機会制約組合せ最適化問題の効率的厳密解法

一森哲男†

機会制約のある組合せ最適化問題を解く，計算効率の良いアルゴリズムを 2 つ提案した．ここで扱ったものは，よく知られた機会制約のある最小全域木問題をより一般的な組合せ最適化問題に拡張した問題である．機会制約のある最小全域木問題を解くための従前のアルゴリズムに比べ，ここで開発したアルゴリズムはどちらも計算効率が飛躍的に向上した．

Efficient Algorithms for Combinatorial Optimization Problems subject to a Chance Constraint

Tetsuo Ichimori†

We propose two algorithms for solving combinatorial optimization problems subject to a chance constraint. The combinatorial optimization problems subject to a chance constraint include the minimum spanning tree problem under a chance constraint which is well known in stochastic programming. Our empirical experiments show that our algorithms are far more efficient than the existing algorithm for finding a minimum spanning tree under a chance constraint.

1. はじめに

グラフ上の最適化問題では，多くの場合，辺 j の重みは確定値であるが，本論文では，それは平均 m_j，分散 $v_j > 0$ の正規分布に従うと仮定する．また，これらの重みを表す確率変数 X_j は互いに独立とする．この仮定の下で，グラフの最小全域木問題や最短経路問題，あるいは二部グラフ上の割当て問題などの組合せ最適化問題を取り扱う．具体的にいえば，ある組合せに対応する確率変数の和が目標値 f 以下となる確率が，定められた信頼度 α 以上となる全域木，経路，マッチングなどが存在するという条件下で f を最小化する問題を扱う．

このような問題は，辺の重みが確定値でないため，入手したデータにばらつきがあったり，誤差があったりするような状況下では有効である．たとえば，ノード間を結ぶケーブルの埋設費用などは大きなばらつきが生じる．2 地点間の移動時間も，ある時間帯に限定しても，かなりばらつきが存在する．実際上，誤差のないデータはまれである．また，中心極限定理より，確率変数の和はそれぞれの確率変数がどのような分布に従っていても，(独立性やその他の仮定の下で) 辺の数が数十程度以上のグラフの問題では，正規分布に従うものと見なすことができるので，ここで考察する問題は，多くの現実問題にも有効と思われる．

全域木，経路，マッチングなどを一般に記号 T で表し，T 全体の集合を $\mathcal{F}$ で表すと，我々の問題は以下のように定式化できる．

$$(\mathrm{P}_0)\quad \min_{T\in\mathcal{F}} \quad f \qquad \text{s.t.} \quad \Pr\left\{\sum_{j\in T} X_j \le f\right\} \ge \alpha$$

ここで提案した，問題 (P_0) の解法は，より一般的な次の組合せ最適化問題 $(\mathrm{P}_{\min})$ や $(\mathrm{P}_{\max})$ を解くことができる．すなわち，n 次元 0/1 変数ベクトル $\boldsymbol{y}$ とその実行可能領域 Y が定義されている．さらに，各成分 y_j に対して独立な確率変数 X_j も定義されており，確率変数の和 $\sum_{j=1}^n X_j y_j$ は正規分布に従うとする．

$$(\mathrm{P}_{\min})\quad \min_{\boldsymbol{y}\in Y} \quad f \qquad \text{s.t.} \quad \Pr\left\{\sum_{j=1}^n X_j y_j \le f\right\} \ge \alpha$$

$$(\mathrm{P}_{\max})\quad \max_{\boldsymbol{y}\in Y} \quad g \qquad \text{s.t.} \quad \Pr\left\{\sum_{j=1}^n X_j y_j \ge g\right\} \ge \alpha$$

† 大阪工業大学
Osaka Institute of Technology

しかしながら，以下の議論では，簡単化のため (P_0) を用いて議論する．

問題 (P_0) はよく知られた，機会制約計画[1)~3)] に属する問題と考えられるが，組合せ最適化を対象とした研究は文献 7) が初めてである．これは (P_0) の特別な場合，すなわち，T がグラフの全域木の場合を扱っており，最適な全域木を多項式時間で発見するアルゴリズムを提案している．そのアルゴリズムは，グラフの辺 j の重みがパラメトリックな重み $m_j + v_j\lambda$ として，全域木 T の重み $(\sum_{j\in T} m_j) + (\sum_{j\in T} v_j)\lambda$ を最小にする全域木をすべて列挙することにより最適な全域木を発見している．

この研究は次のように発展した．1 つ目は単にアルゴリズムそのものの改善を目的にした研究[5)] である．2 つ目はパラメトリックな重みを持つグラフの最小全域木の全列挙の効率化の研究[4)] であり，その副産物として，効率的に最適な全域木を発見している．3 つ目は，そのアイデアを拡張し，他の組合せ最適化問題に適応した研究[6),8)] である．

従来のアルゴリズムはパラメトリックな重み $(\sum_{j\in T} m_j) + (\sum_{j\in T} v_j)\lambda$ を最小にする全域木を全列挙しているため，その全域木の数が多項式オーダであっても，計算効率はきわめて悪いことが予想される．著者の知る限り，実際の数値実験を発表した研究は存在しないようである．

本論文の目的は，文献 7) の問題を組合せ最適化問題 (P_0) に拡張し，実際の数値計算で効率的に最適解を発見できるアルゴリズムを開発することである．ただし，グラフの辺の重みが確定値であっても，効率的に解けない NP 困難な問題も多数存在するため，問題 (P_0) そのものがつねに効率的に解けるわけではない．ここでは，(P_0) の確定版問題，つまり，グラフの辺 j の重みを確定値 w_j とした普通の最適化問題：$\min_{T\in\mathcal{F}} \sum_{j\in T} w_j$ のインスタンスをいくつ解くかで，計算効率を議論する．非常に多くの組合せ最適化問題が存在することを考えれば，このような単純な基準は妥当なものと思われる．

本論文で扱った問題での数値実験結果の範囲内では，ここで提案したアルゴリズムは効率が良く，(P_0) の確定版問題のインスタンスを数個解くだけで元問題 (P_0) を解くことに成功した．

2. 解　　法

組合せ T に対応する確率変数 $X(T) = \sum_{j\in T} X_j$ は平均 $m(T) = \sum_{j\in T} m_j$, 分散 $v(T) = \sum_{j\in T} v_j$ を持つ正規分布に従うことに注意しながら適当な変数変換をすると，問題 (P_0) は以下の問題に書き換えることができる[3),7)]．

$$\min_{T\in\mathcal{F}} \quad m(T) + \sqrt{v(T)} \tag{1}$$

各 T に対し，v-m 平面上に点 $(v(T), m(T))$ を定義する．関数 $f(v,m) = \sqrt{v} + m$ は凹関数なので，これの最小値は，$\{(v(T), m(T)) \mid T \in \mathcal{F}\}$ の凸包の端点で実現する．この凸包を $\mathcal{H}$ とすると，元問題 (P_0) は次の最適化問題 (P) に帰着される．

$$\text{(P)} \qquad \min_{(v,m)\in\mathcal{H}} \sqrt{v} + m.$$

各端点 (v,m) は他の端点 (v',m') に対し，$v' \le v$, $m' \le m$ ならば $(v',m') = (v,m)$ となるとき，効率的端点と呼ばれる．明らかに，効率的端点で (P) の最小値が実現する．

正の定数 λ に対し，次の問題を考える．

$$\min_{(v,m)\in\mathcal{H}} \quad \lambda v + m. \tag{2}$$

この問題は

$$\text{(Q)} \qquad \min_{T\in\mathcal{F}} \quad \lambda v(T) + m(T)$$

と同じである．あるいは，$w_j = m_j + \lambda v_j$ を辺 j の重みとした，(P_0) 確定版問題と等価である．問題 (Q) の最適解は最適な組合せ T のことであるが，説明の便宜上，以下の議論では点 $(v(T), m(T))$ を意味することにする．また，この点 $(v(T), m(T))$ は効率的端点であることに注意する．

λ の値をいろいろ変化させることにより凸包 $\mathcal{H}$ の効率的端点を見つけることができる．しかし，すべての効率的端点を見つけるために，変化させるべき λ の異なる値の数は少なくとも効率的端点の数以上に必要である．しかし，必要なものは (P) の最適な端点のみで，すべての効率的端点を見つける必要はない．そこで，問題 (Q) を解くことにより，効率的に (P) の最適な端点を見つける方法を以下で考える．

凸包 $\mathcal{H}$ の異なる効率的端点が総計 $n \ge 2$ 個あるとして，v 座標の小さいものから順に番号を与える．

$$(v^0, m^0), (v^1, m^1), \ldots, (v^{n-1}, m^{n-1})$$

$1 \le k \le n-1$ に対し，2 点 (v^{k-1}, m^{k-1}), (v^k, m^k) を結ぶ線分の傾きの絶対値を s_k とすると

$$s_k = \frac{m^{k-1} - m^k}{v^k - v^{k-1}}$$

となる．このとき，$\mathcal{H}$ の凸性より $s_1 > s_2 > \ldots > s_{n-1}$ となる．また，$s_k > \lambda > s_{k+1}$ とした問題 (Q) の唯一の最適解は端点 (v^k, m^k) である．一方，$\lambda = s_k$ とした問題 (Q) の最適解は，端点 (v^{k-1}, m^{k-1}) と

(v^k, m^k) であるが，これら以外にも，この両端点を結ぶ線分上に (Q) の最適解が存在することもありうる．しかし，このような点は (P) の最適解にはなりえない．理由は次のとおりである．両端点 (v^{k-1}, m^{k-1}) と (v^k, m^k) を結ぶ線分上の点（両端点は除く）を (v', m') とすると，関数 $f(v, m) = \sqrt{v} + m$ はこの線分上で狭義の凹なので

$$\min\left\{\sqrt{v^{k-1}}+m^{k-1}, \sqrt{v^k}+m^k\right\} < \sqrt{v'}+m'$$

となるからである．よって，この場合，両端点 (v^{k-1}, m^{k-1}) と (v^k, m^k) のみを解に採用する．

いま，問題 (Q) を解くことにより，問題 (P) の最適な効率的端点を効率的に見つける方法を考えているが，効率的端点の座標を事前に知ることはできないので，連続する効率的端点を結ぶ線分の傾きも知ることはできない．そこで，順に効率的端点を 1 つずつ，ただし，必要なものだけ，明らかにすることにより最適な端点を探し出す．

2 つの正の数字 λ_A と λ_B は $\lambda_A \geq s_1$，$0 < \lambda_B \leq s_{n-1}$ を満たすとする．$\lambda = \lambda_A$ および $\lambda = \lambda_B$ とした問題 (Q) を解けば，v-m 平面上の端点 (v^0, m^0) および (v^{n-1}, m^{n-1}) がそれぞれ見つかる．いま，$A = (v^0, m^0)$，$B = (v^{n-1}, m^{n-1})$ とする．点 A を通り，傾きが $-\lambda_A$ の直線を l_A とする．同様に，点 B を通り，傾きが $-\lambda_B$ の直線を l_B とする．仮定より $\lambda_A \neq \lambda_B$ なので，この両直線は必ず v-m 平面上で交わる．その交点を H とする．このとき，$\mathcal{H}$ の凸性よりすべての効率的端点は $\triangle ABH$ に含まれる．

v-m 平面上の任意の点 $X = (v, m)$ に対して，記号

$$v = X_v, \quad m = X_m, \quad f(X) = \sqrt{v} + m$$

を定義する．次に，不等式

$$f(H) \geq \min\{f(A), f(B)\} \tag{3}$$

が成立する場合とそうでない場合を考える．まず，式 (3) が成り立つときを考える．このとき，式 (3) の右式の最小値を与える端点 A または B を除き，さらに，等号が成立するときは交点 H を除き，$\triangle ABH$ は曲線

$$\sqrt{v} + m = \min\{f(A), f(B)\}$$

より，完全に上側に存在する．よって，次の定理が成り立つ．

定理 1 不等式 (3) が成り立つとき，(3) の右辺の最小値を与える端点 A または B より完全に良い解（効率的端点）は $\triangle ABH$ の完全な内部には存在しない．

つまり，式 (3) が成り立てば問題 (P) は解決する．式 (3) の右式の最小値を与える端点 A または B が問題 (P) の最適解である．

次に，式 (3) が成り立たない場合を以下に考える．

$$\lambda_A > \lambda_C > \lambda_B$$

を満足する，ある $\lambda = \lambda_C$ に対して，A，B 以外の別の端点 $C = (v^k, m^k)$ が見つかったとする．この仮定は $n \geq 3$ と $s_k \geq \lambda_C \geq s_{k+1}$ を意味する．点 C を通り，傾きが $-\lambda_C$ の直線を l_C とする．

この仮定のもとでは，直線 l_A と l_C は必ず 1 点で交わり，その交点を I，直線 l_B と l_C も必ず 1 点で交わり，その交点を J とするとき，端点 $A = (v^0, m^0), \ldots, (v^k, m^k) = C$ は $\triangle ACI$ に含まれ，端点 $C = (v^k, m^k), \ldots, (v^{n-1}, m^{n-1}) = B$ は $\triangle CBJ$ に含まる．このとき，問題 (Q) を 1 回解くことにより，最適な端点が存在する領域 $\triangle ABH$ を $\triangle ACI$ または $\triangle CBJ$ に限定できる．

上記の手法を再帰的に用いれば，最適な端点を発見することができる．以下で，全体の手続きを詳しく述べる．

3. アルゴリズム A

$\triangle ABH$ の 3 頂点 A，B，H が見つかった後，λ_C の値の選び方が問題となる．λ_C は $\lambda_A > \lambda_C > \lambda_B$ となる無限の値の中から適当なものを選び出せばよいが，ここでは

$$\lambda_C = \text{線分 } AB \text{ の傾きの絶対値} \tag{4}$$

と選ぶ．この選び方だと，簡単な計算より $\triangle ACI$ と $\triangle CBJ$ の面積の和は $\triangle ABH$ の面積の 4 分の 1 以下になり，最適な端点が存在する領域が 4 分の 1 以下に減少する．

問題 ($\mathbf{P}_0$) を解くアルゴリズム A

(1) $w_j = v_j$ として問題 (Q) を解き，端点 $A = (A_v, A_m)$ と直線 l_A を定める．直線 l_A は縦軸つまり m 軸に平行で，$v = A_v$ とする．

(2) $w_j = m_j$ として問題 (Q) を解き，端点 $B = (B_v, B_m)$ と直線 l_B を定める．直線 l_B は横軸つまり v 軸に平行で，$m = B_m$ とする．

(3) $H = (A_v, B_m)$ とし $\triangle ABH$ を定義する．

(4) 探索すべき三角形の集合 $D = \{\triangle ABH\}$ を定義する．

(5) 最適値の上限 $U = \min\{f(A), f(B)\}$.

(6) 最適端点の候補 $E =$ (U を定義する端点) とする．

(7) D の中から三角形を 1 つ選び，それを $\triangle ABH$ とする．$D = D \backslash \{\triangle ABH\}$.

(8) $f(H) < U$ ならば下記のステップを実行する．

(a) $\lambda_C = \left|\text{線分 } AB \text{ の傾き}\right|$ とする．

(b) $w_j = m_j + \lambda_C v_j$ として問題 (Q) を解き，端点 C と直線 l_C を定める．

(c) $C = A$ または $C = B$ でなければ以下のステップを実行する．

(i) $U = \min\{U, f(C)\}$, $E = (U$ を定義する端点 $)$ とする．

(ii) l_A と l_C の交点 I を求める．

(iii) l_B と l_C の交点 J を求める．

(iv) $f(I) < U$ ならば $D = D \cup \{\triangle ACI\}$.

(v) $f(J) < U$ ならば $D = D \cup \{\triangle CBJ\}$.

(9) $D = \emptyset$ ならば最適端点を E と定義する．E に対応する組合せ T を出力しストップ：問題 (P_0) の最適解の発見．そうでなければ，ステップ (7) へ戻る．

4. アルゴリズム B

以前に述べたが，λ_C の選び方は無数にある．そこで，ここでは別な選び方を考えてみる．v-m 平面上で効率的端点 (v^k, m^k) は曲線

$$\sqrt{v} + m = \sqrt{v^k} + m^k \tag{5}$$

上に存在している．効率的端点 (v^k, m^k) での接線は

$$m - m^k = -\frac{1}{2\sqrt{v^k}}(v - v^k) \tag{6}$$

である．

曲線 (5) 上に存在するか，あるいは，この曲線より完全に上側に存在する効率的端点 (v^i, m^i) は明らかに

$$\sqrt{v^i} + m^i \geq \sqrt{v^k} + m^k \tag{7}$$

を満足するし，その逆も成り立つ．不等式 (7) を満足する効率的端点 (v^i, m^i) は，端点 (v^k, m^k) を除き，直線 (6) より完全に上側に存在するので，これらの端点は決して $\lambda_C = 1/(2\sqrt{v^k})$ としたときの問題 (Q) の最適解には選ばれない．よって，次の定理が成り立つ．

定理 2 $\lambda_C = 1/(2\sqrt{v^k})$ とした問題 (Q) を解いたとき，k 番目の端点 (v^k, m^k) が見つかるか，または，k 番目の端点 (v^k, m^k) より完全に好ましい別の端点が見つかる．

この定理の前半で効率的端点 (v^k, m^k) が見つかれば，この端点を不動点と呼ぶ．効率的端点 (v^k, m^k) が問題 (P) の最適解ならば，この端点は不動点であるが，定理の後者の場合は生じないので，この端点が $\lambda_C = 1/(2\sqrt{v^k})$ としたときの問題 (Q) の唯一の最適解であることを意味する．ただし，このことは必ずしも問題 (P) の最適解が唯一であることを意味しない．実のところ，複数の最適解が存在する数値例を作成することは可能である．

さらに，この λ_C の選び方では，次のような大きな長所がある．

定理 3 2 つの正の数 $\lambda_A > \lambda_B$ に対し，端点 $A = (v^0, m^0)$, $B = (v^{n-1}, m^{n-1})$ が見つかったとする．直線 l_A と直線 l_B も以前同様に定義し，その交点を H とする．2 つの不等式 $\lambda_A > 1/(2\sqrt{A_v}) > \lambda_B$, $\lambda_A > 1/(2\sqrt{B_v}) > \lambda_B$ が成り立つと仮定する．$\lambda_C = 1/(2\sqrt{A_v})$ とおいた問題 (Q) を解いたとき，最適解として A 以外の別の端点 $C = (v^k, m^k)$, $k \neq 0$ が見つかったとする．直線 l_C も以前同様に定義する．直線 l_A と直線 l_C の交点を $I = (I_v, I_m)$ とする．このとき $\triangle ACI$ 内部には最適解は存在しない．また，$\lambda_C = 1/(2\sqrt{B_v})$ とおいて B 以外の別の端点 C が見つかったとする．直線 l_C も同様に定義する．直線 l_B と直線 l_C の交点を J とする．このとき $\triangle CBJ$ 内部には最適解は存在しない．

証明 後半も内容は同じなので，前半のみ証明する．$k = 0$ とした接線 (6) は当然のことながら，端点 (接点) $A = (v^0, m^0)$ を通るが，この直線上では関数 $f(v, m)$ は，端点 A で最大値をとる狭義の凹関数である．よって，この接線上で A から離れるほど関数値 $f(v, m)$ は減少する．また，3 点 A, I, C の v 座標 A_v, I_v, C_v（それぞれ）に関して $A_v = v^0 < I_v < v^k = C_v$ であることに注意する．直線 l_C 上に存在する点 C と I を m 軸にそって平行移動させ，点 C と I が，端点 A を通る $k = 0$ とした接線 (6) 上に存在するようにする．両点の移動先の点をそれぞれ C' と I' とする．このとき両者の関数値の差は不変である：

$$f(C') - f(I') = f(C) - f(I)$$

また，接線上で点 C' のほうが I' より A から遠いので $f(C') < f(I')$ となる．つまり，$f(C) < f(I)$ となり，$f(I) \geq \min\{f(A), f(C)\}$ を満たしていることが分かる．よって，定理 1 より $\triangle ACI$ の完全な内部には問題 (P) の最適解は存在しない．(証明終わり)

この定理 3 によれば，$\triangle ABH$ の内部を探索する際

$$\lambda_C = \frac{1}{2\sqrt{A_v}} \tag{8}$$

または

$$\lambda_C = \frac{1}{2\sqrt{B_v}} \tag{9}$$

とすれば，以前と異なり $\triangle ABH$ の探索を $\triangle ACI$ または $\triangle CBJ$ の一方の探索に限定できる．ただし，定理 2 により，効率的端点 A と B が不動点の場合，

$\triangle ABH$ の内部に A, B 以外に別の効率的端点が存在しても，それらの新しい効率的端点が見つからないという欠点もある．だから，式 (8) や (9) のように λ_C を選べば，計算効率が必ず上昇するというものではない．

次に，定理 3 に基づき，λ_C を式 (8) または (9) を満たすように選ぶ部分をアルゴリズム A に追加したアルゴリズム B を提案する．変更部分はアルゴリズム A のステップ (8) に以下の (a)–(b) を追加しただけである．

問題 (P_0) を解くアルゴリズム B

(8) $f(H) < U$ ならば下記のステップを実行する．

(a) 端点 B が不動点と宣言されず，$\lambda_A > 1/(2\sqrt{B_v}) > \lambda_B$ ならば $w_j = m_j + v_j/(2\sqrt{B_v})$ として問題 (Q) を解き，端点 C と直線 l_C を定める．$C = B$ ならば端点 B を不動点と宣言する．そうでなければ以下のステップを実行する．

(i) $U = \min\{U, f(C)\}$, $E = (U$ を定義する端点 $)$ とする．

(ii) l_A と l_C の交点 I を求める．

(iii) $f(I) < U$ ならば $D = D \cup \{\triangle ACI\}$.

(iv) ステップ (9) へ．

(b) 端点 A が不動点と宣言されず，$\lambda_A > 1/(2\sqrt{A_v}) > \lambda_B$ ならば $w_j = m_j + v_j/(2\sqrt{A_v})$ として問題 (Q) を解き，端点 C と直線 l_C を定める．$C = A$ ならば端点 A を不動点と宣言する．そうでなければ以下のステップを実行する．

(i) $U = \min\{U, f(C)\}$, $E = (U$ を定義する端点 $)$ とする．

(ii) l_B と l_C の交点 J を求める．

(iii) $f(J) < U$ ならば $D = D \cup \{\triangle CBJ\}$.

(iv) ステップ (9) へ．

(c) $\lambda_C = \left|$線分 AB の傾き$\right|$ とする．

(d) $w_j = m_j + \lambda_C v_j$ として問題 (Q) を解き，端点 C と直線 l_C を定める．

(e) $C = A$ または $C = B$ でなければ以下のステップを実行する．

(i) $U = \min\{U, f(C)\}$, $E = (U$ を定義する端点 $)$ とする．

(ii) l_A と l_C の交点 I を求める．

(iii) l_B と l_C の交点 J を求める．

(iv) $f(I) < U$ ならば $D = D \cup \{\triangle ACI\}$.

(v) $f(J) < U$ ならば $D = D \cup \{\triangle CBJ\}$.

5. 数値実験

上記のアルゴリズムを評価するために数値実験を行った．各 m_j の値は 5 通りの一様乱数（整数）で与えた：(a) $L \le m_j \le L$+1,000, (b) $L \le m_j \le L + 500$, (c) $L \le m_j \le L + 50$, (d) $L \le m_j \le L + 10$, (e) $L \le m_j \le L + 5$ と順にばらつきの幅を小さくしていった．ここで，乱数の下限 L は以下の数値実験では 450 としたが，この L の値は，すべての m_j が一定値だけ変化するのであれば，どんな値でも結果は同じである．これは構成される凸包 $\mathcal{H}$ が v-m 平面上で上下に平行移動するだけで，目的関数 $f(v, m) = \sqrt{v} + m$ の最小化には影響しないからである．

また，各 $s_j = \sqrt{v_j}$ の値も以下の 5 通りの一様乱数（整数）で与えた．(1) $10 \le s_j \le 200$, (2) $10 \le s_j \le 160$, (3) $10 \le s_j \le 120$, (4) $10 \le s_j \le 80$, (5) $10 \le s_j \le 40$ と分散の値が徐々に小さくなるように，つまり，徐々に問題が確定的な問題に近づくようにした．

これらの組合せは 25 通りあるが，簡単のため，m_j の幅を変化させるときは $10 \le s_j \le 200$ を仮定し，s_j の幅を変化させるときは $450 \le m_j \le 550$ を仮定した．ただし，以下の結果から，計算複雑度の数値は m_j や s_j の幅の変化にあまり依存していない．

それぞれのケースでは 100 個の問題を作成した．以下に示した数値実験結果の時間複雑度は，1 つの問題 (P_0) を解くのに平均いくつの問題 (Q) のインスタンスを解いたかを示している．一方，空間複雑度は 100 問のうちで一度に記憶した三角形の最大数と定義した．

5.1 最小全域木

100 頂点の完全無向グラフ K_{100} を考えた．**表 1** に示したように，アルゴリズム B ではたかだか 8 つの問題 (Q) のインスタンスを解いている．グラフの辺の総数は 4,950 であるので，アルゴリズム[7] では $4{,}950 \times (4{,}950\text{-}1)/2 \doteqdot 1.2 \times 10^7$ 個の問題 (Q) のインスタンスを解く．よって，ここで開発したアルゴリズム B はアルゴリズム[7] と比べて，約 150 万倍計算効率が優れていることが分かる．

5.2 最短経路

70×70 の正方格子グラフを考えた．2 つの自然数 $1 \le i \le 70$, $1 \le j \le 70$ に対し，頂点 v_{ij} は平面上の格子点 (i, j) に存在する．右向き有向辺 $(v_{ij}, v_{i\,j+1})$ と上向き有向辺 $(v_{ij}, v_{i+1\,j})$ の 2 種類を考える．有向

表 1 アルゴリズム A と B の複雑度
Table 1 Complexity of Algorithms A and B.

m_j, s_j の範囲	時間	空間	時間	空間
$450 \leq m_j \leq 1{,}450$	11.20	3	7.17	1
$450 \leq m_j \leq 950$	11.65	3	7.24	1
$450 \leq m_j \leq 500$	11.37	3	8.10	1
$450 \leq m_j \leq 460$	11.45	2	8.08	1
$450 \leq m_j \leq 455$	11.45	2	7.90	1
$10 \leq s_j \leq 200$	11.89	4	7.91	1
$10 \leq s_j \leq 160$	11.32	3	7.91	1
$10 \leq s_j \leq 120$	11.55	3	7.72	1
$10 \leq s_j \leq 80$	11.56	3	7.48	1
$10 \leq s_j \leq 40$	11.29	3	6.93	1

完全グラフ K_{100} 上の最小全域木問題

表 2 アルゴリズム A と B の複雑度
Table 2 Complexity of Algorithms A and B.

m_j, s_j の範囲	時間	空間	時間	空間
$450 \leq m_j \leq 1{,}450$	8.17	1	4.46	1
$450 \leq m_j \leq 950$	8.16	1	4.67	1
$450 \leq m_j \leq 500$	8.44	2	5.78	1
$450 \leq m_j \leq 460$	8.25	2	6.03	1
$450 \leq m_j \leq 455$	8.18	2	5.80	1
$10 \leq s_j \leq 200$	8.09	2	5.29	1
$10 \leq s_j \leq 160$	8.31	2	5.19	1
$10 \leq s_j \leq 120$	8.25	2	5.09	1
$10 \leq s_j \leq 80$	8.02	2	4.97	1
$10 \leq s_j \leq 40$	8.17	2	4.69	1

正方格子グラフ上の最短経路問題

表 3 アルゴリズム A と B の複雑度
Table 3 Complexity of Algorithms A and B.

m_j, s_j の範囲	時間	空間	時間	空間
$450 \leq m_j \leq 1{,}450$	9.68	3	5.87	1
$450 \leq m_j \leq 950$	10.06	3	6.45	1
$450 \leq m_j \leq 500$	10.02	3	7.37	1
$450 \leq m_j \leq 460$	10.02	3	7.42	1
$450 \leq m_j \leq 455$	9.32	2	7.33	1
$10 \leq s_j \leq 200$	10.24	3	7.32	1
$10 \leq s_j \leq 160$	10.25	3	7.39	1
$10 \leq s_j \leq 120$	10.28	3	7.04	1
$10 \leq s_j \leq 80$	10.09	2	6.89	1
$10 \leq s_j \leq 40$	9.73	3	6.24	1

完全二部グラフ $K_{120\,120}$ 上の割当問題題

辺は合計 $70 \times (70-1) \times 2 = 9{,}660$ 本ある．始点を頂点 $v_{1\,1}$ とし終点を $v_{70\,70}$ とした．結果を**表 2** に示す．

5.3 割当問題

完全二部グラフ $K_{120\,120}$ を考えた．辺の総数は $120^2 = 14{,}400$ である．結果を**表 3** に示す．アルゴリズム B ではせいぜい 7 個の問題 (Q) のインスタンスを解いているだけである．また，同時に記憶する三角形も常時 1 つしかない

6. あとがき

機会制約組合せ最適化問題を解く効率の良いアルゴリズムを 2 つ提案した．ここでの数値実験結果では，アルゴリズム B は総じてアルゴリズム A よりも，効率が良かった．特に，空間複雑度が 1 になっており，常時記憶した三角形の数は 1 つだけである．

さまざまなタイプの組合せ最適化問題に，ここで提案したアルゴリズムを適用したわけではないので，断言はできないが，ここので数値実験結果はアルゴリズム A，B の効率の良さを強く示唆している．

最後に，問題 (Q) が多項式時間で解ける場合の，問題 (P_0) を解く我々のアルゴリズムの計算量について述べる．問題 (Q) を 1 つ解くことは凸包 $\mathcal{H}$ の端点を 1 つ見つけることに対応している．だから，この凸包の端点の総数が多項式オーダであれば，我々のアルゴリズムは多項式時間になる．最小全域木問題は文献 6)，7) に示されているように，凸包の端点の総数が多項式オーダである．しかしながら，一般には，端点の総数が多項式オーダとはいえないので，現時点では，ここでのアルゴリズムは多項式時間とはいえない．

参考文献

1) Charnes, A. and Cooper, W.W.: Chance–Constrained Programming, *Management Science*, Vol.6, No.1, pp.73–79 (1959).
2) Charnes, A. and Cooper, W.W.: Chance Constraints and Normal Deviates, *Journal of the American Statistical Association*, Vol.57, No.297, pp.134–148 (1962).
3) Charnes, A. and Cooper, W.W.: Deterministic Equivalents for Optimizing and Satisficing under Chance Constraints, *Operations Research*, Vol.11, No.1, pp.18–39 (1963).
4) Fernandez-Baca, D., Slutzki, G. and Eppstein, D.: Using Sparsification for Parametric Minimum Spanning Tree Problems, *Nordic Journal of Computing*, Vol.3, No.4, pp.352–366 (1996).
5) Geetha, S. and Nair, K.P.K.: On Stochastic Spanning Tree Problm, *Networks*, Vol.23, pp.675–679 (1993).
6) Ichimori, T., Shiode, S., Ishii, H. and Nishida, T.: Minimum Spanning Tree with Normal Variates as Weights, *Journal of the Operations Research Society of Japan*, Vol.24, No.1, pp.61–65 (1981).
7) Ishii, H., Shiode, S., Nishida, T. and Namasuya, Y.: Stochastic Spanning Tree Problem, *Discrete Applied Mathematics*, Vol.3, No.4, pp.263–273 (1981).

8) Katoh, N. and Ibaraki, T.: A Polynomial Time Algorithm for a Chance–Constrained Single Machine Schduling Problem, *Operations Research Letters*, Vol.2, No.2, pp.62–65 (1983).

(平成 18 年 5 月 19 日受付)
(平成 18 年 12 月 7 日採録)

一森 哲男（正会員）

昭和 28 年生．昭和 57 年大阪大学大学院工学研究科応用物理学専攻博士後期課程修了．同年広島大学工学部助手．昭和 60 年より大阪工業大学工学部専任講師．昭和 63 年より大阪工業大学工学部助教授．平成 8 年より大阪工業大学情報科学部教授．システムの最適化関する研究に従事．工学博士．昭和 62 年日本オペレーションズ・リサーチ学会事例研究奨励賞受賞．日本オペレーションズ・リサーチ学会，日本応用数理学会，日本経営工学会各会員．