1E-02

フリーセルの解の存在判定アルゴリズムの設計

Design of an Algorithm for Determining Existence of a Solution to an Initial Position of FreeCell

神保 秀司†
Shuji JIMBO

1 はじめに

フリーセルは，完全情報のソリティアカードゲームであり，初期局面をランダムに作成した場合に解をもたないものが出現する確率が非常に小さいことが知られている．運の要素が無いので古典的プランニング問題のドメインになり得る．

著者らは，効率的にフリーセルの最適解を求めるソルバーの開発を計画しているが，その主な目的は，教師あり学習により局面の難易度をニューラルネットに学習させるときの訓練データを作成することである．フリーセルの局面は，そこから勝利までの最小手数が小さい程易しいと考えられる．十分に学習させたニューラルネットを探索アルゴリズムと組み合せることにより，試行錯誤の回数が極めて少ないプログラムを設計できることを期待している．しかしながら本論文執筆時点で勝利までの最小手数を求めるプログラムが未完成のため，主にランダムな試行に基づいた別の観点からの局面の難易度判定方法について述べる．

2 準備

初めにフリーセルのルールを述べる．

(a) 初期局面では，52 枚の全てのカードが表が見えるようにタブローと呼ばれる 8 つの列に上下方向に並べられている．各列のカードの枚数は，4 つの列が 7 枚で他の 4 つの列が 6 枚である．

(b) タブロー以外にフリーセルと呼ばれる 1 枚のカードが置ける場所が 4 箇所あり，さらにファウンデーションと呼ばれる上がりのカードを積み重ねる場所がカードの種類 (クラブ，ダイヤ，ハート，スペード) 毎に 1 つずつ，合計 4 箇所ある．

(c) フリーセルの手は，フリーセルかタブローの一番下にあるカードを 1 枚だけ取り上げ，次に，取り上げたカードをフリーセルかタブローかファウンデーションに置くという 2 段階から成っている．ただし，取り上げたカードが置けるタブロー内の場所は，空の列か，一番下のカードの順位が取り上げたカードよりも 1 大きく，かつ，取り上げたカードと色が異なる列である．ファウンデーション内の場所は，一番上のカードが取り上げたカードと同じ種類で順位が 1 小さい場所である．

(d) ゲームの目的は，52 枚のカード全てをファウンデーションに移動することである．

フリーセルのような問題は，原理的には整数計画ソルバーや SAT ソルバーで解けるが，フリーセルの問題例を単純に整数計画問題に書き直すと制約不等式の個数が膨大になり求解が極めて困難である．

Paul らは，フリーセルの最小手数の勝利を A* アルゴリズムで求めるときの局面 n についてのヒューリスティック関数値 $h(n)$ としてフリーセルとタブロー内のカード枚数に関数値 $m_e(n)$ を加えたものを提案し，その有効性を主張している [1]．先ず，各カードを点とし次で定義する有向辺集合をもつ 52 点の有向グラフを $G(n)$ とおく．$G(n)$ の有向辺は定常的なものと封鎖的なものの 2 種類からなる．辺 vw が定常的な辺であるのは，v と w の種類が同じで w の順位が v の順位よりも 1 小さいときであり，封鎖的な辺であるのは，同一のタブロー列内で w が v の直ぐ下にあることである．$G(n)$ の定常的な辺に ∞ の重みを与え，封鎖的な辺に 1 の重みを与えたときの最小帰還辺集合 (Minimum feedback arc set 問題の解) のサイズ，最小帰還辺数は，Paul らが定義した関数 $m_e(n)$ の理想値と一致する．

著者による実験では，フリーセルの初期局面から導かれる有向グラフに対する重み付き最小帰還辺集合問題を一般的なノート PC を使って整数計画ソルバーの SCIP に解かせたところ，数十分の一秒程度で解くことができた．このことは，整数計画ソルバーや SAT ソルバーを探索プログラムに組み込んで性能の向上を図れる可能性を示唆している．

† 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University

3 ランダム試行による局面の難易度の判定

著者は，ニューラルネットにフリーセルの局面の難易度を推測させることを教師あり学習により学習させることを計画し，そのためのデータを与えられた局面 n からのランダム試行中の勝利割合等に基づいて以下のように作成することを計画している．

(a) 最大手数を 1000 程度に設定し，同一の試行内では，ハッシュ表を使って同一局面を繰り返さないようにする．

(b) 10 万回程度ランダム試行中の勝利した試行の割合 (勝率) $W(n)$，平均手数 $L(n)$，平均合計分枝数 $B(n)$ を求める．基本的には勝率 $W(n)$ で難易度を推定する．

(c) 10 万回程度のランダム試行で勝利できなければ，さらに最大 40 万回程度までランダム試行を追加して勝利を求める．例えば，局面 n からの合計 m 回のランダム試行で初めて勝利したときの勝率は，$W(n) = 1/m$ で与える．

(d) ランダム試行の最終局面が勝利でない場合は，Paul らのヒューリスティック関数値を整数計画ソルバーを使うなどして計算し難易度判定に使うことを計画しているが，現時点で実装できていない．

次に，プログラムに使ったハッシュ技法について述べる．今回の実験では，局面の ID 番号として次に述べる 63 ビットのゾブリストハッシュ値を採用した．ハッシュ表は，開番地法で実装した．ハッシュ表の各要素は，64 ビット符号無し整数とし，下位 63 ビットでゾブリストハッシュ値を表し，最上位ビットでハッシュ表への登録の有無を表した．

与えられた局面内の 52 枚のカードの内，フリーセルまたはタブローにある各カード x について，その「直下にあるもの」$y(x)$ が何かに対する 2^{63} 未満の正整数の乱数値 $r(x, y(x))$ を割り当て，局面内の該当するカード x すべてについての乱数値 $r(x, y(x))$ の排他的論理和

$$\bigoplus_x r(x, y(x))$$

を局面のハッシュ値とした (ゾブリストハッシュ)．ただし，直下にあるものを表す整数値 $y(x)$ は，0 から 51 までのカード番号にタブローの先頭としての 52 とフリーセルとしての 53 を加えた 54 個の中から選ばれる．ファウンデーションに置いてあるカードは，フリーセルとタブローに置いてあるカードから特定できるので，ハッシュ値に関与させる必要がない．このハッシュ関数により，フリーセル内のカードの並びとタブロー内の列の並びだけが異なる 2 つの局面には，同一のハッシュ値が割り当てられる．

4 実験結果

マイクロソフトのフリーセルの第 1 ゲームから第 20 ゲームまでにランダム試行を $2^{17} = 131072$ 回適用したとき (実験 A) と CentOS 7 の `/dev/urandom` から得た乱数を使って作った 20 個の初期局面にランダム試行を同回数適用したとき (実験 B) の勝利回数の一覧を下に挙げる．平均勝利回数は，実験 A の場合が 112.15 で，実験 B の場合が 32.25 でありマイクロソフトの小さい番号のゲームがランダムに作った局面よりも容易になっていることが考えられる．

実験 A

0	308	538	0	4	18	14	596	0	0
74	1	22	7	63	58	44	91	33	372

実験 B

2	0	0	0	0	1	95	24	16	109
2	0	11	1	6	30	2	216	1	129

5 おわりに

本論文執筆時点で与えられた局面からの最小手数の勝利を求める探索プログラムが未完成のため，主にランダム試行に基づいた局面の難易度判定方法について述べた．この方法は，原始的モンテカルロ法と見做せる．ランダム試行により通過した局面の難易度に関する情報をゾブリストハッシュ値 (ID 番号) と共にハッシュ表に登録し，その情報に基づいて着手の確率を動的に変化させることにより勝利確率を向上させることを今後検討したい．

謝辞 本研究は JSPS 科研費 JP15K00018 の助成を受けたものです．本研究は主に九州大学情報基盤研究開発センターの研究用計算機システムを利用しました．

参考文献

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