平成 13 年度

学士学位論文

最大利得部分木問題に対する遺伝的アプローチ

A Genetic Approach for Maximum Profitable Subtree Problem

1041016　　栃木 隆道

指導教員　　坂本 明雄

2002 年 2 月 8 日

高知工科大学 情報システム工学科

要 旨

最大利得部分木問題に対する
遺伝的アプローチ

栃木 隆道

１点が根として指定されている連結無向グラフにおいて，各枝にはコスト，各頂点には利得が与えられていて，さらに総コスト上限が定められているとする．最大利得部分木問題（Maximum Profitable Subtree Problem: MPSP）とは，枝のコストの和が総コスト上限以下である根を含む連結部分木での中で，頂点の利得の和が最大となるものを求める問題である．

本論文では，この問題に対して遺伝的アルゴリズム（Genetic Algorithm: GA）を用い，厳密解法である分枝限定法（Branch and Bound method: B&B）と比較して実行時間を，近似解法であるヒューリスティックアルゴリズム（Heuristic Algorithm: HA）と比較して解の精度を，それぞれ検証し考察する．

キーワード　最大利得部分木問題，遺伝的アルゴリズム，ヒューリスティックアルゴリズム，分枝限定法

Abstract

A Genetic Approach for Maximum Profitable Subtree Problem

Takamasa TOCHIGI

Suppose we are given a graph that is undirected and connected. In this graph, one vertex is specified as a root. A cost and a profit are also given to each corresponding edge and vertex, respectively. Then, maximum profitable subtree problem is a problem of finding a subtree that maximizes total profits within a prescribed amount of total cost. Of course, the resultant subtree must be connected and include the root.

In this paper, we describe an implementation of genetic algorithm for the maximum profitable subtree problem. The fastness of execution time is verified compared with a branch and bound method, and the accuracy of solutions is verified compared with a heuristic algorithm.

key words maximum profitable subtree problem, genetic algorithm, heuristic algorithm, branch and bound method

目次

図目次

図目次

表目次

第 1 章

はじめに

限られた予算以内で，なるべく大きな都市をより多く結ぶ高速道路網を構築する場合や，できるだけ多くの端末を低コストで繋ぐ通信ネットワークの構築をする場合など，あるコスト以内で最大の利得を得たいとき，最大利得部分木問題 (Maximum Profitable Subtree Problem: MPSP) が応用できる．

最大利得部分木問題とは，連結無向グラフにおいて，1 点が根として指定されおり，各枝にはコスト，各頂点には利得が与えられていて，さらに総コスト上限が定められているとき，根を含む連結部分木で，枝のコストの和が総コスト上限以下で，頂点の利得の和が最大となるものを求める問題である．

この問題は多項式時間では解くことができないとされている NP 困難な問題に属する．

最大利得部分木問題は，基本的な問題で応用も豊富と考えられるにも関わらず，この問題に対しての研究事例はあまり見つからない．

一方，最適化手法の一つである遺伝的アルゴリズム (Genetic Algorithm: GA) とは，生物集団の進化と淘汰を模倣したアルゴリズムで，世代を重ねるに従い，遺伝子が交叉と突然変異を繰り返しながら進化していき，より環境に適した良質な個体を解とする手法である．

本研究では，最大利得部分木問題に対して遺伝的アルゴリズムを用いて解く方法を提案し，厳密解法である分枝限定法 (Branch and Bound method: B&B) と比較して実行時間を，近似解法であるヒューリスティックアルゴリズム (Heuristic Algorithm: HA) と比較して解の精度を，それぞれ検証する．

本論文の構成は以下の通りである．2 章で最大利得部分木問題の詳細を説明し，本研究における解の探索手法を説明する．以降，3 章ではヒューリスティックアルゴリズムによる解

法の説明，4 章では分枝限定法による解法の説明をする．5 章では遺伝的アルゴリズムの概要を説明し，6 章では遺伝的アルゴリズムを用いた解法について説明する．そして 7 章では実験の結果および比較と考察をする．8 章は結論である．

第 2 章

最大利得部分木問題

2.1 最大利得部分木問題の定義

頂点集合 V，枝集合 E を持つ連結無向グラフ $G = (V, E)$ において，根 $r(\in V)$ が指定されているとする．各頂点には正の値を持つ利得 $p_i(i \in V)$，各枝には同じく正の値を持つコスト $c_{ij}((i, j) \in E)$，さらに総コスト上限 C が与えられているとする．このとき，最大利得部分木問題とは，根 r を含む連結部分木の中で，枝のコストの総和が C 以下であり，木に含まれる頂点の利得の総和が最大になるようなものを求める問題である [1]．図 2.1 に最適解の例を示す.

類似した問題にスコアオリエンテーリング問題があるが，その問題は部分木の代わりに部分巡回路を求める問題である.

総コストの上限を与えない (上限を無限大にする) 場合は，スコアオリエンテーリング問題では，巡回セールスマン問題の決定問題と等価になり，NP 完全な問題になるのに対して，最大利得部分木問題では，全ての頂点を含む最小木を求める問題になり，非常に簡単に解くことができる．また，全ての頂点を含むことができない場合は，図 2.2 のように全ての枝が根に接続されている場合を考えると，ナップザック問題に帰着できるため，この問題は NP 困難な問題であることが証明される.

2.1　最大利得部分木問題の定義

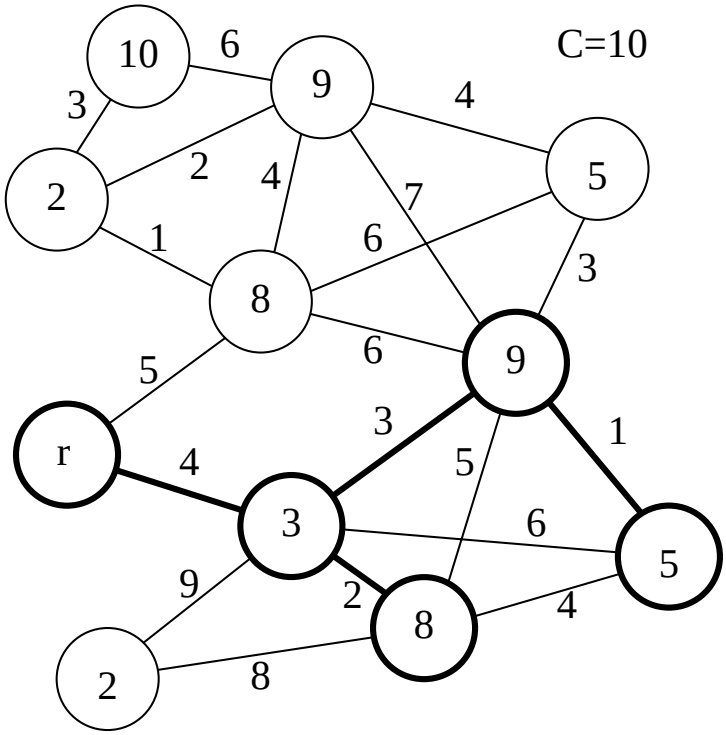

図 2.1　最大利得部分木問題の最適解

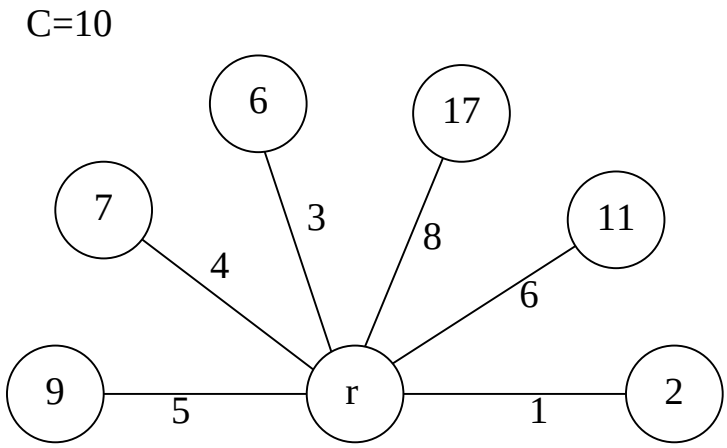

図 2.2　ナップザック問題への帰着

2.2　MST 関数

この最大利得部分木問題においては，連結性の保たれた頂点集合から誘導される枝集合は，そのグラフの最小木を構成するものが最も総コストが少ない．

そこで，本研究では，入力した頂点集合から最小木を構成する枝集合を返す関数，MST(Minimum Spanning Tree) 関数を使用する．

MST 関数は，入力された頂点集合から構成される最小木の枝集合を返すが，最小木を連結できない場合は全枝集合を返す．

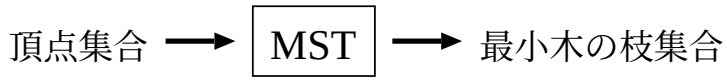

図 2.3　MST 関数の入出力

つまり，頂点集合を与えるとその頂点集合に対する最適な枝集合は MST 関数により決められるので，本研究では最大利得部分木問題の解を，最適な頂点集合を探索する方法をとる．

2.3　問題の作成

本研究において，実験に使う問題は乱数により自動で作成した．入力パラメータは，頂点数 V，枝数 E，総コスト上限 C である．頂点は一辺が $2V$ の正方形の格子にランダムで配置し，頂点の利得は乱数により 1 から 20 までの整数を与える．枝のコストは頂点間の距離の整数部+1 にした．枝は連結性を保つ必要があるため，最小木を選択し残りはコストの小さいものから順に選択する．

2.4　問題の特徴

最大利得部分木問題は，頂点数，枝数，総コスト上限によって様々な特徴を持つ問題になる．

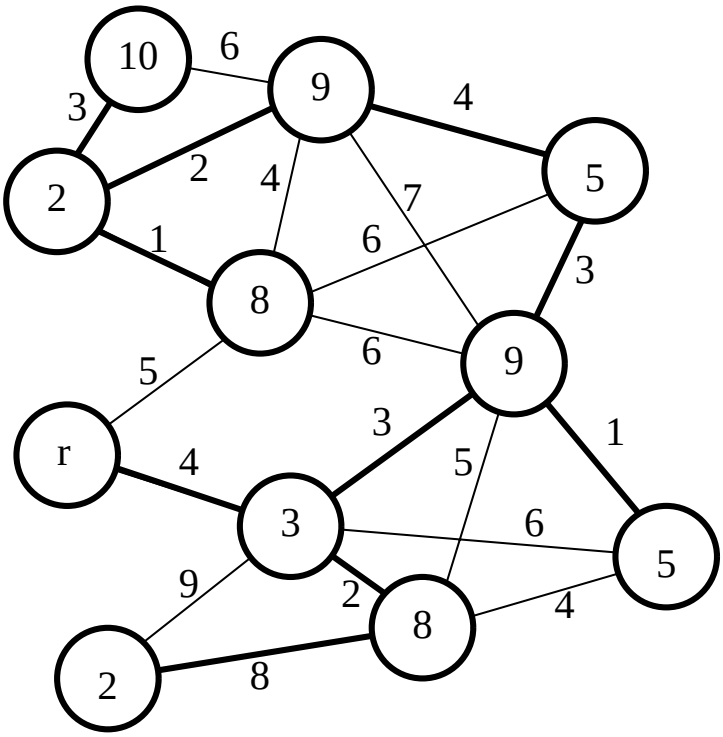

図 2.4　最小木の例

頂点数は問題全体の大きさに関係する．少なければ解に含む頂点の組合せが少ないので探索範囲は狭くなり，頂点数が多ければ組合せパターンも多くなる．

枝数は，頂点集合から構成される最小木の探索範囲に影響する．例えば，頂点数が少なくてもそれらを連結する枝候補が多い場合は，最小木を求める上で時間がかかる．逆に枝が少なければ，短時間で最小木を求められる．

総コスト上限は解に含まれる頂点数に影響する．上限が大きいときには多くの頂点を含むことになり，上限が小さければ，少ない頂点集合になる．よって，一般的に上限が大きい程，解候補の頂点集合は多くなる．

また，総コスト上限が非常に小さい場合，いくら頂点数が多かったとしても，根から近い範囲のみしか探索しなくてよいので，可能解の探索範囲は狭くなることもある．ところが，上限が無限大ならば，全ての頂点を連結する最小木問題になるので，非常に簡単になる．

第 3 章

ヒューリスティックアルゴリズムによる解法

人間が探索していく方法に比較的近い探索方法をとるヒューリスティックアルゴリズムでは，短時間で近似解を得ることができる．

3.1　ヒューリスティックアルゴリズムの概要

最大利得部分木問題に対するヒューリスティックアルゴリズムは，根のみの木から始めて，総コスト上限を超えないように一つずつその段階で最も良いと思われる頂点を取り込んで木を拡張していく戦略をとる．

3.2　取り込む頂点の選択方法

この問題では，新たな頂点を一つ取り込むと必ず利得は増加するが，ここで最小木を構成し直すと，図 3.1 の例のようにコストは逆に減少する場合もある．

従って，得られる利得が同じ場合は，総コストが少ない頂点を優先的に取り込むのが良い．

そのことを考慮して，取り込むべき頂点は，取り込み候補の頂点全てに対して，取り込まれた後の最小木の総コストと総利得の割合が以下のように，最も高くなる頂点を選択する．

$$\frac{\text{総利得}}{\text{総コスト}} \quad \rightarrow \quad max$$

つまり，取り込んだ後の最小木において，コストが少なく利得が多くなるような頂点を選

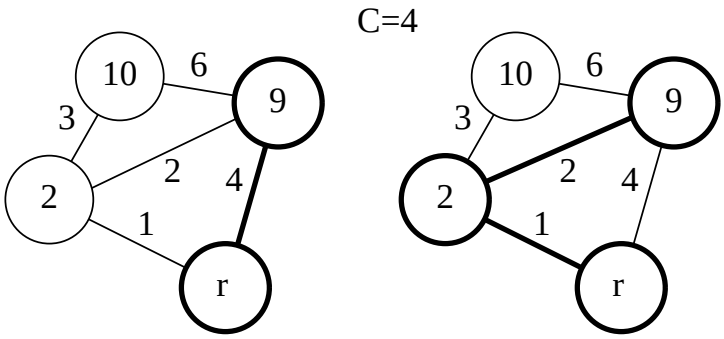

図 3.1　頂点を加えることでコストの総和が減少する場合

択することになる．ただし，ここでコストが総コスト上限を超えてしまうものは選択できない．

こうして，取り込める頂点がなくるまで拡張してできた部分木を解とする．

3.3　ヒューリスティックアルゴリズムの特徴

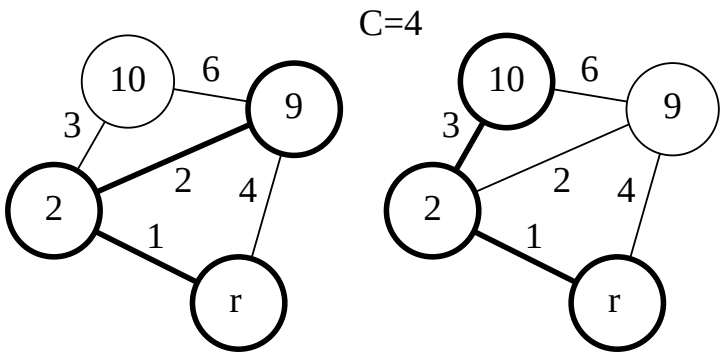

図 3.2　最適解を求めることができない場合

ヒューリスティックアルゴリズムは，比較的短時間に解を出すことができるが，図 3.2 のような例題では，左側の図のような利得 11 という解で終了してしまう．この例題では右側の図に示すように利得 12 という最適解が存在することは明らかである．

3.3　ヒューリスティックアルゴリズムの特徴

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

第 4 章

分枝限定法による解法

最大利得部分木問題は NP 困難な問題なので，多項式時間で解けないと考えられる．従って厳密に解くには全パターンを当たればよいが，一般的に問題の探索範囲が膨大になるに伴い，その探索時間も非常に膨大になる．

そこで，ある程度探索不要な部分を発見し，問題を緩和させることで時間を減らす厳密解法を考える．

本問題に対する分枝限定法は，根のみの木から始めて，連結可能な頂点全てに対して，その頂点以下の最大利得部分木問題に分散させ，以下，再帰的に分散し，小さな問題を厳密に解くことで全体の問題を解く．

4.1 分枝限定法による解探索の概要

まず，根のみの木から連結可能な頂点群を挙げ，各頂点数以下の最大利得部分木問題を解く問題に分ける．同様にして，分けられた最大利得部分木問題を更に分割する．これを繰返し，問題を細かく分散させ，その全ての問題に対し厳密解を得ることで全体の厳密解を得る．

図 4.1 の上側の例題では，元の問題を，下側のサイズの小さい二つの問題に分割することになる．この処理を繰り返すことで，簡単な問題に分割してそれぞれを解くことで，元の問題を厳密に解く．

次節では入力された頂点集合以下の最大利得部分木問題を解くための再帰的な BGH 関数について考案する．

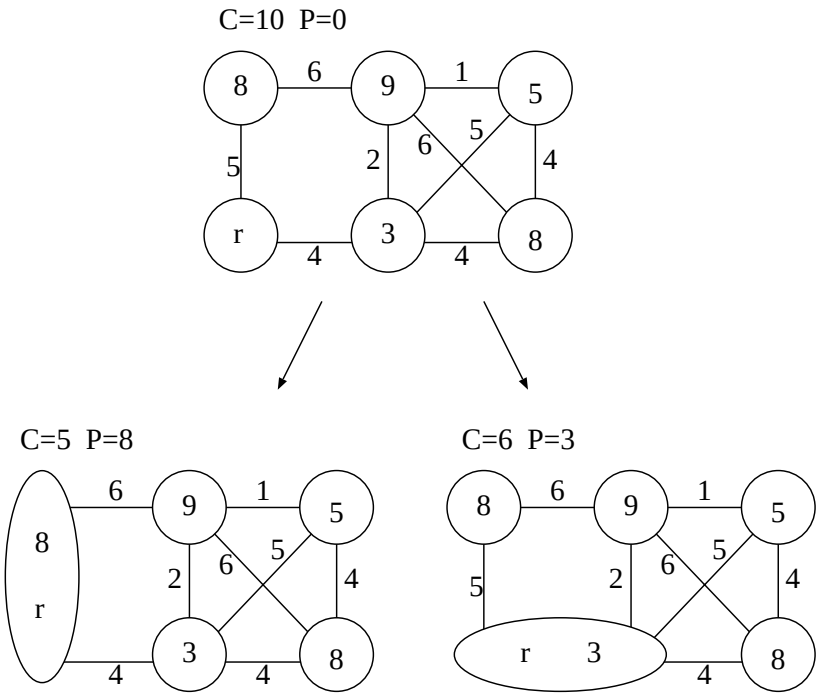

図 4.1　問題を二つに分散する

4.2　BGH 関数

BGH 関数は，以下に示すような処理を行なう．

1. 入力された頂点集合に連結可能な頂点を探す．
2. 連結可能な頂点が一つも無い場合，入力された頂点集合をそのまま返す．
3. 連結可能な頂点がある場合，それぞれの連結可能な頂点において，その連結可能頂点を入力された頂点集合に付加し，BGH 関数に入れて返ってきた頂点集合の総利得が最大のものを返す．

ここで連結可能な頂点とは，入力された頂点集合と枝で連結している頂点で，かつ，その新たな頂点を加えて最小木を構成したとき，総コストが上限を超えていないものを指す．

なお，別の分枝頂点で同じ頂点集合をもう一度探索しようとしている場合は，その先の探索を省略するようにした．図 4.2 に示すように，左側で探索終了し，右側から同じ問題がで

きた場合，一方のみを探索すればよいので，探索を省略する．こうすることで，探索範囲を大幅に減少できる．

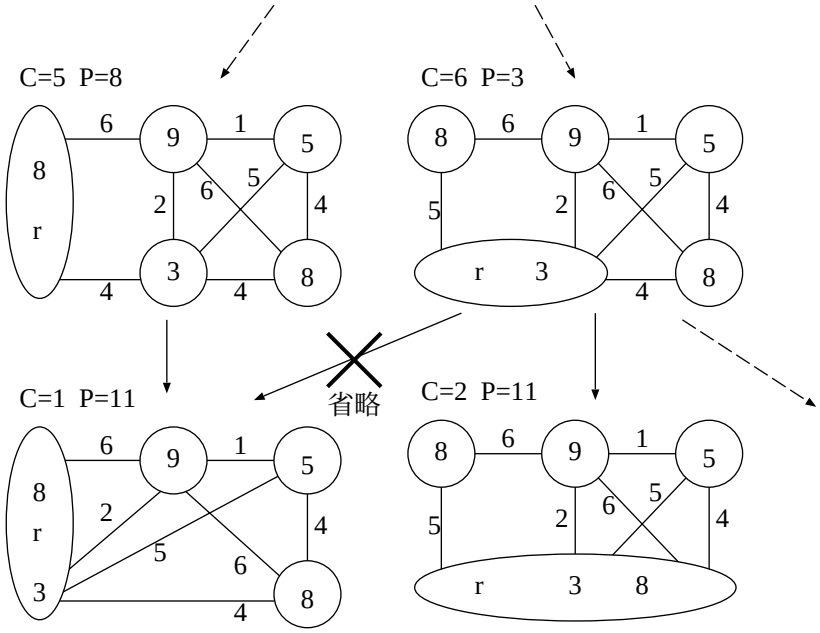

図 4.2　重なった問題の探索不要な例

4.3　分枝限定法の特徴

全頂点パターンを探索するにあたり，連結可能な頂点のみを探索すれば重複することが無いので，比較的探索時間が早く厳密な解を求めることができるが，問題が大きくなると，やはり多くの探索時間が必要となる．

特に，枝の数が多かったり総コスト上限が高く設定されている問題は，連結パターン，連結頂点の量が大きくなる場合に探索時間を多く費やすと考えられる．

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は，生物集団の進化と淘汰の過程を模倣した最適化手法で，各個体の染色体が交叉と突然変異を繰り返しながら世代を重ねるに従って，環境にそぐわないものは消えて，より環境に適した個体が生み出されていく過程を模倣したアルゴリズムである．

自然界では，環境に適した能力を持った生物が生き残り，次の世代にその遺伝子を伝えていくと考えられる．つまり，生き残って繁殖している生物は環境に適した良質な遺伝子を持っているといことになる．

こうした過程を計算機上で行なうことで，問題に対してより良い解を持つ個体を求める手法である．

5.2　遺伝的アルゴリズムの流れ

計算機上で遺伝的アルゴリズムを行なうにあたって，問題のコーディングをして染色体の形を決め，環境への適応度の算出式を決める．遺伝的アルゴリズムの大まかな流れは図 5.1 に示すようになる．

各項目の詳細は以下に示す．

1. 乱数を使用して様々な遺伝子を持った個体の初期集団を生成する．
2. 各個体に対して，その個体が持つ染色体情報から，環境に適応している度合を表現する適応度を算出する．

5.2　遺伝的アルゴリズムの流れ

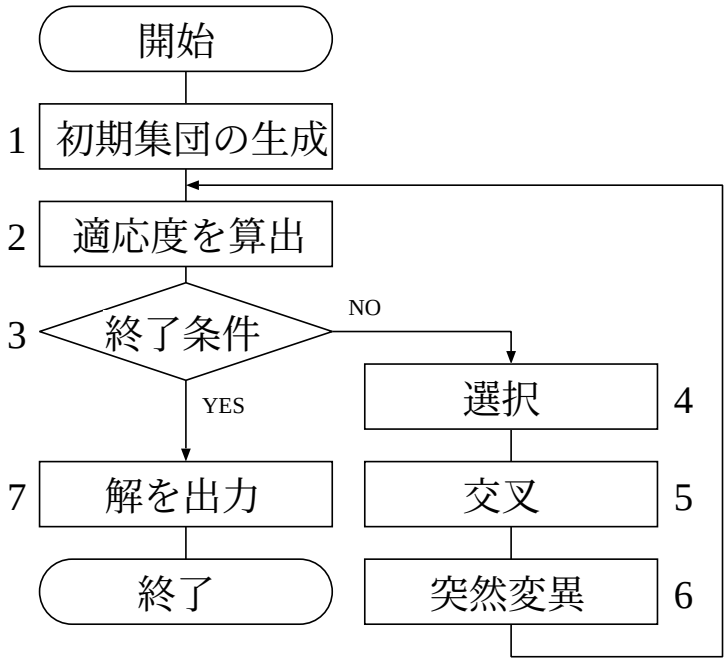

図 5.1　遺伝的アルゴリズムの流れ

3. 終了条件を満たしてない場合は 4，5，6 を繰り返す．満たした場合は 7 へ．
4. 適応度に応じて，次の世代に残す個体を選択する．適応度の低いものは選択されにくいので，消滅しやすい．
5. 選択された染色体の遺伝子を乱数を利用して交叉し，新たな集団を作る．交叉とは，二つの個体の遺伝子の一部を交換して，新たな二つの遺伝子を持つ染色体を作ることである．
6. 各個体の遺伝子に乱数を利用し突然変異を起こす．突然変異とは，ある遺伝子座の一部を，別のものに変化させることである．
7. 生まれた個体の中で最も適応度が高いものを解とする．

第 6 章

遺伝的アルゴリズムによる解法

6.1　コーディングと適応度

先に述べたように，本研究における最大利得部分木問題の解法は，頂点集合から求めるものである．枝情報は，頂点集合から構成される最小木を求める．

従って，遺伝的アルゴリズムを最大利得部分木問題に適用するためには，頂点の情報があればよいので，染色体の各遺伝子座には各頂点番号を対応させ，1 か 0 の情報でその頂点を解に含むかを否か決めるコーディング方法を用いた．染色体は一次元配列で，長さは問題の総頂点数である．染色体は全頂点の情報を持っていることになる.（図 6.1 参照）

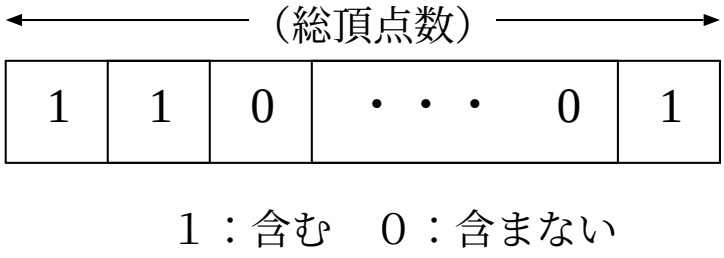

図 6.1　染色体のコーディング

解に含む頂点集合が決まると，その頂点を全て含む最小木を MST 関数で求めることで，その頂点集合における最良の枝集合を求める．

なお，この問題は，解の総コストが上限を超えていなければ評価は利得のみに依存するので，適応度はコストを考慮しないように以下のようにした．

$$\text{適応度} = \frac{\text{部分木の総利得}}{\text{問題中の全頂点の総利得}}$$

ただし，同じ適応度を持つ個体に対しては，コストが低い方を優先して最適解候補にする．

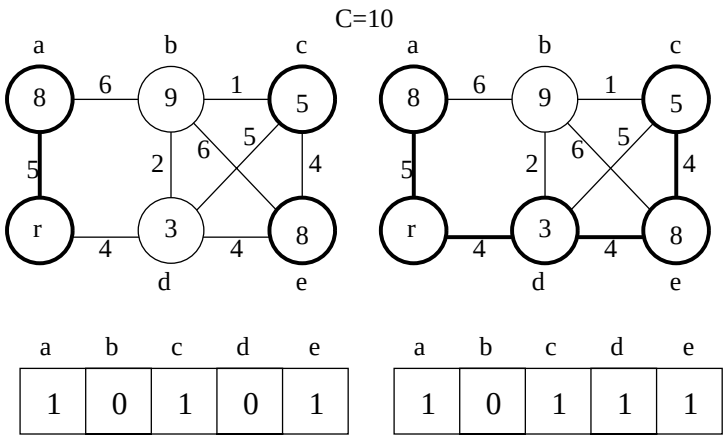

図 6.2　最小木を構成できない例

構成された最小木の総コストが上限を超えた場合や，連結できない頂点がある場合などは解にはならない．図 6.2 の左側の染色体の場合では a，c，d の頂点を含むことになるが，c と d の頂点には枝が無いため連結できない．右側の染色体の場合は，全て連結できているが，総コストが上限を上回っている．

これらを致死遺伝子と呼び，適応度は 0 とされ，次の世代には子を残せないことが確定となる．

6.1.1　MSTC 関数

前述したような MST 関数で最小木を求めると，問題中の枝の総数が多くない限り，乱数で決められる頂点集合では，連結性を保たれている木を構成することができないものが生まれやすいと考えられる．また，総コスト上限が高く設定されていない問題の場合では，最小木が構成できたとしても，総コスト上限を超えてしまうものが生まれやすいと考えられる．

そういった最小木を構成できない致死遺伝子は，次の世代に残ることはない．多くの染色体が致死遺伝子となってしまうと，交叉そのものができなくなり，世代を重ねること自体ができない．

そこで，致死遺伝子を出さないために，連結できる頂点集合のみを考慮し，根からコスト上限を超えない範囲で，低いコストで連結できる頂点を優先して取り込んで最小木を構成していくように，MST 関数を緩めた条件付き MST，MSTC(MST-Conditioned) 関数を考案した．

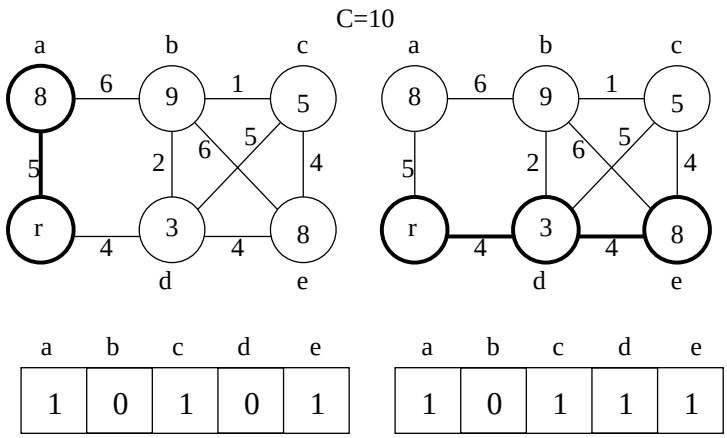

図 6.3　MSTC 関数での最小木

MST 関数では致死遺伝子となっていた図 6.2 の両染色体を，MSTC 関数では図 6.3 のように，左側の染色体では，枝が無く連結ができなかった頂点は無視し，右側の染色体では，コストの低い枝を優先的に取り込み，上限を超えた以降の枝を無視し，頂点の取り込みを終了している．両染色体は取り込むべき頂点の情報の一部を無視してはいるが，致死遺伝子ではなくなっている．

MSTC 関数で最小木を構成すれば，連結性を保てない頂点集合も連結している部分のみを考慮して最小木を構成し，多くの頂点を含んだ頂点集合でも総コスト上限を越す直前までに構成された最小木を出すことができるので，致死遺伝子は含まなくなる．

6.1.2　頂点の重要度

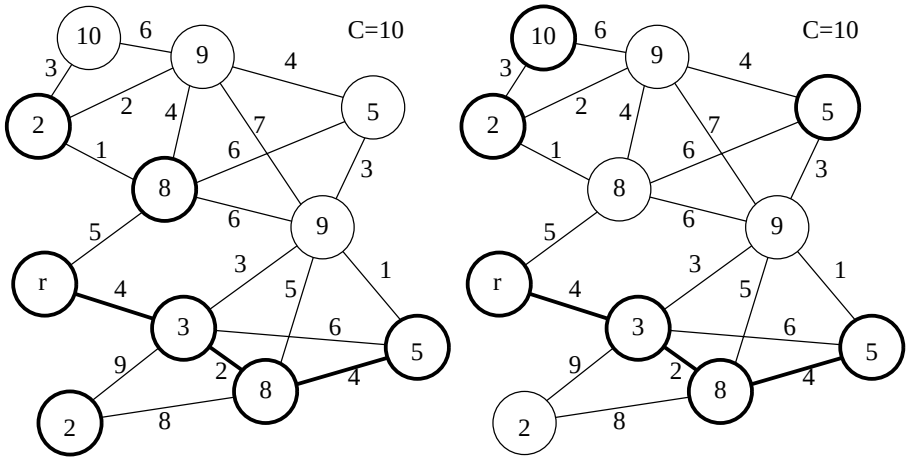

図 6.4　同じ木を構成する二つの頂点集合の例

前述したように MSTC 関数を使うと，図 6.4 の例に示すように，様々な頂点集合が同じ木に丸められて構成されると考えられる．これはすなわち，丸められる木を左右する鍵となる頂点があると思われる．つまり，各頂点により重要度が異なると考えられる．

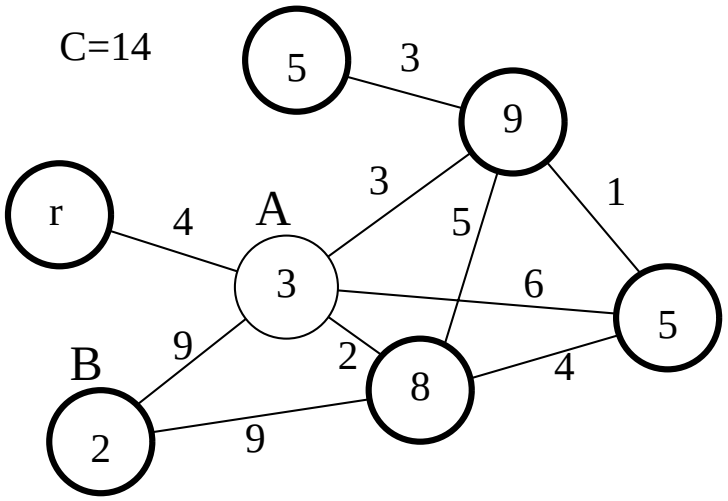

図 6.5　頂点の重要度

例えば図 6.5 に示すようにある部分を連結する枝が一つしかない場合などは，染色体がその枝に連結している頂点 A を含んでいない場合，根から頂点 A より離れたところにある頂点群は一切含まれることがない．こういった場合，頂点 A にはその先に接続されている頂点群の重みも含まれるため，頂点 A の重要度は高いと言える．

逆に，図 6.5 の頂点 B を選択するには，頂点 A に接続される他の頂点すべてを含んでいないときで，頂点 A と頂点 B を含んでいるときのみに限る場合など，頂点 B のようにコストが高すぎて選択されにくい上，利得が低いため選択しても適応度の上昇に結び付くことの無い頂点は重要度が低いと言える．こういった頂点に関しては，実質上，考慮すべき頂点から除外できると考えると，問題が縮小したと考えられる．

しかし，こういった選択されにくい頂点を含んだものが最適解であったりする場合も考えられる．図 6.6 のような場合，問題は非常に難解であると言える．

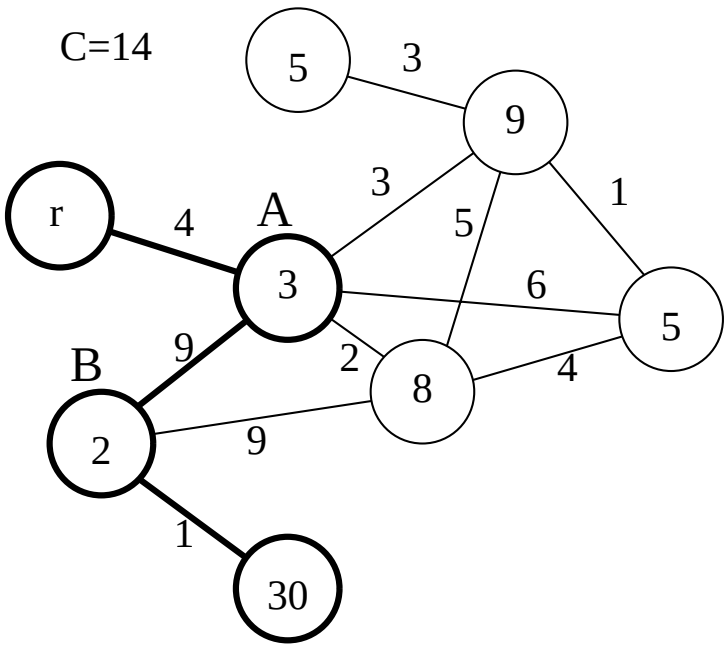

図 6.6　難解な問題の最適解の例

6.2　選択方法

選択方法はルーレット戦略を用いた．

ルーレット戦略とは，ルーレットを回し，一定の確率で一つが選ばれるように，ランダムに個体を選択する方法であるが，その選択される確率は，各個体の持つ適応度に比例するように決められる．

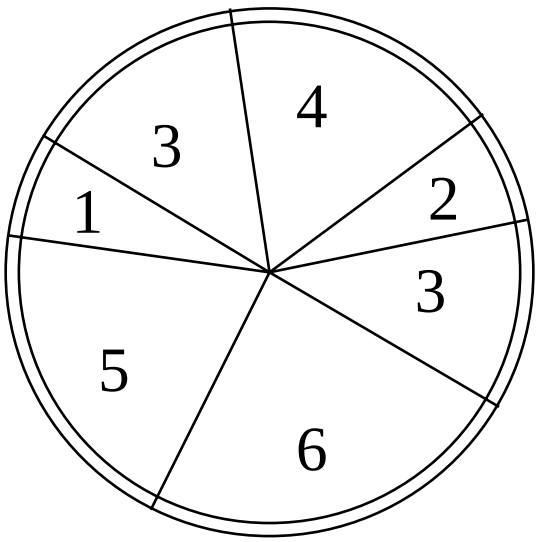

図 6.7　ルーレット戦略

適応度が低い個体は選択されにくく，適応度が高い個体は選択されやすい．

6.3　交叉方法

交叉方法は，一様交叉の方式を用いた．

一様交叉とは，二つの染色体の同じ遺伝子座の遺伝子を，ランダムで交換するか否かを決める方式である．

本問題におけるコーディングでは，遺伝子の並びに規則性は無いので，一様にランダムで交換するようにした．しかし，一様に半分の確率で交換すると，全ての個体において，約半

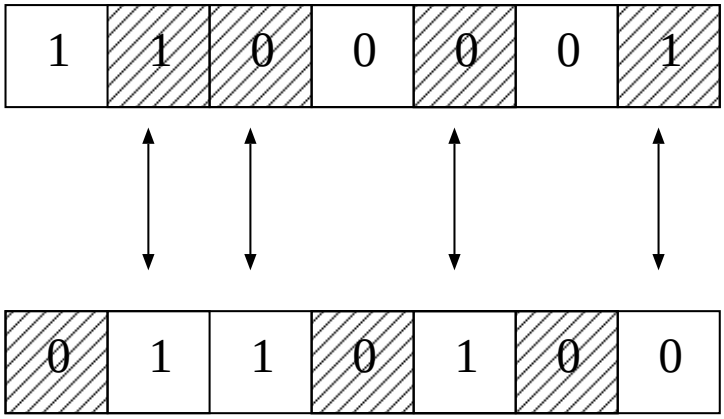

図 6.8　一様交叉

分の情報が入れ替わるので，元の個体が持つ特徴を大きく破壊してしまいかねない．

そこで，大体，遺伝子の中でどれくらいの割合を交換するかを，ランダムで決めることにした．こうすることで，遺伝子の殆んど交換しないものや，半分程度を交換するものなど，多様性が増加すると考えられる．

6.4　突然変異

突然変異は，変異させる遺伝子座をランダムに決め，そのビットを反転するという単純なものにした．

6.5　終了条件

終了条件は，最良の解が更新されない世代数が連続してある世代数まで続いたとき，または，適応度が 1 になるような全頂点を含む最適解が発見されたとき終了する．

6.6　上界値を利用した遺伝的アルゴリズムへ

6.6.1　染色体中の 1 と 0 の割合

一般的な方法で，初期集団の各遺伝子座が 1 か 0 に決まるのは，乱数を用いて 50 %でランダムに決められる．すると，当然遺伝子中の 1 と 0 の割合は，大体同じくらいになる．これは，大体半分の頂点を含むということである．

以降，交叉により 1 の多い個体が生まれて生き延びたとしても，突然変異によって 1 と 0 の割合は均等化へと向けられると考えられる．なぜなら，遺伝子中の 1 の割合が 70 %の個体が生まれたとして，突然変異が起こる確率を 10 %としたとき，変異するビットの 70 %は 1 から 0 に変化し，30 %は 0 から 1 に変化する．従って，全ビット中 10 %の 1 と 0 の割合が反転するので，$(7-3=4)$ %の 1 が 0 に変わり，変異後の 1 の割合は約 66 %に減少すると思われる．つまり，これが繰り返されると，次第に 1 と 0 の比率は等比に近づいていくことになる．

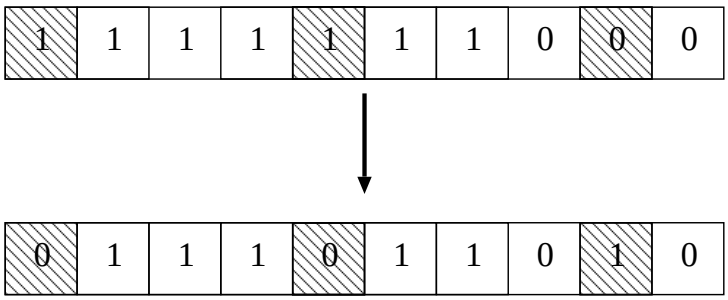

図 6.9　突然変異による 1 と 0 の割合の変化

図 6.9 の上の染色体の 1 の割合は 70 %であるが，30 %の遺伝子に突然変異を起こさせると，下の染色体のように 1 の割合は 60 %になっている．1 の方が多いため，変異する確率も上がってしまうからである．

従って，世代を重ねても全頂点中，半分の頂点を含むという個体が生まれやすいという結果になると考えられる．

6.6.2　1 と 0 の割合の初期設定

最大利得部分木問題では，最適解の頂点数は全頂点数の内どれくらいの割合であるか未知である．もし，最適解が全頂点中の殆どの頂点を含めるとしたとき，この遺伝的アルゴリズムでは半分の頂点を含める解が生まれやすいため，最適解に近づくには時間がかかると思われる．

また，MSTC 関数の性質上，遺伝子の頂点情報の全てが含まれるわけではなく，幾つかの含まれるべき頂点は連結されていない場合やコストが高く選択されない場合が考えられるので，更に取り込む頂点数が減少すると考えられる．

では，意図的に初期集団の個体の遺伝子を殆どの頂点を含むように設定したとするとどうだろうか．最適解が全頂点中のわずかしか含まれない場合で，困難な局所解がある問題などに対して，前述した頂点の重要度によって選択されにくい頂点はより選択されにくくなるため，その局所解から抜けにくいと考えられる．

そこで，ちょうど良い割合の頂点情報を持った個体の初期集団を生成するため，最適解に含まれる頂点の割合の上界値を求める．初期集団の各個体が持つ 1 の割合を，上界値に合わせて調節し，世代を重ねる過程においても，突然変異による 1 の割合の緩和に対して上界値に合わせた確率を調整する．こうすることで，無駄な探索を減らし，局所解に陥る可能性を減少させ，解の精度の向上と実行時間の短縮をはかる方法を考える．

6.6.3　上界値 1

全枝中で，コストの小さいものから k 個の枝を選び，その総コストを計算したとき，それは，解が k 個の枝を含んだときのコストの下限値となり，考えられる最も低い場合のコストである．つまり，k 個の枝を含む場合，その下限値より少ないコストに抑えられる木は絶対構成されないと言える．

0 から順に各 k に対してその下限値を求めていったときその値が，初めて総コスト上限を超えたとき，この値を超える頂点数を持つ個体は総コスト上限を超えない最小木を決して

構成することはできないと言えるので，最適解が発見されることはあり得ないことになる．よって，頂点数 $k-1$ が解の上界値となる．

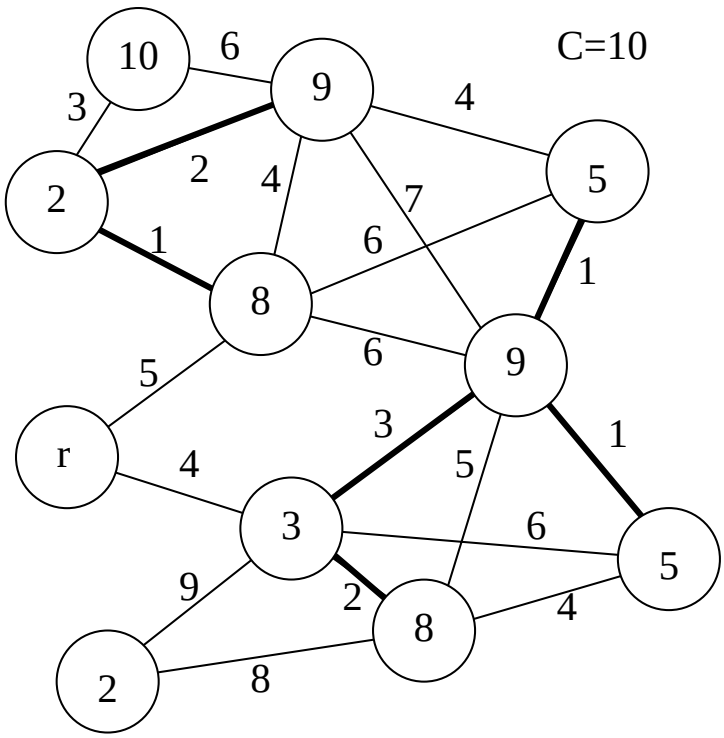

図 6.10　上界値 1

図 6.10 では，6 本の枝が選択できたので上界値は 6 ということになる．

だが，実際には最適解の枝集合は根から順に連結していて，閉路を作らないようなものでなければならない．そのことを考慮して，更に詳しい上界値を求める方法を考える．

6.6.4　上界値 2

解に含まれる枝は，必ず根から連結されている．つまり，k 個の頂点を含む解からは，根から枝 k 本分離れている枝より遠い枝を含むことはないと言えるので，その枝は下限値を求める上で選択しなくてよい．

図 6.11 は，枝の選択範囲を根からの距離によって分けたものである．最適解が k 個の頂

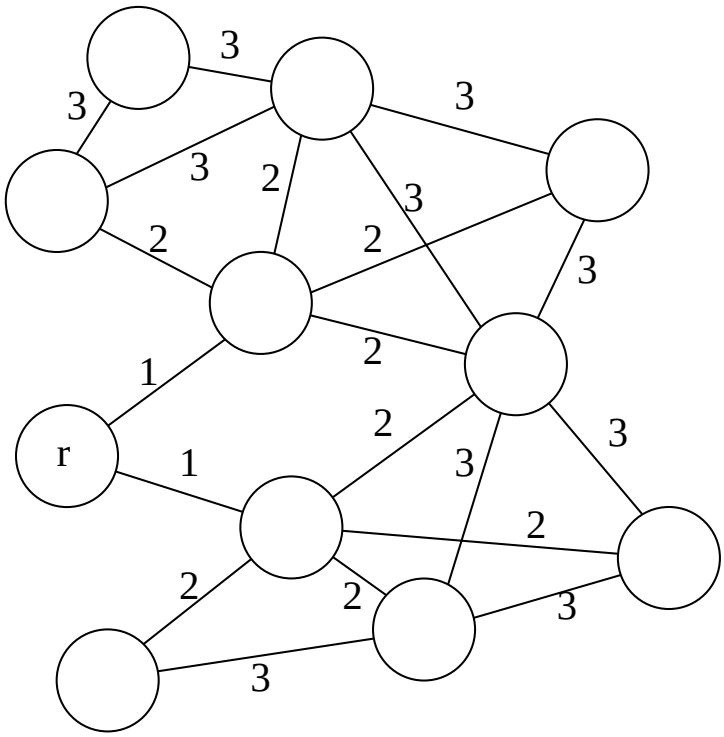

図 6.11　枝の根からの距離

点を含む場合，最初の枝は距離 1 の枝から選択し，次の枝は距離 1 と距離 2 の枝から選択する．以下同様にして選択していき，総コスト上限を超える直前の枝数が上界値となる．

また，枝は閉路を構成してはならないので，閉路が構成される枝は選択対象から除外して選択する．

以上を考慮し，順に各 k に対する総コストの下限値を求めていき，初めに総コスト上限を超えた k 以上からは解が生まれないことがわかるので，頂点数 $k-1$ が解の上界値となる．

選択する枝に制限を設けいてるので，前述の上界値より詳しい値が期待できる．

図 6.12 は，図 6.10 と同じ問題を距離を考慮した選択方法によって求めた上界値である．先ほどの場合の上界値は 6 だったのに対し，距離を考慮した場合の上界値は 5 である．距離を考慮した方がより小さな上界値を求めることができるため，より詳しい解の頂点数の上界値が求められる．

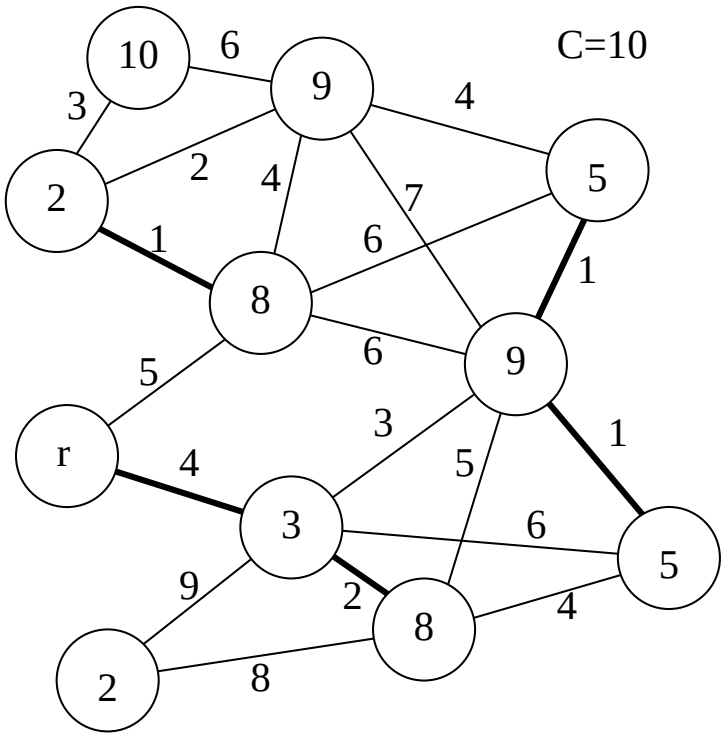

図 6.12　上界値 2

6.6.5　上界値による初期集団及び突然変異の調整

上界値を超える数の頂点数を持つ最適解はあり得ないことがわかるので，初期集団に与える頂点数の割合を上界値になるように調整する．合わせて，突然変異の確率も上界値を考慮し，1 と 0 の割合が変わらないようにする．

こうすることで，初期集団ですでに解に近いと思われる個体が多く含まれると考えられ，更に世代を進める上で頂点数の割合が大きく変化することを抑えているので，解の発見が短時間になり，かつ，精度の高いものが発見しやすくなると思われる．

6.7　遺伝的アルゴリズムの特徴

ヒューリスティックアルゴリズムと同様に必ず最適解を発見できる保証はないが，交叉による多様性と突然変異による局所解の回避で，より良い解が期待できる．また，上界値を利

用するなど探索範囲を抑圧しているため，より効率的に最適解に近づくことができる．

遺伝的アルゴリズムは，一世代中の個体の数を多く設定すれば，一世代中の計算時間が多くかかるが多様性が広がり，終了条件を厳しくすれば，実行時間そのものが長くなるが，解の信頼性の向上になると思われる．この二つのパラメータの調節は，解の探索時間と解の精度にも影響すると思われるので重要である．

第 7 章

実験結果と考察

7.1　遺伝的アルゴリズムと上界値考慮遺伝的アルゴリズム

上界値を考慮して初期集団と突然変異確率を設定した遺伝的アルゴリズムと，従来の遺伝的アルゴリズムの解の精度による比較の結果を表 7.1 に示す.

最適解に含まれる頂点の割合別に比較するため，総コスト上限別に実験し，データは 8 回の実験での利得の最大値を求めた.

実験に使用した問題は，頂点数 30，枝数 80 で，遺伝的アルゴリズムのパラメータは，個体数 90，終了世代数 60 である.

表 7.1　遺伝的アルゴリズムと上界値考慮遺伝的アルゴリズム

総コスト上限	頂点の割合	従来法	上界値考慮法
50	0.20	76	76
100	0.40	157	157
200	0.77	261	268

最適解に含まれる頂点数が全体の頂点数に対して約半分以下の場合は，両アルゴリズムにおいて違いは見られないが，頂点数の割合が半分を超えている場合は上界値を考慮した遺伝的アルゴリズムの方が解の精度が良いという結果になったため，以降，上界値を考慮した遺伝的アルゴリズムを使うことにする.

7.2　遺伝的アルゴリズムの解更新の頻度

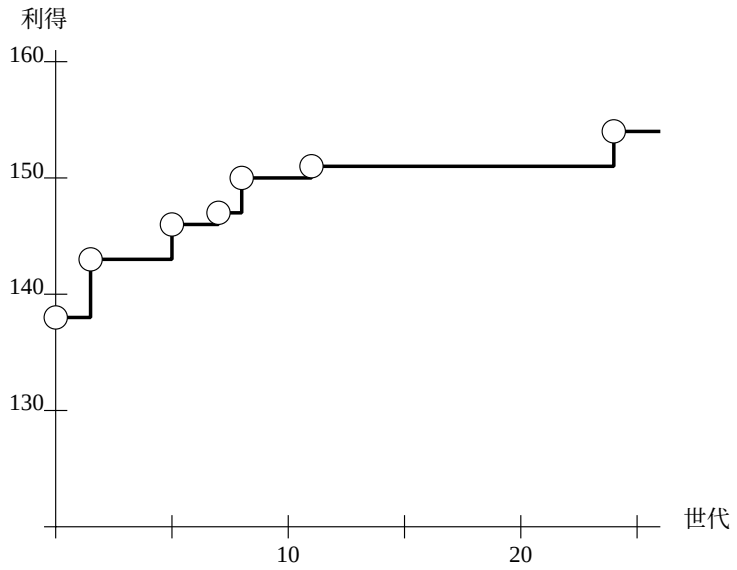

図 7.1　各世代における最良解の利得

図 7.1 は，その世代までの最良解の利得を表したグラフである．世代が進むにつれて最良解の更新の間隔が広くなっているのがわかる．ある程度世代が進むと，なかなか解の更新が行なわれないことを意味している．

解の更新が頻繁に起こる期間は特に交叉によって解が更新していて，解の更新が止まってきたら突然変異による更新になってきていると思われる．なぜなら，世代が進むと似たような個体が集中して発生してくるため，交叉による多様性が望めなくなってくるのである．そうなると突然変異が重要になる．

7.3　遺伝的アルゴリズムの入力パラメータ

遺伝的アルゴリズムを実行するためのパラメータには，個体数，終了世代数，交叉確率，突然変異確率がある．どういったパラメータで実行するのが最も効率が良いのか実験した．

7.3.1　交叉確率と突然変異確率

表 7.2 は，頂点数 40，枝数 150，総コスト上限 150 の問題に対し，個体数 90，終了世代数 60 での交叉確率と突然変異確率の組合せにによる実験の結果である．8 回の試行を行ない，利得が最大のものをとった．

表 7.2　交叉確率 (X) と突然変異確率 (M) の組合せによる利得の変化

X\M	0.2	0.4	0.6	0.8
0.2	208	206	206	203
0.4	209	205	205	203
0.6	208	204	206	203
0.8	208	208	207	205

全体的に大きな差は見られないが，若干，突然変異確率は 0.2 が高い利得が得られている．大きく変異させると，残すべき情報が大きく壊れるためだと思われる．

頂点数 50，枝数 200，総コスト上限 200 に問題を大きくして，突然変異を 0.2 で固定にしたときの交叉確率別に実行時間と利得を調べる実験を行なった．結果を表 7.3 に示す．実験は，8 回の試行で最大値の利得と平均時間を表している．

交叉確率の小さいものは，世代があまり進まないうちに終了していると見られる．交叉をあまり行なわないので，世代が進んでも解の更新がなかなか起こらないため，良い解が生まれにくいと思われる．

一定以上の交叉確率ならば実行時間にも利得にもあまり差が現れていないが，若干交叉確率 0.6 が効率が良いように見られる．しかし，解に差が見られないのは，局所解が少なく簡単な問題であるからとも考えられる．

表 7.3　交叉確率による利得と実行時間の変化

X	利得	時間
0.2	258	43.5
0.4	261	56.1
0.6	263	51.2
0.8	263	56.5

続いて，局所解が多いであろうと予想される大きな問題にて同じ実験を行ない，利得と時間を調べる．問題は頂点数 70，枝数 400，総コスト上限 300 である．同様に 8 回の試行で利得は最大値，時間は平均値を表す．結果は表 7.4 に示す．

表 7.4　難解な問題においての交叉確率による利得の変化

X	利得	時間
0.2	337	73.2
0.3	343	74.3
0.4	346	78.9
0.5	343	78.0
0.6	343	69.0
0.7	345	80.3
0.8	342	73.4

先程と同様に交叉は確率による差はあまり見られなかった．難解な問題に対しては，解の更新が減ってきてから交叉による解の更新が望めず，突然変異による更新が重要になってくるためだと考えられる．

最大利得部分木問題に対して，本研究の手法では，交叉確率による性能の違いがあまり見られないため，以降は交叉確率 0.6 で実験を行なうことにする．

7.3.2　個体数と終了世代数

個体数を多くすれば多様性が生まれ，終了世代数を多くすれば局所解から抜けやすくなり解の精度が上がりやすい．従って，両パラメータはなるべく大きい方が良い解が期待できる．しかし，個体数を多くすると一世代中の演算量が多くなり，終了世代数を多くすると終了までの時間が大きくなるので，両パラメータの増加は実行時間の増大につながる．

そこで，少ない実行時間で最良の解を得る最も良いパラメータを考える．頂点数 40，枝数 200，総コスト上限 200 の問題において，個体数と終了世代数の組合せによる実験を行ない，結果を表 7.5 に示す．また，同問題で総コスト上限を 100 にした場合の実験結果を表 7.6 に示す．実験は 8 回の試行で最大の利得と平均時間をとった．

表 7.5　個体数 (C) と終了世代数 (G) の組合せによる利得と時間 (総コスト上限 200)

C\G	40		80		120		160	
	利得	時間	利得	時間	利得	時間	利得	時間
40	262	4.8	269	14.7	262	16.8	266	22.3
80	263	16.2	263	22.5	262	27.3	263	34.2
120	268	19.9	263	34.0	270	69.0	265	75.5
160	266	27.2	265	40.5	268	99.3	269	110.1

総コスト上限が 200 の場合は，個体数 120，終了世代数 40 以上のパラメータの結果には解に大きな違いは見られない．よって，実行時間が最も短い，個体数 120，終了世代数 40 のパラメータが最も良いと思われる．

表 7.6　個体数 (C) と終了世代数 (G) の組合せによる利得と時間 (総コスト上限 100)

C\G	40		80		120		160	
	利得	時間	利得	時間	利得	時間	利得	時間
40	135	1.0	138	1.9	141	2.5	141	3.9
80	142	1.6	142	3.3	141	5.2	141	5.9
120	142	2.4	142	4.2	142	7.4	142	12.6
160	138	4.5	141	6.2	142	14.9	142	18.1

同様に，総コスト上限が 100 の場合では全体的に大きな差は見られないが，個体数 80, 終了世代数 40 が若干良いパラメータであると思われる．

この結果より，同じ問題でも総コスト上限によってパラメータは変更することが望ましいと思われる．総コスト上限が小さい場合，解になりうる木の種類が少なくなると考えられるため，探索範囲が減り問題が簡単になると思われる．

この問題では，頂点数が大きくなれば染色体の長さも大きくなる．それは，解候補の頂点集合の範囲が広くなることを意味する．また，総コスト上限が大きい場合は問題は難解になり，少ない場合は簡単になると思われる．従って，探索規模は問題の頂点数と総コスト上限に依存すると考えられる．

つまり，パラメータの値は問題の頂点数と総コスト上限に応じて効率が良いものを考える必要があると思われる．

7.3.3　入力パラメータに関する考察

問題は頂点数，総コスト上限以外にも，枝の連結や利得によって様々な特徴を持った問題になる．局所解が非常に多い場合は突然変異確率や終了世代数を高めなくては最適解を発見できなかったり，大きな問題でも非常に簡単な問題であれば個体数も終了世代数も少なくて良いなど，入力パラメータも一概にどれが良いとは言い切れない．

一度の試行のみで良い解を得ようとするならば，個体数，終了世代数を高く設定するべきであるが，何度か試行した中で最良の解を採用するなら，一回の実行時間がそれほど長くないパラメータの方が良い．その場合，様々なパラメータを試しながら，利得や実行時間，解に含まれる頂点数，解発見世代数を考慮しながら，パラメータを変更していく方法が有効であると思われる．

7.4　他のアルゴリズムとの比較

問題別にヒューリスティックアルゴリズム (HA) と分枝限定法 (B&B) と遺伝的アルゴリズム (GA) で実験し，解の利得と実行時間を比較した．結果は図 7.7 に示す．

遺伝的アルゴリズムの実行におけるパラメータは，問題の頂点数の 3 倍の個体数と問題の頂点数の終了世代数に設定してして行なった．8 回の試行の中で求められた最大の利得を解とし，実行時間は平均時間 [msec] を表している．

ヒューリスティックアルゴリズムは非常に短時間に解を求めているが，問題が大きくなるに従いその精度は落ちていることがわかる．問題が大きいほど局所解が多く含まれていると考えられるので，難解な問題になりやすく，局所解に陥っていると思われる．

また，分枝限定法では厳密な解を求めることができるが，問題が大きくなると探索範囲が深くなり，非常に時間がかかるため，ある問題以上から解を求められなくなっている．本研究における分枝限定法では実行上のメモリが影響して実行できないと考えられるが，充分なメモリが確保されていたとしても，実行された結果からは桁外れな実行時間になると予想される．

遺伝的アルゴリズムをヒューリスティックアルゴリズムと比較すると，時間はかかっているが解の精度は常にヒューリスティックアルゴリズム以上の精度である．また，分枝限定法と比較すると，実行時間は非常に短く，更に最適な解を出していることがわかる．

表 7.7　各アルゴリズムとの利得と時間比較

頂点数	枝数	総コスト上限	HA		GA		B&B	
			利得	時間	利得	時間	利得	時間
10	30	20	42	0.0	42	0.0	42	0.0
		40	68	0.0	83	0.1	83	3.1
20	70	50	91	0.0	96	0.1	96	19.6
		100	153	0.0	153	0.6	-	-
30	100	70	137	0.0	138	0.9	-	-
		120	205	0.2	212	3.6	-	-
40	150	100	136	0.1	144	1.9	-	-
		200	273	0.4	282	13.5	-	-
50	200	150	151	0.1	172	8.3	-	-
		250	236	0.3	305	32.1	-	-
60	250	200	230	0.7	230	22.0	-	-
		350	346	1.7	368	91.8	-	-

7.5　各アルゴリズムの比較による考察

実行時間が少なくて済むような小さな問題に対しては，難解な問題にも厳密に解を求めることができる分枝限定法が有効である．また，局所解がないような簡単な問題には大きな問題でも短時間に解を発見できるヒューリスティックアルゴリズムが有効である．

しかし，大きな問題に対しては，一般的に問題が大きい場合は局所解が多くなると思われるので，分枝限定法では時間がかかり過ぎて解が求まらないし，ヒューリスティックアルゴリズムでは発見した解の精度が疑わしい．

遺伝的アルゴリズムならば，大きな問題にも比較的短時間に局所解を脱しながら解を発見することができるため，非常に有効であると思われる．

第 8 章

結論

本論文では最大利得部分木問題に対して遺伝的アルゴリズムを利用する方法を提案し，ヒューリスティックアルゴリズム，分枝限定法と比較し，性能を評価した．

頂点数，枝数，総コスト上限による様々な特徴を持つ問題に対して実験を行なった結果，簡単な問題に対してはヒューリスティックアルゴリズムが有効で，小さな問題に対しては分枝限定法が有効であった．

しかし，様々な工学的問題に適用する場合，一般にその規模や難易度は未知であるため，規模の大きな問題や難解な問題に対して，比較的短時間で良質な解を発見することができる遺伝的アルゴリズムが最大利得部分木問題に対して非常に有効であると言える．

この最大利得部分木問題における解探索の方向性には厳密解法と近似解法があるが，問題の性質上必ず厳密解を求めたい場合は，より高速な分枝限定法を考案せねばならないと思われる．しかし，その時間が実用的にならない以上，本研究が提案する遺伝的アルゴリズムで近似解を求めるのが良いと思われる．

謝辞

本論文は，著者が 2000 年 7 月から 2002 年 3 月までの高知工科大学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なった研究活動の成果を記したものである．

まず，本論文を書き上げるにあたり，多忙の中，全面的に御指導頂き，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．

また，昨年度，プログラムや遺伝的アルゴリズムに留まらず，研究室を一人で管理なされていたりと，本当に色々と御指導頂いた橋本学氏に深く感謝致します．

常に忙しそうでしたが，よく助けてくれた登伸一さんと，助けて頂く前にいなくなってしまいましたが，色々と楽しませてくれた折橋祐一さんに感謝致します．

そして，結構邪慳にされたけれど，たまに麻雀に誘ってもらったりバーチャ 4 で対戦したりと楽しく過ごし，一生懸命バイトをする傍ら最後まで頑張って研究してきた亀本学君，たまに研究室に訪れ，土佐弁が光っていた田村和香那さんに感謝致します．

昨年，色々お世話になりました，卒業された先輩方，研究室を変更された同期の方々に感謝致します．

また，福本研では毒舌で愛の鞭を下さった福本昌弘先生，発表用のノートパソコン，製本のラベルなど様々な面でお世話になりました秋山由佳さん，共に徹夜した嶋岡哲夫君，プリンプリンの横でパチンコをしたという山本真弘君に感謝致します．

人が減って淋しかった研究室を少し賑やかにしてくれた 3 年生，赤間寛君，河野兼裕君，西村章君に感謝します．就職活動，研究活動，大変でしょうけど頑張って下さい．

そして，本研究を完成させるにあたり，色々な面でお世話になり，支えて下さった全ての人に深く感謝致します．

参考文献

[1] 星崎康広，片岡靖詞，森戸晋，“最大利得部分木問題に対する近似解法および厳密解法”，情報処理学会論文誌，Vol.42，No.2，p.318，Feb.2001.