数理解析研究所講究録
1043巻 1998年 25-31

ファジィネットワーク上での最適化問題

島田 文彦 Fumihiko Shimada
石井 博昭 Hiroaki Ishii
伊藤 健 Takeshi Itoh

大阪大学大学院工学研究科
Graduate School of Engineering, Osaka University

1 はじめに

ネットワーク上の最適化問題は、現実問題の中でも特に二要素間の関係を中心としたものに対し、その情報を基にして意思決定者の満足の行く結果を導くのに役立つ。現在では最大フロー問題や最短経路問題、そしてシェアリング問題、スパニングツリー問題などが考えられ、それらに対する多くの解法が提案されてきた。

しかし、このようなネットワーク上の問題は、殆どが現実を一つの目的関数を持つモデルとしてモデル化しているため、意思決定者の満足するような解が得られない場合も有り得る。現在のネットワーク問題は、その多くが一つの目的を中心に考えた一目的問題であるが、これは現実の問題に対して、最も重要とされる一つの目的のみを考慮に入れ、その他の諸々の条件を切り捨てた物となっているからである。

そこで、ここでは、二つの要素間におけるその他の条件に対して意思決定者が総合的な満足度を設定し、その値と元の主目的を、両方ともある程度満足させるような結果を求める二目的問題を提案する。そのために、通常のグラフの変わりにアークの存在可能性がファジィであるファジィグラフを用い、存在可能性をアークの満足度と見ることで、この問題を解くことを提案する。

2 定式化

$G=\{V,E\}$ を、頂点の集合 $V=\{1,2,\cdots,n\}$ とアークの集合 $E=\{e \mid e\in V\times V\}$ からなるグラフとする。

また、グラフの各アーク e に、重み $w(e)$ および存在可能性$\mu(e)$ を付加する。

ここで、通常の問題に次の目的を加えることによって二目的問題とする。

$$\text{Maximize}\min_{e\in E'}\mu(e)$$

ただし、E' は用いるアークの集合である。

2.1 ファジィネットワーク

グラフ G は、各点間の関係を表す隣接行列 R_G によって、次の様に表す事が可能である。

$$R_G = \{r(i,j)\}$$
$$= \begin{pmatrix} 0 & r(1,2) & r(1,3) & \cdots & r(1,n) \\ r(2,1) & 0 & r(2,3) & \cdots & r(2,n) \\ r(3,1) & r(3,2) & 0 & \cdots & r(3,n) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r(n,1) & r(n,2) & r(n,3) & \cdots & 0 \end{pmatrix},$$

ここで、

$$r(i,j) = \begin{cases} 1 & e=(i,j) \in E \\ 0 & e=(i,j) \notin E \end{cases}$$

であるとする。また、使用するグラフが無向グラフの場合には、$r(i,j) = r(j,i)$ となる。
この隣接行列を拡張した関係行列 R_G によって、ファジィグラフを定義する事が可能になる：

$$R'_G = \{r'(i,j)\}$$
$$= \begin{pmatrix} 0 & r'(1,2) & r'(1,3) & \cdots & r'(1,n) \\ r'(2,1) & 0 & r'(2,3) & \cdots & r'(2,n) \\ r'(3,1) & r'(3,2) & 0 & \cdots & r'(3,n) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r'(n,1) & r'(n,2) & r'(n,3) & \cdots & 0 \end{pmatrix},$$

ここで、$r'(i,j)$ を枝 $e=(i,j)$ の存在可能性とし、0 から 1 間での値を取り得るものとする。また、$r'(i,j)$ の値が大きいほど、それに対応する枝がグラフ上に存在する割合が大きいものとする。

3 全体の流れ

始めに一方の目的のみを考慮に入れて問題を解き、後にもう一方の目的に沿うように更新を行うが、その順序によって、次のような二つの流れが存在する。

パターン 1

1. まず、アークの存在可能性を無視した状態で本来の目的が最適になる解を求める。この時点では一目的問題であるから、通常のアルゴリズムを用いて解く事ができる。

2. その時の解を基準に、使用するアークの存在可能性の最小が大きくなる順に解を更新して行く。

3. 2 を繰り返す事で得た解の中から、意思決定者が適当と思われる物を選び出す。非劣解の概念等から解の候補を減らす事は可能だが、最終的な判断は、アルゴリズム内でなく意思決定者の判断によって行われる。

パターン　2

1. まず、存在可能性が最大のアークだけを取り出し、元のグラフの部分グラフとする。この時、部分グラフの頂点は元のグラフと同等である。この部分グラフによるネットワークに対して本来の目的が最適になる解を求める。この時点では一目的問題であるから、通常のアルゴリズムを用いて解く事ができる。

2. その時の解を基準に、使用するアークの存在可能性が次に大きいものから順に加え、新しくできた部分グラフに対して一目的問題を解く。

3. 2を繰り返す事で得た解の中から、意思決定者が適当と思われる物を選び出す。非劣解の概念等から解の候補を減らす事は可能だが、最終的な判断は、アルゴリズム内でなく意思決定者の判断によって行われる。

ここで、非劣解とは、「複数の条件があった場合に対し、その解に優越する解、つまり、全ての目的に対して同等か、より目的に沿う値をとり、尚且つ一つ以上の目的では明らかに良い値を取るような解、が存在しないもの」指す。

4　具体例

ここで、幾つかの例を用いて、通常のモデルのファジィネットワークへも拡張方法を説明する。

4.1　ファジィスパニングツリー問題（パターン1）

ここではまず最小コストを持つスパニングツリーを求め、それをある条件の下で更新する、と言う考えに基づいた効率的解法を提案する。

E の各要素 $e=(i,j)$ に重み $w(e)$ 及び存在可能性 $\mu(e)$ を割り当てる。また、グラフ G より、スパニングツリー T を作る事ができるものとする。

ここで、グラフ G のスパニングツリーとは、

- 閉路を含まないグラフ
- G 上の全ての点を連結する極大グラフ

という条件を満たす、G の部分グラフを指す。

以上のことより、次のような二目的スパニングツリー問題 **BSTP** を提案する。

BSTP:

$$\begin{array}{ll} \text{Minimize} & \sum_{e\in T} w(e) \\ \text{Maximize} & \min_{e\in T} \mu(e) \\ \text{s.t.} & T\text{はスパニングツリー} \end{array}$$

[解法]

まず全てのアークを対象に最小スパニング問題を解く。その後、一定の条件の下でツリーを更新して行き、それぞれのツリーを非劣解として出力する。

はじめのツリーを求めるために、ここではKruskalのアルゴリズムを用いる。

次に、存在可能性の低いアークから順に変換し、ツリーを更新して行く。これは、従来の方法("Algorithm for finding K minimum weigth spanning trees"、[2]) を拡張した物である。

但し、ここでは、二目的問題に対応させるために次のような変更を従来の方法に施したアルゴリズムを用いる。

- ツリーを更新した後に、変更禁止のアークの集合 IN 、 OUT に追加するアークを加えたアーク f でなく削除したアーク e にする。
- 更新時に新たに入れ替えるアーク (e,f) を検索する際、 $\mu(e)=\mu^l$ となるアークの組のみを対象とする。

[定義]

W^l ：ツリー $T^l(1)$ の総コスト
$W^l=\sum_{e\in T^l(1)} w(e)$

$IN^l(i)$ ：ツリー $T^l(i)$ を更新する際、削除の対象に入らないアークの集合

$OUT^l(i)$ ：ツリー $T^l(i)$ を更新する際、追加の対象に入らないアークの集合

$P^l(i,j)$ ：$T^l(j)$ まで求まった後、 $T^l(i)$ から更新可能なツリーの集合
$P^l(i,j)=\{T^l(k)\mid k>j, IN^l(i)\subset T^l(i), OUT^l(i)\subset E-T^l(i)\}$

$Q^l(i,j)$ ：$P^l(i,j)$ で表される更新のうち、総コストの増加量が最も小さい変換
$Q^l(i,j)=\{([e,f],r)\}$ と書くと各 $f\in E-T^l(i)-OUT^l(i)$ 、 $e\in T^l(i)-IN^l(i)$ に対して、$r=w(f)-w(e)$ となる

$m^l(i,\mu)$ ：$T^l(i)$ において、存在可能性が μ であるアークの数

μ^l ：$T^l(1)$ において、使用したアークの存在可能性の下限

$\mu_{\max}$ ：元のネットワークの存在可能性の最大値

前に述べた条件付けによって、アルゴリズムは更に計算量が少なくなる。まず、ツリーの更新後、元のツリー $T^l(i^*)$ は、次回からの検索対象から外れる。これは、 $IN^l(i^*)$ に存在可能性が μ^l のアークが含まれてしまう為、そのツリーから更新したツリーは全て存在可能性の下限が μ^l となってしまうからである。次に、この様に順に更新して求めた $T^{l+1}(1)$ は、必ず「アークの存在可能性が μ^{l+1} 以上であるスパニングツリーの中で総コストが最小のもの」となっている。このことは、次の定理 ([1]) によって証明される。

(定理)

> T を G における最小スパニングツリー、(e,f) を全ての T から交換可能なアークのうちコストの差が最小のものとする。この時、 $T-e\cup f$ は、 $G-e$ における最小スパニングツリーである。

この定理は、条件として「(e,f) は全ての T から交換可能なアークのうちコストの差が最小のもの」としているが、これは「(e,f) は T から交換可能なアークの中である e に対してコストの差が最小になるもの」の様に緩和する事ができる。つまり、任意の e に対してコストの差が最小になるように f を選べば、それが全てのアークの組の中で最小でなくてもよいのである。

$\mu(e)=\mu^l$ と言う条件で e を選び、それに対応する f を選べば、$T-e\cup f$ は $G-e$ における最小スパニングツリーとなっている。これを $\mu(e)=\mu^l$ となる全てのアークで行えば、求まったツリーはアークの存在可能性が μ^{l+1} 以上であるスパニングツリーの中で総コストが最小のものとなる。

[全体の解法の流れ]

Step 0 $l=1$、$i=1$、$IN^1(1)=\phi$、$OUT^1(1)=\phi$ とする。

Step 1 $T^1(1)$（最小スパニングツリー）を求める。
その際、$m^1(1,\mu)$、μ^1 も同時に求める。

Step 2 $(T^l(1),\mu^l)$ を非劣解の一つとして記録する。

Step 3 $\mu(e)\leq\mu^l$ となるアークの内、$T^l(i)$ に含まれないものを $OUT^l(i)$ に追加する。

Step 4 $T^{l+1}(1)=T^l(i)$、$P^{l+1}(1,1)=P^l(i,i)$、$Q^{l+1}(1,1)=Q^l(i,i)$、$IN^{l+1}(1)=IN^l(i)$、$OUT^{l+1}(1)=OUT^l(i)$ とする。

Step 5 $l=l+1$、$i=1$。

Step 6 $Q^l(i,i)$ を求める。もしも有限の r を持つ $Q^l(i,i)$ が存在しなければ、ここで終了。

Step 7 $T^l(i+1)=T^l(i)-e\cup f$、$IN^l(i)=IN^l(i)\cup e$、$OUT^l(i+1)=OUT^l(i)\cup e$ とする。

Step 8 $i=i+1$。

Step 9 $m^l(i,\mu)$ を求める。

Step 10 もしも $m^l(i,\mu^l)>0$ ならば、**Step 5** へ戻る。

Step 11 もしも $\mu^{l+1}=\mu_{\max}$ ならばここで終了。そうでなければ、μ^{l+1} を $|T^l(i)$ に含まれるアークの存在可能性の下限| とし、**Step 2** に戻る。

また、このアルゴリズムは、$O(m^2)$ のオーダーで計算が可能である。これは、元のアルゴリズム ([2]) が $O(Km)$ のオーダーで求められる事による。ここで、K は求めるスパニングツリーの数である。

4.2 ファジィネットワーク上のシェアリング問題（パターン2）

ここで提案するのは、はじめに経路の存在可能性が最高の弧のみを用いてシェアリング問題を解き、使用する弧の存在可能性の下限を下げながらその解を更新していく方法である。

ここで、A の各要素 (i,j) に容量 $c(i,j)$ 及び存在可能性 $0\leq\mu(i,j)\leq 1$ が割り当てられている。V には、供給点の集合 $S=\{s_1,s_2,\cdots,s_k\}$ と需要点の集合 $T=\{t_1,t_2,\cdots,t_l\}$ が含まれる。また、弧 (i,j) を流れるフローを $f(i,j)$、t_j に流れ込むフローを f_j とする。

以上のことより、次のような二目的シェアリング問題 **BSP** を提案する。

BSP:

$$\begin{aligned} &\text{Maximize} \quad \min_{j\in T} f_j/w_j \\ &\text{Maximize} \quad \min_{(i,j)\in A_F} \mu(i,j) \\ &\text{s.t.} \qquad \textstyle\sum_{i\in V-j} f(i,j) = \sum_{i\in V-j} f(j,i) \quad (\ j \in V \cap \overline{S} \cap \overline{T}\ ) \\ &\qquad\quad 0 \le f(i,j) \le c(i,j) \end{aligned}$$

但し、$w_j > 0$ をシンク t_j の重要度とし、$f(i,j) > 0$ である弧 (i,j) の集合を A_F と置く。

[解法]

まず、シェアリング問題を最大フロー問題として解く事を考える。

はじめに、$V' = V \cup \{s,t\}$ 、$A' = A \cup \{(s,s_i) \mid i = 1,2,\cdots,k\} \cup \{(t_j,t) \mid j = 1,2,\cdots,l\}$ なる拡張ネットワーク $G(V',A')$ を考える。ここで、スーパーソース s を各ソース s_i にフローを流すソース、スーパーシンク t を各シンク t_j からのフローを受けるシンクとする。

この様にして定義した新しいネットワークに対して最大フロー問題を解き、それを、シェアリング問題の初期実行可能解とする。ここでは、最大フローを求める方法として、プリフロー-プッシュ法 ([6])、f_j/w_j の最小値を最大にする方法として、Brown のアルゴリズムを用いる。

（プリフロー-プッシュ法）

まず、シンクから順番に高さを設定し、グラフ上の高い方（ソース）から低い方（シンク）にフローを流していく、と考える。

ソースから一段低い（距離ラベルの小さい）点にプリフローを流す。すると流した先には余分なフロー（残存量）ができる。これを減らすためにさらに低い点にプリフローを流す。どこにも流せなくなった場合には、相手が見つかるまで距離ラベルを増加させる。これにより、フローがソース・シンクの内いずれかに全て流れた時点で終了となる。

これを、ファジィネットワークでのシェアリング問題に拡張する。

[全体の流れ]

Step 1 存在可能性が最大のアークのみを用いて部分グラフを作る。

Step 2 **Step 1** で作成したグラフに対して最大フロー問題を解く。

Step 3 **Step 2** で求めた解を初期実行解として、シェアリング問題を解く。

Step 4 次に存在可能性が大きいアークをすべて追加し、新たな部分グラフを作る。

Step 5 **Step 4** で作成したグラフに対して最大フロー問題を解く。この時、初期解として前回のシェアリング問題の解を用いる。

Step 6 **Step 4** のグラフに対するシェアリング問題を解く。

Step 7 全てのアークが追加されていれば終了。そうでなければ、**Step 4** に戻る。

ここでは例としてBrownのアルゴリズムによるMax-Minシェアリング問題を拡張したが、他にもMin-Maxシェアリング問題([5])等の拡張も考えられている。その場合、全体の流れは上記のものと同様だが、問題が次の様に定義される。

BSP:

$$\begin{aligned}
&\text{Minimize} \quad \max_{j\in T} f_j/w_j v^* \\
&\text{Maximize} \quad \min_{(i,j)\in A_F} \mu(i,j) \\
&\text{s.t.} \quad \sum_{i\in V-j} f(i,j) = \sum_{i\in V-j} f(j,i) \quad (\, j \in V \cap \overline{S} \cap \overline{T}\,) \\
&\qquad 0 \le f(i,j) \le c(i,j) \\
&\qquad v^* = \sum_{j\in T} f_j
\end{aligned}$$

但し、v^*はネットワーク上の総フローとする。

目的関数が「Maximize max f_j/w_j」ではなく「Maximize max $f_j/w_j v^*$」となっているのは、前者では、総フローを減少させる事で最終的には0まで第一目的関数を減少させる事が可能になるためである。

5 おわりに

ここでは、現実の問題をモデル化する際、より現実に近いモデルの一つとして、満足度を導入した二目的問題のモデルと、それを解く為の大まかな流れを提案した。2、3の問題については実際のアルゴリズムも考えているが、その外にも、様々な問題の拡張の一つとして考える事ができる。

これからの研究課題としては、まず、最終的な判断の効率化が挙げられる。極端な解を減らす事により、意思決定者の負担を減少させる事が考えられるが、意思決定者の多様な判断に対応するために、最終決定を完全にアルゴリズム化する事は困難だと思われる。また、アークへの付加する数値や目的関数をファジィ概念化する事により、さらに実際的なモデル化を行う事も必要である。

参考文献

[1] Harold N. Gabow, "Two Algorithms for Generating Weighted Spanning Trees in Order", *SIAM J. COMPUT.* 6, pp.139-150, 1977

[2] N, Katoh, T. Ibaraki, and H. Mine, "An Algorithms for Finding K Minimum Spanning Trees", *SIAM J. COMPUT.* 10, pp.247-255,1981

[3] J. R. Brown, "The sharing problem", *Operations Research* 27, pp.324-340, 1979

[4] N. Christofides, *"Graph theory: an algorithm approach"*, Academic Press, New York, NY, 1975

[5] T. Ichimori, "Optimal sharing", *Mathematical Programming* 23, pp.341-348, 1982

[6] Ahuja, R.K., and J.B. Orlin, "A fast and simple algorithm for the maximum flow problem", Working paper 1905-87, Sloan School of Management, M.I.T., Cambridge, MA, to appear in *Oper. Res.*, 1987