整列問題のネットワークフローモデル

萩原 斉　　中森 眞理雄
東京農工大学 工学部 情報工学大講座
〒184 東京都 小金井市 中町 2-24-16

整列を割り当て問題として記述し，電気回路として構成する方法を示す．単純な方法では、n 個のデータに対する整列問題のためのネットワークを作るのに，少なくとも入力だけでも $\Omega(n^2)$ の複雑度を伴う．この問題点を解決するために，データを分解するネットワークを作る方法を示す．これにより，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．このような分解は，事前に行う訳にはいかず，ネットワークを分解してはその下での最適流れを求め，その最適流れに基づいて次のネットワーク分解を行うという手順を繰り返すが，それらの手間の総和が $O(n\log n)$ であることを示す．このことは，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と比較しても，興味深い．

Network Flow Model for Sorting of Data

Hitoshi HAGIWARA and Mario NAKAMORI
Department of Computer Science
Tokyo A & T University
(Tokyo University of Agriculture and Technology)
2-24-16, Nakacho, Koganei, Tokyo, 184 Japan

It is shown that the problem of sorting data is described as an assignment problem. It is also shown that an electric circuit is possible to solve this assignment problem. However, if a simple method is used to construct the circuit, computational complexity of at least $\Omega(n^2)$ will be required for input of data. A network that decomposes data is proposed that overcomes this problem. Thus, we can sort n data with a network of $O(n)$ nodes and $O(n\log n)$ branches by solving the minimum cost flow problem. We have to repeat constructing a network that decomposes data and picking data from the optimal flow in the network. The total time complexiy is $O(n\log n)$. This corresponds the fact that the optimal sorting algorithm is of time complexity $O(n\log n)$.

1 はじめに

コンピュータを利用して問題を解決することは，問題をモデル化し数理的に定式化することから始まる．問題の良いモデル・良い定式化を定義することは困難であるが，一般的に，簡潔であり，良いアルゴリズムが得られる見通しのあるものは良いと考えられることが多い．特に，**線形計画問題**([1])は著しい成功をおさめた典型的な例であり，今日でも，線形計画法と関連づけられる問題には良いアルゴリズムが存在する場合が多い．

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 **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 **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 **NP 完全**であるという([2], [3])．

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．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([4], [5])．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([1],[6],[7])．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([8])．

筆者の一人は，かつて，**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した．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した．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 $O(n^2)$ で表すことを提唱した([9])．しかし，このネットワークを作るのに，少なくとも入力だけでも $O(n^2)$ の複雑度を伴うことは，容易に想像できる．そこで，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した．このことは，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 ([10], [11]) と比較すると，興味深いものであった．しかし，このような分解は，事前に行う訳にはいかず，ネットワークを分解してはその下での最適流れを求め，その最適流れに基づいて次のネットワーク分解を行うという手順を繰り返すため，筆者らの提唱する複雑度の定義が曖昧となっていた．本論文では，この問題に検討を加え，ネットワークを構成したり最適流れからデータを読み取ることに伴う手間のことを複雑度と定義することにより，整列問題のネットワークフローモデルにおける複雑度を明確に定義する．

2 整列と割り当て問題

2.1 線形計画法としての整列

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([9])．

問題P1

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

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

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

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

問題P1において変数 x_{ij} は，a_i が $a_1, a_2, \ldots, a_n$ の大きい方から j 番目のときに1でその他のとき

に0をとる．問題P1が整列問題であることは，一般に，$a_1 \leq a_2 \leq \ldots \leq a_n$, $b_1 \leq b_2 \leq \ldots \leq b_n$ のとき，1, 2, ..., n のいかなる順列 $p_1, p_2, \ldots, p_n$ に対しても，

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

であること，この問題が n 人の従業員を n 個の仕事に1対1に割り当てる問題であり(従業員 i を仕事 j に割り当てるときのコストが ja_i，図1参照)，最適解において変数 x_{ij} の値は自然に0または1となることから分かる．

問題P1の双対問題は次の通りである．

問題D1

maximize

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

subject to

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

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P2

maximize

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

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P2の双対問題は次の通りである．

問題D2

minimize

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

subject to

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

2.2 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 箇所の生産地と n 箇所の消費地があり，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([12])．図2の電気回路には $n^2 + 2n + 1$ 本の辺が含まれる．したがって，このようなネットワークを作るには，少なくとも $\Omega(n^2)$ の手間がかかることは明らかである．整列の最良アルゴリズムの時間複雑度が $O(n \log n)$ であることを考えると，上記の結果は不満足なものである．

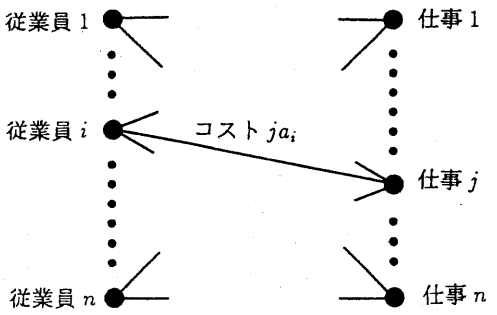

図 1 整列問題の割り当て問題としての解釈

3 整列問題の分解

問題P1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 $c_j(j=1,2)$ を次のとおりに定めて, 最小費用流問題を考えてみる.

$$c_0 = 0, \quad c_1 = 1$$

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)

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

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P1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.

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.

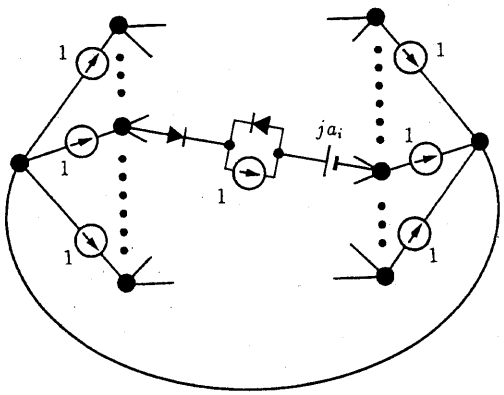

図 2 整列問題を解く回路

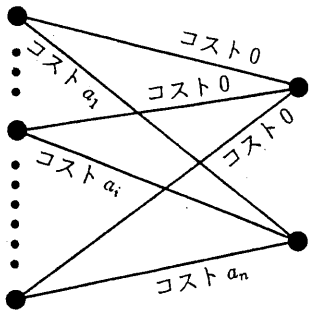

図 3 数値を大きい方と小さい方に分ける回路

4 おわりに

整列問題を解く電気回路を構成する方法を示した．この電気回路によってデータを大きい方と小さい方の2つに分割することができる．この回路を作り出したりこの回路の最適流れから次の段階の回路を作るための情報を取り出したりする手間が$O(n)$であること，分割の段数が$\log n$であることから，手間が全体で$O(n \log n)$であることが示された．回路の動作時間は無視できるので，結局，上記の手間$O(n \log n)$が回路を用いた整列の手間であることになる．

この方法を他の問題に一般化することによって，新たな計算複雑度の概念を定義することが期待される．

本研究は，平成7年度文部省科学研究費補助金(一般研究(C)07680339)の援助を受けた．

参考文献

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

[2] 茨木俊秀, "アルゴリズムとデータ構造," 昭晃堂, 1991.

[3] 小林考次郎, "計算の複雑さ," 昭晃堂, 1988.

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

[5] 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).

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

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

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

[9] 中森 眞理雄, "線形計画問題としての整列," 情報処理学会研究報告, **AL**36-9 (1993).

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

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

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

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