平成 14 年度

学士学位論文

集合 2 分割問題の解法とその評価

Evaluation of Several Solution Methods of a Set Bipartition Problem

1030293　　西村 章

指導教員　　坂本 明雄

2003 年 2 月 12 日

高知工科大学 情報システム工学科

要 旨

集合 2 分割問題の解法とその評価

西村 章

非負整数の集合が与えられたとき，この集合を 2 つの部分集合に分割することを考える．それぞれの部分集合に含まれる要素数を同じにするという条件のもとで，部分集合に含まれる要素の総和の差を最小にする問題を集合 2 分割問題という．

この問題を解くために，最も簡単なヒューリスティックアルゴリズムである欲張り法 (Greedy Method: GM)，欲張り法の解を初期解として少しずつ良い解に改善していく逐次改善法 (Sequential Improvement Method: SIM)，計算時間は指数関数的に増大するが最適解を保証する完全列挙法 (Complete Enumeration Method: CEM)，最適化手法の 1 つとして有効であると言われている遺伝的アルゴリズム (Genetic Algorithm: GA) の 4 つの解法を取り上げて，それらの評価値の精度，計算時間を比較し検証する．

キーワード　集合 2 分割問題，ヒューリスティックアルゴリズム，欲張り法，逐次改善法，完全列挙法，遺伝的アルゴリズム

Abstract

Evaluation of Several Solution Methods of a Set Bipartition Problem

NISHIMURA Akira

We consider a partition of given set of non-negative integers into two equal size subsets. The sum of a subset is the summation of all elements included in the subset. A set bipartition problem is to seek the best partition where the difference of the sum of two subsets is minimum.

In order to solve this problem, four solution methods are used. Those are a greedy method which is the simplest heuristics algorithm, a sequential improvement method which improves the solution of the greedy method, a complete enumeration method which guarantees the optimal solution, and a genetic approach which is one of the effective optimization techniques. These are compared and verified in the accuracy of a solution and calculation time. Experimented results are shown and the evaluation of those four methods is discussed.

key words Set Bipartition Problem, Heuristics Algorithm, Greedy Method, Sequential Improvement Method, Complete Enumeration Method, Genetic Algorithm

目次

図目次

表目次

第 1 章

序論

LSI などのマイクロチップをレイアウト設計する際，部品の適切な配置とそれらの間の配線を定め，できるだけ面積の小さいマイクロチップを設計することが重要な課題となる．この部品配置の設計ということに重点をおいて考えるとき，それぞれの部品の重み (長さや面積など) の違うものをいくつかに分割する問題に帰着する．また，分割方法は 2 分割が基本となる．このような観点から，本研究では次に述べる集合 2 分割問題に着目し，4 種類の解法をプログラミングしてその実験結果からこれらの解法を評価した.

集合 2 分割問題とは，非負整数の集合が与えられたとき，この集合を 2 つの部分集合に分割にする問題である．ただし，それぞれの部分集合に含まれる要素数が同じであるという条件のもとで，それぞれの部分集合の総和の差を最小にする分割を最適なものと考える.

この集合 2 分割問題は NP-困難な問題の一つであり, 問題のサイズの多項式時間で解けるアルゴリズムは存在しないだろうと言われている [1].

本研究ではこの問題を解くために利用した解法として，最も簡単なヒューリスティックアルゴリズムである欲張り法 (Greedy Method: GM)，欲張り法の解を初期解として少しずつ良い解に改善していく逐次改善法 (Sequential Improvement Method: SIM)，計算時間は指数関数的に増大するが最適解を保証する完全列挙法 (Complete Enumeration Method: CEM)，最適化手法の 1 つとして有効であると言われている遺伝的アルゴリズム (Genetic Algorithm: GA) の 4 つの解法を取り上げて，それらの評価値の精度，計算時間を比較し検証する.

GM とはある順序で集合の要素を 1 つずつ取り上げ，その段階で良いと思われる部分集合へその要素を割り当てる．ここでは，値が大きい要素から順序に調べることにしている.

SIM とは 1 つの解から出発し，何らかの方法でその解を改良していく方法である．ここでは，現在の解のそれぞれの部分集合から同じ個数の要素を取り出し，より良い解となるように取り出された要素を 2 分割して元へ戻すようにしている．

CEM とはすべての 2 分割を列挙し，それぞれの解の良さを計算し，それらの最小値を与える 2 分割を解とする．この解法は最適解が保証される．

GA[2][3][4] とは，生物の進化と淘汰の過程を模倣した最適化手法で，初期染色体集団から選択された染色体が交叉と突然変異を繰り返し世代を重ねることで，より環境に適応した個体を生みだしてゆく過程を模倣したアルゴリズムである．

本論文の構成は以下の通りである．2 章で集合 2 分割問題の詳細を説明し，本研究における解の計算過程を説明する．以降，3 章ではヒューリスティックアルゴリズムに属する欲張り法と逐次改善法の説明，4 章では完全列挙法の説明，5 章では遺伝的アルゴリズムの概要と解法について説明する．6 章ではそれぞれの解法の実験結果をふまえた上で比較と評価を述べる．最後に 7 章は結論を述べている．

第 2 章

集合 2 分割問題

2.1 集合 2 分割問題の定義

集合 2 分割問題とは，はじめに非負整数の集合が与えられ，この集合を部分集合に 2 分割にする．このときそれぞれの部分集合の要素数を同じ値にし，それぞれの部分集合の総和の差を最小にする問題である．

集合 2 分割問題で与えられる集合を X とし，その集合のサイズを n とする．すなわち

$$X = \{x_1, x_2, \ldots, x_n\}$$

$$|X| = n$$

である．一般に集合 A の要素の総和を $s(A)$ と表す．

$$s(A) = \sum_{a_i \in A} a_i \tag{2.1}$$

2 分割された集合を Y，Z とする．これらは同じ個数の要素からなるので

$$|Y| = |Z| = n/2 \tag{2.2}$$

$$Y \cup Z = X$$

である．X の 2 分割 Y，Z の評価値を $f(Y, Z)$ とすれば，

$$f(Y, Z) = |s(Y) - s(Z)| \tag{2.3}$$

となる．集合 2 分割問題は与えられた集合 X の 2 分割 Y，Z の評価値が最小となるものを求める問題である．

2.1　集合 2 分割問題の定義

次に具体例を示す.

$$X = \{217, 594, 110, 129, 438, 286, 540, 277, 752, 668\}$$

上に示した集合 X が与えられたとき，2 分割 Y，Z の評価値が最小となる部分集合 Y，Z は

$$Y = \{217, 129, 438, 540, 668\}$$

$$Z = \{594, 110, 286, 277, 752\}$$

となる．このときの集合 Y，Z のそれぞれの総和は式 (2.1) より

$$s(Y) = 1992$$

$$s(Z) = 2019$$

となり，2 分割 Y，Z の評価値は

$$f(Y, Z) = 27$$

となる.

また，集合 2 分割問題は NP-困難な問題の一つであり, 問題のサイズの多項式時間で解けるアルゴリズムは存在しないだろうと言われている [1]．n 個の要素をもつ集合を等しいサイズの 2 つの部分集合に分割する方法は全てで ${}_nC_{n/2}$ 通りある．スターリングの公式を用いれば ${}_nC_{n/2} \approx 2^n$ となることが知られている．[1]

第 3 章

ヒューリスティック解法

3.1　ヒューリスティックアルゴリズムの概要

ヒューリスティックアルゴリズムは，人間が探索していく方法に比較的近い探索方法である．このアルゴリズムの利点として短時間で近似解を得ることができるということ，欠点としては発見的な手法のため最適解を求める保証がないことが挙げられる．

集合 2 分割問題に対するヒューリスティック解法は，最も単純なヒューリスティックアルゴリズムである欲張り法，欲張り法で得られた解を初期解として改善させていく逐次改善法を用いている．

3.2　欲張り法

入力データとして集合 X が与えられたとき，X の要素を一つずついずれかの部分集合に割り当てていく．割り当てていく際，選択される要素は評価値に大きく関連しているため，集合 2 分割問題ではそれぞれの要素が割り当てられる段階で評価値が小さくなるように部分集合に割り当てる．最終的な評価値の差が最小となるためには要素の値が大きいものから割り当てる方が良いと考えられる．これを X の要素が全て割り当てられると終了とする．

次に欲張り法のアルゴリズムを説明する．

1. まず分割する集合を与える．
2. 与えられた集合に含まれる要素の大きい順に選択し部分集合に割り当てるが，このとき部分集合の評価値が小さくなる方に割り当てる．

3. いずれかの部分集合のサイズが与えられた集合のサイズの半分になったら 4 へ進む．そうでなければ 2 へ戻る．
4. 部分集合のサイズが与えられた集合のサイズの半分になっていない方の部分集合に残っている要素を全て割り当て終了となる．

欲張り法のアルゴリズムを図で示すと図 3.1 に示すようになる．

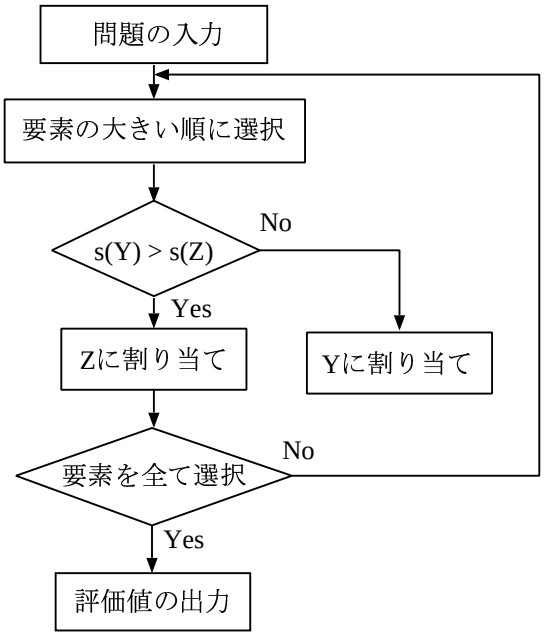


図 3.1　欲張り法の計算過程

次に集合 2 分割問題に対する欲張り法の計算過程の例を示す．

$$X = \{17, 11, 7, 13, 5, 12, 6, 10, 1, 8\}$$

このような集合 X が始めに与えられたとき，まず始めに選ばれる要素は集合 X の一番大きな値の要素 17 である．このとき部分集合 Y，Z の中身は空集合ため $s(Y)$，$s(Z)$ ともに 0 である．そのため選ばれた要素 17 は部分集合 Y に付加され以下のようになる．

$$Y = \{17\} \qquad Z = \emptyset$$

$$s(Y) = 17 \qquad s(Z) = 0$$

次に選ばれる要素は集合 X から要素 17 がなくなるため次に大きい値である要素 13 となる．付加する際，欲張り法では分割される段階で評価値を小さくしなければいけないため $s(Y)$, $s(Z)$ を見て小さい方の部分集合に付加する．例の場合は $s(Z)$ の方が小さいため部分集合 Z に要素 13 が付加され以下のようになる．

$$Y = \{17\} \qquad Z = \{13\}$$

$$s(Y) = 17 \quad s(Z) = 13$$

次は要素 12 が選ばれ以下のようになる．

$$Y = \{17\} \qquad Z = \{13, 12\}$$

$$s(Y) = 17 \quad s(Z) = 25$$

このように次々に大きな値の要素を付加していき，全ての要素が付加されると終了される．付加される際，$s(Y)$，$s(Z)$ ともに同じ値であれば部分集合 Y に付加されるようになる．例の場合は以下のような部分集合 Y，Z と部分集合 Y，Z の要素の総和 $s(Y)$，$s(Z)$ が得られる．

$$Y = \{17, 11, 8, 6, 5\}$$

$$Z = \{13, 12, 10, 7, 1\}$$

$$s(Y) = 47 \quad s(Z) = 43$$

よって，例の場合の評価値は式 (2.3) より 4 となる．

今回の欲張り法での計算方法は，集合 X に含まれる要素の大きい順に各分割段階で選択されている．これは分割された部分集合の最終的な差を小さくするためである．分割段階で後になるにつれて要素の値が小さいものが残るため誤差が小さくなると期待できる．逆に要素の値が小さいものから選択されてしまうと，後に残るものが大きな値となってしまい誤差が大きくなる可能性がある．

3.3　逐次改善法

欲張り法で得た 2 分割を初期解とする．現在の解の部分集合から要素を k 個ずつ取り出し，取り出された要素でできる全ての 2 分割を考え，元の集合に戻したときのに評価値が最小となる解を求める．それらの最適な 2 分割を用いて評価値を改善する．あらかじめ定めておく解の未更新限度回数だけ繰り返しても改善されないとき終了する．

次に逐次改善法のアルゴリズムを説明する．

1. 初期解として欲張り法で得た 2 分割を解として入力する．
2. 逐次改善法の評価値の計算を行う．
3. 元となる解よりも新しい解が良ければ更新し，同じであればばそのまま未更新となり解の未更新回数を 1 増やし 2 へ戻る．
4. 解の未更新回数があらかじめ定められた回数になると解を出力し終了．

逐次改善法のアルゴリズムを図で示すと図 3.2 のようになる．

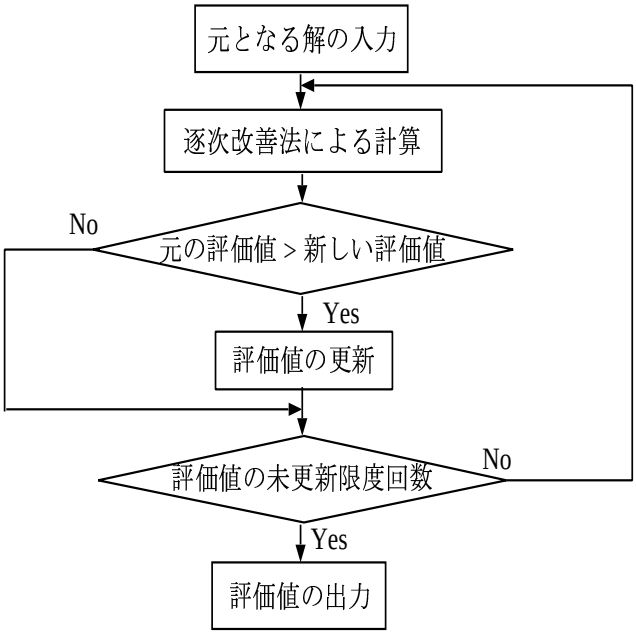


図 3.2　逐次改善法の計算過程

次に集合 2 分割問題に対する逐次改善法の計算過程を例で示す．例は 3.2 で示したものを

利用する.

欲張り法で求められ，逐次改善法の元となる 2 分割 Y，Z，評価値は以下に示すとおりである.

$$Y = \{17, 11, 8, 6, 5\} \qquad Z = \{13, 12, 10, 7, 1\}$$

$$f(Y, Z) = 4$$

次にランダムに要素を取得する k を 2 とし 2 分割 Y，Z からそれぞれランダムに 2 個ずつ取得し，配列 x に格納するとこのようになる.

$$x = \{11, 8, 7, 12\}$$

このときの取得方法は配列 X の左右半分から 2 個ずつ取得するようになる．この配列 x の配列パターンは,

$${}_nC_{n/2} = {}_4C_2 = 6$$

となり，6 通り存在し以下のようになる.

$$x = \{11, 8, 7, 12\}$$
$$x = \{11, 7, 8, 12\}$$
$$x = \{11, 12, 7, 8\}$$
$$x = \{7, 8, 11, 12\}$$
$$x = \{7, 12, 11, 8\}$$
$$x = \{8, 12, 11, 7\}$$

配列 x を元の部分集合 Y，Z に戻したものとその評価値は以下のようになる.

$$Y = \{17, 11, 8, 6, 5\} \quad Z = \{13, 12, 10, 7, 1\} \quad f(Y, Z) = 4$$
$$Y = \{17, 11, 7, 6, 5\} \quad Z = \{13, 12, 10, 8, 1\} \quad f(Y, Z) = 2$$
$$Y = \{17, 12, 11, 6, 5\} \quad Z = \{13, 10, 8, 7, 1\} \quad f(Y, Z) = 12$$
$$Y = \{17, 8, 7, 6, 5\} \quad Z = \{13, 12, 11, 10, 1\} \quad f(Y, Z) = 4$$
$$Y = \{17, 12, 7, 6, 5\} \quad Z = \{13, 11, 10, 8, 1\} \quad f(Y, Z) = 4$$
$$Y = \{17, 12, 8, 6, 5\} \quad Z = \{13, 11, 10, 7, 1\} \quad f(Y, Z) = 6$$

上の結果，評価値が 2 の場合が 6 通りの中で一番最小となり元となっている評価値よりも良いためここで得られた 2 が次の元となる評価値となる．またこの結果を得ることができた 2 分割 Y,Z が次の元となる 2 分割となる．この操作をあらかじめ定めておいた解の未更新限度回数繰り返し行うと終了となる．

3.4　ヒューリスティックアルゴリズムの特徴

ヒューリスティックアルゴリズムは，比較的短時間で解を求めることができる．しかし，以下の例のような集合 X が与えられたときには，

$$X = \{3, 5, 10, 1, 9, 4\}$$

欲張り法では以下のような評価値が得られてしまう．

$$Y = \{10, 4, 3\} \qquad Z = \{9, 5, 1\}$$

$$f(Y, Z) = 2$$

この例では以下に示すような最適解が存在する．

$$Y = \{9, 4, 3\} \qquad Z = \{10, 5, 1\}$$

$$f(Y, Z) = 0$$

また，逐次改善法ではランダムに要素を選択しているため，評価値を改善できる要素が選ばれないと上のような最適解を得ることができない．

このように，ヒューリスティックアルゴリズムは局所解に陥りやすく，最適解が得られない場合が考えられるため，最適性は保証できない．

第 4 章

完全列挙法

4.1　完全列挙法

完全列挙法は最適解を保証する解法である．計算方法は集合が与えられたときに全ての 2 分割を列挙しその評価値を計算する．全ての組合せパターンの計算が終了した時点で一番良い評価値が，与えられた集合を 2 分割にしたときの最適解である．しかし，集合 2 分割問題では 2 分割にされた部分集合が逆になっているだけのものは同じ部分集合とみなすため，計算時間を短縮できる．この詳細については次節で述べる．また，全ての組合せパターンを求めるために，計算過程の中で新たに 0 と 1 を $n/2$ 個ずつ入れてできる配列パターンを作成する関数を作り，計算の簡略化をしている．この関数についても次節で述べる．

4.2　0-1 配列作成関数

集合 2 分割問題は NP-困難な問題に属するため要素数が多くなると膨大な計算時間が必要となる．このため完全列挙法などの最適解を保証する解法では計算時間短縮が重要な課題となってくる．本節では計算時間短縮のために作成した 0-1 配列作成関数について説明する．

今回利用する 0-1 配列とは，集合 X が与えられたときの集合 X に含まれる要素数 n を配列の大きさとし，その中に 0 と 1 を $n/2$ 個ずつ入れてできる配列のことである．これを具体例を挙げて示す．

集合 X が与えられたとき，集合 X の要素数を 4 とすると，以下のような 0 と 1 が 2 個ず

つ入った配列が作られる．

$$[1 \quad 1 \quad 0 \quad 0]$$
$$[1 \quad 0 \quad 1 \quad 0]$$
$$[1 \quad 0 \quad 0 \quad 1]$$
$$[0 \quad 1 \quad 1 \quad 0]$$
$$[0 \quad 1 \quad 0 \quad 1]$$
$$[0 \quad 0 \quad 1 \quad 1]$$

これが要素数 4 の 0-1 配列の全 6 パターンである．この解法の分割方法は 0，1 がある場所と集合 X を対応させて分割する．要するに 1 の場所の要素を部分集合 Y に，0 の場所の要素を部分集合 Z とする．また，与えられる集合 X の要素数が 6 ならば配列パターンは ${}_6C_3 = 20$ 個作成され，要素数が 8 ならば ${}_8C_4 = 70$ 個作成される．しかし，この配列全パターンを計算していると要素数が増えるにつれて膨大な計算時間が必要となる．この計算時間を少しでも短縮させるために前節で述べたように 0 と 1 が逆の配列パターンは削除する．これを上の例の 0-1 配列の全パターンと集合 X を与えて説明すると以下のようになる．

$$X = \{3, 5, 2, 4\}$$

この集合 X と上で作成した 0-1 配列を対応させて部分集合 Y，Z に分割し式 (2.2) より評価値を求めると以下のようになる．

$$[1 \quad 1 \quad 0 \quad 0] \qquad Y = \{3, 5\} \qquad Z = \{2, 4\} \qquad f(Y, Z) = 2$$
$$[1 \quad 0 \quad 1 \quad 0] \qquad Y = \{3, 2\} \qquad Z = \{5, 4\} \qquad f(Y, Z) = 4$$
$$[1 \quad 0 \quad 0 \quad 1] \qquad Y = \{3, 4\} \qquad Z = \{5, 2\} \qquad f(Y, Z) = 0$$
$$[0 \quad 1 \quad 1 \quad 0] \qquad Y = \{5, 2\} \qquad Z = \{3, 4\} \qquad f(Y, Z) = 0$$
$$[0 \quad 1 \quad 0 \quad 1] \qquad Y = \{5, 4\} \qquad Z = \{3, 2\} \qquad f(Y, Z) = 4$$
$$[0 \quad 0 \quad 1 \quad 1] \qquad Y = \{2, 4\} \qquad Z = \{3, 5\} \qquad f(Y, Z) = 2$$

このように 0，1 が逆になっている配列は部分集合 Y，Z も逆になるだけで評価値は同じ値になる．このためプログラム上では 0-1 配列の 1 番目を 1 と固定し配列を生成する．

4.3　完全列挙法の計算過程

完全列挙法の計算過程を図で示すと図 4.1 のようになる.

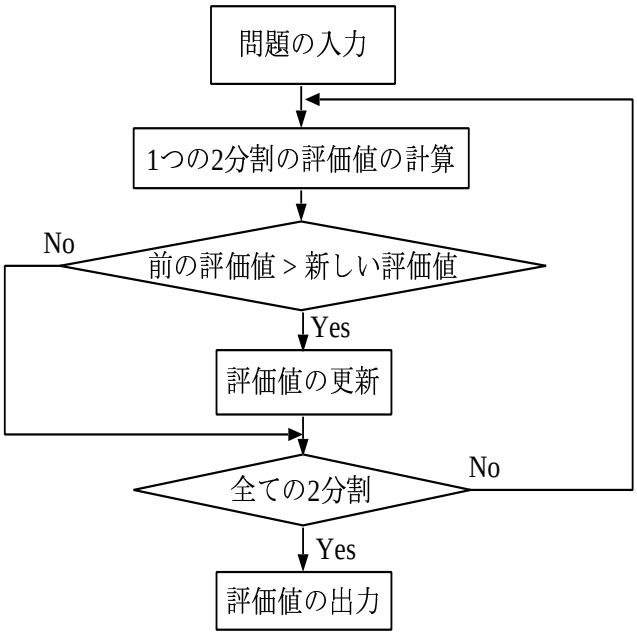


図 4.1　完全列挙法の計算過程

次に例を挙げて完全列挙法の計算過程を示す. 以下のような集合 X が与えられたとき,

$$X = \{3, 5, 6, 2, 1, 9, 7, 4\}$$

要素数は 8 となるため前節 4.2 で述べたように 0-1 配列のパターンは ${}_8C_4/2 = 35$ 通り作成される. 作成された 35 通りの 0-1 配列パターンと集合 X を対応させて得られた評価値集団の中で, 一番値の小さい評価値が, 与えられた集合 X の最適解となる. 以下に示したものが最適解を得ることができる 0-1 配列である.

$$[1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0]$$

これと集合 X とを対応させてできた部分集合 Y, Z と式 (2.3) より得られた評価値は以下のようになる.

$$Y = \{3, 5, 2, 9\} \qquad Z = \{6, 1, 7, 4\}$$

$$f(Y, Z) = 1$$

4.4　完全列挙法の特徴

今回の完全列挙法は，4.2 節で述べたように全ての組合せを計算させる時間の半分の計算時間で最適解を求めることが可能である．しかし，与えられる集合の要素数が大きくなると膨大な時間が必要となる．

このため完全列挙法は，最適性は保証されるが問題サイズが大きくなると短時間で解を得ることができないという特徴が挙げられる．

第 5 章

遺伝的アルゴリズム

5.1　遺伝的アルゴリズムの概要

遺伝的アルゴリズム [2][3][4] とは，ダーウィンの「自然選択説」を模倣したアルゴリズムである．「自然選択説」とは環境条件に最も適応している種が生き残り，さらに進化する可能性を秘めているとされている．要するに，進化の結果変異した個体がその環境から拒まれれば悪い個体とされその個体は淘汰される．一方，その個体が環境に受け入れられると良い個体とされる．環境に適応した個体は交叉を行い子孫を残すことができる．また，突然変異によりまれに良い個体が生まれことがある．交叉や突然変異により良い個体を次世代の集団として世代を重ねるにつれて，環境への適応度が高い遺伝情報は優先的に伝えられ，適応度の低い遺伝情報は自然淘汰される．つまり集団の中には，適応度の高い個体が増えていくことになる．

遺伝的アルゴリズムはこのように自然の生物の進化と淘汰の過程を簡略化した最適化手法である．

5.2　遺伝的アルゴリズムの構成と流れ

遺伝的アルゴリズムを計算機で行うための構成を図 5.1 に示す．各段階の項目についても簡単に説明する．

- 初期染色体集団の生成

  ランダム関数を利用し様々な遺伝子を持った染色体の初期集団の形成.

- 適応度の計算

5.2　遺伝的アルゴリズムの構成と流れ

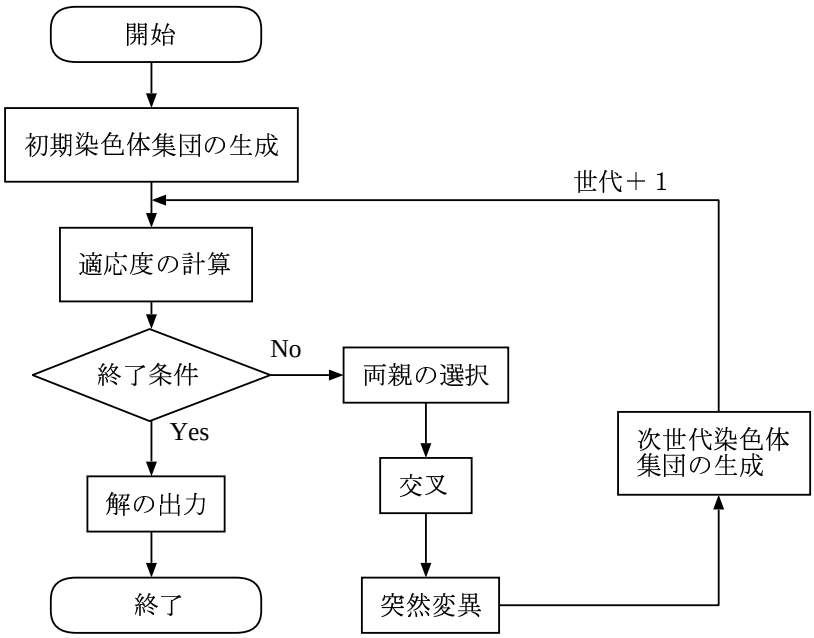


図 5.1　遺伝的アルゴリズムの流れ

各染色体に対して，その染色体が持っている遺伝子情報から得られた環境に対する適応の度合を表した値の計算.

- 終了条件

  条件を満たすと世代の繰り返しが終了し解の出力へ進む.

- 両親の選択

  適応度に応じた染色体を染色体集団から選択する．次世代に良質な染色体を残すために適応度の高い染色体が選択されやすく，適応度の低い染色体は選択されにくい.

- 交叉

  選択された染色体を交叉させ新たな染色体を形成する．交叉とは 2 つの親となる染色体の遺伝子を一部交換させ，新しく 2 つの子となる染色体を作ることである.

- 突然変異

  ランダム関数を利用し染色体に含まれる遺伝子を交換する.

- 次世代染色体集団の生成

  次の世代に残す染色体集団の形成.

5.3　遺伝的アルゴリズムの解法

遺伝的アルゴリズムの解法には著者と同じ坂本研究室に所属している赤間 寛君が作った順列符号化 GA の開発環境を利用し実験を行っている.

5.3.1　初期染色体集団の生成

初期染色体集団の生成は染色体の個数 n と染色体に含まれる遺伝子数 m よって決定される．染色体の個数はあらかじめ決めることができ，遺伝子数は与えられた集合の要素数となる．また，遺伝子は 0 から $m-1$ までの順列によってランダムに生成される．例えば $n=5$ とし，$m=6$ としたときの初期染色体集団を図で示すと図 5.2 のようになる.

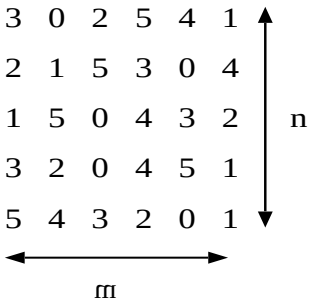


図 5.2　初期染色体集団

5.3.2　適応度

次世代に良質な染色体を残すために適応度の計算は重要なものとなる．このため今回の遺伝的アルゴリズムの適応度の計算に欲張り的な方法を取り入れて，ある程度良い適応度を求められるようにしている．計算過程は次の通りである.

1. 集合 X を与える.
2. 染色体集団の情報を取り入れる.
3. 各染色体の遺伝子と集合 X を対応させ適応度を計算する.

この計算過程を例を挙げて説明する．集合 X として下のような集合を与える．この時, 集合 X を 1 つの配列とする．

$$X = \{15, 17, 12, 16, 18, 14, 10, 11, 19, 13\}$$

次に染色体集団の情報を取り入れ，配列 X と各染色体に含まれる遺伝子情報を対応させる．遺伝子と配列との対応は以下のようになる．

染色体　：　9　2　0　5　3　6　1　4　7　8

上のように染色体が与えられるため，染色体に含まれる遺伝子の値と配列 X の配列番号とを対応させる．対応図は図 5.3 のようになる．

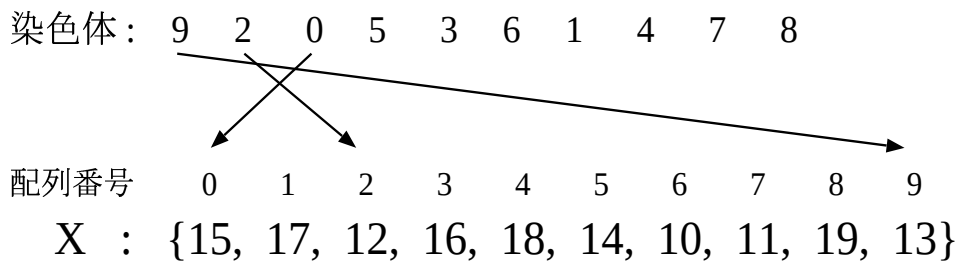


図 5.3　染色体と配列の対応図

このことをふまえた上で適応度の計算をする．計算方法は欲張り的な方法を取り入れているため次のようになる．

染色体に含まれる遺伝子の順番に要素を割り当てていく．まず遺伝子 9 に対応する集合 X の配列番号に格納されている要素 13 が選択され割り当てられる．このとき 3 章で述べた欲張り法と同じく評価値が小さくなる方に割り当てる．始めは部分集合 Y，Z ともに空集合のため部分集合 Y に要素 13 が入る．よって部分集合 Y，Z と部分集合の要素の総和 $s(Y)$, $s(Z)$ は以下のようになる．

$$Y = \{13\} \qquad Z = \emptyset$$

$$s(Y) = 13 \qquad s(Z) = 0$$

次に選択される要素は遺伝子 2 と対応する集合 X の配列番号に格納されている要素 12 が選択され割り当てられる．割り当ては，部分集合の要素の総和が小さい方になるため Z が

選択される.

$$Y = \{13\} \qquad Z = \{12\}$$

$$s(Y) = 13 \qquad s(Z) = 12$$

次に遺伝子 0 に対応する要素 15 が選択され，総和の小さい Z が選ばれる.

$$Y = \{13\} \qquad Z = \{12, 15\}$$

$$s(Y) = 13 \qquad s(Z) = 27$$

このように次々に遺伝子と対応する要素が割り当てられていくと以下のようになる.

$$Y = \{13, 14, 16, 17, 11\}$$

$$Z = \{12, 15, 10, 18, 19\}$$

$$s(Y) = 71 \qquad s(Z) = 84$$

このように分割された結果式 (2.3) より評価値は 13 となる．この操作を染色体集団全ての染色体に行い適応度の計算をする.

5.3.3　両親の選択方法

両親の選択方法としてルーレット選択，トーナメント選択の 2 種類選択できるようにしている.

ルーレット選択

ルーレット選択は適応度の計算によって得られた適応度により両親として選択される割合が決定される．全体に占める適応度の割合が多ければ面積が増え両親として選択される確率が高くなる．適応度の割合が少なければ両親として選ばれる確率は低くなるが，両親として選択されないということは無い．ルーレット選択の具体的な例を図 5.4 に示す.

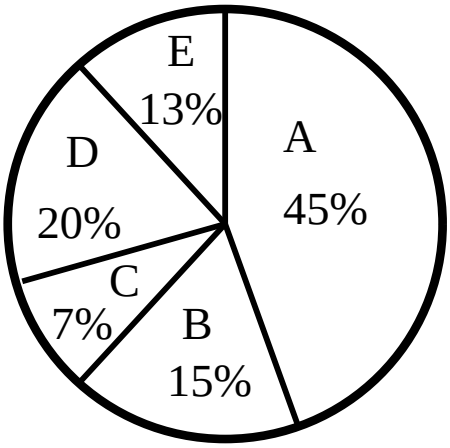
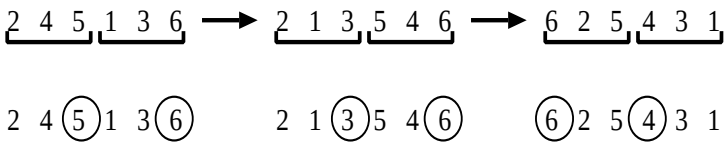


図 5.4　ルーレット選択

トーナメント選択

トーナメント選択は適応度の計算によって得られた適応度の集合をランダムに並べ換え，この集合を 2 分割にし部分集合を生成する．それぞれの部分集合から最も適応度の高い値を親として選択する．選択後，適応度の集合を再度ランダムに並べ換え同じ操作を行う．これを親の数が適応度の集合中の解の数だけ繰り返し実行する．トーナメント選択は図 5.5 のように最悪の解は選択されない．



＊注 ：最も適応度が低い値を1、
最も適応度の高い値を6とする

図 5.5　トーナメント選択

5.3.4　交叉方法

交叉方法として順序交叉，部分写像交叉，サイクル交叉，一様順序交叉の 4 種類を選択できるようにしている．なお順序交叉，部分写像交叉については切断点が 1 つか 2 つで 2 種類

に分ける．切断点が 1 つならば操作する遺伝子を左か右で，また切断点が 2 つならば操作する遺伝子を切断点の内側か外側かで分けている．そのため順序交叉で 4 種類，部分写像交叉で 4 種類となっている．これらのことを図で示すと図 5.6 のようになる．また切断点とは染色体のある部分で区切る場所を表す点である．図 5.6 の k は切断点を表す.

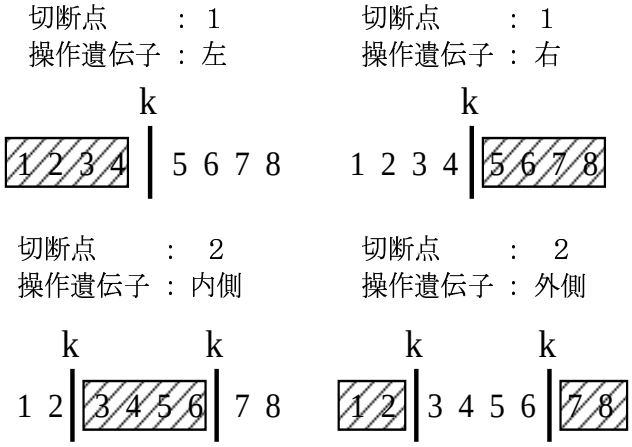


図 5.6　切断点と操作遺伝子

順序交叉

順序交叉は対立遺伝子の相対位置に注目し操作を行う．まず両親となる 2 つの染色体 (P_1, P_2) を選択しランダムに切断点を決定する．次に親 (P_1) の切断点から左部分をそのまま子 (O_1) に写す．その後，親 (P_1) の残る右部分を親 (P_2) で左端から出現する順序にしたがって子 (O_1) に写す．この操作を図 5.7 に示す.

部分写像交叉

部分写像交叉は交叉する親によって部分的に決まる遺伝子の順序の情報を持つ子孫が生成される．まず両親となる 2 つの染色体 (P_1, P_2) を選択しランダムに切断点を決定する．親 (P_1) の切断点から右側にある遺伝子と親 (P_2) の切断点から左側の部分に同じ遺伝子がないか調べる．もし同じ遺伝子があればその遺伝子座の対立遺伝子を見る．その対立遺伝子がさ

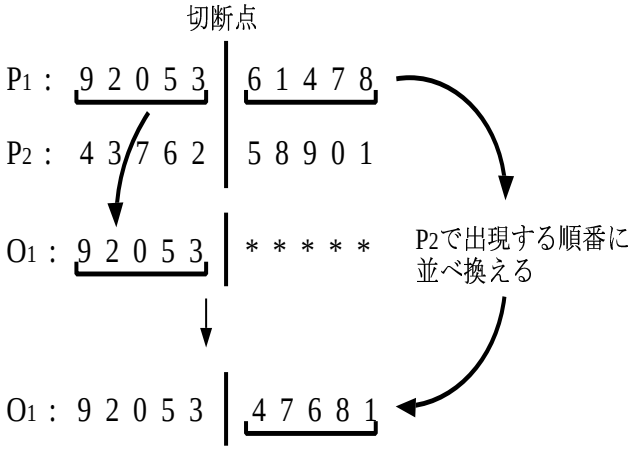


図 5.7　順序交叉

らに親 (P_2) の切断点より左側の部分に同じ遺伝子がないか調べる．この操作を同じ遺伝子が発見できなくなるまで行う．最終的に選ばれた遺伝子を子 (O_1) に入れ，右部分を決定する．次に子 (O_1) の左部分は親 (P_2) の切断点より左側をそのまま写す．この操作を図 5.8 に示す．

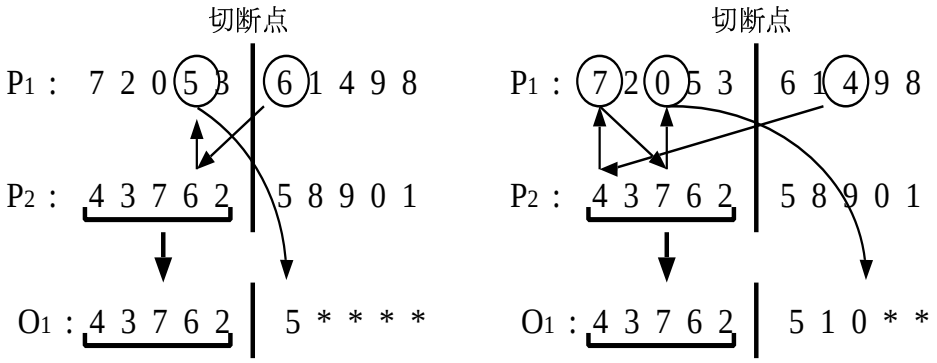


図 5.8　部分写像交叉

サイクル交叉

サイクル交叉は位置が一致する対立遺伝子からなる，2 つの親に共通する部分集合を含んでいる．まず親 (P_1) の 1 番目の遺伝子から始め，その遺伝子を子 (O_1) の 1 番目に写す．

次に子 (O_1) の 1 番目の遺伝子は割り当てられたため，親 (P_2) の 1 番目の遺伝子と同じ遺伝子を親 (P_1) から探す．見つかればその遺伝子を親 (P_1) の遺伝子の位置と同じ場所の子 (O_1) に写す．この操作を一番初めの遺伝子に戻るまで行う．次に親 (P_2) のサイクルに含まれていない場所から開始し同じように作業をする．これを全ての遺伝子が選択されるまで行う．この操作を図 5.9 に示す.

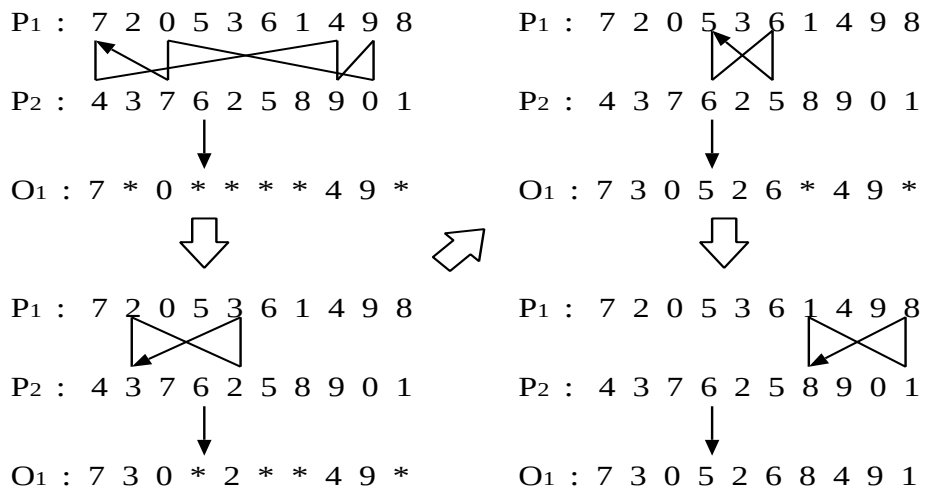


図 5.9　サイクル交叉

一様順序交叉

一様順序交叉はランダムな 2 進記号列のテンプレートを作成し，その配列を用いて交叉を行う．テンプレートの 1 がある場所は親 (P_1) から子 (O_1) へ，0 がある場所は親 (P_2) から子 (O_2) へと写される．次に子 (O_1，O_2) へと写されていない遺伝子集合については，親 (P_1) の遺伝子集合は親 (P_2) の，親 (P_2) の遺伝子集合は親 (P_1) の現れる順番に従い子 (O_1，O_2) の空いている場所に写される．この操作を図に表すと図 5.10 のようになる．なお，図 5.10 の 2 進記号列のテンプレートを T で表す.

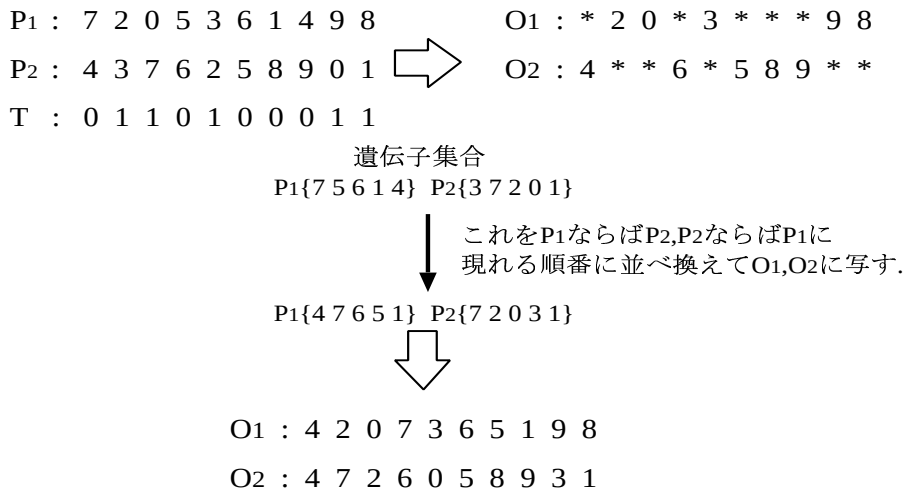


図 5.10　一様順序交叉

5.3.5　突然変異

突然変異にはランダムな 2 点を入れ替える，ランダムな 2 点間を逆順にする，ランダムな 2 点間をランダムに撹拌するものがある．ランダムな 2 点を決定した後内側，外側を操作対象にするかはプログラム実行中に決定される．これらの突然変異を図 5.11 に示す.

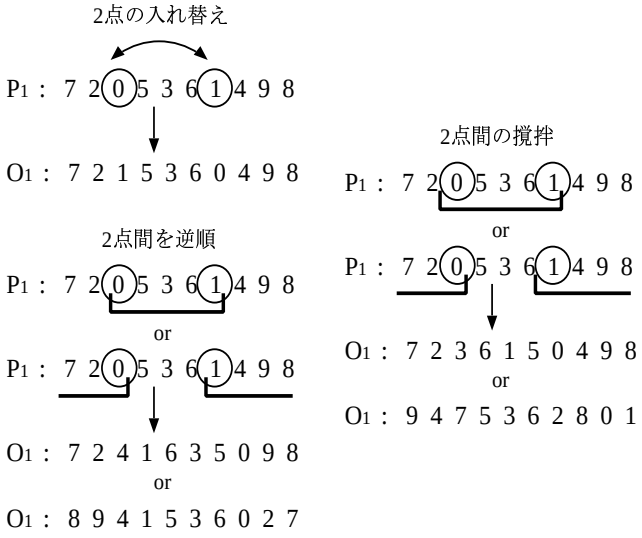


図 5.11　突然変異

5.3.6　終了条件

終了条件はあらかじめ評価値の未更新限度回数を設定しておき，評価値の未更新がその回数続くと終了するように設定している．

5.3.7　集合 2 分割問題に対する GA のアルゴリズム

集合 2 分割問題に対して遺伝的アルゴリズムを適応する際のアルゴリズムは以下のようになる．

1. 乱数を用いて初期染色体集団を生成する．
2. 適応度を計算する．
3. 適応度に応じた確率で親を選択する．
4. 交叉確率 P_c で，交叉を行う．
5. 突然変異確率 P_m で，突然変異を行う．
6. 終了条件を満たすまで 2〜5 を繰り返し行う．終了条件を満たせば最良解を提示し終了する．

終了条件は「50 世代の間，最良解が更新されない」とする．

5.4　遺伝的アルゴリズムの特徴

遺伝的アルゴリズムは，今回利用した欲張り法，逐次改善法のようなヒューリスティックアルゴリズムと同様に最適解は保証されないが，両親の選択方法，交叉方法によりいくつかの組合せができ，突然変異を行うことにより局所解に陥ることが少なく良い解を期待することができる．

また，1 世代の染色体数の増加や終了条件を厳しくすることで，計算時間は必要となるが最適解に近付く可能性が増える．この 1 世代の染色体数，終了条件は解の精度，計算時間に大きく関わり他の解法と比較する際に遺伝的アルゴリズムの重要なパラメータとなる．

第 6 章

実験結果と解法の比較と評価

6.1　実験と比較について

ヒューリスティックアルゴリズム，完全列挙法，遺伝的アルゴリズムの解法のそれぞれの実験を行い，最後に全ての解法を解の精度，計算時間で比較し評価を行う．なお実験データについては要素数 10，20，30，40，50，60，70，80，90，100 の 10 通りのデータを用いて実験を行う．

ヒューリスティックアルゴリズムについては欲張り法と逐次改善法の比較をし，3 章で述べたように逐次改善法では欲張り法よりも良い評価値を得ることを前提として実験しているためどの程度改善されているかを検証する．また，遺伝的アルゴリズムでは解の未更新限度回数を増やすことによって解の精度，計算時間がどのように変化するか検証する．

6.2　ヒューリスティックアルゴリズム

問題に対する欲張り法と逐次改善法の解の精度，計算時間を比較する．逐次改善法の k の値については $k = 3$，$k = 5$ の 2 通り行っており，改善率 $(\%)$ については欲張り法の評価値から逐次改善法の評価値では改善率分改善されたということを示している．改善率の計算方法は以下のようになる．

$$\begin{aligned} \text{改善率}\ (\%) &= (1 - \frac{\text{逐次改善法の評価値}}{\text{欲張り法の評価値}}) \times 100 \\ &= \frac{\text{欲張り法の評価値} - \text{逐次改善法の評価値}}{\text{欲張り法の評価値}} \times 100 \end{aligned}$$

実験結果を表 6.1 に示す．ただし時間が 0.0000 となっているものについては 0.0001 未満

であることを意味している．

表 6.1　問題に対するヒューリスティックアルゴリズムの評価値と計算時間 (s)

要素数	欲張り法		逐次改善法 ($k=3$)			逐次改善法 ($k=5$)		
	評価値	時間	評価値	改善率	時間	評価値	改善率	時間
10	42	0.0000	22	47.7	0.0036	22	47.7	0.0065
20	420	0.0000	4	99.0	0.0037	4	99.1	0.0132
30	250	0.0029	18	92.8	0.0041	2	99.2	0.0140
40	336	0.0030	2	99.4	0.0042	2	99.4	0.0143
50	314	0.0031	24	92.4	0.0045	2	94.4	0.0144
60	328	0.0031	6	98.2	0.0048	2	99.4	0.0145
70	228	0.0033	12	94.7	0.0051	2	99.1	0.0148
80	65	0.0034	11	83.1	0.0052	3	95.3	0.0155
90	130	0.0035	6	95.4	0.0054	2	98.5	0.0156
100	220	0.0037	6	97.3	0.0058	2	99.1	0.0157

6.2.1　考察

実験結果より欲張り法については逐次改善法よりも計算時間は早いが評価値は悪く，また与えられた問題によって解の精度にバラツキが見られる．また，逐次改善法については，改善率を見ると一定ではないが多くの場合で 90%を越える値を出している．このため逐次改善法は欲張り法で得た近似解をより最適解に確実に近付けることができるということが言える．また，k 値の比較実験について，$k=5$ は探索要素を $k=3$ に比べ増やしているため，$k=3$ のときよりも改善率も良くなっており解の精度が上がっている．$k=3$ については $k=5$ に比べ荒い探索になっているため，問題の大きさによってバラツキが見られる．

6.3　完全列挙法

完全列挙法に対して要素数 10，20，30，40，40 の 4 通りのデータを用いて実験を行った．なお 50 以上のデータについては膨大な時間が必要と考えられるため計測していない．表 6.2 に実験結果を示す．ただし時間が 0.00 となっているものについては 0.01 未満であることを意味している．

表 6.2　問題に対する完全列挙法の評価値と計算時間 (s)

要素数	評価値	時間
10	22	0.00
20	0	0.03
30	0	30.06
40	0	26782.52

6.3.1　考察

完全列挙法は 4.4 でも述べたように最適解が保証されているが表 6.2 からも分かるように，要素数が少なければ短時間で最適解を得ることができる．しかし，要素数が増えると計算時間が必要となっている．要素数 50 以上については膨大な計算時間が必要になるため実験を行うことができなかった．

6.4　遺伝的アルゴリズム

遺伝的アルゴリズムでは両親の選択 (2 通り)，交叉方法 (10 通り)，突然変異方法 (3 通り)，交叉確率は (9 通り)，突然変異確率 (9 通り) などによりいくつもの組合せができる．交叉確率，突然変異確率については 0.1〜0.9 の 9 通りである．その組合せ数は,

$$\text{両親選択} \times \text{交叉方法} \times \text{突然変異方法} \times \text{交叉確率} \times \text{突然変異確率}$$

以上 4860 通りの組合せが存在する．これを要素数 50 のデータで実験し最良解の一番良い値でパラメータ設定を行うつもりであったが，殆んどの場合で最良解が同じ値を得たため，乱数の種 (0〜9) を用いてそれぞれのパラメータ毎の 10 個の評価値の平均値で一番良い値のパラメータを全てのデータに適応した．採用したパラメータを以下に示す．

- 両親選択　：　ルーレット選択
- 交叉方法　：　一様順序交叉
- 突然変異方法　：　2 点間のランダムな撹拌
- 交叉確率　：　0.9
- 突然変異確率　：　0.5

なお解の未更新限度回数，染色体の個体数については両方とも 50 と設定している．遺伝的アルゴリズムの実験結果を以下の表 6.3 に示す．なお実験は解の未更新回数を 50 と 100 について行っている．表中の Avg については乱数の種を 0 から 9 に変化させたときの 10 回の評価値の平均値であり，最良解については 10 回の内の一番良い解である．時間については 10 回の総計である．

表 6.3　問題に対する遺伝的アルゴリズムの評価値と計算時間 (s)

要素数	50			100		
	最良解	Avg	時間	最良解	Avg	時間
10	22	22.0	0.405	22	22.0	0.850
20	0	5.4	0.930	0	4.4	1.839
30	0	3.0	1.206	0	1.8	2.434
40	0	4.2	2.045	0	5.6	4.107
50	0	7.6	2.743	0	3.6	5.853
60	2	5.8	4.782	0	1.8	6.552
70	4	9.4	4.885	0	5.4	7.483
80	1	10.8	5.742	1	4.4	10.043
90	2	7.6	6.985	2	7.6	13,342
100	0	3.0	7.317	0	5.6	13.693

6.4.1　考察

表 6.3 からも分かるように，全てのデータにおいて比較的短かい計算時間で良い結果を得ることができている．また，解の未更新限度回数を増やすことによって計算時間は増えるが評価値は平均値にしても最良解にしても良い評価値が得られた.

6.5　それぞれの解法の比較と評価

以下の表 6.4 はそれぞれの解法の実験結果である．逐次改善法については $k = 5$ の場合である．また，遺伝的アルゴリズムについては解の未更新限度回数が 50 の場合である．Avg については乱数の種を 0〜9 までにしたときの 10 回の評価値の平均である．最良解については 10 回の内の一番良い解で時間は 10 回の統計である．逐次改善法の改善率の単位は%である．ただし計算時間が 0.00 になっているものについては 0.01 未満であることを意味している．

表 6.4　問題に対する各解法の評価値と計算時間 (s)

要素数	欲張り法		逐次改善法			完全列挙法		遺伝的アルゴリズム		
	評価値	時間	評価値	改善率	時間	評価値	時間	最良解	Avg	時間
10	42	0.00	22	47.7	0.006	22	0.000	22	22.0	0.405
20	420	0.00	4	99.1	0.013	0	0.038	0	5.4	0.930
30	250	0.02	2	99.2	0.014	0	30.069	0	3.0	1.206
40	336	0.03	2	99.5	0.014	0	26782.527	0	4.2	2.045
50	314	0.03	2	99.4	0.014	-	-	0	7.6	2.743
60	328	0.03	2	99.4	0.014	-	-	2	5.8	4.782
70	228	0.03	2	99.2	0.014	-	-	4	9.4	4.885
80	65	0.03	3	95.4	0.015	-	-	1	10.8	5.742
90	130	0.03	2	98.5	0.015	-	-	2	7.6	6.985
100	220	0.03	2	99.1	0.015	-	-	0	3.0	7.317

6.5.1　考察

今回，様々な大きさのデータに対して実験を行った結果，小さな問題に対しては完全列挙法が有効であった．しかし，大きな問題になるにしたがって膨大な計算時間が必要となって

いる．欲張り法や逐次改善法などのヒューリスティックアルゴリズムに関してはデータによって解の精度は異なるが小さな問題に対しては有効であった．また，逐次改善法の解の精度という点では欲張り法の評価値からの改善率は高く，完全列挙法よりは評価値の精度は低いが最適解に非常に近い値を得ることができている．遺伝的アルゴリズムについては小さな問題，大きな問題に関係なく，ヒューリスティックアルゴリズムより計算時間は必要となるが解の精度は良く，また，完全列挙法と比較すると計算時間は短く，更に最良解も完全列挙法と同じように最適な解を得ることができている．

第 7 章

結論

本研究では，集合 2 分割問題に対して最も単純なヒューリスティックアルゴリズムである欲張り法，欲張り法の解を初期解として少しずつ良い解に改善していく逐次改善法，計算時間は指数関数的に増大するが最適解を保証する完全列挙法，最適化手法の一つとして有効であると言われている遺伝的アルゴリズムを用いて解の精度，計算時間で比較した．

表 6.4 の実験結果より完全列挙法よりも高速にかつ，欲張り法や逐次改善法よりも精度の高い解を得ることができた遺伝的アルゴリズムを利用した解法が，この集合 2 分割問題に対して有効であることを確認することができた．

遺伝的アルゴリズムが他の解法よりも有効である要因として遺伝的アルゴリズムの特性があると考えられる．まず両親の選択，交叉方法，突然変異方法や各種確率などの組合せによって深く探索できるということである．次に突然変異により局所解に陥ることが少ないという特性が挙げられる．この特性によりヒューリスティック解法よりも解の精度が高く，また完全列挙法よりも計算時間が早く最適解に非常に近い近似解を得ることができるということを確認できた．

今回は，逐次改善法での実験で k の値を 3，5 にしたときの比較をした．その結果，探索要素を増やした $k = 5$ は $k = 3$ のときより改善率は良くなるが計算時間は増加するという結果を得た．このことより k と計算時間の間にはトレード・オフの関係が成り立つことを確認した．

また，本研究では与えられた集合を分割する際に部分集合の要素の数を同じ数にするということを前提とし実験を行ったが，今後，多様な工学的諸問題に適応するためには要素数を固定しないということを今後の課題として挙げる．

謝辞

本論文は，著者が 2001 年 7 月から 2003 年 2 月までの高知工科大学工学部情報システム工学科在学中に，同学科坂本研究室において行った研究活動の成果を記したものである．

はじめに，著者の大学生活が有意義に楽しく送ることができたことに感謝致します．

GA からプログラム，本論文を書き上げるまで，多忙の中，御指導頂き，また，就職活動にもアドバイスを頂いた坂本明雄教授に深く感謝致します．

常に余裕を持ち，様々な相談にのってもらい，励ましの言葉，アドバイスを下さった登伸一さん，忙しい中研究活動についてアドバイスを下さった友池貴之さん，助けて頂きたい時にはいつもいなく，飲み会の席ではいつもの顔を見せて皆を癒してくれた折橋祐一さんに深く感謝致します．

そして，同研究室の仲間として友に励まし合い，時には友人として相談にものってもらい，アドバイスをもらった赤間寛くん，河野兼祐くん，河内友彦くんに感謝致します．

また，福本研では愛の鞭を下さった福本昌弘先生に感謝致します．

研究室を賑やかにしてくれた 3 年生諸君に感謝致します．これからの就職活動，研究活動頑張って下さい．

最後に，著者が本大学入学時から今まで過ごしやすい環境を整えて頂いた情報システム工学科の先生方，また支えて下さった全ての方々に深く感謝致します．

参考文献

[1] 白石洋一訳, 組合せ最適化アルゴリズムの最新手法, 丸善, 2002

[2] 社団法人 電気学会，遺伝アルゴリズムとニューラルネット-スケジューリングと組合せ最適化-，コロナ社，1998

[3] 北野宏明編，遺伝的アルゴリズム，産業図書，1993

[4] 萩原将文著，ニューロ・ファジィ・遺伝的アルゴリズム，産業図書，1994