フリーセルの着手を学習するディープラーニングのためのデータセットの作成

Construction of Data Set for Deep Learning of Moves in FreeCell

玉井 慎悟† 今村 奨† 上野 一樹† 神保 秀司†

Shingo Tamai Sho Imamura Kazuki Ueno Shuji Jimbo

† 岡山大学大学院自然科学研究科

1 はじめに

近年十分に学習を積んだニューラルネットと従来型の探索部分の組み合わせにより強力なゲーム AI が実装できることが実証された．著者らは，一人用トランプゲームであるフリーセルの質の良い着手を求めるニューラルネットの開発を試みている．ニューラルネットで短時間で十分な学習成果を挙げるためには深さ優先探索などを用いた従来型のアルゴリズムを用いて十分な質と量のデータセットを作成する必要がある．得られたデータセットを用いてニューラルネットが教師あり学習を行う．本研究では，従来型の探索手法を用いたフリーセルソルバーを利用したデータセットを作成するための方法について，現時点までに得られた結果に基づいて考察し，今後の課題としている最短手順又はそれに近い解を高速に求めるための構想についても言及する．

2 フリーセルの求解

2.1 フリーセルとは

フリーセルは一人用トランプカードゲームであり，タブローと呼ばれる 8 つの山に表向きに並べられたジョーカーを除く 52 枚のカードをフリーセルと呼ばれる 4 つのスペースをうまく活用して，ホームセルと呼ばれる場所にスート（カードのマークの事）ごとにエースからキングの順番で移動させることを目的とする．初期盤面は図 1 のようにカードを 8 つのタブローにランダムに分割し 7 枚のタブローを 4 つと 6 枚のタブローを 4 つ用意する．移動させることの可能なカードは各タブローの天（一番上，図上の一番下）のカードかフリーセルに置かれているカードである．フリーセルとは 4 つ用意されている空のセルで，フリーセルに置くことの可能なカードは各セルに 1 枚までである．タブローの上に置くことの可能なカードは各タブローの天のカードと異なる色の数字が一つ小さいカードのみである（A=1, J=11, Q=12, K=13 とする）．図 1 について例をあげれば，♡7 の上に♣6 を，♢K の上に♣Q を置くことができるが，♢A の上に♣K を置くことはできない．以上のルールに則り初期盤面からすべてのカードをスートごとに A から K の順にホームセルに移動させれば勝利となる．

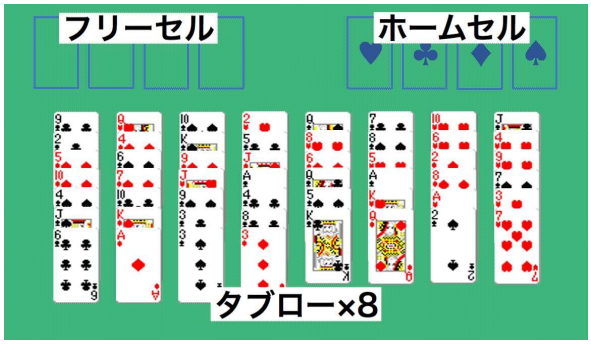

図 1：フリーセルの初期盤面の一例．左上にフリーセルが 4 つ，右上にホームセルがありすべてのカードをスート毎に A から K の順にホームセルに移動させると勝利

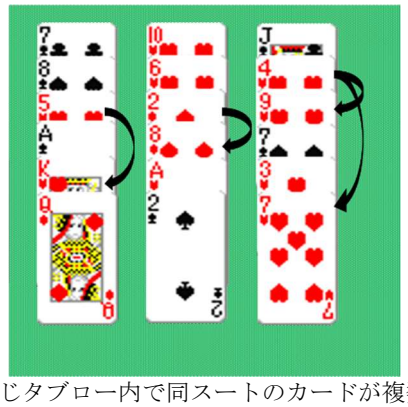

図 2:. 同じタブロー内で同スートのカードが複数あり，数字の大きいカードの方が上にある場合，上にあるカードを“封鎖カード”と呼ぶ

2.2 フリーセルソルバ

様々な初期盤面に対しての解を得るために，初期盤面をランダムに作成する．ランダムに作成した初期盤面をフリーセルソルバに与えて解を求める．使用するソルバではヒューリスティック関数で探索空間を制限し，深さ優先探索を行っている[1][2]．関数には以下の 4 つのパラメータを使用した．“現在までの手数(T)”，“ホームセルに移動していないカードの数(N)”，“封鎖カードの数(B)”，“動かしていないカードの数(M)”の 4 つである．ここでは，図 2 に示したように同じタブロー内に同スートのカードが複数あり，数字の大きいカ

ードの方が上にあるとき上にある数字の大きいカードを封鎖カードと呼ぶ[4].

今回はヒューリスティック関数を次のように定義した.

$$1500 * T + alph1 * N + alph2 * B + alph3 * M$$

ソルバを動かす際に alph1，alph2，alph3，delt(公差)，手数の上限の 5 つの条件を与え，ヒューリスティック関数があらかじめ定めたコストの上限を超えないような着手の探索を行う．コストの上限の初期値は初期盤面のヒューリスティック関数を超えないように指定し，コストの上限より小さいコストで解が見つからない場合は一定数(公差)ずつコストの上限を上げ再び範囲内で解くことの可能な解を探索する．ある程度上限を上げても(今回は 20 回とした)手数の上限より短い手数の解が見つからない場合は解くことを諦める．今回は次に打てる手を見つけるときの高速化手法として 48×2 の 2 次元配列の表を使用し，表を引くだけでタブローからタブローへの移動の手を見付けることが可能なようにしている．内容は各スートの A から Q までの計 48 枚 (K は他のカードの上に置くことはできないので登録する必要がない) につき，現在の盤面で上に置くことが出来るタブローの番号を登録している．表を初期盤面について設定した後，各手毎に微修正を繰り返し，空でないタブローの天のカードそれぞれについて表を引くだけで移動の手を高速で見付けることが出来る．また同一局面の排除のためにゾブリストハッシュを利用している．登録内容は 63 ビットのハッシュ値の他に，16 ビットのその局面までの最短手数を含めている（合計 10 バイト)．出力される解は盤面に対する着手を 16 進数で表記する．一つの着手を 16 進数表現で各フリーセルを 0 から 3，各タブローを 4 から b，各ホームセルを c から f とし，移動元と移動先を組み合わせた 2 文字で表す（例：6b とあれば 6 の位置のカードを b の位置に移動することを表す)．

3 学習データ

今回の目的は十分な質と量をもつディープラーニングのためのデータセットを作成し、ニューラルネットに教師有り学習をさせる[3]ことである．ニューラルネットは現在の盤面に対してどのような着手が有効かを学習する．そのため与えるデータは“現在の盤面＋着手”とする．ソルバで得られた解と初期盤面を与えることで，初期盤面から全てがホームセルに返った状態までの 1 手ごとの盤面を再現し，着手と組み合わせて一つのデータとする．

4 実験

ここではデータセットの作成にかかる時間についての実験結果を述べる．今回は 2500 通りの初期盤面に対する解のデータセットの作成を行った．条件は表 1 に様に定めた.

表 1 ：与えた変数

alph1	alph2	alph3	delt	手数の上限	n
2000	1000	1000	1000	120	12

2500 通りの初期盤面の解を求める前に，まずは何通りかの変数の組み合わせを試した．同一の 50 通りの初期盤面に対して様々な組合せで実験をしたところ表 1 の組み合わせが最も早く解を求めることができたため今回はこの値を採用している．また引数として n を指定し，ハッシュ表のサイズを指定している．サイズは n を与えた時に$10 * 2^{n+12}$バイトとなるようにしている（例:n=14 の時は$10 * 2^{26} \fallingdotseq 6.4 * 10^8$ バイト=640MB)．使用したマシンの仕様を以下に示す.

表 2 ：実験に使用したマシンのスペック

CPU:	Intel(R) Xenon(R) CPU E5-2630 v2 @ 2.60GHz を 2 個搭載
Memory:	256GB
OS:	Cent OS Linux release 7.5.1804
コンパイラ:	gcc 4.8.5

上記のマシンで 125 個の初期盤面に対しての解を求めてデータセットを作成する操作を 20 並列で行い，合計 2500 通りの初期盤面の解を求めるのにかかった時間は約 12 分となった．1 つの初期盤面に対して解を求めるのにかかった時間の平均は約 5.8 秒となった．Paul ら[4]のソルバでは 1 ゲームの最短手順を求める時間の平均が 39.9 秒となっている．Paul らのソルバでは最短手順を求めているため直接的な比較にはならないが，目的が学習に必要なデータを大量に作るということだと考えると今回の手法で十分だと考えた.

5 より良い解を得るために

学習の精度を上げるためには，ソルバが見つける解は可能な限り最短手順に近いことが望ましい．現在使用しているソルバは深さ優先探索に基づいているため，最長手順を指定するとその範囲内で解が見つかるとすぐに出力されるため解がアルゴリズムで制限されている手数の上限に近い長いものになりがちである．以下にソルバに最短手順又は最短手順に近い解を求めさせる方法の構想について述べる.

求まった解の手順から無駄な手を省くための次に述べる操作は，容易に実装することができるので取り入れるべきである.

場所 a にあるカードを別の場所 b に移動する手を (a, b) で表す．手 (a, b) が現れたとしたら，その後次の 3 つの目的のどれかを達成するためである.

(a1) a がタブローの天であり，a の位置に a にあったカードとは別のカードを置くため．従って，その後 (x, a) の形の手が現れる.

(a2) a がタブローの天であり，a にあったカードが移動したことにより現れたカードを別の場所に移動するため．従って、その後 (a, x) の形の手が現れる.

(b) a にあったカードが移動した先 b がタブローの天であり，b の場所に a にあったカードが置かれた後さらに別のカードを置くため．従って，その後 (x, b) の手が現れる。

これらの目的達成の前に (b, y) の形の手が現れていれば，最初の (a, b) の手は無駄だったことになり，解の手順から最初の (a, b) を省き， (b, y) を (a, y) に変えることによって 1 手短かい解の手順が得られる．この操作を解の手数がそれ以上短くならなくなるまで続ける.

深さ優先探索で最短手順を求めるには，解が求まり，その解に対して上に述べた無駄を省く操作を施した後，手数の制限値を得られた解の手数より 1 小さい値に変更した上で探索を再開すればよい．このようにして探索が完了すれば，最後に求まった手順が最短手順である．ただし，ハッシュ表のサイズや実行時間等の計算資

源に関する制約から完了する前に探索を打ち切る必要があるかもしれない. その場合でも, 計算資源を十分大きく取ることによって最短手順に近い手順が得られることが期待できる.

さらに，深さ優先探索における選択肢の順序付の工夫と優れた許容的ヒューリスティック関数の作成により大幅に実行時間が短縮できると考えている．ここでは，変数値として与えられた局面に対して関数値として非負整数を割り当てる関数で関数値が変数値の局面から勝利までの最短手数以下であることが保証されているものを許容的ヒューリスティック関数と呼ぶ．探索途中の局面 p の許容的ヒューリスティック関数値が制限手数から p までの手数を引いた差よりも大きければ，p 以降の探索を打ち切りバックトラックを起こすことができる．従って，許容的ヒューリスティック関数は，変数値の局面から勝利までの最短手数と関数値の差が小さい程優れている．現在，上に述べた構想に従ってフリーセルソルバの改良を進め，より質の高い，より大規模なデータセットの作成に取り組んでいる．

6 まとめ

今回はフリーセルの初期盤面からソルバを用いて解を探索し，その時点での盤面と着手を組合わせた物を一つのデータとして大量に生産するための手法について述べた. その過程でフリーセルをなるべく使わない(4つのうち 2 つしか使用しないなど)ようにする方法など前節に述べていない手法の検討も行ったが，その手法の有効性は不明である．今後 5 章で述べた構想の実現や A*アルゴリズムを用いた探索[4]などの応用に取り組み，より質の高いデータを大量に作成するために短時間で最短手順に近い解を求めることの出来るソルバの開発を行っていきたい.

謝辞

本研究の一部は，株式会社アースライズカンパニーの支援により実施された.

参考文献

[1] Elyasaf, Achiya, Ami Hauptman, and Moshe Sipper. "GA-FreeCell: Evolving solvers for the game of FreeCell." Proceedings of the 13th annual conference on Genetic and evolutionary computation. ACM, 2011.

[2] Elyasaf, Achiya, Ami Hauptman, and Moshe Sipper. "Evolutionary design of freecell solvers." IEEE Transactions on Computational Intelligence and AI in Games 4.4 (2012): 270-281.

[3] Chan, Chris. "Helping Human Play Freecell."

[4] Paul, Gerald, and Malte Helmert. "Optimal solitaire game solutions using A* search and deadlock analysis." Ninth Annual Symposium on Combinatorial Search. 2016.