式である。

$$\sum_{\mathrm{i}} \frac{D_{ij}}{L_{ij}} \left(p_i - p_j\right) = \begin{cases} -\mathrm{I}_0(\mathrm{j}=1)\ \text{スタート} \\ \mathrm{I}_0(\mathrm{j}=2)\text{ゴール} \\ 0(\mathrm{j}\neq 1,2)\text{その他} \end{cases} \quad (5)$$

２．管の成長方程式（管の太さの適応方程式）

前述したように、現実の粘菌において流量の多い管は太くなり、流量の少ない管は減衰し、消滅する。これを適応過程(Adaptation Process)と呼ぶ。この管の太さの適応(Adaptation)を

表すため、時間変化する伝導率D_{ij}がある。一方、適応過程において長さL_{ij}は一定に保たれている。この特性を下記の式(6)で表すことができる。この関数fは$f(0)=0$を満たす単調増加関数である。rは管の減衰率(decay rate)を示す。流量$|Q_{ij}|$が大きくなるほど、管の伝導率(通りやすさ)D_{ij}は大きくなる。$|Q_{ij}|$が 0 に近づくほど、D_{ij}は 0 に近づく。

$$\frac{d}{dt}D_{ij} = f(|Q_{ij}|) - rD_{ij} \quad (6)$$

数学的導出は割愛するが、式(6)を適当な無次元化を行うことで下記の式(7)を得ることができる。この方程式は手老氏の論文では適応方程式(Adaptation Equation)と呼ぶ。

$$\frac{d}{dt}D_{ij} = f(|Q_{ij}|) - D_{ij} \quad (7)$$

また、関数fは$f(|Q_{ij}|) = |Q_{ij}|^{\mu}$のような指数関数である。関数のパラメータ$\mu$は異なる値によって 3 種類の違う挙動を示す。$\mu = 1$のとき、迷路の最短経路を得ることができる。

7. 付録 B：強化学習詳説

この付録では強化学習が成り立つ理由や更新式の導出方法を解説する。数式及び一部の表現は次の書籍[xix] [xx]を参考にした。

１．マルコフ性

前述した相互作用の説明において、なぜ現在の状態によって行動が決まり、行動によって将来の状態と報酬が決まるのか説明がされていなかった。これは強化学習の環境がマルコフ性(Markov property)という性質を持つからである。マルコフ性に関して本論文では厳密な数学的定義ではなく、資料を参考し定義した便宜上のものを用いる。

> 定義：マルコフ性
>
> マルコフ性とは時刻 t＋1 における環境の応答、つまり報酬と状態は時刻 t における状態と行動のみに左右されるという性質である。
>
> 全てのs_{t+1}，r_{t+1}，s_t，a_tに対して、確率分布 $\Pr = \{s_{t+1}, r_{t+1} \mid s_t, a_t\}$のみを指定することで環境のダイナミクスを定義できる。
>
> （参考：前掲書『強化学習入門』p68-69）

このマルコフ性という性質により、式を反復計算することにより現在の状態のみから将来の期待される報酬を全て予測することができる。マルコフ性を持たない場合は後述する

強化学習の根本が成り立たない。また、マルコフ性においていくつかの注意点がある。一つは、状態を含める範囲である。ポーカーの例で説明すると、状態はプレイヤー1 名の手札であり、カードの山や他のプレイヤーの手札を状態に含めてはならない。二つは、強化学習で良い性能を出すために厳密なマルコフ性ではなく近似を満たせばよい点である。ポーカーにおいて、プレイヤーの表情は判断要素の一つだが、学習には手札の情報で十分である。(前掲書『強化学習入門』p69 より参考)

２．マルコフ決定過程(MDP)

マルコフ性が満たされた強化学習タスクのことをマルコフ決定過程(Markov Decision Process：MDP)と言う。また、状態空間と行動空間が有限の場合に、それは有限マルコフ決定過程((有限 MDP)と言う。ここで有限 MDP の最も重要な特徴を指定する 2 つの概念を紹介する。(前掲書『強化学習入門』p72 より参考)

状態遷移確率(Transition probability)：ある状態と行動において、次の状態へ遷移できる確率。環境によっては、行動しても必ず次の状態を得られない場合がある。FrozenLake という有名な環境の例では、氷の上を歩くと滑って方向を変えてしまう要素を取り入れ、それを状態遷移確率で設定している。状態遷移確率は一般的に$T(s_{t+1} \mid s_t, a_t)$ で表される。

期待収益（Expected return)： 前述したように、強化学習においてエージェントは 1 エピソードの報酬の総和が最大になるように学習する。この最終的な報酬を期待収益または収益と言い、文字R_tで表す。また、相互作用(2.5-図 9)やマルコフ性の定義で使われていた報酬rを即時報酬(Immediate reward)と言う。収益は一般的に即時報酬の関数として、収益関数または報酬関数(Reword function)として表わされる。最も基本的な収益の形は即時報酬の合計である。

$$R_t := r_{t+1} + r_{t+2} + r_{t+3} + \cdots + r_T \quad (1)$$

T は最終時間ステップである。各エピソードが最終時間ステップ T において、終端状態と言う特殊な状態で終了し、開始状態にまたリセットするようなタスクをエピソード的タスク(episodic tasks)と言う。一方、最終時間ステップ$T = \infty$になるようなタスクを連続タスク(continuing tasks)と言う。

上記の式(1)は連続タスクにおいて、報酬を+1 としたときR_tは無限の値をとる。これを解決するため、割引(discounting)と言う概念を導入する。これは同じ価値のものでも、遠く将来にあるものほど価値を低く感じるという、行動経済学の考えをもとにている。下記の式(2)に割引率γ（discount factor)というパラメータを導入し、割引収益R_tを求める。

$$R_t := r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^{T-t-1} r_T = \sum_{k=0}^{T-t-1} \gamma^k r_{t+k+1} \quad (2)$$

$0 \leq \gamma \leq 1$であるため、$\gamma = 0$のときは即時報酬のみに注目するような短期的な視点に立つようになる。$0 < \gamma < 1$のとき、将来になるほどγの指数が大きくなり、即時報酬が小さくなるため、期待収益は減衰する。このとき、無限に加算しても期待収益は有限の値になる。(前掲書『強化学習入門』p63 より参考)

3．価値

前節で期待収益に関して解説したが、有限 MDP において即時報酬は状態と行動に依存する。そのため、エージェントはある状態にいることがどのくらい良いのか、またある状態においてある行動を取ることがどのくらい良いのかを評価する必要がある。「どのくらい良いのか」を価値(value)と言い、前者は状態価値V_π、後者は行動価値Q_πと呼ぶ。価値は期待収益をもとに定義され、最も簡単な価値の式は下記の式(3)(4)のようになる。

$$V_\pi = R_t = \sum_{k=0}^{T-t-1} \gamma^k r_{t+k+1} \quad (3)$$

$$Q_\pi = R_t = \sum_{k=0}^{T-t-1} \gamma^k r_{t+k+1} \quad (4)$$

また、価値は一般的に価値関数の形で表わされる。方策πに対する状態価値関数(state-value function)を$V_\pi(s)$と表す。方策πに対する行動価値関数(value function)を$Q_\pi(s, a)$と表す。ほとんどの強化学習アルゴリズムにおいて、エージェントは 1 エピソードの価値が最大になるように学習する。(前掲書『強化学習入門』p74 より参考)

$$\begin{aligned} R_t &:= r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^{T-t-1} r_T \\ &= r_{t+1} + \gamma(r_{t+2} + \gamma r_{t+3} + \cdots + \gamma^{T-t-2} r_T) \\ &= r_{t+1} + \gamma G_{t+1} \quad (5) \end{aligned}$$

4．ベルマン方程式

しかし、式(3)(4)の価値を利用し強化学習を行う場合二つの問題点がある。

一つは、将来の即時報酬を求める必要があるため、エピソード終了まで価値を得られないこと。この問題は式を下記の式(5)のように再帰的に定義し、価値の推定値を利用することで解決する。

二つは、報酬を必ず得られない場合もある。前述した状態遷移確率を踏まえて考えると、環境に設定されている状態遷移確率によっては、ある状態においてある行動をとったとしても違う方向に移動する、またはそもそも移動できない場合がある。移動ができなければ、即時報酬は得られず、式が成立しなくなる。この問題は下記の式(6)(7)のように価値をR_tの期待値$E_\pi\{\}$で表すことで解決する。{A | B} は「A は B の条件のもとである」という意味を表す。(前掲書『強化学習入門』p75 より参考)

$$V_\pi(s_t) = E_\pi\{R_t|s_t\} = E_\pi\left\{\sum_{k=0}^{T-t-1}\gamma^k r_{t+k+1} \mid s_t\right\} \quad (6)$$

$$Q_\pi(s_t,a_t) = E_\pi\{R_t|s_t,a_t\} = E_\pi\left\{\sum_{k=0}^{T-t-1}\gamma^k r_{t+k+1} \mid s_t,a_t\right\} \quad (7)$$

式(5)と(6)、(5)と(7)を組み合わせると、期待収益を期待値で表し再帰的に定義した下記の式(8)(10)が求まる。これをV_πまたはQ_πに対するベルマン方程式(Bellman Equation)と呼ぶ。ベルマン方程式は強化学習アルゴリズムにおいて最も基本的な式の一つである。

また、期待値は方策$\pi(a|s)$と状態遷移確率$T(s_{t+1}|s_t,a)$を用いて、下記の式(9)(11)のように表すことができる。

$$V_\pi(s_t) = E_\pi[r_{t+1} + \gamma V_{t+1}|s_t] \quad (8)$$

$$= \sum_a \pi(a|s_t)\sum_{s_{t+1}} T(s_{t+1}|s_t,a)\left(r_{t+1} + \gamma V_\pi(s_{t+1})\right) \quad (9)$$

$$Q_\pi(s_t) = E[r_{t+1} + \gamma Q_{t+1}|s_t,a_t] \quad (10)$$

$$= \sum_{s_{t+1}} T(s_{t+1}|s_t,a_t)\ (r_{t+1} + \gamma \sum_{a_{t+1}} \pi(a_{t+1}|s_{t+1})Q_\pi(s_{t+1},a_{t+1})) \quad (11)$$

また、式(8)では行動は方策πによって確率的に選択されるが、価値最大化するような行動をだけを選択したい場合は下記の式(12)で表す。前者の手法は価値を学習するため Value ベースと呼び、後者の手法は価値と方策両方を学習するため Policy ベースと呼ぶ。

$$V(s_t) = \max_a\left\{\sum_{s_{t+1}} T(s_{t+1}|s_t,a)\left(r_{t+1} + \gamma V(s_{t+1})\right)\right\} \quad (12)$$

5. モデルベースとモデルフリー

環境のモデル(model)とは、厳密な定義ではないがエージェントと環境間の相互作用において、エージェントが行う行動に対する環境の反応(報酬と次の状態)を作り出すものである。また、参考のために下記に環境のモデルの定義を引用する。

> 定義：環境のモデル
> エージェントが自分の行動に対してどのように応答するかを予測出来る、あらゆる対象を意味するものとする。1 個の状態や 1 個の行動が与えられたとき、結果として生じる次の状態と報酬の予測がモデルによって作り出される。
>
> (前掲書『強化学習入門』p247 より引用)

モデルがあり、状態遷移確率・収益をベースとする手法をモデルベース(Model-based)と呼ぶ。つまり、状態遷移確率・収益が分かっている場合のみ使える。例として、後述する動的計画法などがある。一方、モデルを持たず、つまり状態遷移確率・期待収益がない手法をモデルフリー(Model-free)と呼ぶ。状態遷移確率・期待収益を推定して学習を行う。例として、後述する TD 学習やモンテカルロ法などがある。

6. 動的計画法(DP)

動的計画法(Dynamic Programming：DP)とは、対象となる問題を複数の部分問題に分割し、帰納的に解き、計算結果を再利用し同じ計算を避けるような最適化アルゴリズムの総称である。代表として、フィナボッチ数列、ナップサック問題、ハノイの塔などがある。

また、先述したベルマン方程式は動的計画法の最適性の必要条件を表す方程式である。詳しい説明は割愛するが、動的計画法を用いればベルマン方程式を逐次計算で効率よく解くことができる。

次に、動的計画法の各手法に関して説明する。

・価値反復法(Value Iteration)：

$$V_{i+1}(s_t) = \max_a \left\{ \sum_{s_{t+1}} T(s_{t+1}|s_t, a) \left(r_{t+1} + \gamma V_i(s_{t+1}) \right) \right\} \quad (13)$$

上記の式(13)は前述した式(12)と同じものであり、i 回目の価値V_iを利用して i+1 回目の価値V_{i+1}を求める。これを更新前後の差$|V_{i+1}(s) - V_i(s)|$が一定値より小さくなるまで繰り返す。エージェントは価値を最大化するような行動をだけを選択し、価値を学習する。これは前述した Value ベースの手法である。

・方策反復法(Policy Iteration)：

$$V_\pi(s) = \sum_a \pi(a_t|s_t) \sum_{s_{t+1}} T(s_{t+1}|s_t, a_t) \left(r_{t+1} + \gamma V_\pi(s_{t+1})\right) \quad (14)$$

上記の式(14)は前述した式(9)と同じものであり、価値と方策両方を交互に更新する。エージェントは方策に従って確率的に行動し、価値と方策両方を学習する。これは前述したPolicy ベースである。

7. モンテカルロ法(MC)

モデルフリーの手法には状態遷移確率と期待収益がないため、ベルマン方程式を繰り返し解くことで、その状態の遷移は近似的に状態遷移確率に従うことができる。そして、モデルベースと同じように価値を求めることができる。この繰り返しによって得られた経験(experience)、またはシュミレーション上の経験(Simulated Experience)から価値を学習する手法をモンテカルロ法(Monte Carlo Methods：MC)と呼ぶ。また、MC の特徴として方策を 1 エピソード終了後に改善するため、価値の更新に実際の割引収益を利用する。従って、下記のように更新式(15)を表す。(前掲書『強化学習入門』p119 より参考)

$$V(s_t) \leftarrow V(s_t) + \alpha((r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^{T-t-1} r_{T-t}) \quad (15)$$

8. TD 学習

TD 学習(Temporal Difference Learning)は動的計画法とモンテカルロ法を組合せた手法である。本論文紹介するものは厳密には TD(0)と呼ぶ手法である。下記の図 2 に TD 学習とモンテカルロ法の比較を示す。動的計画法(DP)と同様に、式を再帰的に定義することでエピソード終了まで待たず価値の推定値を利用する。また、モンテカルロ法と同様に環境のモデルを用いず、経験から学習する。

行動前は価値を$V(s_t)$と推定し、行動後実際の価値は$r_t + \gamma V(s_{t+1})$となり、その差異は$r_t + \gamma V(s_{t+1}) - V(s)$である。これは推定と実際との差異であり、時刻間(t と t+1) の差異でもあるから TD 誤差(Temporal Difference Error)とも呼ぶ。これが経験の正体である。また、TD 学習の特徴として、1 ステップ後(1 回の行動後)に方策を改善する。従って、学習率αを取り入れて下記のように更新式(16)を表す。(前掲書『強化学習入門』p142 より参考)

$$V(s_t) \leftarrow V(s_t) + \alpha(\ r_t + \gamma V(s_{t+1}) - V(s)) \quad (16)$$

8. 謝辞

本論文を作成するにあたり、早い段階から卒業論文指導教員の早稲田大学高等学院 秋山和広教諭より相談及び助言を頂いたことに対し心より感謝致します。本研究の実験部分において、アルゴリズムの実装の疑問点やエラーを親身になって解決法を考えて下さった、WBA 若手の会所属 東北大学 3 年生の美濃佑輝氏に深く御礼申し上げます。また、本論文は 2020 年度早稲田大学高等学院同窓会学術研究奨励金の研究成果の一部となります。

---

9. 参考文献

[i] 中垣俊之「粘菌　その驚くべき知性」　株式会社 PHP 研究所（2010）p1

[ii] Yamada, Hiroyasu & Toth, Agota & Nakagaki, Toshiyuki. (2000). Intelligence: Maze-Solving by an Amoeboid Organism. Nature. 407. 470-470. 10.1038/35035159.

[iii] 前掲書「粘菌　その驚くべき知性」p3

[iv] 中垣俊之, 小林亮　「原生生物粘菌による組合せ最適化法―物理現象として見た行動知―」人工知能学会誌 26 号 p482-493 (2011)

[v] 同論文

[vi] Atsushi Tero, Ryo Kobayashi, Toshiyuki Nakagaki,
A mathematical model for adaptive transport network in path finding by true slime mold,
Journal of Theoretical Biology, Volume 244, Issue 4, 2007, p553-564

[vii] 小林亮, 手老篤使, 中垣俊之　「真性粘菌変形体の運動と情報処理について」　盛岡応用数学小研究集会報告集 p26-35 (2006),

[viii] 京都大学理学部 3 回生　スイス連邦工科大学ローザンヌ校　小南裕介　「強化学習入門」p8

[ix] Richard S. Sutton and Andrew G. Barto. (1998) REINFORCEMENT LEARNING: An

Introduction.（リチャード S．サットン 三上貞芳・皆川雅章(訳)(2000)．「強化学習」 森北出版株式会社) p56

[x] 増井敏克 「Python ではじめるアルゴリズム入門 伝統的なアルゴリズムで学ぶ定石と計算量」 翔泳社（2020）

[xi] “Python で粘菌ネットワーク". Qiita. 2019/05/14.
（https://qiita.com/STInverSpinel/items/8ced06ea7881613a3e2c）

[xii] 横山優希，小藤俊幸 「生物モデルを用いた最短経路問題の解法」 南山大学大学院 理工学研究科 2015 年度 修士論文要旨集（2015）

[xiii] 前掲論文 A mathematical model for adaptive transport network in path finding by true slime mold, (2007) p9-11

[xiv] Numpy だけで書いたガウスの消去法で連立 1 次方程式を解いてみた WATLAB 2020.05.01
（https://watlab-blog.com/2020/05/01/gaussian-elimination/）

[xv] 株式会社電通国際情報サービス 小川雄太郎 「つくりながら学ぶ！ 深層強化学習 PyTorch による実践プログラミング」 マイナビ出版（2019）

[xvi] Ziwei Zhang, Peng Cui, & Wenwu Zhu. (2020). Deep Learning on Graphs: A Survey.

[xvii] Binitha, S., and S. Siva Sathya. "A survey of bio inspired optimization algorithms." International journal of soft computing and engineering 2.2 (2012): 137-151.

[xviii] 前掲論文 「真性粘菌変形体の運動と情報処理について」(2006)

[xix] 前掲書 REINFORCEMENT LEARNING: An Introduction.

[xx] 久保隆宏「機械学習スタートアップシリーズ Python で学ぶ強化学習 [改訂第二版] 入門から実践まで」 講談社(2019)