アルゴリズム 5−17
(1989. 1. 26)

2部グラフ上の最適完全マッチングを列挙するアルゴリズム

福田　公明　　　松井　知己
東京工業大学　理学部　情報科学科

この論文では、割当問題における最適解を全て求めるようなアルゴリズムを提案する。このアルゴリズムは1つの最適解を出力してからもう1つを出力するまでにO(n^3)の手間を必要とし、また用いる記憶容量は全体でO(n^2)となっている。割当問題における全ての最適解を求める問題は、与えられた2部グラフ上の完全マッチングを全て求めるという問題を子問題として含んでおり、このことより本アルゴリズムは完全マッチングの数え上げを行うアルゴリズムと見ることもできる。

Finding All Minimum Cost Perfect Matchings in Bipartite Graphs

Komei FUKUDA and Tomomi MATSUI
Department of Information Sciences, Tokyo Institute of Technology
Oh-okayama, Meguro-ku, Tokyo 152, Japan

The Hungarian method gives an efficient algorithm for finding a minimal cost perfect matching. This paper describes an efficient algorithm for finding all minimal cost perfect matchings. The computational effort required to generate each additional perfect matching is $O(n^3)$. And it requires $O(n^2)$ memory storage. Here we will show that the enumeration of all minimal cost perfect matchings can be reduced to the enumeration of all perfect matchings in some bipartite graph. Therefore our algorithm can be seen as an algorithm for enumerating all perfect matchings in a given bipartite graph.

1 Introduction

Let us consider a complete bipartite graph $B_n = (U, V, E)$ with row vertex set U, column vertex set V, satisfying that $|U| = |V| = n$, and edge set $E = U \times V$. A *matching* M is a subset of edges no two of which are incident with a common vertex. A matching M is *perfect* if each vertex is incident with exactly one member of M. Given a cost function $w : E \mapsto \boldsymbol{Q}$, where $\boldsymbol{Q}$ denotes the set of rational numbers, we define the *assignment problem* (**AP**) as the problem of finding an optimal matching, i.e., a perfect matching M which minimizes the total cost $\sum_{e \in M} w(e)$.

Recently many algorithms for finding an optimal matching of assignment problems are developed. Perhaps the best known algorithm is the Hungarian method [2]. In this paper we introduce an algorithm for finding all optimal matchings of the assignment problem. This problem can be solved by the algorithm for finding the Kth-best solution of assignment problems developed by Murty [3]. However Murty's algorithm requires $O(n^4)$ computational effort to generate each additional perfect matching and $O(Kn^2)$ memory storage. In the worst case, the number K becomes to $n!$. Our algorithm requires $O(n^3)$ computational effort to generate each additional perfect matching, and it saves the memory storage to $O(n^2)$ by using the notion of lexicographic ordering. It will become clear below that the problem finding all optimal matchings can be reduced to the problem of finding all perfect matchings in a given bipartite graph. It implies that our algorithm can be seen as an algorithm for enumerating all perfect matchings in a given bipartite graph.

2 Foundmental Properties

In this section we describe a good characterization of optimal matchings.

For any finite set S, we identify the function $f : S \mapsto \boldsymbol{R}$ with the corresponding vector in $\boldsymbol{R}^S$. For $s \in S$, we abbreviate $\{s\}$ by s. For any $v \in U \cup V$, $\delta(v)$ denotes the set of edges incident to the vertex v.

AP may be stated as an integer linear program in the following way.

(2.1) **AP:** $$\text{minimize} \quad \sum_{e\in E} w(e)\, x(e)$$

(2.2) $$\text{subject to} \quad \sum_{e\in\delta(u)} x(e) = 1, \quad \forall u \in U,$$

(2.3) $$\sum_{e\in\delta(v)} x(e) = 1, \quad \forall v \in V,$$

(2.4) $$x(e) \geq 0, \quad \forall e \in E,$$

(2.5) $$x(e) \text{ is integer}, \quad \forall e \in E,$$

where x is a real-valued vector in $\boldsymbol{R}^E$.

It is well-known that the integrality constraints (2.5) are superfluous to **AP**. The convex hull of all characteristic vectors (in $\boldsymbol{R}^E$) of the perfect matchings is determined by the inequalities (2.2), (2.3), and (2.4). Hence any feasible extreme point solution of the linear program (2.1), (2.2), (2.3), and (2.4) is a feasible integer solution of **AP**.

The dual of this linear program is

D: $$\text{maximize} \quad \sum_{u\in U} y(u) + \sum_{v\in V} y(v)$$
$$\text{subject to} \quad y(u) + y(v) \leq w(e), \quad \forall e = (u,v) \in E,$$

where y is a real valued vector in $\boldsymbol{R}^{U\cup V}$.

The *reduced cost* of an edge $e = (u,v)$ is defined by

$$\overline{w}(e) = w(e) - y(u) - y(v), \quad e = (u,v) \in E.$$

Given a dual solution $y \in \boldsymbol{R}^{U\cup V}$ (not necessarily feasible), the *admissible set* is a subset of edges $E(y) = \{e \in E : \overline{w}(e) = 0\}$, and the *admissible graph* is a bipartite graph $B_n(y) = (U, V, E(y))$.

Lemma 1 *Let $y^* \in \boldsymbol{R}^{U\cup V}$ be an optimal solution of the dual linear program* **D**. *Then a perfect matching M of B_n is optimal to* **AP**, *if and only if $M \subseteq E(y^*)$.*

Proof. Let $x^M \in \boldsymbol{R}^E$ be the characteristic vector of a perfect matching M. If M is optimal to **AP**, then

$$\begin{aligned}\sum_{u\in U} y(u) + \sum_{v\in V} y(v) &= \sum_{e\in E} w(e)\, x^M(e)\\ 0 &= \sum_{e\in E} w(e)\, x^M(e) - \sum_{e=(u,v)\in E} x^M(e)\,(y(u)+y(v))\\ 0 &= \sum_{e=(u,v)\in E} x^M(e)\,(w(e)-y(u)-y(v))\\ 0 &= \sum_{e\in E} \overline{w}(e)\, x^M(e)\,.\end{aligned}$$

From the feasibility of y, $\overline{w}(e) \geq 0$ for any edge e in E. It implies that $M \subseteq E(y^*)$. We can easily verify the converse implication. □

From the above lemma, it is obvious that the problem to find all optimal matchings is equivalent to the problem to find all perfect matchings in the admissible graph $B_n(y^*)$, induced from a dual optimal solution y^*. A dual optimal solution y^* can be obtained easily by the Hungarian method. In the next section, we describe an algorithm for finding all perfect matchings in a bipartite graph.

3 Main Framework for the Algorithm

Here we describe the main framework of our algorithm from the point of view of Kth-best solution.

Let y^* be an optimal solution of **D**, and let E^* be the admissible set $E(y^*)$. We fix an ordering of the edges in the admissible set as $E^* = \{e_1, e_2, \cdots e_q\}$. Given a sufficiently small positive number ε, the *perturbed problem* **PP** can be defined as follows.

(3.1) **PP:** minimize $-\sum_{e_i\in E^*} \varepsilon^i\, x(e_i)$

(3.2) subject to $\sum_{e\in\delta(v)\cap E^*} x(e) = 1, \quad {}^\forall v \in U,$

(3.3) $\sum_{e\in\delta(u)\cap E^*} x(e) = 1, \quad {}^\forall u \in V,$

(3.4) $x(e) \geq 0, \quad {}^\forall e \in E^*,$

(3.5) $x(e)$ is integer, $\quad {}^\forall e \in E^*,$

where x is a real-valued vector in $\boldsymbol{R}^{E^*}$.

Any extreme point solution of the linear program (3.1), (3.2), (3.3), and (3.4) is an integer solution of **PP**. The basic idea of our algorithm is to find the best, 2nd-best, 3rd-best, $\cdots$, and Kth-best matchings of the perturbed problem **PP** consequently.

For any vector $x, x' \in \boldsymbol{R}^{E^*}$, we say x *is lexicographically greater than* x', denoted by $x >_{lex} x'$, or $x' <_{lex} x$, if there exists i $(1 \leq i \leq q)$ satisfying that:

$$x(e_j) = x'(e_j), \quad {}^{\forall}e_j \in E^*,\ 1 \leq j < i\,,$$
$$x(e_i) > x'(e_i).$$

If ε is a sufficiently small positive number, then it is obvious that for any $x, x' \in \{0,1\}^{E^*}$

$$\sum_{e_i \in E^*} \varepsilon^i x(e_i) < \sum_{e_i \in E^*} \varepsilon^i x'(e_i) \quad \text{iff} \quad x <_{lex} x'.$$

Now let $x^{(K)}$ be the characteristic vector of the Kth-best matching of **PP**, if exists. Then it is clear that $x^{(K)} >_{lex} x^{(K+1)}$. For simplicity, we abbreviate $x(e_i)$ by $x(i)$ for any $x \in R^{E^*}$. Let $i_j^{(K)}$ be the index of the jth edge in the Kth-best matching. More precisely,

$$x^{(K)}(i_j^{(K)}) = 1, \quad {}^{\forall}j \in \{1, 2, \cdots, n\},$$
$$1 \leq i_1^{(K)} < i_2^{(K)} < \cdots < i_n^{(K)} \leq q.$$

Now we define the *perturbed subproblem* $\mathbf{PP}_r^{(K)}$ for $r = 1, 2, \cdots, n$ to obtain the $(K+1)$st-best solution from the information of Kth-best solution.

$$\mathbf{PP}_r^{(K)}: \qquad \text{minimize} \quad -\sum_{e_i \in E^*} \varepsilon^i\, x(e_i)$$
$$\text{subject to (3.2), (3.3), (3.4), (3.5), and}$$
$$x(e_i) = x^{(K)}(e_i)\,, \qquad 1 \leq i < i_r^{(K)}\,,$$
$$x(e_i) = 0\,, \qquad i = i_r^{(K)}\,.$$

Then the following lemma yields the method for obtaining the $(K+1)$st-best solution.

Lemma 2 *Let s be the smallest number satisfying:*

$$x^{(K)}(i_s^{(K)}) = 1, \quad x^{(K+1)}(i_s^{(K)}) = 0.$$

Then the following two statements hold.

(1) If $s < r \leq n$, then the problem $\mathbf{PP}_r^{(K)}$ is infeasible.

(2) If $r = s$, then $x^{(K+1)}$ is the unique optimal solution of $\mathbf{PP}_r^{(K)}$.

Proof. (1) Let us assume that the problem $\mathbf{PP}_r^{(K)}$ is feasible. Then the boundedness of the feasible region of $\mathbf{PP}_r^{(K)}$ implies that there exists an optimal solution x^* of $\mathbf{PP}_r^{(K)}$. The assumption $s < r \leq n$ implies that:

$$\begin{aligned}
&x^{(K+1)}(e_i) = x^*(e_i) = x^{(K)}(e_i), && 1 \leq i < i_s^{(K)},\\
&x^{(K+1)}(e_i) = 0 \text{ and } x^*(e_i) = x^{(K)} = 1, && i = i_s^{(K)},\\
&x^*(e_i) = x^{(K)}(e_i), && i_s^{(K)} < i < i_r^{(K)},\\
&x^*(e_i) = 0 \text{ and } x^{(K)} = 1, && i = i_r^{(K)}.
\end{aligned}$$

Then $x^{(K+1)} <_{lex} x^* <_{lex} x^{(K)}$, and it is a contradiction.

(2) From the definition of $\mathbf{PP}_s^{(K)}$, it is clear that: (i) if x is a feasible solution of $\mathbf{PP}_s^{(K)}$, then $x <_{lex} x^{(K)}$: (ii) $x^{(K)}$ is not feasible to $\mathbf{PP}_s^{(K)}$: and (iii) $x^{(K+1)}$ is feasible to $\mathbf{PP}_s^{(K)}$. It follows that $x^{(K+1)}$ is optimal to $\mathbf{PP}_s^{(K)}$. From the form of the objective function of $\mathbf{PP}$, it is obvious that the objective values of all matchings feasible to $\mathbf{PP}$ are different, and it implies the uniqueness of the optimal solution of $\mathbf{PP}_s^{(K)}$. □

Now we describe our algorithm.

Algorithm

Step 0: [Initialization];

Solve the original assignment problem **AP**.

Let x^* and y^* be the optimal solutions of **AP** and **D**.

Index the admissible set as:

$E(y^*) = \{e_1, e_2, \cdots, e_q\}$,

so that: $x^*(e_i) = 1, \ 1 \leq i \leq n,$ and

$x^*(e_i) = 0, \ n < i \leq q.$

Set $K := 1$.

Set $x^{(K)} := x^*$.

Step 1: [Out put the Kth-best solution];

Out put $x^{(K)}$ as the Kth-best solution of **PP**.

Set $K := K + 1$.

Set $r := n + 1$.

Step 2: [Stopping Rule];

If $r = 1$, then stop.

Else, set $r := r - 1$.

Step 3: [Solve the perturbed subproblem];

Solve the perturbed subproblem $\mathbf{PP}_r^{(K)}$.

If $\mathbf{PP}_r^{(K)}$ is infeasible, then go to step2.

Else, set $x^{(K)}$ be the unique optimal solution of $\mathbf{PP}_r^{(K)}$ and go to step1.

4 Discussion of the Algorithm

In this section, we discuss about the computational bound of our algorithm.

The assignment problem **AP** in the step 0 and the perturbed subproblems $\mathbf{PP}_r^{(K)}$ in the step 3 can be solved by the Hungarian method in $O(n^3)$ computational bound [2]. Then between the occurrences of the step1, $O(n^4)$ computation are required. It implies that to generate each additional perfect matching, this algorithm requires $O(n^4)$ computational effort. Here we describe the method to reduce the computational bound of the step 3.

In the step 3, first we only need to know whether the perturbed subproblem $\mathbf{PP}_r^{(K)}$ is feasible or not. Next if the perturbed subproblem $\mathbf{PP}_r^{(K)}$ is feasible, then we need to solve the problem $\mathbf{PP}_r^{(K)}$. It implies that the step 3 can be replaced by the following one.

Step 3': Step 3.1 Check the feasibility of the perturbed subproblem $\mathbf{PP}_r^{(K)}$.

If $\mathbf{PP}_r^{(K)}$ is infeasible, then go to step2.

Step 3.2 Solve the perturbed subproblem $\mathbf{PP}_r^{(K)}$.

Set $x^{(K)}$ be the unique optimal solution of $\mathbf{PP}_r^{(K)}$ and go to step1.

Now we consider the computational bound of step3.1 . Let E_1 be the subset of admissible set which correspond to the variables fixed to 1 in the perturbed subproblem $\mathbf{PP}_r^{(K)}$. And let E' be the subset of admissible set which correspond to the variables unfixed in the perturbed subproblem $\mathbf{PP}_r^{(K)}$. If there exists a perfect matching M in the admissible graph satisfying that $E' \cup E_1 \supseteq M \supseteq E_1$, then the characteristic vector of M is a feasible solution of the problem $\mathbf{PP}_r^{(K)}$. And the converse implication also holds. Then the step 3.1 can be reduced to the maximum cardinality matching problem. The maximum cardinality matching problem can be solved in $O(n^{2.5})$ computation [1]. However we can reduce the computational complexity of the step 3.1 even more. Let $M^{(K)}$ be the Kth-best perfect matching of $\mathbf{PP}$. Then the edge subset $M^{(K)} \setminus \{e_{i_r^{(K)}}\}$ is a $(n-1)$ cardinality matching, satisfying that $E' \cup E_1 \supseteq M^{(K)} \setminus \{e_{i_r^{(K)}}\} \supseteq E_1$. By using this matching as an initial solution of the maximum cardinality matching algorithm, it can be solved in $O(n^2)$ computation (see the chapter 10 of [4]). It implies that between the occurrences of the step 3.2, $O(n^3)$ computation is required. In the step3.2, the perturbed subproblem $\mathbf{PP}_r^{(K)}$ can be solved by Hungarian method in $O(n^3)$ computation. Therefor the computational steps required between the occurrences of the step1 is $O(n^3)$.

In this algorithm, to find the $(K+1)$st-best solution of $\mathbf{PP}$, we only need to maintain the admissible graph and the Kth-best solution. At this point, our algorithm differs from Murty's algorithm for finding Kth-best solution. And it is clear that our algorithm requires only $O(n^2)$ memory storage.

References

[1] J. E. Hopcroft and R. M. Karp. A $n^{5/2}$ algorithm for maximum matching in bipartite graphs. *Journal of SIAM Computing*, 2, 225–231, 1973.

[2] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2, 83–97, 1955.

[3] K. G. Murty. An algorithm for ranking all the assignments in order of increasing cost. *Operations Research*, 16, 682–687, 1968.

[4] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization*. Prentice-Hall, New Jersey, 1982.