Ⅳ-88　最適ネットワーク問題のアルゴリズムの改良について

神戸大学 工学部　正員　枝村 俊郎
神戸大学 工学部　正員　○森津 秀夫

1. はじめに

まず最適ネットワーク問題の厳密解法について、backtrack programmingの系統に属する手法のアルゴリズムの改良について述べよう。つぎにそれによる例題の計算結果の考察をもとに開発した新しい近似解法のアルゴリズムについて述べる。

2. 問題の定式化

アルゴリズムの比較・改良が目的なので、最も単純な次の問題を対象とする。

問題1

$$\min \quad Z = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij} \ (x_1, x_2, \cdots, x_n) \qquad (1)$$

$$\text{s.t.} \quad \sum_{k=1}^{m} l_k x_k \leqq L^c \qquad (2)$$

$$x_k = 0 \text{ or } 1 \quad (k = 1, 2, \cdots, m) \qquad (3)$$

ここに、d_{ij}：ijノード間の最短距離、x_k：kリンクの状態を表わす0－1変数で、kリンクがネットワークに含まれないとき$x_k = 0$とする。l_k：kリンクの長さ、L^c：ネットワークに含まれるリンクの長さの合計の上限（制約長）、m：リンク数、n：ノード数、すなわち、ネットワーク長が制約長以内のネットワークからノード対の最短距離の合計が最小のものを選択する問題である。

3. 厳密解法のアルゴリズム

この研究はHoangのbranch search algorithm[1]を発展させたものである。Hoangの方法の特色は組み合わせトリーの各節点で、目的関数の下限値と分枝変数を定めるために次のナップサック問題が導入されていることである。

問題2

$$\min \quad F = \sum_{k=1}^{m} f_k y_k \qquad (4)$$

$$\text{s.t.} \quad \sum_{k=1}^{m} l_k y_k \geqq \sum_{k=1}^{m} l_k - L^c \qquad (5)$$

$$y_k = 0 \text{ or } 1 \quad (k = 1, 2, \cdots, m) \qquad (6)$$

ここに、f_k：kリンクを除くことによって生じる式(1)の目的関数の増加の下限値、y_k：kリンクの状態を表わす0－1変数で、kリンクがネットワークに含まれないとき$y_k = 1$とする。実際には式(6)を、

$$0 \leqq y_k \leqq 1 \quad (k = 1, 2, \cdots, m) \qquad (7)$$

に置き換えた線形計画問題を問題3とし、問題2の代わりに使用する。この問題3の最適解は1変数だけが小数で、他は0か1のどちらかである。この最適解の小数の変数を1にすると、それは問題2の実行可能解であり、しかも問題1の比較的良い実行可能解であることがわかった。これから組み合わせトリーの節点で固定変数を制約に加えて問題3を解き、その解を使って実行可能解の節点まで一度に分枝するようにする。これをアルゴリズム1としよう。

問題1の目的関数は1組でも経路のないノード対があればその値は無限大になる。そこで、計算過程でネットワークをつねに連結に保てば、目的関数値が無限大になる解を調べることなくすみ、とくに制約長が小さいときに計算時間を短縮できよう。このことから組み合わせトリーの節点で、それまでに$x_k = 1$に固定されたリンクでできるネットワークが非連結ならば、連結成分を1つのノードと考えてこれらを経済木（minimal spanning tree）でつなぐように分枝する。そして連結になればアルゴリズム1と同じ計算手順を使う。これをアルゴリズム2としよう。

4. 計算例と結果の考察

図－1の最大ネットワークに対し、さまざまな制約長を与えた問題を、Hoangのアルゴリズムとアルゴリズム1，2で解いてみた。これらの手法において最適解を得るのに必要な計算時間を比較すると、図－2のようになる。

他にも例題を解いたが、一般にアルゴリズム1ではHoangのアルゴリズムよりも計算時間ははるかに短くなった。また制約長が小さくなればHoangのアルゴリズムなどでは計算時間が急激に長くなる傾向を有するのに反し、アルゴリズム2によればかえって短くなるという成果を得た。

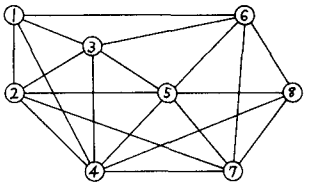
最適ネットワークに含まれるリンクは表－1のようになった。この表－1のリンク番号は、近似解法であるbackward法によって最大ネットワークからリンクを除いてゆくとき、除かれる順に番号をつけたものである。この表を見れば、第1にある最適ネットワークからリンクを除くことにより、それよりも制約長の小さい最適ネットワークが得られることが多いこと、第2にbackward法の解の近傍に最適解がある可能性が高いことがわかる。これは他の例題の計算結果でも同じであった。

5. 近似解法

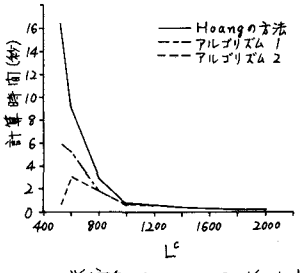
4.の考察における第1の性質により、最大ネットワークからある刻み分だけ制約長の小さい最適ネットワークを求め、つぎにこのネットワークを最大ネットワークとして、制約長をある刻みだけ減らした最適ネットワークを求めるという繰り返しを行ない解を求める。これをアルゴリズム3とする。つぎに第2の性質を利用し、backward法の解の近傍に検討リンクを設け、その組み合わせを厳密解法で調べるアルゴリズムが考えられる。これをアルゴリズム4とする。

アルゴリズム3は制約長の刻み、アルゴリズム4は検討リンク数と、それぞれパラメーターを持っており、その値の選び方によってはこれらは単なるbackward 法にも厳密解法にもなる。

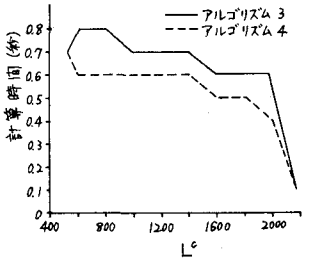
アルゴリズム3，4によって先程の例題を解いたときの計算時間を解いたときの計算時間が図－3である。これからわかるように計算時間は極めて短く、得られた解も最適解かそれに近いものであった。



図－1．例題の最大ネットワーク



図－2 厳密解法による計算時間



図－3 近似解法による計算時間

表－1 最適ネットワークに含まれるリンク

リンク	端点		Lc 527	600	800	1000	1200	1400	1600	1800	2000	2136
1	2	7										
2	1	6										
3	4	8										
4	1	4										
5	6	7										
6	2	5										
7	3	4										
8	3	6										
9	4	7										
10	6	8										
11	7	8										
12	1	2										
13	2	4										
14	2	3										
15	5	6										
16	5	8										
17	1	3										
18	3	5										
19	4	5										
20	5	7										

6. おわりに

経済木構成手法を応用し、ネットワークをつねに連結に保つアルゴリズム2は予想通り十分満足できるものであった。また2種類の近似解法はパラメーターの値によって、計算時間と解の精度を調整することができるという特徴を持ち、しかも最適ネットワークの検討を基礎にしたものである。従来の近似解法の考え方に厳密解法を持ち込んだものと言えるが、これらは十分に実際の問題を扱えるものである。

参考文献

1) Hoang Hai Hoc : A Computational Approach to the Selection of an Optimal Network, Management Science, Vol. 19, No. 5, January, 1973.