根付き部分木ネットワークの利益最大化

ABE, Yusuke / 安部, 友輔

(発行年 / Year)
2013-03-24
(学位授与年月日 / Date of Granted)
2013-03-24
(学位名 / Degree Name)
修士(工学)
(学位授与機関 / Degree Grantor)
法政大学 (Hosei University)

2012年度　　修士論文

論文題名　根付き部分木ネットワークの利益最大化

-Maximizing the profit of a rooted subtree-

指導教員　五島洋行

大学院工学研究科

システム工学専攻修士課程

11R6202

氏名　安部友輔（アベ　ユウスケ）

Abstract In this thesis, we address *maximum-profit-rooted-subtree problem* ($MPRS$). In the problem, given a connected undirected graph, a root vertex, each vertex's non-negative income, and each edge's non-negative cost, we aim to find a rooted tree in which the total profit is maximized. The problem can be applied to various real-life situations such as cable television service and supply of energy (gas, water and electricity etc.). This important problem had been proposed by " Kobayashis(2003) " for the first time.

The main contributions of this research are to prove that the $MPRS$ is $NP-hard$, formulate the $MPRS$ as a mixed integer programming problem, and propose a heuristic algorithm.

The $MPRS$ is classified as an optimization problem. In optimization problems, an objective function and constraint equations are given. Under the constraints, we find a maximum or minimum value of the objective function. Combinatorial optimization problem is also classified as an optimization problem, and the solutions are represented by combination or permutation. Moreover, a part of combinatorial optimization problems can be formulated as a network. Our problem is positioned here. Using the ideas and terminologies in graph theory, this problem is referred to as *maximum-profit-rooted-subtree problem*, and we call this $MPRS$ for short.

Chapter 1 is an introduction, and we outline the background and organization of this thesis. Next, in chapter 2, we definite terminologies. In chapter 3, we formulate the $MPRS$ as a combinatorial optimization problem. Moreover, practical usages and previous researches are overviewed. In chapter 4, we formulate the $MPRS$ as a mixed integer programming problem. In chapter 5, we prove that the $MPRS$ is $NP-hard$. In chapter 6, we introduce three heuristic algorithms, one of which has been proposed by us. Though the other algorithms are existing, we rewrite these in detail. In chapter 7, we show the results of computational experiments and perform considerations. Finally, in chapter 8, we conclude our research giving future works.

論文要旨　本研究では「最大利益根付き部分木問題 (*maximum-profit-rooted-subtree problem* (*MPRS*))」の問題のクラスや，効率的な解法などについて検討する．具体的には，点の集合，枝の集合に加え，点に付された収入，枝に付されたコスト，根の指定が与えられる．それに対して，最大利益根付き部分木を求めるという問題である．ケーブル TV やガス，水道事業など様々な事業がこの問題として定式化できる．この重要な問題は「古林ら（2003)」によってはじめて提案された．

本研究の主な貢献は，NP 困難性の証明，混合整数計画問題としての定式化，新たなアルゴリズムの提案を行ったことである．

なお，「最大利益根付き部分木問題」は最適化問題の一つである．最適化問題とは，ある制約条件と目的関数が与えられる．そして，その制約条件の下で，目的関数を最大化，もしくは最小化する解を求める問題である．最適化問題の中でも組合せ最適化問題は，解が順列，組合せで表される問題であり，さらに組合せ最適化問題の中にはネットワークとして表現することができる問題がある．ネットワークとは，グラフ理論でいう“ グラフ ”に，重みを持たせたものである．「最大利益根付き部分木問題」はここに位置付けられる．

第 1 章では，研究背景と論文構成を述べる．次に第 2 章で用語の定義を行う．なお，グラフ理論における用語の定義は，文献によって異なることが多い．本論文では第 2 章での定義を使用することとする．それを用いて第 3 章で $MPRS$ を組合せ最適化問題として定式化する．さらに，応用と先行研究について述べる．第 4 章では $MPRS$ を混合整数計画問題として定式化する．第 5 章では NP 困難性の証明をする．なお，NP 困難である問題は，多項式時間で最適解を求めることが難しい問題である．次の第 6 章では近似解を求めるための提案手法を三つ紹介する．一つは我々の提案したものであり，二つは既存のアルゴリズムである．本論文では二つの既存アルゴリズムを詳しく書き直している．そして，第 7 章で計算実験の結果と考察を示す．最後に第 8 章で結論と今後の課題を述べる．

目次

第 1 章

序章

1.1 研究背景

本研究では「最大利益根付き部分木問題 ($maximum\text{-}profit\text{-}rooted\text{-}subtree\ problem$ ($MPRS$))」の問題のクラスや，効率的な解法などについて検討する．この重要な問題は古林ら [1] によってはじめて提案された．さらに [1] ではアルゴリズムを提案しており，そのアルゴリズムの優れた特徴と，しかし必ずしも最適解を求められないことを述べている．その後，古林らは [2] にて，高速なアルゴリズムも提案しているものの，これも最適解を必ずしも求められない．

そこで本研究では，この問題の NP 困難性を証明した．さらに，混合整数計画問題としての定式化を行った上で，新たなアルゴリズムの提案と計算実験を行った．

なお，「最大利益根付き部分木問題」は最適化問題の一つであり，図 1.1 のように位置付けられる．最適化問題とは，ある制約条件と目的関数が与えられる．そして，その制約条件の下で，目的関数を最大化，もしくは最小化する解を求める問題である．最適化問題の中でも組合せ最適化問題は，解が順列，組合せで表される問題であり，さらに組合せ最適化問題の中にはネットワークとして表現することができる問題がある．ネットワークとは，グラフ理論でいう “ グラフ ” に，重みを持たせたものである．「最大利益根付き部分木問題」はここに位置付けられる．

他にここに位置付けられるものとして，有名な巡回セールスマン問題がある．この問題は，すべての都市を巡る最短距離の巡回路を求める問題である．言い換えると，どの道路を組合せるべきかという問題であり，組合せ最適化問題である．さらに，巡回セールスマン問題はネットワークとして表現することができる．都市を点，各都市間の道路を枝とし，そして，枝に距離の重みを持たせることによって表現される．

また，「最大利益根付き部分木問題」は部分木を求める問題である．公共サービスのようにすべての顧客を対象としたサービスは，全点木問題として定式化できる．一方，企業

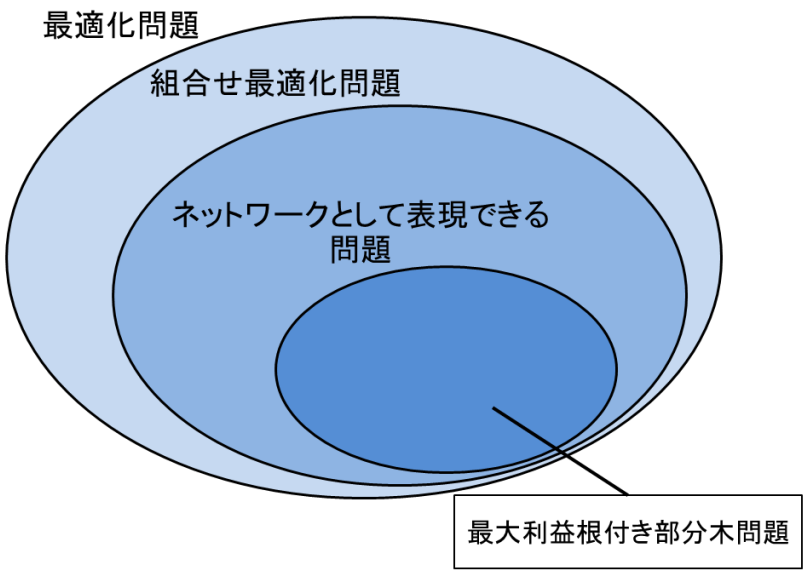

図 1.1　最大利益根付き部分木問題の位置付け

のように利益が最大になるよう顧客を選ぶサービスは，部分木問題に定式化できる．そのような状況は数多くあると考えられ，部分木の関連研究はいくつかある．しかし「最大利益根付き部分木問題」に関する研究は少なく，本研究ではこれに取り組む．

1.2　論文構成

本章では，研究背景と論文構成を述べる．次に第 2 章で用語の定義を行う．なお，グラフ理論における用語の定義は，文献によって異なることが多い．本論文では第 2 章での定義を使用することとする．それを用いて第 3 章で $MPRS$ を組合せ最適化問題として定式化する．さらに，応用と先行研究について述べる．第 4 章では $MPRS$ を混合整数計画問題として定式化する．第 5 章では NP 困難性の証明をする．なお，NP 困難である問題は，多項式時間で最適解を求めることが難しい問題である．次の第 6 章では近似解を求めるための提案手法を三つ紹介する．一つは我々の提案したものであり，二つは既存のアルゴリズムである．本論文では二つの既存アルゴリズムを詳しく書き直している．そして，第 7 章で計算実験の結果と考察を示す．最後に第 8 章で結論と今後の課題を述べる．

第 2 章

グラフ理論

グラフ理論でいうグラフとは，円グラフや棒グラフなどの図ではなく，つながりを表現するものである．また，グラフ理論の定義は文献により異なることが多く，本研究では本章で述べるものを使用する．多くの定義は，文献 [3]，[4] によるものである．

無向グラフは，点の集合 V と，V の 2 点を結ぶ枝の集合 E からなる (図 2.1)．各枝 $e \in E$ は V の 2 要素の部分集合であり，ある $u, v \in V$ を用いて $e = \{u, v\}$ と表せる．さらに，$e = \{u, v\}$ において，u と v は隣接するといい，u と v を e の端点という．また，e は u と v に接続するという．点 u に隣接している枝の本数を次数という．

E は点同士の対称な関係を表しているため，$\{u, v\}$ と $\{v, u\}$ は同じものである．非対称な関係を表したいときには，有向グラフという概念を用いる．有向グラフは，点の集合 V と，V の 2 点を結ぶ枝の集合 A からなる (図 2.1)．つまり，各枝 $a \in A$ は V の 2 要素

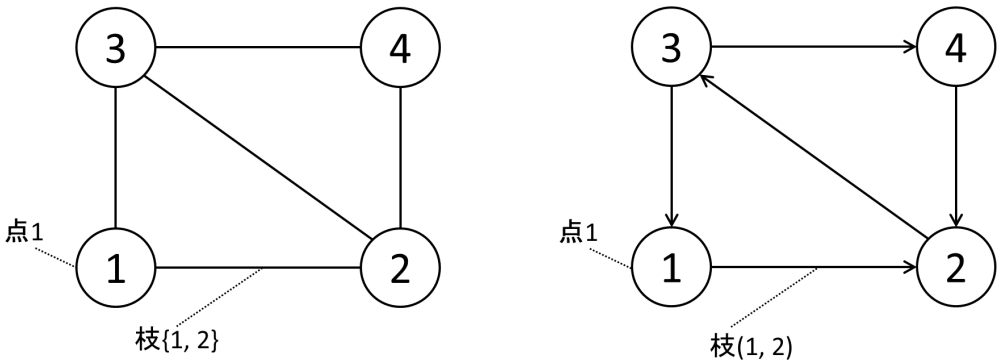

図 2.1　左図：無向グラフ．点 1 と点 2 は隣接している．枝 $\{1, 2\}$ の端点は点 1 と点 2 である．また，枝 $\{1, 2\}$ は点 1 と点 2 に接続している．点 1 の次数は 2 である．右図：有向グラフ．枝 (1,2) において，点 1 は枝 (1,2) の始点，点 2 は枝 (1,2) の終点．また，枝 (1,2) は点 1 を出て点 2 に入る．点 1 の入次数は 1，出次数は 1 である

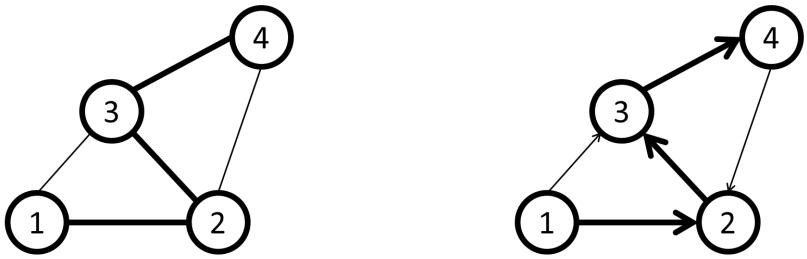

図 2.2　左図：パス $P = 1, \{1,2\}, 2, \{2,3\}, 3, \{3,4\}, 4$. 右図：有向パス $P^{'} = 1, (1,2), 2, (2,3), 3, (3,4), 4$

の部分集合であり，ある $u, v \in V$ を用いて $a = (u, v)$ と表せる．ただし，(u, v) と (v, u) は異なる．さらに，$a = (u, v)$ において，u を a の始点，v を a の終点という．また，a は u を出て，v に入るという．点 u に入る枝の本数を入次数，点 u から出る枝の本数を出次数という．無向グラフと有向グラフを総称して，グラフという．

有向グラフに対して，無向基礎グラフという概念もある．無向基礎グラフ G' は，点集合が有向グラフ G と同じであり，枝集合が G の各枝 (u, v) の向きを除いた枝 $\{u, v\}$ である無向グラフである．

なお，グラフの部分集合を部分グラフという．さらに，グラフ上で定義される次のような用語がある．

- パス，有向パス

  - 無向グラフ G において，$v_1, e_1, v_2, e_2, v_3 ..., v_k, e_k, v_{k+1}$ からなる点と枝の列 P に対して，各枝 e_i が $\{v_i, v_{i+1}\}$ であるときに，P を G のパスという．有向グラフ $G^{'}$ において，$v_1^{'}, e_1^{'}, v_2^{'}, e_2^{'}, v_3^{'} ..., v_k^{'}, e_k^{'}, v_{k+1}^{'}$ からなる点と枝の列 $P^{'}$ に対して，各枝 $e_i^{'}$ が $(v_i^{'}, v_{i+1}^{'})$ であるときに，$P^{'}$ を $G^{'}$ の有向パスという (図 2.2).

- 閉路，有向閉路

  - 無向グラフ G において，パス $v_1, e_1, v_2, e_2, v_3 ..., v_k, e_k, v_{k+1}$ が $k > 1$，かつ $v_1 = v_{k+1}$ であり，点と枝の重複が $v_1 = v_{k+1}$ 以外にないとき，このパスを閉路という．有向グラフ $G^{'}$ において，パス $v_1^{'}, e_1^{'}, v_2^{'}, e_2^{'}, v_3^{'} ..., v_k^{'}, e_k^{'}, v_{k+1}^{'}$ が $k > 1$，かつ $v_1^{'} = v_{k+1}^{'}$ であり，点と枝の重複 $v_1^{'} = v_{k+1}^{'}$ 以外にないとき，この有向パスを有向閉路という (図 2.3).

図 2.3　左図：閉路 $P = 1, \{1,2\}, 2, \{2,3\}, 3, \{3,1\}, 1$．右図：有向閉路 $P' = 1, (1,2), 2, (2,3), 3, (3,1), 1$

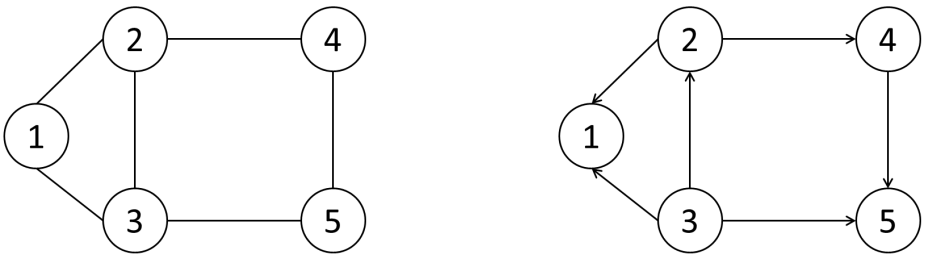

図 2.4　左図：連結な無向グラフ．右図：連結な有向グラフ

- 連結
  - 無向グラフにおいて，任意の 2 点にパスが存在するとき連結であるという．有向グラフでは，その無向基礎グラフにおいて，任意の 2 点にパスが存在するとき連結であるという (図 2.4).
- 部分木
  - 無向グラフ $G = (V, E)$ において，閉路を含まない連結な部分グラフを部分木という．部分木の点集合が V であるとき，全点木という (図 2.5)．ある一つの点を指定して，その点を根とすることがある．根を指定した部分木を根付き部分木という．

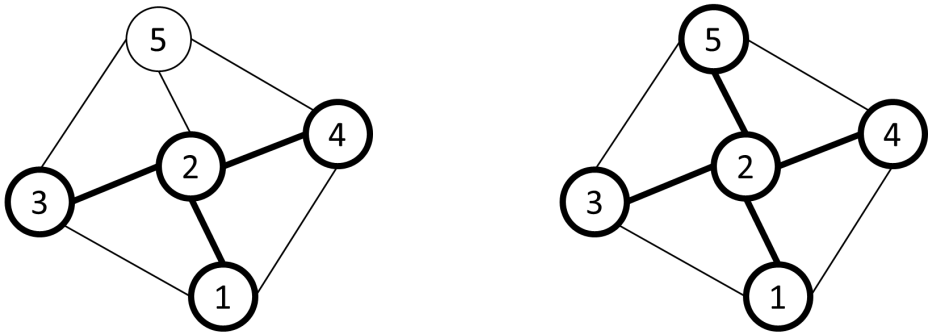

図 2.5　左図：部分木．右図：全点木

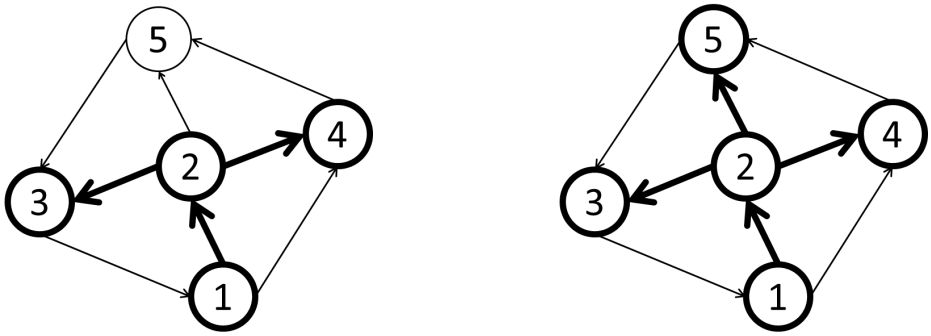

図 2.6　左図：部分有向木．右図：全点有向木．ともに根は点 1

- 部分有向木

  – 有向グラフ $G = (V, A)$ において，有向閉路を含まず連結であり，かつ各点の入次数が高々 1 である部分グラフを部分有向木という．部分有向木の点集合が V であるとき，全点有向木という．なお，無向基礎グラフにおいて部分木であっても，必ずしも部分有向木ではない．部分有向木において，入次数が 0 である点がただ一つだけある．その点を根という (図 2.6).
  また，文献 [3]，[4] に部分木，部分有向木の定義はない．本研究における全点木，全点有向木のみがそれぞれ木，有向木という名称で定義されている．文献 [5] に部分木の定義があるが，全点木の部分グラフのことを部分木と呼んでいる．

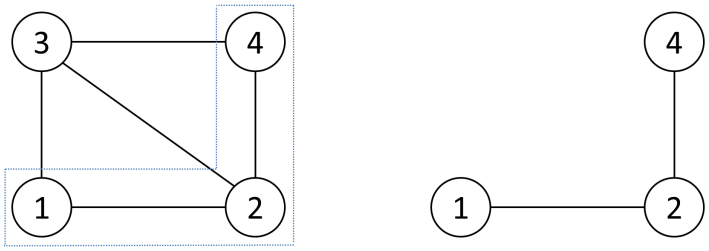

図 2.7　左図：図 2.1 の左図に対して，$S = \{1, 2, 4\}$ とする．右図：図 2.1 の左図に対して，$S = \{1, 2, 4\}$ とした誘導部分グラフ

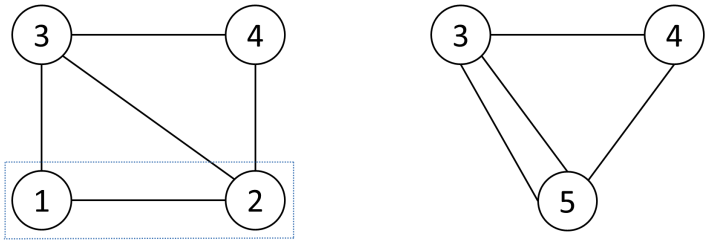

図 2.8　左図：図 2.1 の左図に対して，$S = \{1, 2\}$ とする．右図：点 1,2 と枝 $\{1, 2\}, \{1, 3\}, \{2, 3\}, \{2, 4\}$ を削除し，新しい点 5 を加え，さらに，各枝 $\{1, 3\}, \{2, 3\}, \{2, 4\}$ を枝 $\{5, 3\}, \{5, 3\}, \{5, 4\}$ に置き換えて縮約する

- 誘導部分グラフ
  - 無向グラフ $G = (V, E)$ に対して，点の部分集合 $S \subseteq V$ を取り出す．点集合を S として，枝集合を両端点が S に含まれているすべての枝とする．その G の部分グラフを，誘導部分グラフという (図 2.7)．誘導部分グラフは有向グラフ上でも同様に定義される．

- 縮約
  - 無向グラフ $G = (V, E)$ に対して，点の部分集合を $S \subseteq V$ とする．S のすべての点と S に接続するすべての枝を削除し，新しい点 x を加え，さらに，$v \in S, w \notin S$ となる各枝 $\{v, w\}$ を枝 $\{x, w\}$ に置き換えることを縮約するという (図 2.8)．縮約は有向グラフ上でも同様に定義される．

第 3 章

最大利益根付き部分木問題

3.1 定式化

第 2 章で定義した用語を使い，組合せ最適化問題として定式化を行う．入力は連結な無向グラフ $G = (V, E)$，根の指定，各点に付された収入，各枝に付されたコストである．それに対して出力は，根を含む利益最大の部分木である．グラフ $G = (V, E)$ は各点に番号が付いているとする．根は番号 1 であり，他の点は $2, 3, \ldots, n$ (≥ 2) と正の整数でそれぞれ異なる番号が付いているとする．また，この番号付けによって，V のサイズは n となる．各点 i $(\in V)$ は，実数である非負収入 p_i を持つ．なお，出力は必ず点 r を含むため，$p_1 = 0$ とする．各枝 $\{i, j\}$ $(\in E)$ は，実数である非負コスト c_{ij} を持つ．G は無向グラフであるため，$c_{ij} = c_{ji}$ となる．G における，点 r を含む部分木を $T = (N, A)$ とする．利益である $\mathrm{profit}(T)$ を，

$$\mathrm{profit}(T) = \sum_{i \in N} p_i - \sum_{\{i,j\} \in A} c_{ij}.$$

とする．出力は $\mathrm{profit}(T)$ を最大化する T である．以上をまとめると表 3.1 になる．この問題を最大利益根付き部分木問題 (*maximum-profit-rooted-subtree problem* (*MPRS*)) と呼ぶ．例を図 3.1 に示す．

表 3.1　最大利益根付き部分木問題

入力: 連結な無向グラフ $G = (V, E)$, 根 r の指定, 各点 i $(\in V)$ の持つ収入 $p_i (\geq 0)$, 各枝 $\{i, j\}$ $(\in E)$ の持つコスト $c_{ij} (\geq 0)$. **出力:** $\mathrm{profit}(T)$ を最大化する点 r を含む部分木 $T = (N, A)$.

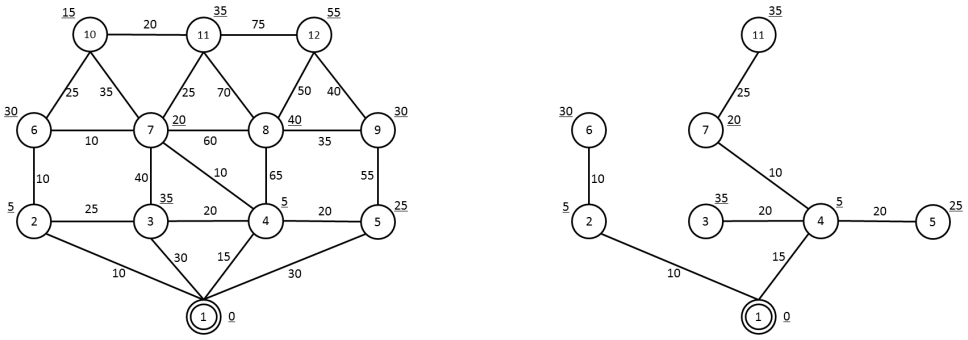

図 3.1　左図は $MPRS$ の入力例．右図はその最適解であり，最適値は 45

3.2　応用

文献 [1] では，ケーブル TV 事業において，利益を最大化するためにはどうすればよいか，という現実の問題があった (図 3.2)．具体的には，ただ一つのセンター，いくつかの顧客の候補，いくつかのケーブルの候補，顧客からの収入，ケーブル工事のコストが与えられる．これに対して，利益が最大となるように，顧客を選び，それらの顧客にケーブルを張りたい．ただし，それらの選び方，張り方にも制約がある．

まず，ケーブル TV であるため，顧客はセンターと繋がっていなければならない．これは，根付き部分木の条件，根を含み連結であることに対応する．次に，明らかに余分なケーブルを張ってはならない．これは，根付き部分木の条件，無閉路であることに対応する．例えば，顧客 3 にコンテンツを届けるために，ケーブル $\{1,2\},\{2,3\},\{3,1\}$ と張ると余分なケーブルを使ってしまう．

$MPRS$ を効率的に解くことによって，大きな収入を得ることができる．また，無閉路であるという条件により，過剰な資源を削減することもできる．さらに，入力を木に限定することにより，多くの NP 完全問題が多項式時間で解けることが知られている [3]．$MPRS$ で求めた顧客とケーブルの集合に対して，何らかの NP 完全問題を多項式時間で解くことができる．

元々のケーブル TV 事業における利益最大化，という現実の問題に対して，$MPRS$ とその解法は，強力な道具となる．さらに $MPRS$ は，ガス，水道，電気などの事業も含んでいると言える．具体的には，枝を，ガス管，水道管，電線と捉えることによって適用できる．

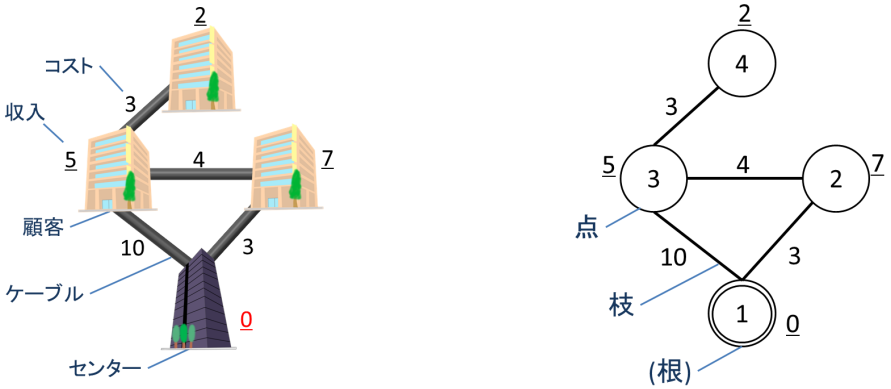

図 3.2　左図：ケーブル TV 事業．右図：$MPRS$ としての定式化

表 3.2　最小重み全点木問題

入力: 無向グラフ $G = (V, E)$, 各枝 $\{i, j\}$ $(\in E)$ の持つ重み c_{ij}.
出力: G の最小重み全点木.

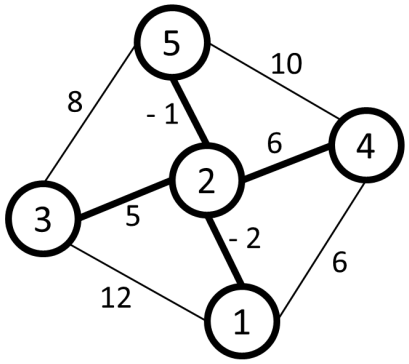

図 3.3　最小重み全点木問題の例．太線が最適解であり，最適値は 8

3.3　先行研究との関係

グラフ理論の古典的な問題に，最小重み全点木問題がある [4](表 3.2)(図 3.3)．この問題に対する多項式時間アルゴリズムは既に存在する．

最小重み全点木問題を一般化した問題に，一般最小重み全点木問題がある．この問題は Myung ら [6] ではじめて提案された．さらに，Feremans ら [7] は 8 つの定式化を行って

表 3.3　シュタイナー木問題

入力: 無向グラフ $G = (V, A)$, 点の部分集合 $S \subseteq V$, 各枝 $\{i, j\}$ $(\in E)$ の持つ重み $c_{ij} (> 0)$. **出力:** S を含む，G の最小重み部分木.

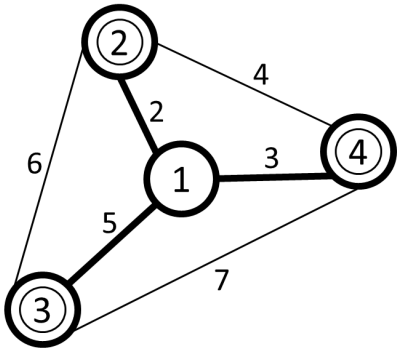

図 3.4　シュタイナー木問題の例．点 2,3,4 が S．太線が最適解であり，最適値は 10

いる．

最小重み全点木問題の拡張は他に，次数制約最小重み全点木問題がある．この問題は Narula ら [8] によってはじめて提案されて，さらに分枝限定法による解法が提案された．その後 Alexandre ら [9] はこの問題に対して，ラグランジュ緩和法を用いたアルゴリズムを提案している．

また，次数制約最小重み全点木問題のよく知られている拡張として，すべての点に同じ k の次数制約のつく，k-次数制約最小重み全点木問題がある．$k = |V| - 1$ のときは最小重み全点木問題となる．

最小重み全点木問題は，すべての点に枝を張らなくてはならない．これに対し，ある点集合を張る部分木を求める問題もあり，シュタイナー木問題 [4](表 3.3)(図 3.4) と呼ばれる．シュタイナー木問題は NP 困難であることが証明されている．

シュタイナー木問題の拡張としては，グループシュタイナー木問題がある．この問題は Reich ら [10] によってはじめて提案され，Hwang ら [11] によって再調査された．[11] にてアルゴリズムも提案されており，Helvig ら [12] が改良を行っている．

$MPRS$ はシュタイナー木問題の特殊なタイプであると考えられる．$MPRS$ において，与えられる点集合がただ一つであるという点は，シュタイナー木問題に含まれる．し

表 3.4　最小重み全点有向木問題

入力: 有向グラフ $G = (V, A)$, 各枝 (i, j) $(\in A)$ の持つ重み c_{ij}. **出力:** G の最小重み全点有向木.

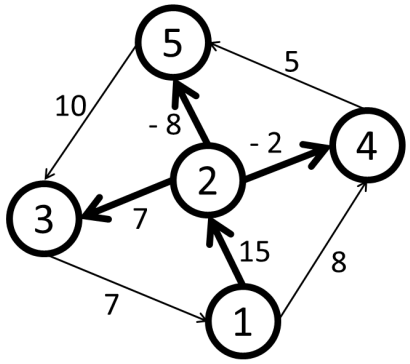

図 3.5　最小重み全点有向木問題の例．太線が最適解であり，最適値は 12

表 3.5　最小重み根付き全点有向木問題

入力: 有向グラフ $G = (V, A)$, 根 r の指定, 各枝 (i, j) $(\in A)$ の持つ重み c_{ij}. **出力:** 点 r を根とする，G の最小重み全点有向木.

かし，点に収入，枝にコストがあり，利益木を求めるという点は，シュタイナー木問題の拡張であると言える．

次に，有向グラフ上の先行研究について述べる．無向グラフ上では，最小重み全点木問題があった．その有向グラフ版ともいえる問題が，最小重み全点有向木問題である (表 3.4)(図 3.5)．さらに，最小重み根付き全点有向木問題がある．この問題は指定された点を根とする，最小重みの全点有向木を求める問題である．最小重み全点有向木問題と最小重み根付き全点有向木問題は，多項式時間で解ける [4].

Rao ら [13] は最小重み根付き部分有向木問題を提案しており，この問題が NP 困難であることを証明している．さらに [13] では，アルゴリズムを提案して良好な実験結果を出している．ただし，この問題の入力は，有向閉路を含まない有向グラフであることに注意が必要である．

第 4 章

混合整数計画問題

4.1 混合整数計画問題

第 1 章では，最適化問題のうち，解が順列や組合せで表されるものを組合せ最適化問題であると述べた．最適化問題を解の種類で分類する以外に，制約条件や目的関数の種類で分類する方法もある [14]．代表的なものを，表 4.1 にまとめた．

さらに線形計画問題のうち，決定変数が連続変数かつ離散変数であるものを混合整数計画問題 (*Mixed Integer Programming*(*MIP*)) と呼ぶ．

4.2 定式化

この章では，$MPRS$ を MIP として定式化する．まずは，$MPRS$ のインスタンスから，双方向である有向グラフ $D = (N_D, A_D)$ を次のようにつくる．例を図 4.1 に示す．

- $N_D := V$.
- $A_D := \{(i,j)|\{i,j\} \in E, j \neq 1\}$.
- 枝 $(i,j) \in A_D$ は重み $w_{ij} := p_j - c_{ij}$ を持つ.

表 4.1　最適化問題の分類

	制約条件	目的関数
線形計画法	線形	線形
二次計画法	線形	2 次関数
非線形計画法	線形 and/or 非線形	非線形
整数計画法	整数 and/or 線形	線形

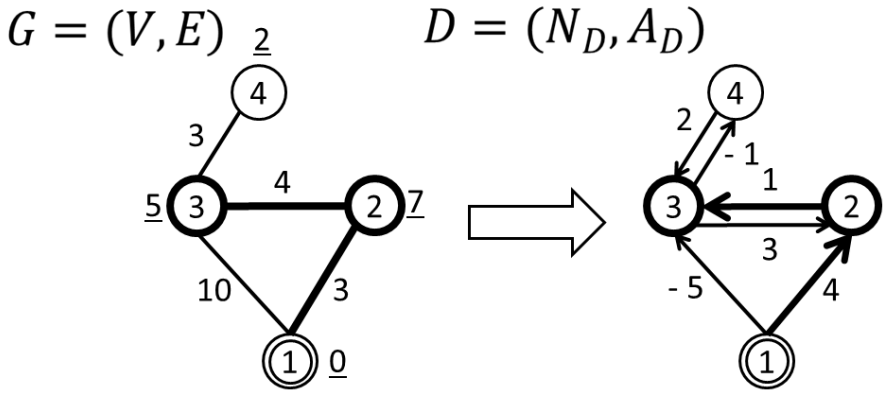

図 4.1　有向グラフ D への変換例．太線が最適解であり，最適値は 5

D 上で最大重み根付き部分有向木を見つけることは，元問題において最適解を見つけることと等価である (図 4.1)．次に，各枝 (i,j) に対応した 0-1 変数 x_{ij} と，各点 i に対応した実数変数 u_i を定義する．u_i は点 i の訪問順序を表す．点 1 における u_1 を 0 と解釈する．

$$
\begin{aligned}
&x_{ij} = \begin{cases} 1 & \text{枝 } (i,j) \text{ を解に含む場合,} \\ 0 & \text{それ以外,} \end{cases} \\
&u_1 = 1, \\
&2 \le u_i \le n, \quad i = 2, \ldots, n.
\end{aligned}
$$

$MPRS$ を次のように定式化する．

$$
\text{maximize} \quad \sum_{(i,j) \in A_D} w_{ij} x_{ij} \tag{4.1}
$$

$$
\text{subject to} \quad \sum_{(i,j) \in A_D} x_{ij} \le 1, \quad \text{for all } j \in N_D \setminus \{1\}, \tag{4.2}
$$

$$
\begin{aligned}
x_{ij} \le & \sum_{(k,i) \in A_D \setminus \{(j,i)\}} x_{ki}, \\
& \text{for all } (i,j) \in A_D \setminus \mathrm{OUT}(1),
\end{aligned} \tag{4.3}
$$

$$u_i + 1 - (n-1)(1 - x_{ij}) + (n-3)x_{ji} \leq u_j, \\ \text{for all } (i,j) \in A_D \setminus \mathrm{OUT}(1). \tag{4.4}$$

式 (4.2)，(4.3)，(4.4) は解が根付き部分有向木になるための条件を表す．式 (4.2) は，解において，選択された点に入ってくる枝が高々一つであることを示している．このため，式 (4.2) を入次数制約と呼ぶ．式 (4.3) は，解において，ある枝が選択されるためには，その先行の枝が少なくとも一つ選択されなければならないことを示している．この制約式により，解は連結なグラフとなるため，式 (4.3) を連結制約と呼ぶ．式 (4.2)，(4.3) は文献 [13] で提案されたものである．

有向閉路の制約式は様々なものがある [15]，[16]．本研究では式 (4.4) を採用する．式 (4.4) は式 (4.5) を改良したものである．

$$u_i + 1 - (n-1)(1 - x_{ij}) \leq u_j, \\ \text{for all } (i,j) \in A_D \setminus \mathrm{OUT}(1). \tag{4.5}$$

式 (4.5) を式 (4.4) に改良する過程を記す．式 (4.5) は，解において，有向閉路が含まれないことを示している．各枝 (i,j) に対し，$x_{ij} = 1$ であるとき $u_j \geq u_i + 1$ となり，有向閉路が含まれない．また，もし解に有向閉路が一つでも含まれるならば，u_i の値が ∞ となってしまう．なお，式 (4.5) は式 (4.6) のように書き換えることもできるが，制約式の数が点の指数サイズになってしまうため実用的ではない．

$$\sum_{i \in S, j \in S} x_{ij} \leq |S| - 1, \quad S \subseteq N_D \text{ かつ } |S| > 1. \tag{4.6}$$

式 (4.5) を改良し，式 (4.4) を得るために，持ち上げと呼ばれる次の操作を行う [16]．まずは x_{ji} を左辺に加えて，その係数を α とする．

$$u_i + 1 - (n-1)(1 - x_{ij}) + \alpha x_{ji} \leq u_j. \tag{4.7}$$

α を実行可能解を除かない範囲で，できるだけ大きくする．$x_{ji} = 0$ のときは式 (4.5) になる．$x_{ji} = 1$ のときには $x_{ij} = 0$ となるため，$u_j + 1 = u_i$ となる．このとき，α は，

$$\alpha \leq u_j - u_i - 1 + (n-1) = n - 3. \tag{4.8}$$

となる．α をできるだけ大きくして，式 (4.4) を得る．

第 5 章

NP 困難性

5.1 計算複雑性

計算機科学の一分野である計算複雑性理論では，計算可能な問題をその難しさによって分けるための，クラス P，NP などの概念がある．これらの概念を正確に定義するためには，計算模型であるチューリング機械や，Yes か No を出力する判定問題などが用いられる [3]．本節では，クラスについての直観的な説明をする．

問題のサイズ n に対して，処理時間の上界を n の多項式で表現できるときに，その問題を多項式時間 (polynomial time) で解けるという．クラス P は多項式時間アルゴリズムの存在する問題のクラスである．

次に，クラス NP について説明する．クラス NP は Not-P でなく，Non-deterministic Polynomial time の略である．クラス NP は，多項式時間アルゴリズムが存在するかはわからないが，与えられた解を多項式時間で検証可能な問題のクラスである．

定義より，$P \subseteq NP$ であるが，$P = NP$ であるかどうかはまだわかっていない．この問題は計算機科学における有名な未解決問題の 1 つである．ただし，これまでの研究から $P \neq NP$ と予想されている．

他にも，NP 完全，NP 困難というクラスがある．NP 完全は，NP の中で最も計算困難な問題である．NP 困難は，NP 完全と同等以上に計算困難な問題である．NP，NP 完全，NP 困難に属する問題は，現実的な時間で最適解を求めることは難しいとされている．クラスの包含関係は，図 5.1 のようになる．

ある問題の計算困難性を証明するために，命題 1 の対偶である命題 2 を用いることが多い [3].

命題 1. 問題 Y は問題 X に多項式時間で帰着可能とする．このとき，問題 X が多項式時間で解けるならば，問題 Y も多項式時間で解ける．

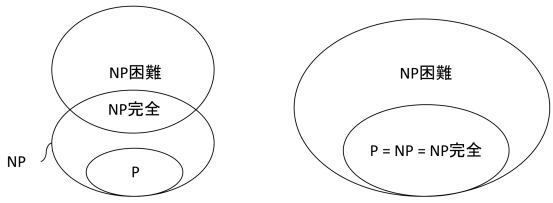

図 5.1　左図：$P \neq NP$ の場合．右図：$P = NP$ の場合

表 5.1　容量制約なし施設配置問題

入力：入力:施設の有限集合 F, 利用者の有限集合 D, 施設 $i \in F$ の開設コスト $o_i \in \mathbb{R}^+$, 施設 $i \in F$ を利用者 $j \in D$ が使うときの利用コスト関数 $s_{ij} \in \mathbb{R}^+$.
出力：開設コストと利用コストの総和が最小となるような, 開設する施設の集合 X と利用者の施設への割り当て $\sigma : D \to X$ を求める.

命題 2. 問題 Y は問題 X に多項式時間で帰着可能とする．このとき，問題 Y が多項式時間で解けないならば，問題 X も多項式時間で解けない.

既に計算困難であることがわかっている問題 Y があるとする．命題 2 により，問題 Y が問題 X に多項式時間で帰着可能であることが証明できれば，問題 X は問題 Y と同等以上に難しいことが証明できる．$MPRS$ の NP 困難性の証明でもこの形式をとっている．問題 Y がシュタイナー木問題，もしくは容量制約なし施設配置問題，問題 X が $MPRS$ に対応する.

証明 1 はシュタイナー木問題，証明 2 は容量制約なし施設配置問題を，問題 Y とした証明である．どちらの問題も NP 困難な問題である [4]．前者の証明はシンプルであり，また後者の証明は文献 [13] を参考にしたものである.

5.2　施設配置問題

利用者の需要が最も満たされる施設を，候補施設から選ぶ問題は，施設配置問題と呼ばれている．施設を適切な場所に開設することは，効率的にサービスを提供するために非常に重要なことである．施設配置問題の中でも最もよく知られているのが，容量制約なし施設配置問題 ($Uncapacitated\ Facility\ Location\ Problem(UFLP)$) である (表 5.1)．なお，$UFLP$ はプラント配置問題，または倉庫配置問題と呼ばれることもある．この問題は NP 困難であり，近似アルゴリズムが既に提案されている [4].

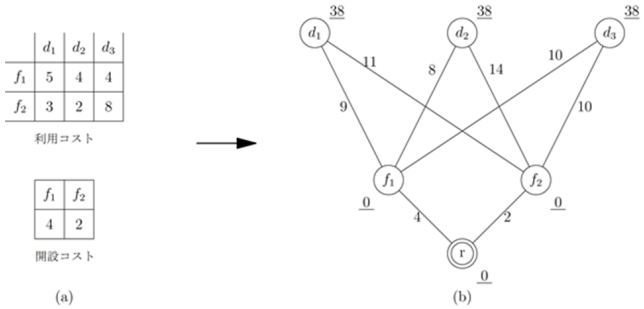

図 5.2　(a) は UFLP のインスタンス例．(b) は (a) を MPRS のインスタンスに変換したもの

5.3　NP 困難性の証明

$MPRS$ が NP 困難であることを示す．

定理 1. $MPRS$ は NP 困難である．

証明 1. シュタイナー木問題を次のように定義する．無向グラフ $G = (V, E)$，各枝 $e \in E$ の持つ非負重み $w(e)$，点の部分集合 $S \subseteq V$ が与えられる．それに対して，S を含む G の最小重み部分木を見つける．

シュタイナー木問題のインスタンスから $MPRS$ のインスタンスをつくる．$MPRS$ のインスタンスのグラフを，シュタイナー木問題と同一である $G = (V, E)$ とする．各枝 $e \in E$ の持つコスト $c(e)$ は $w(e)$ とする．各点 $v \in V$ の持つ収入 $p(v)$ は，$v \in V(G) \setminus S$ であるときは 0，$v \in S$ であるときは $\sum_{e \in E(G)} w(e)$ とする．根 r は S のある一つの点とする．

このように $MPRS$ のインスタンスを構成する．これに対しての，最適な根付き部分木 T は，点集合 S を含む．点 $v \in S$ の持つ収入がとても大きいためである．T は点集合 S を含み，枝重みの合計を最小化している．これはシュタイナー木問題の最適解に対応する．□

証明 2. $UFLP$ のインスタンスからグラフ $G = (V, E)$ をつくる．点集合を $V = F \cup D \cup \{r\}$ とし，枝集合を $E = E_F \cup E_S$ とする．ここで，$E_F = \{\{r, i\} \mid i \in F\}$，$E_S = \{\{i, j\} \mid i \in F, j \in D\}$ とする．

次に, 点と枝の重みを定義する．M_t を $UFLP$ のすべてのコストの総和とし, M_o をすべ

ての開設コストの総和とする．すなわち，$M_t = \sum_{i \in F} o_i + \sum_{i \in F, j \in D} s_{ij}$，$M_o = \sum_{i \in F} o_i$ となる．点 $v \in F \cup \{r\}$ の重み $p_v = 0$ とし，点 $v \in D$ の重み $p_v = M_t + M_o$ とする．また，枝 $\{r, i\}(i \in F)$ の重み $c_{ri} = o_i$，枝 $\{i, j\}(i \in F,\ j \in D)$ の重み $c_{ij} = M_o + s_{ij}$ とする．この変換の様子を図 5.2 に示す．この変換は明らかに多項式時間で行える．

$UFLP$ のインスタンスがコスト K の最適解 (X, σ) を持つことの必要十分条件が，$MPRS$ において $|D|M_t - K$ となる最適な根付き部分木を持つことであることを示せばよい．$UFLP$ の最適解 (X, σ) から，$MPRS$ における根付き部分木 $T = (N, A)$ を次のように構成する．点集合を $N = \{r\} \cup X \cup D$, 枝集合 $A = \{\{r, i\} | i \in X\} \cup \{\{\sigma(j), j\} | j \in D\}$ とする．このとき T の利益は，

$$
\begin{aligned}
&\sum_{j \in D} p_j - (\sum_{i \in X} c_{ri} + \sum_{j \in D} c_{\sigma(j)j}) \\
&= |D|(M_t + M_o) \\
&\quad -(\sum_{i \in X} o_i + |D|M_o + \sum_{j \in D} s_{\sigma(j)j}) \\
&= |D|M_t - K. \qquad (5.1)
\end{aligned}
$$

となり，$UFLP$ のインスタンスに対する最適解のコストが K のとき，$MPRS$ において利益 $|D|M_t - K$ となる T が存在する．

逆を証明する．グラフ G における $MPRS$ の最適解を $T = (N, A)$ とする．もし，点集合 N が D を必ず含んでいて，D の各点の次数が 1 であるならば，T は $UFLP$ のインスタンスの最適解 (X, σ) に対応する．この議論を次に示す．

$\{\{r, i\} \in A | i \in X\}$，$\{\{i, j\} \in A | \sigma(j) = i, i \in F, j \in D\}$ であるとすると，T の利益は $|D|M_t - K$ となる．式 (5.1) から $UFLP$ のインスタンスの最適解 (X, σ) のコストは K である．よって，T はすべての D を含んでいて，かつ，D の各点の次数は 1 である．

D の各点 j は正の重み $p_j = M_t + M_o$ を持つ．根 r から D の各点 j に対して，点 $i \in F$ と 2 本の枝を含んだパスができる．$c_{ri} + c_{ij} = o_i + M_o + s_{ij} < M_t < M_t + M_o = p_j$ であるため，点 $i \in F$ において，2 本の枝 $\{r, i\}$ と $\{i, j\}$ との和は p_j より小さい．よって，点 $j \in D$ は T に含まれている．

次に T において，D の各点の次数が 1 であることを示す．まず，点 $j \in D$ の次数が高々 2 であると仮定する．そうすると，T において，根 r から点 $i \in F$ までのパス P_i に含まれている枝 $\{i, j\}$ の点 i が存在する．ここで，枝集合 $A_{T'} = (A_T \cup \{r, i\}) \setminus \{i, j\}$ とすると，$|A_T| = |A_{T'}|$ となるため，$T' = (N, A_{T'})$ は根付き部分木であり，T' は根 r につながる点 i と点 j の 2 つの点を持つ．さらに，$o_i = c_{ri} < c_{ij} = M_o + s_{ij}$ から，T' の利益は T の利益より大きくなり，T が最適解であることに矛盾する．よって，T における各点 $j \in D$ の次数は 1 である．□

第 6 章

提案手法

6.1 剪定法

このアルゴリズムでは，まず入力である無向グラフ G に対して，第 4 章で述べた双方向の有向グラフ D をつくる．次に D に対して，最大重み根付き全点有向木を求める．なお，最大重み根付き全点有向木問題の多項式時間アルゴリズムは既存である [4].

最後に，手順 2 で求めた全点有向木から余分な枝を削除する．手順 2 では，各点 v を根とする部分木の重みを記憶する変数 $\omega(v)$ を使う．手順 2 の出力は，$\omega(r)$ を最大化する部分木 T である．また，この手法は次の **Remark**1 を持つ.

Remark 1. 手順 2 で削除される枝が一つもない場合は，最大重み全点有向木が最適解となる.

アルゴリズム 1 に手法の全体を示す．手順 2 の計算量は，根付き全点有向木を求める計算量 $O(nm)$ に吸収される．よって，剪定法，アルゴリズム 1 の計算量は $O(nm)$ となる．ただし，本研究では使用していないが，フィボナッチヒープを用いることによって，$O(m + n\log|n|)$ に改良することができる [17].

アルゴリズム 1 剪定法

1: 双方向の有向グラフ D をつくる
2: D 上で，最大重み根付き全点有向木 T を求める
3: T = 枝剪定 (T, w)
4: 部分有向木 T の向きを無くして，部分木にする

手順 2 枝剪定 (T, w)

各点 $v \in N(T)$ に対して，$\omega(v)$ を 0 とする
深さ優先探索で，後行順に点にラベル付けをする
for $1, 2, \ldots, n-1$ とラベル付けされた各点 v **do**
  if $\omega(v) + w(u, v) < 0$ である枝 $(u, v) \in A(T)$ **then**
    T から v を根とする部分木を削除する
  else
    $\omega(u) = \omega(u) + \omega(v) + w(u, v)$
  end if
end for
return T

6.2 付随木連結法

文献 [1] にて提案されたアルゴリズム．各点それぞれに，それを根とする部分有向木を持たせる．各点 $v \in V(G)$ に対して，それを根とする部分有向木を T_v とする．はじめの段階では，それぞれの部分有向木は根だけである．利益が大きくなるように，それぞれの部分有向木は拡大していく．

アルゴリズム 3 付随木連結法

各点 $v \in V(G)$ に対して，部分有向木 T_v をつくる
双方向の有向グラフ D をつくる
repeat
  各枝 $(u, v) \in A(D)$ に対して，$\{\mathrm{profit}(T_v) - c(u, v) | (u, v) \in A(D)\}$ の最大値 max を求める
  if $max > 0$ **then**
    for $u \in N(T_k)$ である各点 $k \in V(G)$ **do**
      $T_k =$ 木連結 (u, T_v, T_k, w, p, c)
    end for
  end if
  $A(D) = A(D) \setminus \{(u, v)\}$
until $A(D) \neq \emptyset$ **and** $max > 0$
元グラフ G 上で，$N(T_r)$ の誘導部分グラフと枝重み c に対する，最小重み全点木を求める

手順 4 木連結 (u, T_v, T_k, w, p, c)

Require: $u \in N(T_k)$

if $u \in N(T_v)$ **then**
  return T_k
else if $N(T_k) \cap N(T_v) == \emptyset$ **then**
  $N(T_k) = N(T_k) \cup N(T_v)$
  $A(T_k) = A(T_k) \cup A(T_v) \cup \{(u, v)\}$
else
  $S = N(T_v) \setminus (N(T_k) \cap N(T_v))$
  S による T_v の部分有向木 T'_v を求める
  $T'_v =$ 枝剪定 (T'_v, w)
  if $\mathrm{profit}(T'_v) > 0$ **then**
    $N(T_k) = N(T_k) \cup N(T'_v)$
    $A(T_k) = A(T_k) \cup A(T'_v) \cup \{(u, v)\}$
  end if
end if
return T_k

枝 $(u, v)(u \in N(T_k))$ を加えて，部分有向木 T_v と部分有向木 T_k をつなげることによって，利益が大きくなる場合を考える．この場合，次の式として表現される．

$$\mathrm{profit}(T_k) < \mathrm{profit}(T_k) + \mathrm{profit}(T_v) - c(u, v).$$

もし，T_v が点 u を含んでいる場合，枝 (u, v) が T_v に閉路をつくってしまう．一方，もし，T_k と T_v が共通の点を持っている場合，それらの点を T_v から削除しなければならない (図 6.1)．しかし，T_v から点を削除した得た T'_v によって，T_k の利益が減少するかもしれない．したがって手順 4 では，剪定法で用いた手順 2 を適用する．

木連結には次の変数，点 u，部分有向木 T_v，T_k，重み w, p, c，出力である部分有向木 T_k を使う．

最後に，元グラフ G 上で，$N(T_r)$ の誘導部分グラフと枝重み c に対する，最小重み全点木を求める．手法の全体をアルゴリズム 3 に示す．STEP3〜11 が $O(m)$ である．手順 4 の計算量は $O(n+m)$ である．よって, STEP6〜8 の計算量は $O(n) \times O(n+m) = O(nm)$ となる．したがって，付随木連結法，アルゴリズム 3 の計算量は $O(nm^2)$ となる．

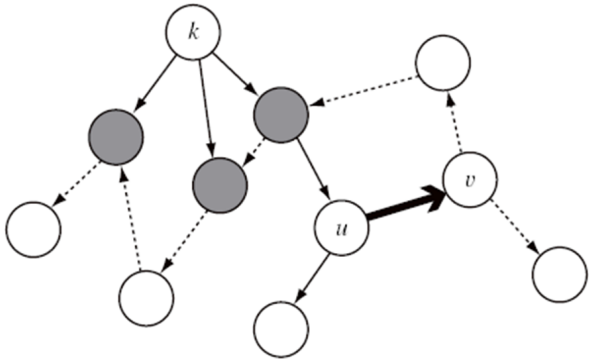

図 6.1　枝 (u, v) で，T_v と T_k がつながる例．T_k は実線，T_v は点線である．$N(T_k)$ と $N(T_v)$ の共通の点は黒で塗られている

6.3　最大重み経路法

文献 [2] にて提案されたアルゴリズム．このアルゴリズムの主なアイデアは，最大経路を繰り返し求めることである．そのために，まず入力である無向グラフ G に正の閉路がないように点を併合する．併合後のグラフ上で，最大経路を繰り返し求め，根付き部分木を得る．次に，剪定法で行った枝の削除を行う．最後に，得られた点集合に張る，最小重み全点木を求める．

アルゴリズム 5 最大重み経路法

1: $\bar{G} =$ 点併合 (G, r, p, c)
2: 双方向の有向グラフ $\tilde{G}$ をつくる
3: $T =$ 最長経路 $(\tilde{G}, r, \tilde{w}, \bar{p}, \bar{c})$
4: $T =$ 枝剪定 $(T, \tilde{w})$
5: 元グラフ G 上で，N_T の誘導部分グラフと枝重み c に対する，最小重み全点木 $G_T = (N_T, A_T)$ を求める

6.3.1　点併合

入力グラフ G において，$p_i \geq c_{ij}$ かつ $p_j \geq c_{ij}$ である点 i と点 j が存在する場合，それらを併合する．併合は本論文で用いる名称であって，一般的な名称ではない．併合は第 2 章で述べた縮約とほとんど同じ操作である．違いは次のとおりである．縮約を行うと，二

手順 6 点併合

Input: 無向グラフ $G = (V, E)$，根 r の指定，収入 p_i，コスト c_{ij}.

Output: 併合後のグラフ $\bar{G}$

1: 初期化 すべての k に対して，$S_k := \{k\}$

2: **if** $p_i \geq c_{ij}$ かつ $p_j \geq c_{ij}$ である点 i，点 j に対して **then**

3: 　点 i と点 j を，点 k に置き換えて $\bar{G}$ とする

4: 　$S_k := S_i \cup S_j$

5: 　$\bar{G}$ に対して再帰的にこれを行う

6: **end if**

つの同じ点に接続する枝が二つ以上つくられる場合がある．第 2 章の図 2.8 の場合，点 3 と点 5 の間に二つの枝が存在している．このアルゴリズムは，二つの同じ点に接続する枝は一つとする．その際には，コストが最も小さいもの以外を削除する．さらに，併合点 k の重みは $p_k = p_i + p_j - c_{ij}$ とする．また，併合点 k の点集合 S_k に点 i と点 j を記憶する．併合後のグラフを $\bar{G}$ とする．

6.3.2 双方向の有向グラフ

$\bar{G}$ に対して，双方向の有向グラフ $\tilde{G} = (\tilde{N}, \tilde{A})$ をつくる．$\tilde{G}$ の点集合は $\bar{G}$ と同一である．$\tilde{G}$ の枝集合は $\bar{G}$ の枝を双方向にしたものである．それぞれの枝 $(i, j) \in \tilde{A}$ において，重みは次のようにする．

$$\tilde{w}_{ij} = \begin{cases} -\infty, & j = 1 \text{ である場合}, \\ p_j - c_{ij} & \text{それ以外}. \end{cases} \tag{6.1}$$

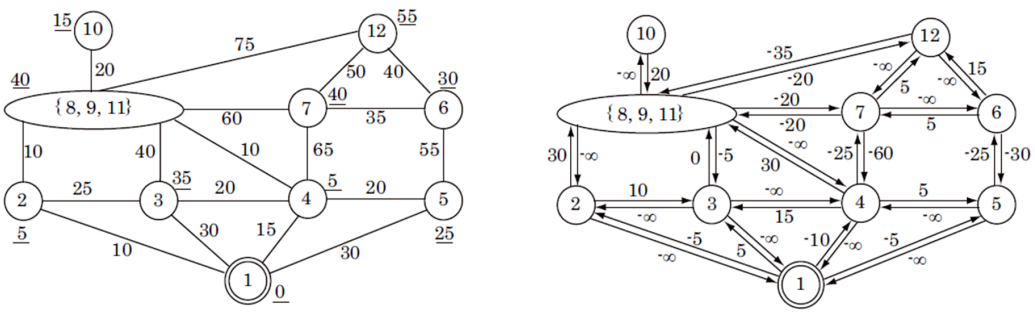

図 6.2　左図：図 3.1 の $\bar{G}$ の例．右図：左図の $\tilde{G}$ の例

$\tilde{G}$ において，$\tilde{w}_{ij} > 0$ かつ $\tilde{w}_{ji} > 0$ となる枝 (i,j) は存在しない．理由は次のとおりである．

$\tilde{G}$ において，$\tilde{w}_{ij} > 0$ かつ $\tilde{w}_{ji} > 0$ となる枝 (i,j) が存在すると仮定する．そのとき，$p_i > c_{ij}$ かつ $p_j > c_{ij}$ となる．$\bar{G}$ において，点 i または点 j は併合されているため，これは矛盾である．

次に，$\tilde{G}$ の枝 (i,j) の重み $\tilde{w}_{ij} > 0$ となるものに対して，枝 (j,i) の重み $\tilde{w}_{ji}$ を $-\infty$ に更新する．これにより，$\tilde{G}$ の特徴である次の補題 1 が得られる．

補題 1. $\tilde{G}$ は正の閉路を含まない．

証明： L を $\tilde{G}$ における有向閉路とする．N_L と A_L をそれぞれ L の点集合，枝集合とする．さらに，$\tilde{w}(L) = \sum_{(i,j)\in A_L} \tilde{w}_{ij}$ を L の重みとする．$\tilde{G}$ は双方向の有向グラフであるから，L の枝をすべて逆にした L' が存在する．もし L' に重み $\tilde{w}_{ji} > 0$ である枝 (j,i) が含まれるとする．そのとき，$\tilde{w}_{ij} = -\infty$ となり，$\tilde{w}(L) = -\infty$ となる．もし L' が正の枝を持たないとき，次の式を得る．

$$\begin{aligned}\tilde{w}(L) &= \sum_{(i,j)\in A_L} \tilde{w}_{ij} = \sum_{(i,j)\in A_L} (p_j - c_{ij}) = \sum_{j\in N_L} p_j - \sum_{(i,j)\in A_L} c_{ij} \\ &= \sum_{j\in N_{L'}} p_j - \sum_{(i,j)\in A_{L'}} c_{ij} = \sum_{(i,j)\in A_{L'}} (p_j - c_{ij}) = \sum_{(i,j)\in A_{L'}} \tilde{w}_{ij} \le 0.\end{aligned}$$

上の式では，$c_{ij} = c_{ji}$ であることに注意する．□

6.3.3 最長経路

$\hat{G} = (\hat{N}, \hat{A})$ を導入する．$\hat{G}$ の点集合と枝集合は $\tilde{G}$ と同一である．枝 $(i,j) \in \hat{G}$ の重みは $\hat{w}_{ij} = \tilde{w}_{ij}$ である．さらに，$d(i,j)$ を $\hat{G}$ における，点 i から点 j への最長経路とする．また，点集合 $U \subset \hat{N}$ とし，$d(U,j)$ を $\hat{G}$ における，U から点 j への最長経路とする．すなわち，$d(U,j) = \max_{i\in U} d(i,j)$ となる．最長経路を求めるとき，その重みを $\hat{w}$ で表す．

Remark 2. 補題 1 によって $\hat{G}$ に正の閉路が存在しないことが証明された．STEP7〜9 で枝の重みが更新される．しかし，重みを更新する上で $\hat{G}$ は正の閉路を含まない．よって，手順 7 を通して $\hat{G}$ は正の閉路を含まない．

Remark 3. STEP4 における最長経路の計算は，枝重みを正負逆転させたグラフにおいて最短経路を求めることに対応する．枝重みを正負逆転させたグラフは，負の有向閉路を含まない．よって，最短経路は多項式時間で求められる [4].

手順 7 最長経路

Input: 有向グラフ $\hat{G}$，重み $\hat{w}$，収入 p_i，コスト c_{ij}

Output: 全点木 $\hat{T} = (N_{\hat{T}}, A_{\hat{T}})$

1: 点集合 $N_{\hat{T}} := \hat{N}$，枝集合 $A_{\hat{T}} := \emptyset$ とする．
2: 点集合 $\hat{N}_1 := \{1\}$，$\hat{N}_0 := \hat{N} \setminus \{1\}$ とする
3: **while** $\hat{N}_0 \neq \emptyset$ **do**
4: 　各点 $j \in \hat{N}_0$ に対して，$d(\hat{N}_1, j)$ を計算する
5: 　各点 $j \in \hat{N}_0$ に対して $d(\hat{N}_1, j)$ を最大とするパスの点集合と枝集合をそれぞれ，$\hat{N}_2$ と $\hat{A}_2$ とする．$\hat{N}_1 := \hat{N}_1 \cup \hat{N}_2$，$\hat{N}_0 := \hat{N}_0 \setminus \hat{N}_2$
6: 　$A_{\hat{T}} := A_{\hat{T}} \cup \hat{A}_2$
7: 　$i \in \hat{N}_1$ かつ $j \in \hat{N}_1$ である枝 (i, j) に対して，$\hat{w}_{ij} = 0$ とする
8: 　$i \in \hat{N}_1$ かつ $j \in \hat{N}_0$ である枝 (i, j) に対して，$\hat{w}_{ji} = -\infty$ とする
9: 　$i \in \hat{N}_1$ かつ $j \in \hat{N}_0$ である枝 (i, j) に対して，もし $\hat{w}_{ij} = -\infty$ であるならば，$\hat{w}_{ij} = p_j - c_{ij}$ とする
10: **end while**

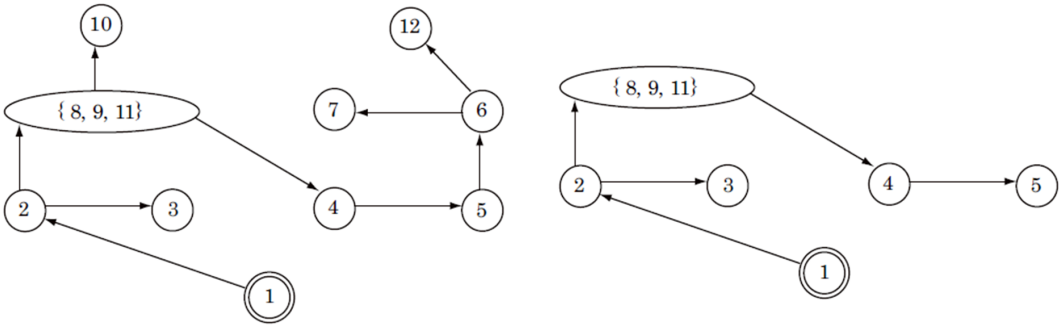

図 6.3　左図：図 6.2，右図の根付き全点有向木の例．右図：左図の根付き部分有向木の例

6.3.4　枝の削除

手順 7 の後に得られた根付き部分有向木から，剪定法の手順 2 を用いて，余分な枝を削除する．枝の削除後に，点 1 と連結である点集合を N_{WT} とする．N_T は次のようになる．

$$N_T = \bigcup_{i \in N_{WT}} S_i.$$

6.3.5 G の最小重み全点木を求める

元グラフ G 上で，N_T の誘導部分グラフと枝重み c に対する，最小重み全点木 $G_T = (N_T, A_T)$ を求める．

6.3.6 計算量

手順 6 の計算量は $O(m^2)$ である．手順 7 における最長経路の計算量は，$Moore - Bellman - Ford$ アルゴリズムにより，$O(nm)$ となる [4]．したがって，手順 7 の計算量は $O(n^2m)$ となる．STEP5 の最小重み全点木はグリーディに求めることができる [4]．よって，最大重み経路法，アルゴリズム 5 の計算量は $O(n^2m)$ となる．

6.4 総括

どのように根付き部分木を求めるかの，主となるアイデアが 3 つのアルゴリズムでそれぞれ異なる．

剪定法は，全点木をまず求めることが大きなポイントである．このことにより求めた部分木と全点木との関連を考えることができる．例えば，枝剪定において削除される枝が一つもない場合は，最大重み全点有向木が最適解と言える．また，第 7 章では，全点有向木から枝が削除されて，目的関数値がどれだけ増加を示す改善率を計算している．

付随木連結法は，各点に部分木を持たせることが大きな特徴である．ある点と隣接する点をつなぐと損であっても，さらに先までつなげれば利益がでる場合を見落とさないようにしている．そのため，良好な目的関数値が得られることが考えられる．

最大重み経路法は，最長経路を繰り返し求めて，それらをつなぐことによって，部分木を得る．また，はじめに点を併合するところがポイントである．点併合により，後のすべての手順を通して正の閉路が存在しない．

剪定法, 付随木連結法, 最大重み経路法の計算量は, それぞれ $O(nm)$, $O(nm^2)$, $O(n^2m)$ となる．一般的に，点の数より枝の数の方が多いため，剪定法，最大重み経路法，付随木連結法の順に計算量は少ない．

第 7 章

計算実験

7.1 テストデータ

第 6 章で提案したアルゴリズムを実装し，計算実験を行った．実装は C 言語とそのライブラリである LEDA(Library for Efficient Data types and Algorithms)[18] で行った．入力グラフは，網グラフとランダムグラフの 2 種類を用いた．網グラフは，格子グラフに斜めの枝を加えたものである (図 7.1)．点数を n とすると，枝数 $m = 4n - 6\sqrt{n} + 2$ となる．なお，網グラフは，本論文内での名称であって一般的なものではない．ランダムグラフは，LEDA に入っている関数で発生させた．ただし，その関数は，非連結グラフを生成してしまうことがある．連結グラフをつくるために，非連結グラフを生じたときには，もう一度関数を呼び出してグラフを生成した．連結の判定には，探索を行って全点を訪問できるか調べた．なお，ある点 i, j に接続する枝 $\{i, j\}$ を二つ以上生成することはない．

グラフの大きさは点数 9 個，枝数 20 個〜点数 2,500 個，枝数 9,702 個とし，それぞれの大きさで 100 回ずつ実験を行った．また，第 4 章で MIP として定式化したため，ソ

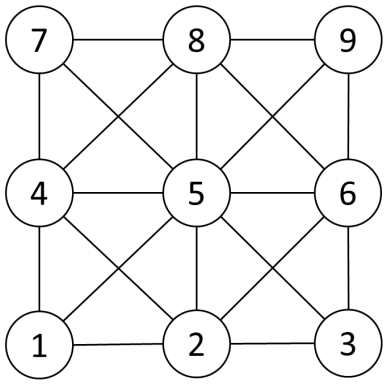

図 7.1　点 9 枝 20 の網グラフ

表 7.1　実験環境

Machine: DELL Bostro 220s
CPU: Intel Core2 Duo E8400 @ 3．00GHz
OS: Windows XP Professional
Memoly: 4GB
Compiler: Visual C++ 2010

ルバー CPLEX を用いて最適値を求めた．CPLEX においては，点数 9 個，枝数 20 個〜点数 49 個，枝数 156 個で実験を行った．点数 225 個，枝数 812 個以上のグラフでは，最適解を求めるために 7 日以上かかってしまうためである．網グラフ，ランダムグラフともに，点重み 1〜1,000，枝重み 1〜2,200 の整数とし，一様分布で発生させた．実験環境は表 7.1 のとおりである．

7.2　結果と考察

剪定法は，まず全点木を求めて，余分な枝を削除して部分木を得る．改善率 = |(profit(部分木) − profit(全点木))/profit(全点木)| と定義した．また，近似比 = 最適値/近似値 と定義した．なお，点 9 枝 20 のサイズで近似値が 0 になってしまう場合は，ケースを作り直した．近似比が 1 のとき，近似解は最適解となる．近似比が 1 に近ければ近いほど，良い値となる．

まず，目的関数の平均値 (表 7.3)，最適解との一致回数 (表 7.7)，近似比の平均値 (表 7.8) から，網グラフ，ランダムグラフともに，付随木連結法，剪定法，最大重み経路法の順に，値が良いことがわかる．また，近似比の値はすべてのアルゴリズムで良好であると言える．さらに，近似比の標準偏差 (表 7.9) から，サイズが大きくなるにつれて近似比が収束してきている．

計算時間の平均値 (表 7.4) から，網グラフ，ランダムグラフともに，最大重み経路法が最も速く，次に付随木連結法，剪定法の順に速いことがわかる．ただし，計算時間の標準偏差（表 7.5）とそれぞれのヒストグラム（図 7.2〜7.7）から，剪定法の値が大きく，ケースによって速さにバラつきがあると言える．特に，大きいサイズになると値が大きい．付随木連結法と最大重み経路法はバラつきが小さいと言える．これらをまとめると表 7.2 になる．

改善率の平均値，標準偏差 (表 7.6) から，サイズが大きくなるにつれて，値が収束してきていることがわかる．

表 7.2　性能比較表

手法名	剪定法	付随木連結法	最大重み経路法
速さ	○	○	◎
目的関数値	○	◎	△

表 7.3　各サイズ 100 回の目的関数の平均値

		網グラフ			
点	枝	剪定法	付随木	重み経路	CPLEX
9	20	1012.54	1033.99	1039.62	1049.48
25	72	3929.02	3993.44	3879.12	4028.51
49	156	8773.26	8840.69	8458.30	8964.01
225	812	46097.52	46487.03	44770.30	-
625	2,352	136689.75	137246.90	132734.53	-
841	3,192	183626.76	184626.45	178741.47	-
1,600	6,162	357120.71	359216.74	347859.92	-
2,500	9,702	561821.47	565496.90	547136.62	-

		ランダムグラフ			
点	枝	剪定法	付随木	重み経路	CPLEX
9	20	1063.29	1055.04	1048.29	1070.70
25	72	4226.87	4246.96	4198.41	4313.79
49	156	9503.57	9572.65	9413.36	9652.02
225	812	48566.56	48747.86	47891.54	-
625	2,352	141716.91	141925.72	139507.89	-
841	3,192	191864.01	192199.08	189240.30	-
1,600	6,162	370500.81	371180.69	364986.88	-
2,500	9,702	581778.28	582693.17	573030.05	-

表 7.4　各サイズ 100 回の計算時間の平均値

		網グラフ			
点	枝	剪定法	付随木	重み経路	CPLEX
9	20	0.001	0.003	0.002	0.063
25	72	0.008	0.004	0.002	0.230
49	156	0.037	0.006	0.002	1.108
225	812	0.876	0.155	0.006	-
625	2,352	8.584	3.100	0.018	-
841	3,192	19.329	7.650	0.031	-
1,600	6,162	92.591	58.171	0.111	-
2,500	9,702	296.394	225.976	0.265	-

		ランダムグラフ			
点	枝	剪定法	付随木	重み経路	CPLEX
9	20	0.001	0.004	0.001	0.041
25	72	0.005	0.003	0.001	0.068
49	156	0.042	0.006	0.001	0.947
225	812	1.073	0.166	0.005	-
625	2,352	16.786	3.228	0.019	-
841	3,192	39.046	8.176	0.033	-
1,600	6,162	340.316	62.772	0.125	-
2,500	9,702	815.195	243.286	0.333	-

表 7.5　各サイズ 100 回の計算時間の標準偏差

		網グラフ			
点	枝	剪定法	付随木	重み経路	CPLEX
9	20	0.004873	0.005792	0.004523	0.107073
25	72	0.007863	0.006613	0.004717	0.400587
49	156	0.020943	0.007279	0.004899	1.780912
225	812	0.476751	0.010040	0.007279	-
625	2,352	4.688018	0.125510	0.006029	-
841	3,192	12.871319	0.276769	0.004534	-
1,600	6,162	61.730924	1.952790	0.007763	-
2,500	9,702	183.906669	5.600966	0.011816	-

		ランダムグラフ			
点	枝	剪定法	付随木	重み経路	CPLEX
9	20	0.003846	0.006528	0.004314	0.088001
25	72	0.007770	0.005914	0.004090	0.103289
49	156	0.027401	0.007317	0.003580	1.917088
225	812	1.118905	0.010332	0.007032	-
625	2,352	22.156728	0.100146	0.006739	-
841	3,192	50.353502	0.414822	0.004899	-
1,600	6,162	394.996885	1.317980	0.005849	-
2,500	9,702	1234.629418	3.594290	0.010113	-

表 7.6　各サイズ 100 回の改善率の平均値，標準偏差

		網グラフ		ランダムグラフ	
点	枝	平均	標準偏差	平均	標準偏差
9	20	258.07	522.387	945.49	3361.814
25	72	87.17	87.131	77.18	57.234
49	156	55.48	45.227	54.25	32.097
225	812	31.25	6.993	38.40	8.476
625	2,352	25.69	3.386	33.67	4.753
841	3,192	24.87	3.349	34.12	3.549
1,600	6,162	23.47	2.398	32.87	2.546
2,500	9,702	22.74	1.510	31.37	1.911

表 7.7　各サイズ 100 回の最適解との一致回数

		網グラフ			ランダムグラフ		
点	枝	剪定法	付随木	重み経路	剪定法	付随木	重み経路
9	20	92	91	93	95	95	93
25	72	61	72	52	61	64	59
49	156	34	43	15	36	58	32

表 7.8　各サイズ 100 回の近似比の平均値

		網グラフ			ランダムグラフ		
点	枝	剪定法	付随木	重み経路	剪定法	付随木	重み経路
9	20	1.06	1.02	1.83	1.04	1.01	1.02
25	72	1.04	1.01	1.06	1.03	1.03	1.03
49	156	1.03	1.02	1.07	1.02	1.01	1.03

表 7.9　各サイズ 100 回の近似比の標準偏差

		網グラフ			ランダムグラフ		
点	枝	剪定法	付随木	重み経路	剪定法	付随木	重み経路
9	20	0.378716	0.063870	8.249302	0.280777	0.055486	0.098209
25	72	0.114853	0.029109	0.124416	0.063996	0.070111	0.067920
49	156	0.053189	0.028534	0.086653	0.033783	0.018301	0.038910

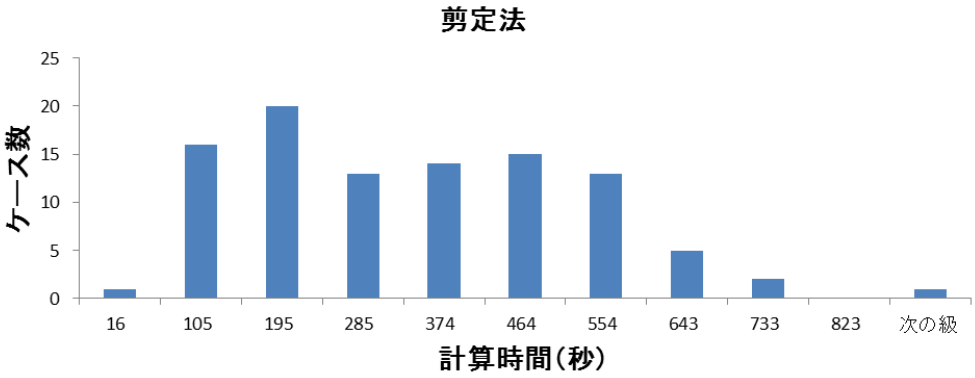

図 7.2　網グラフ n=2,500，m=9,702，剪定法を用いた計算時間のヒストグラム

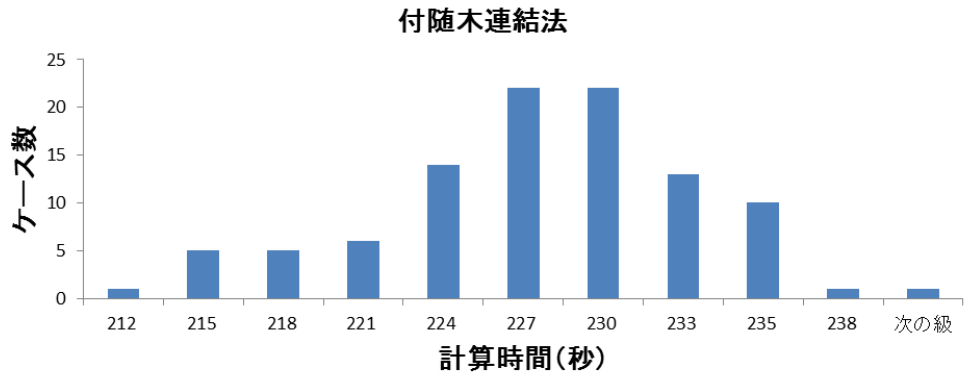

図 7.3　網グラフ n=2,500，m=9,702，付随木連結法を用いた計算時間のヒストグラム

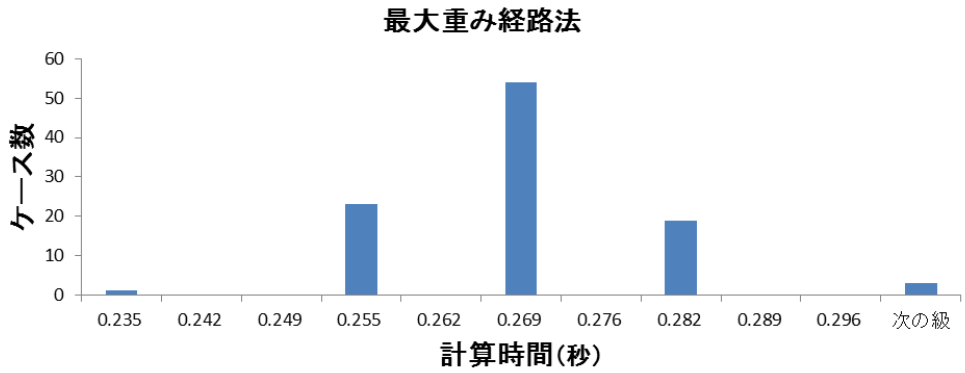

図 7.4　網グラフ n=2,500，m=9,702，最大重み経路法を用いた計算時間のヒストグラム

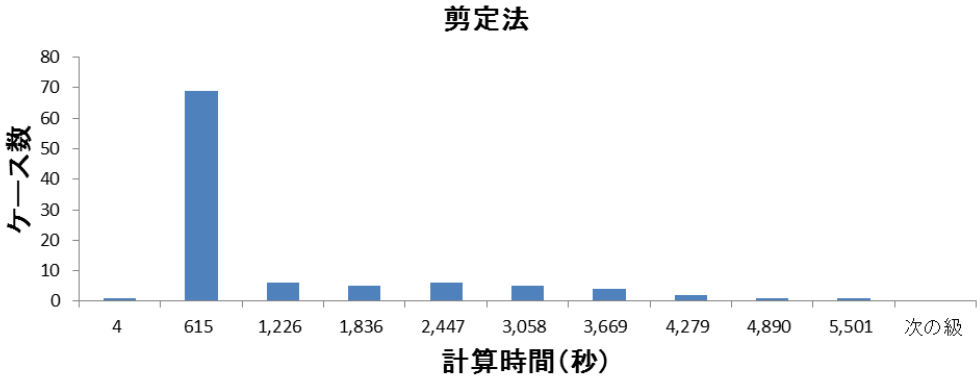

図 7.5　ランダムグラフ n=2,500，m=9,702，剪定法を用いた計算時間のヒストグラム

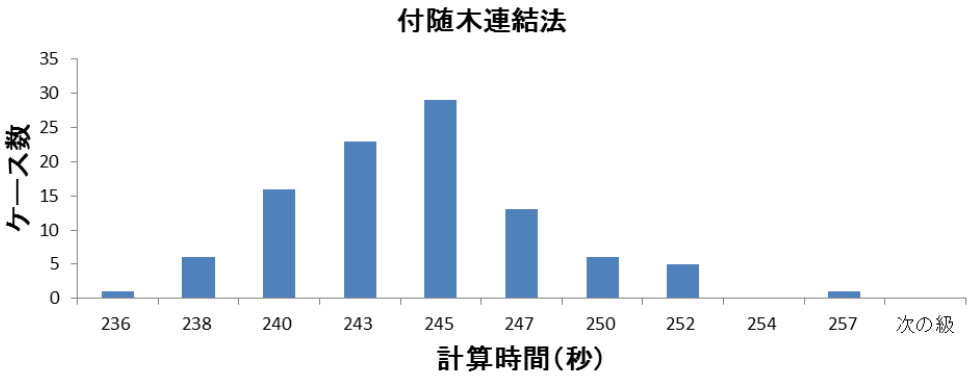

図 7.6　ランダムグラフ n=2,500，m=9,702，付随木連結法を用いた計算時間のヒストグラム

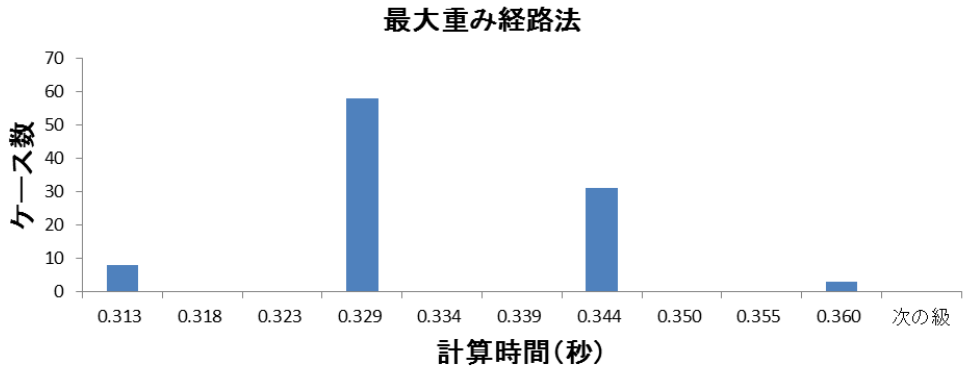

図 7.7　ランダムグラフ n=2,500，m=9,702，最大重み経路法を用いた計算時間のヒストグラム

第 8 章

結論と今後の課題

$MPRS$ は，ケーブル TV，ガス，水道，電気事業など様々な事業に適用できる．さらに，入力を木に限定することにより，多くの NP 完全問題が多項式時間で解けることが知られている．$MPRS$ で求めた木に対して，何らかの NP 完全問題を多項式時間で解くことができる．

$MPRS$ は重要な問題であり，この問題に対するアルゴリズムはいくつか提案されていた．しかし，最適解を必ずしも求められなかった．NP 困難性の証明を行ったことが本研究の 1 つの貢献である．また，$MPRS$ を MIP として定式化した．そのため，ソルバー CPLEX を使うことができた．さらに，すでに提案されている付随木連結法と最大重み経路法のアルゴリズムを詳しく書き直し，剪定法を提案した．剪定法を実装して行った計算実験では，3 つの手法が異なる特徴を持つことと，良い近似比が得られたことを示せた．

今後の課題としては，近似比の保証つきアルゴリズムの開発が考えられる．

参考文献

[1] 古林隆，福馬敏子，榊原英行，露木真理子：“最大利益根付き問題のアルゴリズム，”日本オペレーションズ・リサーチ学会：*2003* 年秋季研究発表会アブストラクト集，pp.166-167，2003.

[2] 古林隆，福馬敏子：“最大利益根付木問題のアルゴリズム II -最大重み経路法-，”日本オペレーションズ・リサーチ学会：*2005* 年春季研究発表会アブストラクト集，pp.116-117，2005.

[3] Jon Kleinberg and Eva Tardos: “Algorithm Design,” First Edition, Chapters 3, 8, 10, Addison-Wesley, 2006.

[4] B. Korte and J. Vygen: “Combinatorial Optimization: Theory and Algorithms,” Fourth Edition, Chapters 6, 7, 22, Berlin, Heidelberg, Springer-Verlag, 2008.

[5] A. V. Aho, J. E. Hopcroft, J. D. Ullman: “The Design and Analysis of Computer Algorithms,” First Edition, Chapter 2, Addison-Wesley, 1974.

[6] Y.S. Myung, C.H. Lee, and D.W. Tcha: “On the generalized minimum spanning tree problem,” *Networks,* vol. 26, No. 4, pp. 231-241, 1995.

[7] Corinne Feremans, Martine Labbe, Gilbert Laporte: “A Comparative Analysis of Several Formulations for the Generalized Minimum Spanning Tree Problem,” *Networks,* vol.39, No. 1, pp. 29-34, 2002.

[8] S.C. Narula and C.A. Ho: “Degree-constrained minimum spanning tree,” *Computers and Operations Research,* vol. 7, No. 4, pp. 239-249, 1980.

[9] Alexandre Salles da Cunha, Abilio Lucena: “Lower and Upper Bounds for the Degree-Constrained Minimum Spanning Tree Problem,” *Networks,* vol. 50, No. 1, pp. 55-66, 2007.

[10] G.Reich and P.Widmayer: “Beyond Steiner’s problem: A vlsi oriented generalization,” Lecture notes in computer science, Vol. 411, pp. 196-211, Springer Verlag, 1989.

[11] F.K. Hwang, D.S.Richards, and P. Winter: “The Steiner tree problem,” North-

Holland, 1992.

[12] C.S. Helvig, Gabriel Robins, Alexander Zelikovsky: “An Improved Approximation Scheme for the Group Steiner Problem,” *Networks,* vol. 37, No. 1, pp. 8-20, 2001.

[13] V.V. Rao and R. Sridharan: “Minimum-weight rooted not-necessarily-spanning arborescence problem,” *Networks,* vol. 39, No. 2, pp. 77–87, 2002.

[14] Saul I. Gass, Carl M. Harris: “Encyclopedia of Operations Research and Management Science,” First Edition, Kluwer Academic Publishers, Boston, Massachusetts, USA, 1996.

[15] C.E. Miller, A.W. Tucker and R.A. Zemlin: “Integer programming formulations and traveling salesman problems,” *Association for Computing Machinery,* vol. 7, pp.326–329, 1960.

[16] 久保幹雄：“ロジスティクスの数理，” 第 6 章，共立出版，2007.

[17] H.N.Gabow, Z.Galil, T.Spencer and R.E.Tarjan: “Efficient algorithms for finding minimum spanning tree in undirected and directed graphs,” *Combinatorica,* vol. 6, pp.109–122, 1986.

[18] 浅野 哲夫，小保方 幸次：“LEDA で始める C/C++ プログラミング，” サイエンス社，2002.

研究業績

- Japan-Korea Joint Workshop on Algorithms and Computation (WAAC) (2012/7/10〜11 東京).
- Industrial Engineering and Engineering Management (IEEM) (2012/12/10〜13 香港).

謝辞

この研究自体は学部時代の指導教員でもある千葉英史助教にご指導いただきました。千葉先生には、学部時代からさまざまなことで本当にお世話になりました。先生のおかげで、計算機とは何か、数学やアルゴリズムとは何か、無矛盾な世界はどんなところか、それらの全体イメージを自分の中に創ることができました。

この論文の体裁や構成、言葉使いなどは大学院の指導教員である五島洋行准教授にご指導いただきました。五島先生にも、色々なことで本当にお世話になりました。また、五島ゼミでは、概念や情報の伝え方に重みをおいています。人に何かを伝えるためには、バランス感覚と、いかに自分のことを客観的に見れるかが、重要であると学びました。

大学院一年生時、私の指導教員は、現在ご退職された古林隆教授でした。$MPRS$ は古林先生の提案した問題であり、最大重み経路法は先生が開発したアルゴリズムです。以前、古林先生は「アルゴリズムは美しくなければいけない」とおっしゃっていました。私は当時なんとなくでしか了解していなかったですが、最大重み経路法を理解するにしたがい意味がわかってきたような気がします。

三人の先生方には、学問の深い森を見せていただきました。その経験はとても貴重なものです。深く感謝いたします。

千葉ゼミの後輩、五島ゼミの後輩、仲間にもお世話になりました。それぞれのキャラクターが濃くて、思い出がたくさん頭に浮かびます。

また、論文には書かれていない光の当たらない苦労が、研究にはあります。本研究では、プログラムのバグとりや計算実験の諸々、データのまとめ作業などです。バグがなかなか取れなかったり、データの整理に手違いが生じたりして、イライラしてしまったこともありました。そのような理不尽な気持ちを受け入れてくれた家族や彼女、友達に感謝します。

最後に、この論文を読んでくれるかもしれない未来の「あなた」にも感謝します。ぜひ、新しいページを書き加えてください。