絵合わせパズル

～人間はコンピュータに勝てるのか～

プログラミング演習 A グループ：朝山　絵美
Emi ASAYAMA

1　はじめに

近年，コンピュータは急速な発展を遂げている．1997 年，IBM 社のスーパーコンピュータ Deep Blue がチェスの世界チャンピオンであるガルリ・カスパロフ氏に勝利し，世間の注目を浴びた．このように，技術進歩に伴い，人間の能力を上回る性能を持つコンピュータは次々と生まれている．[1)]

本報告では，人間及びコンピュータが対戦できる絵合わせパズルシステムを作成し，そのシステムについて詳細を述べる．本システムは 2 つの機能を有している．1 つは人間がパズルを行うことのできる機能であり，1 つはコンピュータがパズルの配置を自動的に求める最適化機能である．後者では，絵合わせをパズルの最適な配置を求める組合せ最適化問題とみなし，汎用最適化手法であるシミュレーテッドアニーリング (Simulated Annealing:SA) を用いて最適化を行う．本システムではこれらの 2 つの機能を用い，人間とコンピュータで絵合わせパズルを完成させるまでの時間の速さを競う．

2　絵合わせパズルのシステム

2.1　絵合わせパズルとは

絵合わせパズルとは，一枚の絵を複数の正方形のパーツに分割しシャッフルしたものを，元の一枚の絵に戻るようにパーツを動かしていくパズルゲームである．絵合わせパズルでは，Fig. 1 のように空のマスが 1 つあり，その空きマスの周りのいずれか 1 つのマスを空のマスに移動させることでゲームが進んでいく．

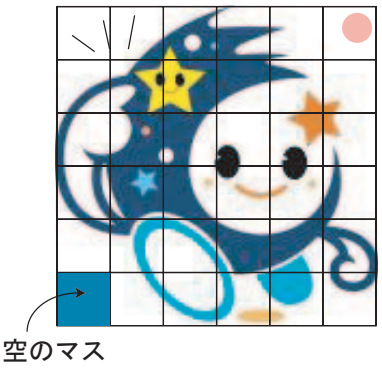

Fig. 1　絵合わせパズル

2.2　システムの概要

本システムは縦 5 マス，横 5 マスの合計 25 マスから構成される．絵のパーツのない空の 1 マスを除いた 24 マスに，分割された絵がはめ込まれている．

本システムには，人間が絵合わせパズルを行う機能と，コンピュータが最適化手法により絵合わせパズルを自動的に行う機能がある．この 2 つの機能を用いることで，人間とコンピュータのどちらが速く絵合わせパズルを完成させることができるかを競うことができる．

2.3　システムの状態遷移モデル

本システムにおける状態は絵全体，すなわちすべてのパーツの配置をまとめて一つの状態としている．状態が遷移する際のトリガとなるものは，ユーザからの入力である．入力は人間が絵のパーツを動かす操作である．それにより状態が遷移する．状態遷移図を Fig. 2 に示す．

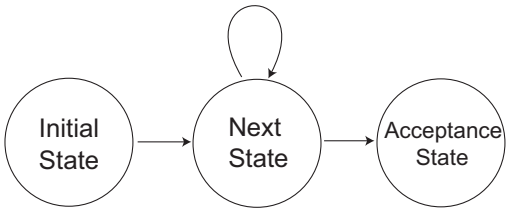

Fig. 2　状態遷移図

Fig. 2 における初期状態は，絵をシャッフルした状態であり，ゲームはこの状態から開始される．ゲーム開始後は，人間によるパーツの移動によって状態が遷移し，絵が完成すれば終了する．

2.4　システムの流れ

本システムでは，まずパーツのシャッフルを行い，絵のパーツがランダムに配置された状態でゲームをスタートさせる．本システムのフローチャートを Fig. 3 に示す．

ユーザがパーツを動かす際のシステムの流れについて詳しく説明する．ユーザによって絵のあるパーツがクリックされると，システムはそのパーツが移動可能であるかを判定する．その判定は，クリックされたパーツの上下左右の 4 つのマスに空のマスがあるか否かで決定する．次に，現在のパーツの配置が終了条件を満たしているかを判定する．それらの処理を繰り返し，絵が完成すればシステムは終了となる．

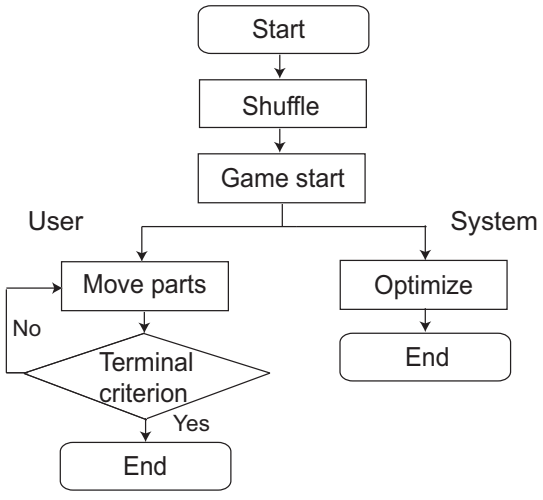

Fig. 3　システムの流れ

3　SA による最適化機能

3.1　SA とは

SA とは，Kirkpatrick[2] らによって提案された最適化問題のための近似解法の 1 つである．SA は，高温で加熱した金属の温度を徐々に下げて冷却することによって，もとの金属より欠陥の少ない優れた結晶構造を生成する物理プロセス (焼きなまし) を計算機上で模倣した最適化手法である．

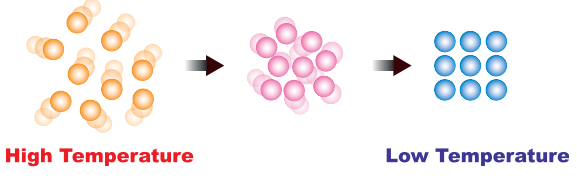

Fig. 4　焼きなまし

SA では，現在の解から計算されるエネルギー (目的関数値) と，その解を微小に変化させた後のエネルギーを用いて，式 (1) に示す Metropolis 基準に従い，この微小な変化を採用するか否かの判断を行う．

$$P_{AC} = \begin{cases} 1, & \text{if } \triangle E < 0 \\ \exp\left(-\dfrac{\triangle E}{T}\right), & \text{otherwise} \end{cases} \tag{1}$$

現在の状態 x を微小に変化させて新しく生成された状態 x' は，そのエネルギー E' が現在の状態 x のエネルギー E よりも低ければ，確率 1.0 で受理される．そうでない場合は，x' のエネルギー E' と x のエネルギー E との差分 $\triangle E(= E' - E)$，および温度パラメータ T を用いて，その摂動が受理される確率を計算し，その確率に従って受理判定がなされる．

このとき温度パラメータ T の値が高い場合には，前の状態よりもエネルギーの大きな状態を選択する可能性が高いが，T が低くなるとエネルギーが小さい状態のみが選択される確率が高くなる．したがって十分高い温度から，徐々に温度を冷却することによって，最小ではない極小 (局所的最小状態) に陥らずに，大域的最小状態に到達することができる．

3.2　SA のアルゴリズム

SA のアルゴリズムを Fig. 5 に示す．

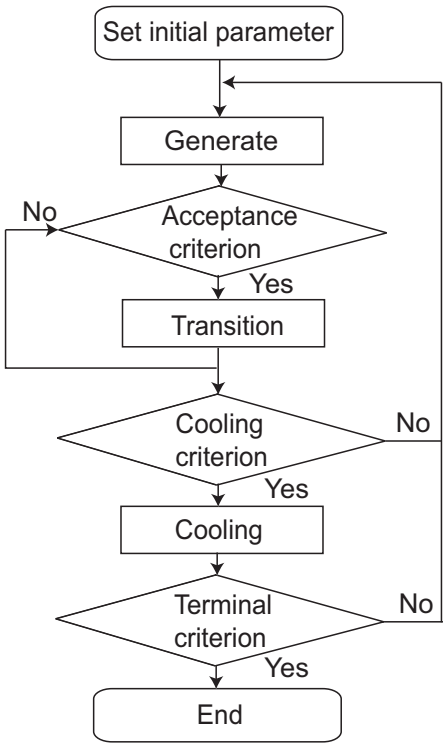

Fig. 5　SA のアルゴリズム

SA のアルゴリズムは次の通りである．

1. 初期設定
   温度を初期化し, 初期状態を与え, 初期状態でのエネルギーを計算する．
2. アニーリング（一定期間繰り返す）
   現在の状態からランダムに次状態を生成し, 次状態でのエネルギーを計算する．エネルギーの変化量と温度を用いて, 次状態を受理するか否かの判定を行う．受理する場合は次の状態に遷移する．
3. クーリング
   一定期間アニーリングを行った後に, 温度を下げる．
4. 終了
   2，3 を十分に繰り返し，終了条件を満たせば終了する．

3.3 絵合わせパズルへの SA の適用

本システムでは，絵合わせパズルをパーツの最適な配置を求める組合せ最適化問題と捉え，最適化アルゴリズムとして SA を適用する．本節では，SA を適用する際の設計変数，目的関数，及び生成処理の方法について述べる．

3.3.1 設計変数

本システムでは，絵の全 25 パーツのそれぞれの座標 $(x_k, y_k)(1 \leq k \leq N)$ を設計変数とする．絵のパーツの座標は状態遷移に従って変化していく．全パーツの座標をまとめて行列 M とし，式 (2) のように与える．なお，N はマスの総数である．

$$M = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ : & : \\ x_k & y_k \\ : & : \\ x_N & y_N \end{bmatrix} \tag{2}$$

3.3.2 目的関数

本システムにおける目的関数を定義する．まず，全 25 マスの座標，すなわちそれぞれの絵のパーツが目標とする座標 $(x_k, y_k)(1 \leq k \leq N)$ をまとめて行列 D とし，式 (3) のように与える．なお， N はマスの総数である．

$$D = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ : & : \\ x_k & y_k \\ : & : \\ x_N & y_N \end{bmatrix} \tag{3}$$

設計変数である現在のパーツの位置座標 M と目標とする位置座標 D のユークリッド距離 $d(k)$ をとり，重み $w(k)$ を乗算する．それらの総和を目的関数とする．式 (4)〜(6) に示す．

$$f = \sum_{k=1}^{N} d(k)w(k) \tag{4}$$

ただし，

$$d(k) = \sqrt{(M_{k1} - D_{k1})^2 + (M_{k2} - D_{k2})^2} \tag{5}$$

$$w(k) = \begin{cases} \sqrt{(D_{k1} - D_{k25})^2 + (D_{k2} - D_{k25})^2} & (k < 25) \\ 1 & (k = 25) \end{cases} \tag{6}$$

とする．

目的関数設計の際，ユークリッド距離 $d(k)$ の総和のみとした場合では，最適な値を得ることができなかった．そのため，距離 $d(k)$ に重み $w(k)$ を乗算した． $w(k)$ は，パズルが正しい配置となった際の空マスの位置とあるパーツとの距離である．これにより，空マスの位置から遠い位置にあるパーツの目的関数値は大きくなる．これは，パズルが正しい配置となった際の空マスの位置から遠い位置にあるパーツを優先して絵合わせを行うと，より速く解くことができるという人間の知恵を利用している．

3.3.3 生成処理

生成処理は，空のマスの近傍で行われる．空のマスの近傍とは，空のマスの上下左右に位置するマスのことである．Fig. 6 のように，近傍は空のマスの場所によって異なる．生成処理では，空のマスの近傍内のどのマスを空のマスへ遷移させるかを確率的に決定する．

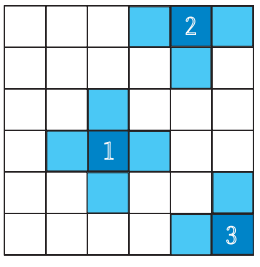

Fig. 6　空のマスの近傍

Fig. 6 の 1〜3 の番号をふっているマスを空のマスとする．

4 数値実験

4.1 実験概要

人間が絵合わせパズルを行うことのできる機能と SA が絵合わせパズルを完成させる最適化機能をそれぞれ実行し，実行時間を計測した．なお，人間が絵合わせパズルを実行する機能では，被験者 10 人を対象として実験を行い，SA では試行回数を 10 試行とした．

最適化機能に用いたパラメータを Table 1 に示す．また，実験に用いた環境は Table 2 の通りである．

Table 1　SA のパラメータ

パラメータ	値
最高温度	100
最低温度	0.1
総アニーリング数 [万]	120
クーリング周期	160

Table 2　実験環境

CPU	Pentium4 530J 3.0GHz
メモリ	1024MB
OS	WindowsXP Professional

4.2　実験結果

実験で得られた結果を Table 3 に示す．なお，それぞれの結果は 10 試行の中央値を示している．

Table 3　比較結果

	実行時間 [sec]
人間	583.5
コンピュータ	8.0

Table 3 より，両者の実行時間を比較すると，コンピュータで絵合わせパズルを行うほうが圧倒的に速いことがわかる．10 人を対象に実験を行ったが，最速で 162[s] であった．このことから考えても人間が実行速度でコンピュータに勝つことは不可能であると考えられる．

5　実行例

絵柄がシャッフルされた際の実行画面を Fig. 7 に示す．この状態は，状態遷移図における初期状態であり，この状態からパズルは開始されることになる．

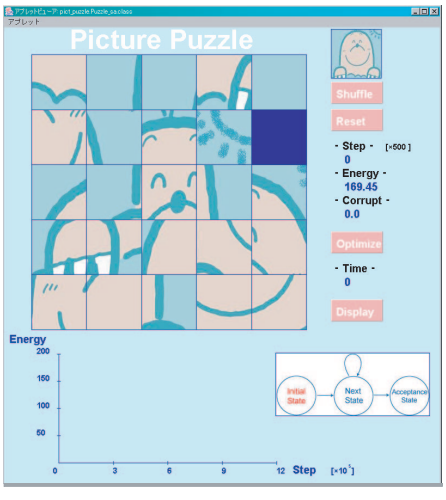

Fig. 7　初期状態

Fig. 7 中の下部のグラフは，SA におけるエネルギー履歴を示している．縦軸をエネルギー，横軸をステップ数としている．

パーツが最適な配置となり，絵が完成した状態 (受理状態) を Fig. 8 に示す．

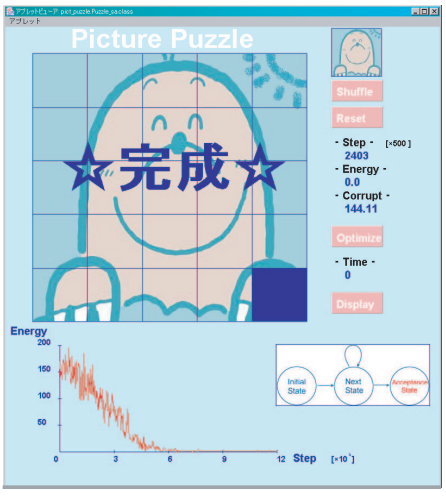

Fig. 8　受理状態

Fig. 8 は，最適化機能を実行した場合の結果を示しており，図中のエネルギー履歴より，探索が進むにつれてエネルギー値が下がり，ある程度の改悪を認めながら最適な値を求められていることがわかる．

6　まとめ

本報告では，絵合わせパズルを行うシステムを作成し，その詳細を述べた．本システムは 2 つの機能を有しており，人間が絵合わせパズルを実行する機能および最適化手法を用いて自動的に絵合わせパズルを行う機能を正しく実装した．実装したシステムは，一枚の絵全体を状態とすることで，状態遷移の様子を視覚的に捉えることができた．SA による最適化機能に関しては，ユークリッド距離のみで目的関数を設計した場合では良好な解を得ることができなかった．しかし，人間が絵合わせパズルを行う際の知恵を取り入れ，目的関数の設計を工夫することで，絵合わせパズルの最適な配置を求めることができた．

また，2 つの機能を有する絵合わせパズルシステムを用いて，両者の実行時間を計測し比較した．コンピュータが圧倒的に勝った．本問題に関しては，人間はコンピュータに勝つことが不可能であると考えられる．

参考文献

1) `http://www.ywad.com/books/161.html`

2) Kirkpatrick, S., Gelett Jr.C.D., Vecchi, M.P.Optimization by simulated annealing. Science, Vol.220, No.4598, pp.671-680, 1983.

3) Steven Holzner，武藤健志，トップスタジオ. JAVA プログラミング Black Book 2nd Edition, 2003.