ネットワーク流問題とそのアルゴリズム

筑波大学 電子・情報工学系　　久 野　誉 人

1997 年　4 月
2001 年 12 月改訂

1　ネットワーク流問題とは

数理計画法 (mathematical programming) の中核をなす**線形計画問題** (linear program) の中には，問題のもつ特殊な構造に着目することで，シンプレックス法などの汎用アルゴリズムを用いるよりも簡単に解けてしまうものが少なくない．**ネットワーク流問題** (network flow problem) はその代表例であり，このことは制約式の係数行列がもつ特殊な性質に由来する．本稿の目的は，ネットワーク流問題を，その特殊構造を活かして効率よく，しかも厳密に解決するアルゴリズムを紹介することにある．

この節では，まずネットワーク流問題を特徴づける**グラフ** (graph) の諸用語を定義したのち，主要なネットワーク流問題を紹介しよう．

1.1　グラフ

グラフ.　有限個の**頂点** (vertex, node) の集合 $V = \{1, 2, \ldots, m\}$ と，頂点対の集合 $E \subseteq V \times V \equiv \{(i, j) \mid i \in V,\ j \in V\}$ の組をグラフといい，$G = (V, E)$ で表す．集合 E に属する頂点対 $e = (i, j)$ をグラフ G の**枝** (arc, edge)，頂点 i と j を枝 e の**端点** (end node) とよび，枝 e は頂点 i, j に**接続する**という．どの枝の向きも考えないときは G を**無向グラフ** (undirected graph)，枝の向きを考えて (i, j) と (j, i) を区別するときには**有向グラフ**とよぶ．

有向グラフでは枝 $e = (i, j)$ を有向枝 (directed arc) ともいい，頂点 i, j をそれぞれ e の**始点** (tail)，**終点** (head) という．有向，無向を問わず，頂点 i に接続する枝の本数を i の**次数** (degree) といい，特に有向グラフでは i を始点とする枝の数を i の**出次数** (out-degree)，i を終点とする枝の数を i の**入次数** (in-degree) という．

頂点 n の無向グラフ G ですべての頂点間に枝が存在するとき，G を**完全グラフ** (complete

graph) とよぶ．グラフ $G=(V,E)$ に対して V, E の部分集合をそれぞれ V', E' とするとき，任意の $e' \in E'$ の両端点が V' に属すならば，$G'=(V',E')$ は再びグラフとなる．そのような G' を G の**部分グラフ** (subgraph) という．特に $G=(V,E)$ で $V' \subseteq V$ に両端点をもつ枝の集合を E' とするとき，$G'=(V',E')$ は V' による G の**生成部分グラフ** (induced subgraph) であるという．

路．　有向グラフ $G=(V,E)$ の頂点の列 $P=(i_0,i_1,\ldots,i_p)$ が $(i_k,i_{k+1}) \in E$ $(k=0,1,\ldots,p-1)$ を満たすとき，P を頂点 i_0 から i_p への**有向路** (directed path) とよぶ．**無向路** (undirected path) も同様に定義されるが，無向路 $P=(i_0,i_1,\ldots,i_p)$ では隣接する 2 つの頂点 i_k, i_{k+1} に対して $(i_k,i_{k+1}) \in E$ か $(i_{k+1},i_k) \in E$ のいずれか $(k=0,1,\ldots,p-1)$ が満たされればよい．有向路と無向路をあわせて単に**路** (path) とよぶが，路 P を頂点あるいは枝の集合とみなして $i \in P$ や $(i,j) \in P$ などの表現を用いる．

始点 i_0 と終点 i_p が同じ頂点である路を**閉路** (circuit)，含まれる頂点 i_0, i_1, …, i_p がすべて異なる路を**単純路** (simple path)，その両方の性質をもつ路を**単純閉路** (simple circuit) とよぶ．グラフ G のすべての頂点を通る単純閉路を**ハミルトン閉路** (Hamiltonian circuit) とよぶ．これらの閉路，単純路に対しても有向，無向が定義される．

連結．　グラフ G の任意の 2 つの頂点間に無向路が存在するとき，G は**連結グラフ** (connected graph) であるという．グラフ G が連結でなくとも，その連結な生成部分グラフ G' で，G' を真に含む連結生成部分グラフが存在しないとき，G' を G の**連結成分** (connected component) という．連結でないグラフは互いに頂点を共有しないいくつかの連結成分に分解される．

木．　グラフ G が閉路を含まないとき，G を**非巡回的** (acyclic) であるという．また，非巡回的な連結グラフ T を**木** (tree) とよぶ．頂点数 n の連結グラフ T が木であるための必要十分条件は，T が $n-1$ 本の枝をもつことである．次数が 1 である木の頂点を**葉** (leaf) とよぶが，木には少なくとも 2 枚の葉が存在する．グラフ $G=(V,E)$ に対して G と同じ頂点集合をもつ部分グラフ $G'=(V,E')$ が木であるとき，G' を G の**全域木** (spanning tree) という．

1.2　ネットワーク流モデル

頂点数 $|V|=n$，枝数 $|E|=m$ の有向グラフ $G=(V,E)$ において，すべての枝 $(i,j) \in E$ に対して費用 c_{ij} と容量 u_{ij} が与えられているものとしよう．それぞれの頂点 $i \in V$ には供給量あるいは需要量を表す値 b_i が対応しており，$b_i > 0$ ならば頂点 i を**供給点** (supply node)，$b_i < 0$ ならば i を**需要点** (demand node)，また $b_i = 0$ ならば頂点 i を**中継点** (transshipment node) とよぶ．このようにグラフ G の頂点や枝に何らかの情報（ここでは c_{ij}, u_{ij}, b_i）が付加されたものを**ネットワーク** (network) という．本稿では，

$$b_i \in \mathcal{Z},\ \forall i \in V; \quad c_{ij}, u_{ij} \in \mathcal{Z},\ \forall (i,j) \in E \tag{1.1}$$

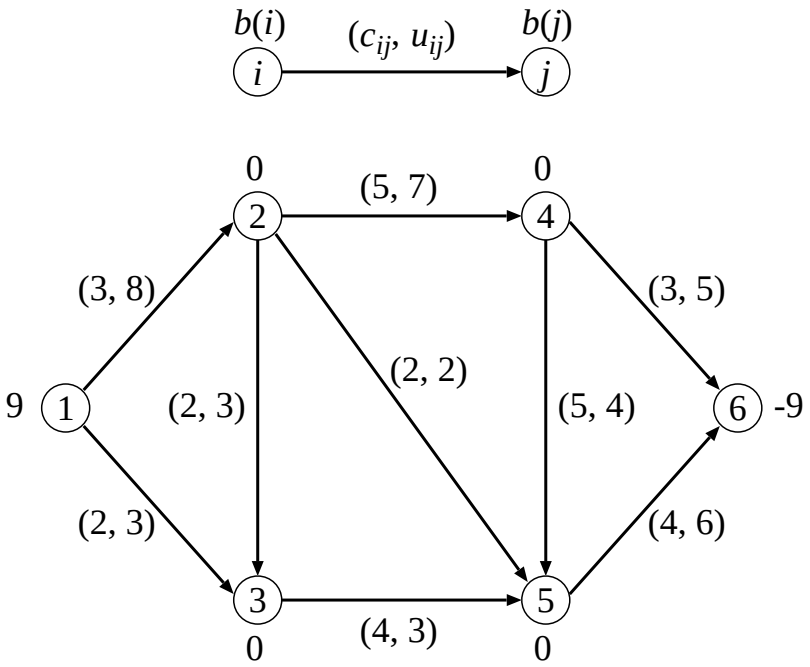

図 1.1: 最小費用流問題.

であることを仮定する（ただし，$\mathcal{Z}$ は整数全体の集合を表す）.

ネットワーク上での最適化問題のほとんどは**最小費用流問題** (minimum cost flow problem) とよばれる次の線形計画問題として定式化される:

$$\left|\begin{array}{lll} \text{最小化} & \displaystyle\sum_{(i,j)\in E} c_{ij}x_{ij} & \text{(1.2a)} \\ \text{条　件} & \displaystyle\sum_{\{j|(i,j)\in E\}} x_{ij} - \sum_{\{j|(j,i)\in E\}} x_{ji} = b_i, \quad \forall i \in V & \text{(1.2b)} \\ & 0 \le x_{ij} \le u_{ij}, \qquad\qquad \forall (i,j) \in E. & \text{(1.2c)} \end{array}\right.$$

式 (1.2a) を**目的関数** (objective function)，式 (1.2b), (1.2c) を**制約条件** (constraints) とよぶ点は一般の数理計画問題と同じだが，変数値ベクトル $\mathbf{x} = (x_{ij})$ はネットワークの**流れ** (flow) とよばれることが多い．制約式 (1.2b) は，頂点 i から出る流れの総量と i に入る流れの総量の差が需給量 b_i に等しくなければならないことを意味し，**流量保存条件** (flow conservation constraint) とよばれる．また (1.2c) は，枝 (i,j) の流れ x_{ij} が与えられた容量 u_{ij} を越えてはならないことを意味し，**容量条件** (capacity constraint) とよばれる．この 2 種類の制約条件を満足する流れ $\mathbf{x}$ を**実行可能流** (feasible flow)，その中で目的関数の値を最小にする流れ $\mathbf{x}^*$ を**最適流** (optimal flow) という.

式 (1.2b) の左辺の係数行列を $\mathbf{A}$ で表し，ベクトル $\mathbf{c} = (c_{ij})$，$\mathbf{u} = (u_{ij})$ を用いれば，問題 (1.2) を

$$\text{最小化}\,\{\mathbf{cx} \mid \mathbf{Ax} = \mathbf{b},\ \ \mathbf{O} \le \mathbf{x} \le \mathbf{u}\} \tag{1.3}$$

のように簡潔に書くことができる．行列 $\mathbf{A}$ はグラフ G の**接続行列** (incidence matrix) とよばれ，その各行は G の各頂点に，また各列は各枝に対応し，G の枝と頂点の接続関係を表す．例えば，A_{ij} を枝 (i,j) に対応する $\mathbf{A}$ の列とすれば

$$A_{ij}^{\mathsf{T}} = [0\cdots 0 \underset{(\text{第 } i \text{ 行})}{+1} 0\cdots 0 \underset{(\text{第 } j \text{ 行})}{-0} 0\cdots 0]$$

であり，第 i, j 基本ベクトル $\mathbf{e}^i$, $\mathbf{e}^j$ を使って

$$A_{ij} = \mathbf{e}^i - \mathbf{e}^j$$

と表すことができる．したがって $\mathbf{A}$ は，mn 個の成分の中で $2m$ 個だけが非ゼロで，非ゼロ成分はすべて $+1$ か -1，また各列には $+1$ と -1 がちょうど 1 つづつ含まれる．この係数行列 $\mathbf{A}$ の特殊性からただちに次の 2 つが導かれる:

(a) 流量保存条件 (1.2b) をすべて足しあわせると

$$0 = \sum_{i\in V} b_i, \quad \text{したがって} \quad \sum_{\{i\in V | b_i > 0\}} b_i = - \sum_{\{i\in V | b_i < 0\}} b_i.$$

言い換えれば，総供給量と総需要量が一致しなければ，流量保存条件 (1.2b) は満たされない．

(b) 逆に，総供給量と総需要量が一致すれば，流量保存条件 (1.2b) の和から自明な等式 $\mathbf{Ox} = 0$ が得られる．つまり，(1.2b) の任意の等式は，それ以外の等式の和に -1 を掛けたもの等しく，**冗長** (redundant) な制約式である．

最短路問題 (shortest path problem)． 与えられた頂点 1 から他のすべての頂点への最短距離の有向路を決定する問題．問題 (1.2) において

$$u_{ij} = n, \ \forall (i,j) \in E; \quad b_i = \begin{cases} n-1, & i = 1 \\ -1, & \forall i \in V \setminus \{1\} \end{cases}$$

とし，c_{ij} を枝 (i,j) の長さとすれば，その最適解は頂点 1 から他の各頂点への最短路に沿って 1 単位の流れを送る．

最大流問題 (maximum flow problem)． 特定の**入口** (source) 頂点 s から**出口** (sink) 頂点 t へできるだけ多くの流れを送る問題．問題 (1.2) のネットワークに

$$c_{ts} = -1; \quad u_{ts} = +\infty$$

なる枝 (t,s) を人工的に追加し，

$$b_i = 0, \ \forall i \in V; \quad c_{ij} = 0, \ \forall (i,j) \in E$$

とすれば，その最適解は枝 (t,s) 上の流れを最大化する．この流れは頂点 s から頂点 t へのグラフ G 上の最大総流量に等しい．

割当問題 (assignment problem)． グラフ G の頂点集合は 2 つの部分集合 V_1, V_2 に等分割され，枝集合は $E \subseteq V_1 \times V_2$ を満たす．このとき，割当費用 c_{ij} の総和が最小となるように V_1 の各頂点を V_2 の 1 つの頂点に割り当てる問題．これは問題 (1.2) で

$$b_i = 1, \ \forall i \in V_1; \quad b_i = -1, \ \forall i \in V_2; \quad u_{ij} = 1, \ \forall (i,j) \in E$$

とした場合に他ならない．

この他にも**ヒッチコック問題** (Hitchcock transportation problem) や**輸送問題** (transshipment problem) などがネットワーク流問題として有名だが，これらもまた (1.2) の特殊ケースとして扱うことができる．こうしたネットワーク流問題の応用は工学をはじめ経済・経営システムまで枚挙にいとまがなく，したがって問題 (1.2) の効率的な解決が実社会に与える影響は極めて大きい．なお，ネットワーク流問題の応用例については参考文献を参照のこと．

参考文献

ネットワーク流問題に関する教科書は洋書，和書とも星の数ほど出版されている．その中で定評のあるものを挙げると:

[1] R.K. Ahuja, T.L. Magnanti and J.B. Orlin, *Network Flows – Theory, Algorithms, and Applications* (Prentice Hall, 1993).

[2] 伊理 正夫, 藤重 悟, 大山 達雄, グラフ・ネットワーク・マトロイド(産業図書, 1986).

[3] C.H. Papadimitriou and K.S. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity* (Prentice Hall, 1982).

いずれもマニアックな内容だが，[1] は比較的新しく，また読みやすい．しかし，800 ページを越す大著なのでエッセンスだけを知りたい場合は，

[4] R.K. Ahuja, T.L. Magnanti and J.B. Orlin, “Network Flows” in: G.L. Nemhauser, A.H.G. Rinnooy Kan and M.J. Todd eds., *Optimization* (North-Holland, 1989), 211–369
(伊理 他監訳, 最適化ハンドブック(朝倉書店, 1995), 206–358).

でも十分に間に合う (このダイジェスト版ですら 150 ページを越える! 普通の和書なら 1 冊分のボリューム)．線形計画法の教科書にもネットワーク流問題が特殊ケースとして登場することが多く，

[5] M.S. Bazaraa, J.J. Jarvis and H.D. Sherali, *Linear Programming and Network Flows* (Second ed., John Wiley and Sons, 1990).

[6] V. Chvătal, *Linear Programming* (Freeman and Company, 1983).

などではその半分近くのページがネットワーク流に割いてある．どちらも線形計画法の教科書としては非常に優れたもので，ネットワーク流に関する記述も下手な (ネットワーク流の) 専門書よりもずっと詳しい．本稿では主に [1,3,4,5] を参考に話を進める予定である．