Predict State of the Incomplete Information Game with Game Agent

ゲームエージェントを使った不完全情報ゲームの盤面予測

遠田 英嗣*　小高 知宏*　黒岩 丈介**　白井 治彦***

Predict State of the Incomplete Information Game with Game Agent

Hidetsugu TODA*,　Tomohiro ODAKA*,　Jousuke KUROIWA**and Haruhiko SHIRAI***

(Received February 24, 2017)

In this paper, we can thaw an incomplete information game with artificial intelligence. The incomplete information game game covers a part of the information to a player and is carried out. In this study, It is intended to let the game agent predict the invisible part of the incomplete information game game.

However, we do not know whether you can expect artificial intelligence. Therefore I prepare for a simple incompleteness information game to know whether you can expect artificial intelligence and can thaw it. The simple incomplete information game game searches towards a goal in a small field. The agent searches for the field of this game many times and learns the position of the goal. The agent imitates a successful example once. The agent begins a search with the start. When an agent arrives at the goal by few steps, We think the agent able to learn an incomplete information game. If an agent was able to predict the position of the goal, We may apply the artificial intelligence for the reality world.

Key Words **:** Artificial Intelligence,Game Agent,Machine Learning,Incomplete Information Games

1. はじめに

近年コンピュータを用いて課題を自律的に解かせる試みは多く,20世紀中頃からコンピュータチェスをさせるといった研究が盛んである.

コンピュータが人間をチェスで破り, 現在ではマッチ戦において世界チャンピオンに対して無敗で勝利を収めるなどその成果は目覚ましい [1].

2016年3月にGoogle DeepMindが作成したAlphaGoという囲碁プログラムが人間の中で最も囲碁が強い人物の李世ドルに対して全5戦で4勝1敗という成績を収めた. チェスプログラムがチェスのグランドマスターに勝利したのは1997年であるのに対し, 囲碁プログラムが2016年と遅れている.

囲碁はチェスのようなゲームと比較して, 局面の数が非常に膨大である. よって全探索のようなコンピュータの能力をフルに使った探索の仕方ができず長い間人間に勝つことが困難であると考えられていた. (少なくとも10年先だと言われていた.) しかしながらAlphaGoが勝利したことにより人工知能の発展の速度が尋常でないことがうかがえる [2].

しかし, 近年研究されている自律的なゲームエージェントの多くは完全情報ゲームに対してのものが多く, 相手の手札, 状況などの情報が分からない不完全情報ゲームに対する研究は少ない.

不完全情報ゲームでは相手の状態や行動が不明瞭であり, 予測しなければならない部分が少なからず存在する. 現実には当たり前のように予測しなければならない事がありふれているため, 完全情報ゲームより不完全情報ゲームのほうが現実に近いと言える. そのため現実世界への応用が期待されている [3].

* 大学院工学研究科 原子力・エネルギー安全工学専攻
** 大学院工学研究科 知能システム工学専攻
*** 工学部技術部
* Nuclear Power and Energy Safety Engineering Course, Graduate School of Engineering
** Human and Artificial Intelligence Systems Course, Graduate School of Engineering
*** Technical Division

本研究では, 不確定要素が入る不完全情報ゲームを独自に設定し, 不完全情報ゲームに対して機械学習によって人工知能が不明瞭である部分を予測することができるかどうかの確認を行う.

本論文では 2 章では題材とする不完全情報ゲームの設計を行い,3 章では題材を解くエージェントの設計及びを行う,4 章では実験を行い,5 章で実験により出た結果を示す.6 章で結果の考察を行い,7 章でまとめを述べる.

2. ゲームの種類と題材設定

世界のゲームは完全情報ゲームと不完全情報ゲームの 2 つに分類される. 完全情報ゲームとは全ての情報が開示されているゲームであり, 近年研究されている将棋やチェスはその中でも二人零和有限確定完全情報ゲームという部類に分類される.

不完全情報ゲームは情報の一部しか開示されていないゲームであり, トランプゲームや麻雀などがそれにあたる. 世界にあるゲームの中の大半が不完全情報ゲームとなる.

本章では, 本研究で用いる不完全情報ゲームの設計を行う.

2.1 題材となる不完全情報ゲームの設計

本研究で用いる不完全情報ゲームとして 9 × 9 の限定されたフィールドを作る. そして, そのフィールドの中にランダムに開始位置と終了位置を配置する.

ゲームを行うプレイヤーからは終了位置は分からないようになっており, 開始位置から出たキャラクターをフィールドの中の終了位置を発見するために探索させる. 終了位置を探索した時点でゲームクリア, 題材を解いたとする.

このゲームでの不確定要素としては, プレイヤーはキャラクターの位置は分かるが終了位置が分からないという点と, 開始位置と終了位置がランダムであるから開始位置と終了位置までの距離が一定でないという事にある.

このゲームにおいて評価の基準となるのは移動した回数であり, 少ない手順が高評価になる.

2.1.1 フィールド設定

題材とするフィールドの横方向に X 軸を, 縦方向に Y 軸を設定する.X 軸は左を基準とし,Y 軸は上を基準とする. 図 1 にフィールドのイメージを示す. フィールド内には移動を阻害する障害物のようなものは存在しない. 本論文内ではフィールド内での位置 [X 座標,Y 座標] で表すこととする.

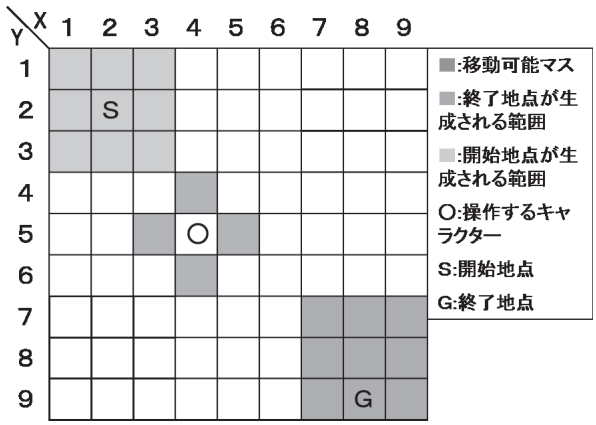

図 1 フィールドイメージ図

2.1.2 開始位置と終了位置

開始位置を [1〜3,1〜3] の中でランダムに生成し, 終了位置を [6〜9,6〜9] の範囲でランダムに生成する. 図 1 内の左上の 9 マスが開始位置の生成範囲となり, 右下の 9 マスが終了位置の生成範囲となる.

開始位置が [1,1] で終了位置が [9,9] だった場合最も開始位置と終了位置が遠くなり, 最短距離は 16 マスとなる. また, 開始位置が [3,3] で終了位置が [6,6] だった場合が最も近くなり, 最短距離は 6 マスとなる. そのため開始位置と終了位置の間の最短距離は 6〜16 マスの間で変化する.

2.1.3 キャラクターの設定

キャラクターは 1 回の手順で上下左右のいずれかに 1 マスだけ動くことができ, 斜め方向には動けない. キャラクターの移動可能範囲は開始位置などと同様に図 1 に示す. 図 1 内において, ○がキャラクターを表し縦横 1 マスがキャラクターが 1 回の手順で移動できる範囲となる.

3. エージェントの設計

エージェントは人間の代わりに題材を解決する存在を指し, 本研究では題材内のキャラクターを操作する存在である. エージェントは題材に対して機械学習を行い, 題材への理解を深めていく.

エージェントのキャラクターを動かすための行動規範や学習手法を本章にて説明する.

3.1 行動規範

基本的に本研究でのエージェントはランダムにキャラクターを動かす. エージェントには題材に取り組む際に各移動方向に対してステータスとして移動値という値を持たせる. このステータスの初期値は各移動方向に対して全て一定で 25 である.

これは初期値ではエージェントは全ての方向への移動確率が同じである事を意味する.

3.2 学習手法

3.1 節で説明したようにエージェントは移動値を持つ. この移動値を変化させることでエージェントは学習を進めていく. 移動値は終了位置にキャラクターが到達し, 題材をするまで値は変化せず, ゲームクリアした時点で移動回数に基づいて移動値を変動させる.

これによりエージェントは前回終了位置に到達した, 成功した事例を倣うように学習を進めていく. これを何度も繰り返すことによってエージェントは機械学習を進めていく.

学習方法のイメージ図を図 2 に示す. 同じ値を持ったエージェントを複数体用意する. これを第一世代とし, これらのエージェントはそれぞれ, 題材に取り組む. 全個体が題材を解いた後, 全個体の中で最も評価値の高いエージェントを選出する.

最も評価値の高いエージェントのステータスを持ったエージェントを, 第一世代と同じ数だけ用意する. これを第二世代とする. これらに同様に題材を解かせる.

この動作を一定回数行った後, 最終世代でのもっとも評価値の高いエージェントのステータスを学習結果とする.

3.3 学習手法の式

t 回目のゲームの時の上方向への移動値を U_t とする. 同様に t 回目のゲームの時の下方向への移動値を D_t,t 回目のゲームの時の右方向への移動値を R_t,t 回目のゲームの時の左方向への移動値を L_t とする. 移動値の初期値は U_0, D_0, R_0, L_0 で表され,3.1 節で説明したようにそれぞれ 25 である. 各方向への移動値の更新式は以下で表される.

$$U_{t+1} = U_t + CU_t$$
$$D_{t+1} = D_t + CD_t$$
$$R_{t+1} = R_t + CR_t$$
$$L_{t+1} = L_t + CL_t$$

CU_t, CD_t, CR_t, CL_t は t 回目のゲームでの各方向への移動回数を表しており,t 回目での移動値に t 回目のゲームでの移動回数を足し合わせることで t+1 回目の移動値を算出する.

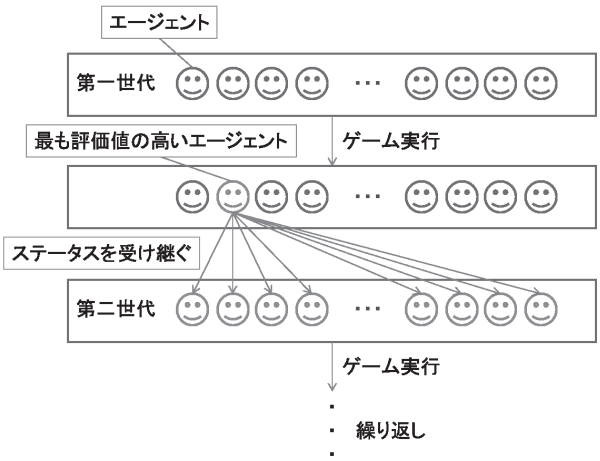

図 2 エージェントの学習方法イメージ

それぞれの移動確率を PU_t, PD_t, PR_t, PL_t で表す. 移動確率の更新式を以下に示す.

$$PU_{t+1} = \frac{U_t}{U_t + D_t} \quad , \quad PD_{t+1} = \frac{D_t}{U_t + D_t}$$

$$PR_{t+1} = \frac{R_t}{R_t + L_t} \quad , \quad PL_{t+1} = \frac{L_t}{R_t + L_t}$$

2 章で説明したように開始位置から見た時の終了位置の位置は右下方向にあるため PR と PD の確率が大きくなる事が予想される.

4. 実験

実際にエージェントに題材を解かせる実験を行った.

今回世代数×個体数が 1,000,000 回となるように学習を行う. これを学習前と学習後のデータで比較し, 学習が行えているかどうかの確認を行う. また, フィールドの 1 マス毎の移動頻度を見ることによっても学習を行っているかの確認を行う.

4.1 学習前

学習を行う前にエージェントに問題を解かせる.

本研究ではエージェントはランダムな方向にキャラクターを操作するため, 偶然終了位置をごく少ない移動回数で探索してしまう可能性がある. そこで 1 回のみの試行ではなく複数回試行させる. 今回は 100 回題材を解かせ, 各試行の終了位置の探索にかかった移動回数を見る.

以下にその時のそれぞれの試行回数時の図を示す.[図 3]

学習前のデータであるので移動確率は全て 25 %となっている. 完全にランダムに動くため, 当然全体的な移動回数は多くなる.

今回 100 回の試行中, 偶然少ない移動回数で終了位置に入ったものでも 25 回の移動が必要だった. 偶然に頼ったものであるので今回最大の移動回数の場合,1462 回もの移動を行った.

最大移動回数と最小移動回数とを比べると最小の 25 回はかなり少ないもののように見えるが, 今回のフィールドは 9 × 9 の全 81 マスであるので全体のマスの約 3 割を移動したことになる. 多くはないが少ないとは言えないほどの移動回数と言える.

4.1.1 学習前の移動頻度

学習前のデータを用いて, フィールド内をエージェントがキャラクターをどのように動かしているか, それぞれのマスへの移動頻度を見る. 図 4 に 100 回の試行のうちから無作為に 1 つ選びだし, その時の移動頻度を示す.

図 4 では, 色が濃いところほどエージェントがマスを訪れているという事になる. 図を見てわかるように, 開始位置の近くを多く探索し, 終了位置付近はあまり探索していないことが分かる.

これは開始位置から出たエージェントが徐々に行動範囲を広げて行っているように見える. そのためか終了位置は開始位置から近い位置の [7,7] に生成された時が比較的少ない移動回数で探索を終了していることが考えられる.

4.2 個体数と世代数の比

個体数と世代数の積が 1,000,000 になるような学習を行わせるが, 個体数と世代数の積が 1,000,000 となるような比は数多く存在する. そこで今回の学習に適した個体数と世代数の比を確認する.

比較を行うのは個体数 10 で世代数 100,000 の学習と, 個体数 100 で世代数 10,000 の学習, 個体数 1,000 で世代数 1,000 の学習, 個体数 10,000 で世代数 100 の学習, 個体数 100,000 で世代数 10 の学習の計 5 つの学習を行う.

得られた移動確率を学習データとする. この学習データを用いて 100 回題材を解かせ, 各試行の終了地点にかかった移動回数のデータをテストデータとする.

学習データの各方向への移動確率とテストデータの最大移動回数, 最小移動回数を比較することで最も適した個体数と世代数の比を探す.

- 個体数 10 ×世代数 100,000

学習後の各方向への移動確率は上 22.50 %下 27.53 %右 27.50 %左 22.47 %となった.

この学習データで 100 回題材を解かせた結果, 最大移動回数は 436 回で, 最小移動回数は 12 回であった. 学習前のデータと比べると多少右下の方向へ行く傾向が強くなっている.

- 個体数 100 ×世代数 10,000

学習後の各方向への移動確率は上 20.31 %, 下 29.67 %, 右 29.68 %, 左 20.35 %となった.

この学習データで 100 回題材を解かせた結果, 最大移動回数は 444 回で, 最小移動回数は 18 回であった. 個体数 10 の時と比較すると, 右下へ行く傾向が強くなっているが最大移動回数と最小移動回数は変化がなかった.

- 個体数 1,000 ×世代数 1,000

学習後の各方向への移動確率は上 18.18 %, 下 31.92 %, 右 31.78 %, 左 18.10 %となった.

この学習データで 100 回題材を解かせた結果, 最大移動回数は 201 回で, 最小移動回数は 9 回であった. 個体数 10 と 100 の時と比べると, 最大移動回数が非常に大きく減っている. 最小移動回数も同様に減少しており学習が進んでいることが分かる.

- 個体数 10,000 ×世代数 100

学習後の各方向への移動確率は上 16.08 %, 下 33.76 %, 右 34.32 %, 下 15.84 %となった.

この学習データで 100 回題材を解かせた結果, 最大移動回数は 155 回で, 最小移動回数は 11 回であった. 個体数 1,000 と比べると, 移動確率の変化はあるものの最大移動回数と最小移動回数の回数はあまり変化がなかった.

- 個体数 100,000 ×世代数 10

学習後の各方向への移動確率は上 14.22 %, 下 34.60 %, 右 36.01 %, 左 15.17 %となった.

この学習データで 100 回題材を解かせた結果, 最大移動回数は 218 回で, 最小移動回数は 9 回であった. 個体数 10,000 と同様に 1,000 からの大きな変化は見られない.

学習データの各方向への移動確率と 100 回題材を解かせたテストデータの最大移動回数と最小移動回数の比較では比較要素が不足していたため, テストデータの偏差, 平均を追加して比較する.

表 1 に各個体数でのテストデータの偏差, 平均, 最大移動回数, 最小移動回数を記す.

表 1 個体数毎の学習結果比較

個体数	100k	10k	1k	100	10	学習前
偏差	38.19	27.50	35.25	62.75	81.85	263.9
平均	50.24	50.79	61.65	87.43	113.03	282.4
最大	218	155	201	444	436	1462
最小	9	11	9	18	12	25

偏差を見ると学習前から比較して個体数 10,000 までは減少している. しかし個体数 100,000 からは増加し, 平均も同様に個体数 10,000 までは減少している. 個体数 100,000 は個体数 10,000 とほぼ横並びとなった. しかし個体数 10,000 の場合, 最小移動回数が少し増加している.

今回のテストデータをとるために学習データで題材を 100 回試行したが, これでは個体数 1,000 と個体数 10,000, 個体数 100,000 の明確な差が見えなかった.

乱数による誤差である可能性があるため試行回数を多くすることで誤差の少ないテストデータを作成する. その後個体数 1,000, 個体数 10,000, 個体数 100,000 の比較を改めて行う.

その結果以下の表のようになった.[表 2]

表 2 個体数 1,000 以降の学習結果比較

個体数	100k	10k	1k
偏差	36.74	33.16	36.06
平均	47.73	50.39	61.10
最大	585	408	449
最小	8	9	8

偏差に大きな差は見られず, 平均, 最大, 最小においても差は見られなかった.

この結果から個体数 1,000 で学習の進行が止まっていることが考えられる. このことから個体数 1,000 で学習を行うのが最適であると考えられる.

5. 結果

実験の結果個体数 1,000 で学習を行うのが最適であった. また, 先に予想した通り全ての学習において右下方向への移動確率が大きくなり右下への重みづけがなされていることが分かる.

図 5 に学習後のデータを使って 100 回試行したときの図を示す.

学習前と学習後のデータとを比較すると全体的に見て大幅に移動回数が減っていることが図 3 と図 5 から見て取れる.

100 回試行したなかでの最大移動回数は 201 回で学習前から比べると最大移動回数は大幅に減少している.

しかし, 今回のフィールドの全マス数は 81 マスであるので [1,1] から [9,9] まで順に全探索したとしても 81 回しか移動しないにもかかわらず 201 回もの移動を行っているので, 非常に移動回数が多い. このことからこの時は図 6 で示す範囲で終了位置を通り過ぎて停滞している事が考えられる.

5.1 学習後の移動頻度

学習後のデータを用いて, フィールド内をエージェントがどのように移動しているか, それぞれのマスへの移動頻度を見る. 学習前と同様に図 7 に 100 回の試行のうちから無作為に 1 つ選びだし, その時の移動頻度を示す.

色が濃いところほどエージェントがマスを訪れているという事になる. 全体的に見て学習前と異なり学習前は開始位置近くを探索していたが, 学習後は終了位置付近を主に探索していることが分かる.

このことから学習が行えていると考えられる. また,[7,7] の位置の G に○がついているところが終了位置であるが,4.2 で予想した通り終了位置を通りすぎて [8,8] から [9,9] 付近を往復し, 停滞していることが分かる.

今回の移動頻度を見ると学習の結果から開始位置から見て終了位置が右下にあると予想し右下方向へ向かう傾向が強い. そのため右端の [9,9] に終了位置が生成された時が比較的少ない移動回数で探索を終了していることが考えられる.

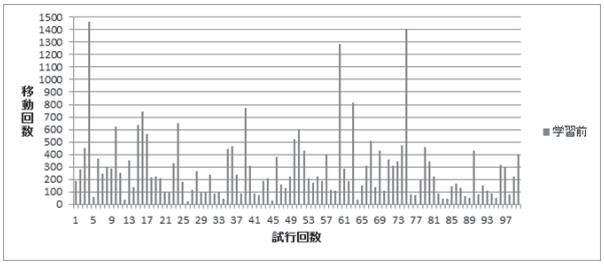

図 3 学習前の移動回数

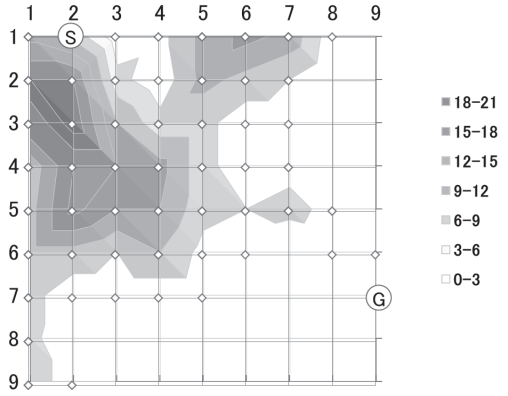

図 4 学習前の各マスへの移動頻度

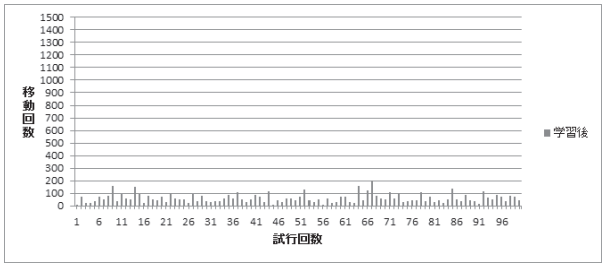

図 5 学習後の移動回数

Y＼X	1	2	3	4	5	6	7	8	9
1									
2		S							
3									
4									
5									
6									
7							G		
8									
9									

図 6 停滞の原因

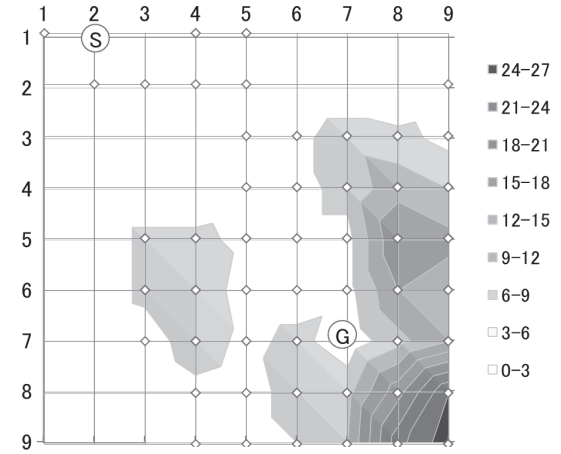

図 7 学習後の各マスへの移動頻度

6. 考察

人工知能を用いて, プレイヤーに対して全ての情報が与えられていない不完全情報ゲームの問題の学習を行わせるに際して総学習回数 1,000,000 の中で個体数と世代数を変化させた.

その結果今回の学習に最も適しているのは個体数 1,000 世代数 1,000 であった.

個体数が 1,000 より少なければ移動回数が多いものに対してのマイナス報酬が適した値にならず, 学習が遅れてしまうことが考えられる.1,000 より多い場合はマイナス報酬が 1,000 付近で一定となり学習の進みが一定になると考えられる.

学習の進みが一定であるため個体数 1,000 より多くなったテストデータにおいて個体数 1,000 と大差ない結果になったと考えられる.

個体数 1,000 で学習を行った結果, 学習前に比べて移動回数は最大が 1462 から 201 へ減少し, 最小が 25 から 9 に減少しており, 移動回数の観点から見ると学習は成功していると考えられる.

また, 各マスへの移動頻度を学習前と学習後で比較してみると学習前は開始位置付近を重点的に探索していることが分かる. 学習後は終了位置の周りを重点的に探索していることが分かる. このことから移動頻度の観点から見ても学習は成功していると考えられる.

移動確率についても同様に, 学習前は全て 25 %であったが, 学習後には上 18.18 %, 下 31.92 %, 右 31.78 %, 左 18.10 %と予想した通り右方向と下方向に対しての確率が上がっており重みづけされていることが分かる. このことからも学習は成功していると考えられる.

また, この学習前と学習後の移動確率の変化から今回の不完全情報ゲームを解くエージェントは開始位置から見て右下方向に終了位置があると予想しているといえる.

しかしながら終了位置を過ぎてしまった場合にはまだ右下の方に終了位置があると考えそのまま右下へ行こうとするため, 右下の端を右往左往し停滞してしまう傾向がある.

これは今までの予想を破棄し別の予想をするというルールを獲得できていないからである. このルールが獲得できればその場の状況にあった判断が可能であると考えられる.

そのため今後の課題としては行き詰った場合の判断と, 選択肢を複数用意し今までのルールを破棄, 複数あるルールの中から 1 つを選択し行動するルールの変更という 2 つの事柄の学習が必要となる.

7. まとめ

近年行われている人工知能のチェスや将棋, 囲碁のような完全情報ゲームへの適用が盛んである.

今回人工知能を完全情報ゲームでなく, 不完全情報ゲームに対して適応させることが可能であるかどうかを不確定要素の存在する独自の不完全情報ゲームを作成し, その確認を行った.

今回の学習においてもっとも上手くいった総学習回数 1,000,000 の中の個体数と世代数の組み合わせは個体数 1,000 で世代数 1,000 である学習方法が適しているという結果になった.

個体数 1,000 未満の時個体数が多くなればなるほど総学習回数が一定の場合エージェントは学習をよりよくおこなっていた. しかし, 個体数 1,000 以上から移動確率の変化は見られたものの, テスト試行を行った時の平均の移動回数や偏差の値に大きな変化が見られなかったため, 個体数 1,000 が適切だと考えた.

また, 個体数を大きくすればするほど学習にかかる時間は飛躍的に増加し, 時間効率の点から言っても個体数 1,000 が適正であると考えられる.

学習の結果このエージェントがフィールド内を探索するための各方向へ移動する確率は上方向 18.18 %, 下方向 31.92 %, 右方向 31.78 %, 左方向 18.10 %となった.

学習前の予想では右下方向への移動確率が高くなることが予想されており, 予想通りの結果となった.

また, 移動確率が変化したことでエージェントの移動傾向が変化し終了位置の付近を探索するようになったことで全体的な移動回数の減少が見られた. しかしながら学習データでの試行を何度もしていると途中で全 81 マスであるのに対しこれを超える非常に移動回数が多い試行が見られた.

これは終了位置を過ぎて端まで到達してしまった場合でこの時エージェントは終了位置を過ぎたことに気付かずまだ右下方向に終了位置があると予想してしまっているためである.

これを改善するためにはルールの追加によって予想の変更を学習させる必要がある.

対策としては停滞に入った場合に, 今までの予想を破棄し別の予想を立ててその予想に従って動くようになることや, 自分の周囲のマスの確認することで終了位置を通りすぎないようにすることなどをルールとして獲得できるような学習を導入することが考えられる.

参考文献

[1] 松原 仁:素朴な疑問:なぜチェス名人はコンピュータチェスに負けなければならなかったのか?, 情報処理,38 巻-8 号,pp.705-706(1997)

[2] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel & Demis Hassabis:"Mastering the game of Go with deep neural networks and tree search".Nature, Vol.529-7587,pp.484-489(2016)

[3] 大佐賀 猛, 野中 秀俊, 吉川 毅, 杉本 雅則:不完全情報ゲームにおける適応的モンテカルロ木探索手法の提案, 情報処理学会論文誌数理モデル化と応用 (TOM),Vol.8-1,pp.38-44(2015)