アルゴリズム 36－9
（1993．11．25）

線形計画問題としての整列

中森 眞理雄

東京農工大学 工学部 情報工学大講座

整列問題を線形計画問題 - 特に，割り当て問題 - として記述する方法を示す．その割り当て問題を電気回路の定常解を求める問題と解釈して，その回路を構成する方法を示す．割り当て問題 (あるいは，一般に，ネットワークにおける最小費用流問題) として記述できる問題の複雑度を表すために，ネットワーク複雑度の概念を提案する．整列問題のネットワーク複雑度を $O(n^2)$ あるいは $O(n \log n)$ とすることができることを示す．

Sorting as Linear Programming

Mario Nakamori

Tokyo University of Agriculture and Technology

The sorting problem is described as a linear programming problem and, especially, as an assignment problem. An electric network is shown that gives the optimal solution of the assignment problem. Assignment problems, as well as the general minimum-cost flow problems, are evaluated from the point of view of network comoplexity. It is shown that the network complexity of the sorting problem is $O(n^2)$, or $O(n \log n)$.

1　はじめに

コンピュータサイエンスの問題には，“良い” アルゴリズムが存在することが知られている問題と，“良い” アルゴリズムが知られていない問題とがある．ここで，アルゴリズムの “良さ” とは，計算時間や記憶場所の大きさを問題の規模を表す指標の関数で表したときの関数形のことと，一応，見なしておく．アルゴリズムの能率の良さを評価したり良いアルゴリズムを考案したりアルゴリズムの良さの限界を調べたりする研究を計算複雑度の研究とよぶ．計算時間，記憶場所の大きさを表す関数をそれぞれ時間複雑度，空間複雑度という．多項式時間の決定性アルゴリズムが存在する問題のクラスを **P** と表し，多項式時間の非決定性アルゴリズムが存在する問題のクラスを **NP** と表す．クラス **P** (または **NP**) に属するいかなる問題も時間複雑度が多項式オーダの決定性アルゴリズムによって **P** (または **NP**) のある部分クラスのいずれかの問題に変換されるとき，この部分クラスの問題は **P** 完全である (**NP** 完全である) という ([1], [2], [3])．

近年の計算複雑度の理論によれば，クラス **P** の問題 (多項式時間の決定性アルゴリズムが存在する問題) は線形計画問題として記述できる．逆に，線形計画問題として記述できる問題には良いアルゴリズムが存在することが多い (線形計画問題に対する Karmarkar のアルゴリズム ([4]) とは異なる意味で)．近年，並列アルゴリズムの理論的研究が進んだのに伴い，良いアルゴリズムが存在する問題と線形計画問題との関係も明らかにされつつある．

特殊な線形計画問題として，ネットワークにおける**最小費用流問題**がある ([5])．最小費用流問題に対しては，理論的な意味でも実用的な意味でも，能率の良い (もちろん，多項式時間の) アルゴリズムが知られている ([6], [7])．特殊な最小費用流問題である**割り当て問題**については，特に高速のアルゴリズムが知られている ([8],[9],[10])．ただし，すべての線形計画問題が最小費用流問題や割り当て問題として記述できるわけではない ([11])．

本稿では，**整列** (与えられた数値を昇順あるいは降順に並べる問題) が最小費用流問題 - 特に，割り当て問題 - として記述できることを示す．すなわち，n 個の数値を整列する問題を，$2n$ 個の点と n^2 本の辺から成るネットワークにおける最小費用流問題として定式化する．さらに，そのようなネットワークを電気回路として構成する方法を示す．このネットワークが $2n$ 個の点と n^2 本の辺を含むことから，この整列問題の **ネットワーク複雑度** を $O(n^2)$ で表すことにする．さらに，このネットワークを分解することにより，整列問題を，点の個数が $O(n)$ で，辺の本数が $O(n \log n)$ のネットワークにおける最小費用流問題に帰着することができることを示す．このことを，整列問題の最良のアルゴリズムの複雑度が $O(n \log n)$ であるという計算複雑度の理論 ([12]) と比較すると，興味深い．

従来，計算複雑度の研究で広く用いられた計算のモデルは Turing 機械やランダムアクセス機械 (RAM) であった ([13])．実際，クラス **P** や **NP** の概念の導入には Turing 機械が大きな役割を果たした．しかし，問題やアルゴリズムの複雑さを表す指標は，Turing 機械やランダムアクセス機械における計算時間だけが唯一のものではない．例えば，筆者は論理式充足可能性問題など各種の困難な問題を連続変数に関する双線形計画問題として記述し，それに要する変数の個数を問題複雑度とみなして考察した ([14], [15])．本稿では，整列問題を割り当て問題とみなし，割り当て問題と電気回路の類似点に注目して，電気回路の定常解を求める問題として定式化する．本稿では，問題の複雑さを，対応する電気回路の複雑さとして捉える．

本稿で述べる計算のモデルやアルゴリズムは，いわゆるアナログコンピュータやアナログ回路の分野に属するものである．アナログコンピュータが今日のコンピュータサイエンスの主流にならなかったのは，計算の精度が確保できなかったためであるが，本稿のアナログ回路の最適解は自然に整数条件を満たすので，精度の問題はない．計算時間は，アナログ回路の定常解が得られるまでの時間だけであり，ディジタルコンピュータをモデルとするアルゴリズム論や計算複雑度の理論における計算時間より高速に最適解が得られる．本稿の計算のモデルでは，対象を表す回路を作り上げるところが最も手間を要する部分であり，その回路の素子数が問題の複雑さを最もよく表すと考えられる．このことが，前記のネットワーク複雑度の概念を提唱した理由である．

2　整列と線形計画問題

複数個の数値の中の最大値や最小値を求めたりそれらを整列したりする問題を線形計画問題として記述してみる．

2.1　最大値 (最小値)

数値 $a_1, a_2, \ldots, a_n$ が与えられたとき, これらの中の最大値を求める問題は, 線形計画問題として次のように記述される.

問題 P1

maximize
$$z = \sum_{i=1}^{n} a_i x_i$$
subject to
$$\sum_{i=1}^{n} x_i \leq 1,$$
$$x_i \geq 0 \quad (i = 1, 2, \ldots, n)$$

問題 P1 の最適解において, a_i が最大値である i に対して $x_i = 1$ となり, それ以外の i に対して $x_i = 0$ となる. もちろん, 最適解における z の値は $a_1, a_2, \ldots, a_n$ の最大値である.

問題 P1 の双対問題は次の通りである.

問題 D1

minimize
$$w = y$$
subject to
$$y \geq a_i \quad (i = 1, 2, \ldots, n)$$

同様に, 数値 $a_1, a_2, \ldots, a_n$ の最小値を求める問題とその双対問題は, 次のように記述される.

問題 P2

minimize
$$t = \sum_{i=1}^{n} a_i u_i$$
subject to
$$\sum_{i=1}^{n} u_i \geq 1,$$
$$u_i \geq 0 \quad (i = 1, 2, \ldots, n)$$

問題 D2

maximize
$$s = v$$
subject to
$$v \geq a_i \quad (i = 1, 2, \ldots, n)$$

2.2　大きい方 (小さい方) から k 個のものの和

数値 $a_1, a_2, \ldots, a_n$ が与えられたとき, 大きい方から k 個のものの和を求める問題は, 線形計画問題として次のように記述される.

問題 P3

maximize
$$z = \sum_{i=1}^{n} a_i x_i$$
$$\sum_{i=1}^{n} x_i \leq k,$$
$$x_i \leq 1 \quad (i = 1, 2, \ldots, n),$$
$$x_i \geq 0 \quad (i = 1, 2, \ldots, n)$$

問題 P3 の最適解において, a_i が大きい方から k 番目以内のとき $x_i = 1$ となり, a_i がそれ以外のときは $x_i = 0$ となる. もちろん, 最適解における z の値は $a_1, a_2, \ldots, a_n$ の大きい方から k 個のものの和に等しい.

問題 P3 の双対問題は次のとおりである.

問題 D3

minimize
$$w = ky_0 + \sum_{i=1}^{n} y_i$$
subject to
$$y_0 + y_i \geq a_i \quad (i = 1, 2, \ldots, n),$$
$$y_i \geq 0 \quad (i = 0, 1, 2, \ldots, n)$$

同様に, 数値 $a_1, a_2, \ldots, a_n$ の小さい方から k 個のものの和を求める問題は, 線形計画問題として次のように記述される.

問題 P4

minimize

$$t = \sum_{i=1}^{n} a_i u_i$$

subject to

$$\sum_{i=1}^{n} u_i \geq k,$$

$$u_i \leq 1 \quad (i = 1, 2, \ldots, n),$$

$$u_i \geq 0 \quad (i = 1, 2, \ldots, n)$$

問題 D4

maximize

$$w = ky_0 - \sum_{i=1}^{n} y_i$$

subject to

$$y_0 - y_i \leq a_i \quad (i = 1, 2, \ldots, n),$$

$$y_i \geq 0 \quad (i = 0, 1, 2, \ldots, n)$$

2.3 大きい方から k 番目のもの

数値 $a_1, a_2, \ldots, a_n$ が与えられたとき, 大きい方から k 番目のものは, 大きい方から k 個のものの和と小さい方から $n-k+1$ 個のものの和の両方に含まれるので, 両者の和から $a_1 + a_2 + \cdots + a_n$ を減じたものに等しい. したがって, 線形計画問題として次のように記述される.

問題 P5

maximize

$$h = \sum_{i=1}^{n} a_i x_i + (n-k+1) y_0 - \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} a_i$$

subject to

$$\sum_{i=1}^{n} x_i \leq k,$$

$$x_i \leq 1 \quad (i = 1, 2, \ldots, n),$$

$$x_i \geq 0 \quad (i = 1, 2, \ldots, n),$$

$$y_0 - y_i \leq a_i \quad (i = 1, 2, \ldots, n),$$

$$y_i \geq 0 \quad (i = 0, 1, 2, \ldots, n)$$

2.4 整列

数値 $a_1, a_2, \ldots, a_n$ が与えられたとき, これらを大きい方から順に並べる問題 (整列あるいはソート) は, 線形計画問題として次のように記述される.

問題 P6

minimize

$$z = \sum_{i=1}^{n} \sum_{j=1}^{n} j a_i x_{ij}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = 1 \quad (j = 1, 2, \ldots, n),$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad (i = 1, 2, \ldots, n),$$

$$x_{ij} \geq 0 \quad (i, j = 1, 2, \ldots, n)$$

問題 P6 の双対問題は次の通りである.

問題 D6

maximize

$$w = \sum_{i=1}^{n} y_i + \sum_{j=1}^{n} q_i$$

subject to

$$q_j + y_i \leq j a_i \quad (i, j = 1, 2, \ldots, n)$$

数値 $a_1, a_2, \ldots, a_n$ を小さい方から順に並べる問題は, 次のように記述される.

問題 P7

maximize

$$t = \sum_{i=1}^{n} \sum_{j=1}^{n} j a_i u_{ij}$$

subject to

$$\sum_{i=1}^{n} u_{ij} = 1 \quad (j = 1, 2, \ldots, n),$$

$$\sum_{j=1}^{n} u_{ij} = 1 \quad (i = 1, 2, \ldots, n),$$

$$u_{ij} \geq 0 \quad (i, j = 1, 2, \ldots, n)$$

問題 P7 の双対問題は次の通りである.

問題 D7

minimize

$$s = \sum_{i=1}^{n} v_i + \sum_{j}^{n} p_j$$

subject to

$$p_j + v_i \leq j a_i \quad (i, j = 1, 2, \ldots, n)$$

問題 P6 (問題 P7) において変数 x_{ij} (変数 u_{ij}) は, a_i が $a_1, a_2, \ldots, a_n$ の大きい (小さい) 方から j 番目のときに 1 でその他のときに 0 をとる. 問題 P6 (問題 P7) が整列問題であることは, 一般に, $0 \le a_1 \le a_2 \le \ldots \le a_n$, $0 \le b_1 \le b_2 \le \ldots \le b_n$ のとき, $1, 2, \ldots, n$ のいかなる順列 $p_1, p_2, \ldots, p_n$ に対しても,

$$\sum_{i=1}^{n} a_i b_{n+1-i} \le \sum_{i=1}^{n} a_i b_{p_i} \le \sum_{i=1}^{n} a_i b_i$$

であることから分かる.

3 整列問題のネットワーク複雑度

最小費用流問題や割り当て問題は特殊な線形計画問題であり, 電気回路の理論とのアナロジーが成り立つ. 以下に示すように、整列問題は最小費用流問題や割り当て問題として記述することができる.

3.1 割り当て問題としての整列問題

整列問題の問題 P6, P7 (D6, D7) としての記述は, 割り当て問題として解釈すると, 理解しやすい. 一般に, 割り当て問題は, n 人の従業員に n 種の仕事を割り当てる問題で,

$$\begin{aligned}
&\text{minimize (あるいは maximize)} \\
&\qquad z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \\
&\text{subject to} \\
&\qquad \sum_{i=1}^{n} x_{ij} = 1 \quad (j = 1, 2, \ldots, n), \\
&\qquad \sum_{j=1}^{n} x_{ij} = 1 \quad (i = 1, 2, \ldots, n), \\
&\qquad x_{ij} \ge 0 \quad (i, j = 1, 2, \ldots, n)
\end{aligned}$$

と記述される. ただし, 従業員と仕事にはそれぞれ $1, 2, \ldots, n$ という通し番号がついているものとする. 変数 x_{ij} は, 従業員 i に仕事 j が割り当てられるときに 1 で, その他のときに 0 をとるものとする. また, c_{ij} は従業員 i に仕事 j を割り当てるときの "コスト" (あるいは "満足度") である. 上記の制約条件の意味は, 各従業員にはただ一つの仕事が割り当てられ, 各仕事はただ一人の従業員によってなされるということである. また, 上記の目的関数の意味は総コストが最小 (あるいは総満足度が最大) になる組み合わせを求めよということである (従業員と仕事はすべて異なるものとする). 変数 x_{ij} は連続変数であるが, 上記の割り当て問題の最適解において, 各変数の値は自然に 0 または 1 になる.

整列問題を割り当て問題として解釈するには, 人 i に仕事 j を割り当てるときの "コスト" (あるいは "満足度") を ja_i として, 総コストが最小 (あるいは総満足度が最大) になる組み合わせを求めると解釈すれば良い (図 1).

3.2 整列問題を解く電気回路

割り当て問題は特殊な最小費用流問題 (輸送問題) である. 最小費用流問題とは, n ケ所の生産地と n ケ所の消費地があり, 生産地 i での生産量 (供給) を s_i, 消費地 j での消費量 (需要) を d_j, 生産地 i から消費地 j へ運ぶときの単位量あたりの輸送費用を c_{ij} とするとき, 各生産地から運び出される量の和はその生産地での生産量を越えず, 各消費地に運び込まれる量の和はその消費地での消費量を下回らないという制約条件の下で, 総輸送費用を最小とするように, 各生産地から各消費地への輸送量を決める問題である. 生産量と消費量をそれぞれ 1 としたものが割り当て問題である.

最小費用流問題は, 図 2 の電気回路で, 等価的に解くことができることが知られている ([5]). 図 2 の電気回路には $n^2 + 2n + 1$ 本の辺が含まれる. このことを, この最小費用流問題のネットワーク複雑度は $O(n^2)$ であるということにしよう. 問題 P6, P7 を割り当て問題あるいは最小費用流問題として解釈した問題のネットワーク複雑度も $O(n^2)$ である.

3.3 整列問題の分解

問題 P6, P7 では, 整列問題を n^2 個の変数を用いて, 割り当て問題として定式化した. ところで, 係数 $c_{ij} (i = 1, 2, \ldots, n; j = 1, 2)$ を次のとおりに定めて, 最小費用流問題を考えてみる.

$$c_1 = \cdots = c_{n/2} = 0, \quad c_{n/2+1} = \cdots = c_n = 1$$

ただし, n は偶数としてある. ここで考える最小費用流問題は

minimize (あるいは maximize)

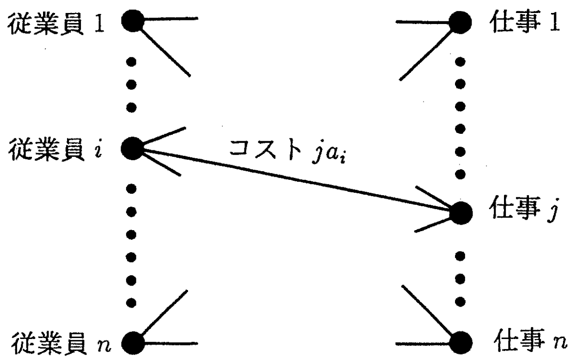

図 1 整列問題の割り当て問題としての解釈

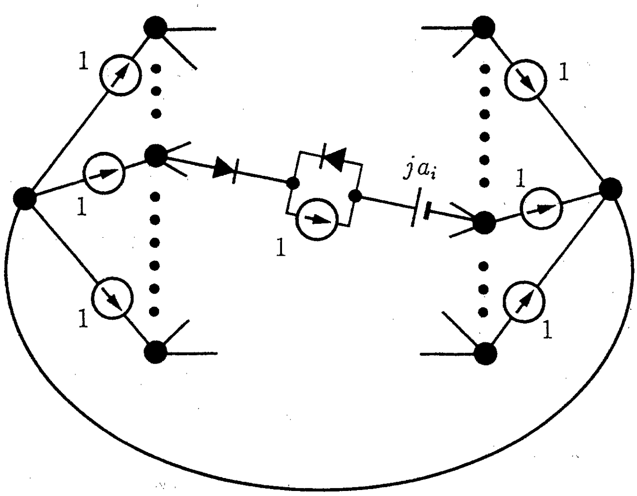

図 2 整列問題を解く回路

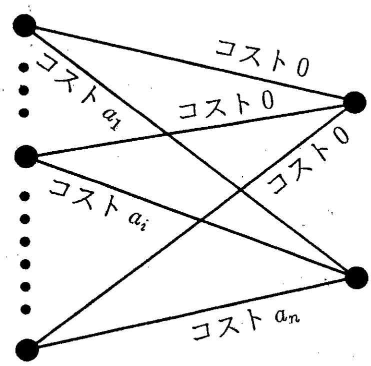

図 3 数値を大きい方と小さい方に分ける回路

$$z = \sum_{i=1}^{n} \sum_{j=1}^{2} c_j a_i x_{ij}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = n/2 \quad (j = 1, 2),$$

$$\sum_{j=1}^{2} x_{ij} = 1 \quad (i = 1, 2, \ldots, n),$$

$$x_{ij} \geq 0 \quad (i = 1, 2, \ldots, n; j = 1, 2)$$

すなわち, 問題 P6, P7 における目的関数中の ja_i が $c_j a_i$ になっている. この最小費用流問題の最適解は $a_1, a_2, \ldots, a_n$ 中の大きい方 (小さい方) 半分を与える. すなわち, 与えられた n 個の数値の大きい方 (小さい方) 半分を求める $O(n)$ のネットワーク複雑度の問題が作られた (図 3).

ここで, 得られた大きい方 (小さい方) 半分に対して同様の問題を考える. それぞれの問題のネットワーク複雑度は前の問題のネットワーク複雑度の半分であるから, 両者を合わせて $O(n)$ となる. この考え方を再帰的に適用すれば, 最終的に $a_1, a_2, \ldots, a_n$ は降順あるいは昇順に整列される. 再帰の段数は $O(\log n)$ である. このことを, n 個の数値を整列する最良のアルゴリズムの計算複雑度が $O(n \log n)$ であることと比較することは興味深い ([12]).

4 おわりに

整列問題を線形計画問題 - 特に, 割り当て問題 - として記述する方法を示した. さらに, その割り当て問題を解く電気回路を示した. 割り当て問題 (あるいは最小費用流問題) として記述できる問題の複雑度を表すために, ネットワーク複雑度の概念を導入した. これにより, 整列問題のネットワーク複雑度を $O(n^2)$ あるいは $O(n \log n)$ とすることができることを示した. 整列問題の入力データから対応する電気回路を構成することが今後の課題として残された.

本研究は, 平成 5 年度文部省科学研究費補助金 (一般研究 (C)05680267) の援助を受けた.

参考文献

[1] 茨木俊秀, "アルゴリズムとデータ構造," 昭晃堂, 1991.

[2] 小林孝次郎, "計算の複雑さ," 昭晃堂, 1988.

[3] E. Börger, *Computability, Complexity, Logic*, North-Holland, 1989.

[4] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, **4**, 374-395 (1984).

[5] M. Iri, *Network Flow, Transportation, and Scheduling – Theory and Algorithms*, Academic Press, 1969.

[6] É. Tardos, "A strongly polynomial minimum cost circulation algorithm," *Combinatorica*, **5**, 247-255 (1985).

[7] S. Fujishige, "A capacity-rounding algorithm for the minimum-cost circulation proglem – a dual framework of the Tardos algorithm," *Mathematical Programming*, **35**, 298-308 (1986).

[8] G. B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, 1963.

[9] L. R. Ford and D. R. Fulkerson, *Flow in Networks*, Princeton University Press, 1962.

[10] V. Chvátal, *Linear Programming*, W. H. Freeman and Company, 1983.

[11] T. Sunaga and M. Iri, "Theory of communication and transportation networks," *RAAG Memoirs*, Vol.2, pp.22-46 (1958).

[12] D. E. Knuth, *The Art of Computer Programming*, Vol.3 (Sorting and Searching), Addison-Wesley, 19.

[13] A. V. Aho, J. E. Hopcroft and J. D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974.

[14] 中森 眞理雄, "論理式充足可能性, 整数計画, および双線形計画について," 情報処理学会第 45 回全国大会講演論文集 (1), 3X-3 (1992).

[15] 荻原斉, 中森眞理雄, "論理式充足可能性、整数計画、双線形計画," 情報処理学会アルゴリズム研究会資料, 35-3, 1993.

付録

本論文では, 整列問題を線形計画問題として記述したり, その双対問題を示したりするので, 復習を兼ねて, 線形計画問題の正準形と双対理論について述べておく.

線形計画問題の正準形とは, 次の形の問題のことである.

maximize

$$z = \sum_{j=1}^{n} c_j x_j$$

subject to

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i \quad (i = 1, 2, \ldots, m),$$

$$x_j \geq 0 \quad (j = 1, 2, \ldots, n)$$

上記の問題を主問題とするとき, 双対問題は次のように書かれる.

minimize

$$s = \sum_{i=1}^{n} b_i y_i$$

subject to

$$\sum_{i=1}^{m} a_{ij} y_i \geq c_j \quad (j = 1, 2, \ldots, n),$$

$$y_i \geq 0 \quad (i = 1, 2, \ldots, n)$$

線形計画法の双対理論によれば, 主問題に最適解が存在するならば双対問題にも最適解が存在する (逆も成り立つ)([8]). 主問題の最適解を x_j^{opt}, 双対問題の最適解を y_i^{opt}とすると,

$$\sum_{j=1}^{n} c_j x_j^{opt} = \sum_{i=1}^{m} b_i y_i^{opt}$$

が成り立つ.