An Algorithm for Fractional Assignment Problems

Maiko Shigeno

Yasufumi Saruwatari †

and

Tomomi Matsui ‡

Department of Management Science
Science University of Tokyo
1-3 Kagurazaka, Shinjuku-ku, Tokyo 162, Japan

†Department of Social Sciences
The National Defense Academy
1-10-20 Hashirimizu, Yokosuka-shi, Kanagawa 239, Japan

‡Department of Mathematical Engineering and Information Physics
Faculty of Engineering, University of Tokyo
Bunkyo-ku, Tokyo 113, Japan

Abstract. In this paper, we propose a polynomial time algorithm for *fractional assignment problems*. The fractional assignment problem is interpreted as follows. Let $G = (I, J, E)$ be a bipartite graph where I and J are vertex sets and $E \subseteq I \times J$ is an edge set. We call an edge subset $X(\subseteq E)$ *assignment* if every vertex is incident to exactly one edge from X. Given an integer weight c_{ij} and a positive integer weight d_{ij} for every edge $(i,j) \in E$, the *fractional assignment problem* finds an assignment $X(\subseteq E)$ such that the ratio $(\sum_{(i,j)\in X} c_{ij})/(\sum_{(i,j)\in X} d_{ij})$ is minimized.

Our algorithm is based on the *parametric approach* and employs the *approximate binary search* method. The time complexity of our algorithm is $O(\sqrt{n}\, m \log D \log(nCD))$ where $|I| = |J| = n$, $|E| = m$, $C = \max\{1, \max\{|c_{ij}| : (i,j) \in E\}\}$ and $D = \max\{d_{ij} : (i,j) \in E\} + 1$.

Key Words. Combinatorial Optimization, Mathematical Programming, Fractional Programming, Assignment Problems, Approximation Optimality, Parametric programming.

1 Introduction

The fractional programming problem has been widely studied in the last two decades (see the survey paper [13] for example). In this problem, an objective function, which is characterized by the ratio of given two linear functions, is to be minimized (or maximized). There are several classes of fractional programming problems. One of the classes consists of 0-1 fractional programming problems in which the variables are restricted to 0-1 integers [12, 6, 2]. In this paper, we treat fractional assignment problems which are special cases of 0-1 fractional programming problems.

Let $G = (I, J, E)$ be a bipartite graph where I and J are vertex sets and $E \subseteq I \times J$ is an edge set; $|I| = |J| = n$ and $|E| = m$. An *assignment* X is a subset of edges such that no two edges in X have a vertex in common and $|X| = n$. In this paper, we assume that the bipartite graph G contains at least one assignment. For each edge $(i, j) \in E$, weights c_{ij} and d_{ij} are associated. The *fractional assignment problem* (**FAP**) is to find an assignment X such that the ratio $(\sum_{(i,j)\in X} c_{ij})/(\sum_{(i,j)\in X} d_{ij})$ is minimized. This problem is formulated as follows:

$$
\begin{aligned}
(\mathbf{FAP}): \quad \text{minimize} \quad & \frac{\sum_{(i,j)\in E} c_{ij}x_{ij}}{\sum_{(i,j)\in E} d_{ij}x_{ij}} \\
\text{subject to} \quad & \sum_{j\in\delta(i)} x_{ij} = 1, \quad \text{for } i \in I, \\
& \sum_{i\in\delta(j)} x_{ij} = 1, \quad \text{for } j \in J, \\
& x_{ij} \in \{0, 1\}, \quad \text{for } (i, j) \in E,
\end{aligned}
$$

where $\delta(i)$ denotes the set of vertices adjacent with the vertex i. Here we assume that all c_{ij} 's are integer numbers and d_{ij} 's are positive integers.

In this paper, we propose an algorithm for the fractional assignment problem whose time complexity is $O(\sqrt{n}\, m \log D \log(nCD))$ where $C = \max\{1, \max\{|c_{ij}| : (i, j) \in E\}\}$ and $D = \max\{d_{ij} : (i, j) \in E\} + 1$. This is the fastest one that solves fractional assignment problems as far as we know. If the weight d_{ij} is equal to 1 for all $(i, j) \in E$, the problem **FAP** is equivalent to the linear assignment problem. In this case, the algorithm proposed in this paper solves the linear assignment problem in $O(\sqrt{n}\, m \log(nC))$ time and this time complexity is the same as that of Orlin and Ahuja's algorithm [11]. Our algorithm is based on the *parametric approach* and employs the *approximate binary search* method.

Section 2 gives an outline of a classical parametric approach to fractional programming problems. We also discuss some parametric methods and their time complexities when they are applied to fractional assignment problems. In Section 3, we introduce the approximate binary search method and show a result that is necessary to apply the method to fractional assignment problems. In Section 4, we propose the new algorithm. Finally, we analyze the time complexity of our algorithm in Section 5.

2 Previous works

For fractional programming problems, there are lots of algorithms based on the parametric approach which is associated with an auxiliary problem having one parameter [7, 3, 8]. When we adopt the approach to the fractional assignment problem, the auxiliary problem is defined as follows:

$$
\begin{array}{lll}
(\mathbf{FAP}(\lambda)): \text{ minimize} & \displaystyle\sum_{(i,j)\in E} (c_{ij} - \lambda d_{ij}) x_{ij} & \\
\text{subject to} & \displaystyle\sum_{j\in\delta(i)} x_{ij} = 1, & \text{for } i \in I, \\
& \displaystyle\sum_{i\in\delta(j)} x_{ij} = 1, & \text{for } j \in J, \\
& x_{ij} \in \{0,1\}, & \text{for } (i,j) \in E,
\end{array}
$$

where λ is a scalar parameter. Note that if λ is fixed, the auxiliary problem $\mathbf{FAP}(\lambda)$ becomes an ordinary linear assignment problem and we know that there proposed several algorithms for the linear assignment problem. For example, Fredman and Tarjan developed a strongly polynomial time algorithm using Fibonacci heaps and its time complexity is $O(n^2 \log n + nm)$ [4]. In 1992, Orlin and Ahuja developed a (weakly) polynomial time algorithm which is based on the auction algorithm and its time complexity is $O(\sqrt{n}\, m \log nM)$ where M is the maximum value of the integer edge weights [11]. These are currently the best available time bounds.

Let $z(\lambda)$ be the optimal value of $\mathbf{FAP}(\lambda)$ and λ^* the optimal value of **FAP**. The function $z(\lambda)$ is continuous, concave and strictly decreasing with respect to λ. We can observe the following properties (detailed see [13] for example).

Observation

(1) *The parameter* λ *is equal to* λ^* *if and only if* $z(\lambda)$ *is equal to 0.*

(2) *An assignment is optimal to* $\mathbf{FAP}(\lambda^*)$ *if and only if it is optimal to* **FAP**.

The property (1) says that when we fix the parameter λ, we can check whether the current value of λ is equivalent to λ^* or not by solving **FAP**(λ). Thus, solving **FAP** is essentially equivalent to finding $\lambda = \lambda^*$ with $z(\lambda) = 0$. These properties suggest search methods which test successive trial values of λ. In the rest of this section, we consider three well-known methods which test the parameter λ iteratively.

The first method is Dinkelbach's method [3] which is essentially equal to the Newton method. Recently, authors [9] showed that when we apply Dinkelbach's method to fractional assignment problems, the number of iterations is bounded by $O(\log(nCD))$, and thus Dinkelbach's method solves **FAP** in $O((n^2 \log n + nm) \log(nCD))$ time or $O(\sqrt{n}\, m(\log(nCD))^2)$ time.

The second method is the binary search method. It is clear from definitions of C and D that the optimal value of **FAP** is in the interval $[-C, C]$, and if the objective values (ratios) of two assignments are not equal, then the difference is at least $1/(nD)^2$. Therefore, the binary search method terminates after $O(\log(nCD))$ iterations, since it reduces the search interval by factor 2 in each iteration. When we employ the strongly polynomial time algorithm for linear assignment problems, the binary search method solves **FAP** in $O((n^2 \log n + nm) \log(nCD))$ time. However, if Orlin and Ahuja's algorithm for linear assignment problems is used in the binary search method, we need to replace the edge weights with $2(nD)^2 c_{ij} - \lfloor 2\lambda(nD)^2 \rfloor d_{ij}$, because every edge weight must be integer in their algorithm. In this case, the binary search method solves **FAP** in $O(\sqrt{n}\, m(\log(nCD))^2)$ time, since every edge weight is less than or equal to $O(n^2 CD^3)$.

In 1979, Megiddo [10] proposed an ingenious method for fractional combinatorial optimization problems. By employing his method, we can construct an algorithm for the fractional combinatorial optimization problem whose time complexity is at most square of the time complexity of an algorithm for the linear version of the combinatorial optimization problem. Thus, by incorporating two algorithms for linear assignment problems mentioned above, Meggido's method solves the fractional assignment problem in $O(n^4(\log n)^2 + n^2 m^2)$ time or in $O(nm^2(\log(nCD))^2)$ time.

3 Basic ideas

Our algorithm employs the approximate binary search method. The idea of the approximate binary search method is due to Zemel [15] and Orlin and Ahuja [11]. The approximate

binary search method is similar to the ordinary binary search method in the following sense. These two methods maintain the search interval $[LB, UB]$ containing the target (i.e. the optimal value λ^* of **FAP**), and reduce the search interval without excluding the target in each iteration. For the reduction of the search interval, the ordinary binary search method solves the auxiliary problem $\mathbf{FAP}(\lambda)$ by fixing λ to $(UB+LB)/2$, and updates $[LB, UB]$ to $[\lambda, UB]$ or $[LB, \lambda]$. In the approximate binary search method, however, we use an approximate solution to $\mathbf{FAP}(\lambda)$ (where $\lambda = (UB + LB)/2$) instead of an optimal solution to $\mathbf{FAP}(\lambda)$.

In the followings, we describe the definition of approximate solutions proposed by Bertsekas [1] and Tardos [14]. Let u_i and v_j be variables indexed by the vertex sets I and J, respectively. For an error bound ε, we say that $X'(\subseteq E)$ is an approximate solution to $\mathbf{FAP}(\lambda)$ if there exist u_i and v_j satisfying the following conditions:

$$\begin{aligned}
\textit{Primal Feasibility}: &\quad X' \text{ is an } \textit{assignment}.\\
\varepsilon\textit{-Dual Feasibility}: &\quad (c_{ij} - \lambda d_{ij}) - u_i - v_j \geq -\varepsilon, \quad \text{for all } (i,j) \in E.\\
\varepsilon\textit{-Complementary slackness}: &\quad (c_{ij} - \lambda d_{ij}) - u_i - v_j \leq \varepsilon, \quad \text{for all } (i,j) \in X'.
\end{aligned}$$

We call these conditions the *ε-optimality conditions,* and the solution X' is called an *ε-optimal solution.* Additionally, if X' is an ε-optimal solution to $\mathbf{FAP}(\lambda)$ for some fixed parameter λ, the objective value $\sum_{(i,j)\in X'} (c_{ij} - \lambda d_{ij})$ is said to be the *ε-optimal value* of X'.

Since the ordinary binary search method solves $\mathbf{FAP}(\lambda)$ exactly, it can reduce the search interval by a factor of 2. In the approximate binary search method, it is hard to reduce the search interval by a factor of 2, because we obtain only an ε-optimal solution of $\mathbf{FAP}(\lambda)$. The following lemma, however, gives an idea to reduce the search interval by using an ε-optimal solution.

Lemma 1 *Let X' be an ε-optimal solution to $\mathbf{FAP}(\lambda)$ for some fixed parameter λ. We denote the optimal value of* **FAP** *by λ^*. Then the followings hold.*

(1) *If* $\sum_{(i,j)\in X'} (c_{ij} - \lambda d_{ij}) \geq 0$, *then* $\lambda - 2\varepsilon \leq \lambda^*$.

(2) *If* $\sum_{(i,j)\in X'} (c_{ij} - \lambda d_{ij}) < 0$, *then* $\lambda + 2\varepsilon > \lambda^*$.

Proof. (1) Let u_i^λ and v_j^λ be variables corresponding to an ε-optimal solution X'. Denote by X^* an optimal solution to **FAP**. It follows from the ε-optimality conditions

that

$$\sum_{(i,j)\in X'} (c_{ij} - \lambda d_{ij}) - \sum_{i\in I} u_i^\lambda - \sum_{j\in J} v_j^\lambda \leq n\varepsilon$$

and

$$\sum_{(i,j)\in X^*} (c_{ij} - \lambda d_{ij}) - \sum_{i\in I} u_i^\lambda - \sum_{j\in J} v_j^\lambda \geq -n\varepsilon.$$

Summing up these inequalities, we get

$$\begin{aligned} \sum_{(i,j)\in X'} (c_{ij} - \lambda d_{ij}) - \sum_{(i,j)\in X^*} (c_{ij} - \lambda d_{ij}) &\leq 2n\varepsilon \\ - \sum_{(i,j)\in X^*} (c_{ij} - \lambda d_{ij}) &\leq 2n\varepsilon \\ -(\lambda^* - \lambda) &\leq \frac{2n\varepsilon}{\sum_{(i,j)\in X^*} d_{ij}} \leq \frac{2n\varepsilon}{n} = 2\varepsilon. \end{aligned}$$

(2) Similarly to (1). //

The above lemma shows that when we obtain an ε-optimal solution to $\mathbf{FAP}(\lambda)$ for some sufficiently small ε, either the upper bound or the lower bound of the search interval can be updated; that is, if the ε-optimal value to $\mathbf{FAP}(\lambda)$ is nonnegative, then the new lower bound becomes $\lambda - 2\varepsilon$, otherwise the new upper bound becomes $\lambda + 2\varepsilon$.

The proof of Lemma 1 directly implies the following corollary.

Corollary 2 *Let X' be an ε-optimal solution to $\mathbf{FAP}(\lambda)$ and X^* an optimal solution to* **FAP**. *Then*

$$\sum_{(i,j)\in X'} (c_{ij} - \lambda d_{ij}) - \sum_{(i,j)\in X^*} (c_{ij} - \lambda d_{ij}) \leq 2n\varepsilon.$$

4 Algorithm

In this section, we will develop an algorithm for fractional assignment problems which employs the approximate binary search method. Our algorithm maintains the scalar parameter, the error bound and the search interval including the optimal value λ^*.

Here, we denote the parameter, the error bound and the search interval in the k-th iteration by λ^k, ε^k and $[LB^k, UB^k]$ respectively. Our algorithm sets λ^k and ε^k as below;

$$\lambda^k = \frac{UB^k + LB^k}{2}, \qquad \varepsilon^k = \frac{UB^k - LB^k}{8},$$

and finds an ε^k-optimal solution X^k of $\mathbf{FAP}(\lambda^k)$ in the k-th iteration. Lemma 1 implies that if the ε^k-optimal value of X^k is nonnegative, then the search interval $[LB^{k+1}, UB^{k+1}]$

becomes $[\lambda^k - 2\varepsilon^k, UB^k]$, otherwise it becomes $[LB^k, \lambda^k + 2\varepsilon^k]$. The definitions of λ^k and ε^k show that $UB^{k+1} - LB^{k+1} = 3/4(UB^k - LB^k)$. Thus the $(k+1)$-th interval length is 3/4 of the k-th interval length. The above is the same technique that Orlin and Ahuja adopted the approximate binary search method into their algorithm [11]. In the first iteration, our algorithm sets $[LB^1, UB^1] = [-C, C]$. Then it is clear that the initial search interval contains the optimal value λ^*.

Next, we discuss the terminal condition of our algorithm. We assume that the interval length becomes less than $1/(nD)^2$ for the first time at ℓ-th iteration; i.e. $UB^{\ell-1} - LB^{\ell-1} \geq 1/(nD)^2 > UB^{\ell} - LB^{\ell}$. Then at the entrance of ℓ-th iteration, the approximate binary search method stops and our algorithm finds an ε^{ℓ}-optimal solution to $\mathbf{FAP}(UB^{\ell})$. The following shows that an ε^{ℓ}-optimal solution to $\mathbf{FAP}(UB^{\ell})$ is also optimal to **FAP**.

Lemma 3 *Assume that* $\mathrm{UB}^{\ell} - \mathrm{LB}^{\ell} < 1/(nD)^2$. *When an assignment* X' *is an* ε^{ℓ}*-optimal solution to* $\mathbf{FAP}(\mathrm{UB}^{\ell})$, *it is also optimal to* **FAP**.

Proof. We denote the ratio $(\sum_{(i,j)\in X'} c_{ij})/(\sum_{(i,j)\in X'} d_{ij})$ by λ'. Let X^* be an optimal solution to **FAP** and λ^* the optimal value of **FAP**.

We assume that X' is not optimal to **FAP**. Then it is clear that $\lambda' - \lambda^* > 0$ and

$$\lambda' - \lambda^* = \frac{\sum_{(i,j)\in X'} c_{ij} \sum_{(i,j)\in X^*} d_{ij} - \sum_{(i,j)\in X^*} c_{ij} \sum_{(i,j)\in X'} d_{ij}}{\sum_{(i,j)\in X'} d_{ij} \sum_{(i,j)\in X^*} d_{ij}} \geq \frac{1}{(nD)^2}.$$

Furthermore, facts that $UB^{\ell} - LB^{\ell} < 1/(nD)^2$ and $LB^{\ell} \leq \lambda^*$ imply $UB^{\ell} < \lambda^* + 1/(nD)^2$. Now we get $UB^{\ell} < \lambda^* + 1/(nD)^2 \leq \lambda'$. Then the properties $\lambda^* \leq UB^{\ell} < \lambda'$ imply that

$$\begin{aligned}
&\sum_{(i,j)\in X'} (c_{ij} - (UB^{\ell})d_{ij}) - \sum_{(i,j)\in X^*} (c_{ij} - (UB^{\ell})d_{ij}) \\
&= \sum_{(i,j)\in X'} d_{ij}(\lambda' - UB^{\ell}) - \sum_{(i,j)\in X^*} d_{ij}(\lambda^* - UB^{\ell}) \\
&\geq n(\lambda' - UB^{\ell}) - n(\lambda^* - UB^{\ell}) = n(\lambda' - \lambda^*) \geq n\frac{1}{(nD)^2} = \frac{1}{nD^2}\,.
\end{aligned}$$

From Corollary 2 and $UB^{\ell} - LB^{\ell} < 1/(nD)^2$,

$$\sum_{(i,j)\in X'} (c_{ij} - (UB^{\ell})d_{ij}) - \sum_{(i,j)\in X^*} (c_{ij} - (UB^{\ell})d_{ij}) \leq 2n\varepsilon^{\ell} = 2n\frac{UB^{\ell} - LB^{\ell}}{8} < \frac{1}{4nD^2}\,.$$

Thus, $1/(nD^2) \leq \sum_{(i,j)\in X'} (c_{ij} - (UB^{\ell})d_{ij}) - \sum_{(i,j)\in X^*} (c_{ij} - (UB^{\ell})d_{ij}) < 1/(4nD^2)$ and it is a contradiction. //

Lemma 3 shows that we can correctly obtain an optimal solution to **FAP** by this terminal condition.

Now, we describe our algorithm.

Algorithm
begin
$[LB, UB] := [-C, C]$;
while $(UB - LB) \geq 1/(nD)^2$ **do**
begin
$\lambda := (LB + UB)/2$; $\varepsilon := (UB - LB)/8$;
find an ε-optimal solution X' of **FAP**(λ); — $(\star)$
if the ε-optimal value is nonnegative
then $[LB, UB] := [\lambda - 2\varepsilon, UB]$
else $[LB, UB] := [LB, \lambda + 2\varepsilon]$
end;
$\lambda := UB$; $\varepsilon := (UB - LB)/8$;
find an ε-optimal solution X' of **FAP**(λ);
output X' as an optimal solution of **FAP**
end.

5 Analysis of the algorithm

Finally, we will show that our algorithm runs in $O(\sqrt{n}\, m \log D \log(nCD))$ time.

Our algorithm reduces the interval length by $3/4$ in each iteration. At the first iteration, the search interval is $[-C, C]$ and its length is $2C$. When the interval length becomes less than $1/(nD)^2$, the approximate binary search method terminates. Thus, the number of iterations of the approximate binary search method is bounded by $O(\log(nCD))$.

Now, we discuss the operations in each iteration. At the line marked $(\star)$ in the above algorithm, we find an ε-optimal solution of **FAP**(λ). In [11], Orlin and Ahuja developed a procedure which finds an ε-optimal solution from a 2ε-optimal solution in $O(\sqrt{n}\, m)$ time. Our algorithm calls this procedure consecutively at the line marked $(\star)$.

The following lemma is convenient to obtain an initial solution.

Lemma 4 *If we set the initial search interval* $[LB^1, UB^1] = [-C, C]$ *and* $u_i = 0$ *for all* $i \in I$ *and* $v_j = 0$ *for all* $j \in J$, *then any assignment is a* $(4\varepsilon^1)$*-optimal solution to* **FAP**(λ^1).

Proof. From the definition of λ^k and ε^k, $\lambda^1 = 0$ and $\varepsilon^1 = C/4$. Let X be any assignment. For each edge $(i,j) \in X$, $(c_{ij} - \lambda^1 d_{ij}) - u_i - v_j = (c_{ij} - 0 \cdot d_{ij}) - 0 - 0 \leq C = 4(\frac{C}{4}) = 4\varepsilon^1$. And for all edge $(i,j) \in E$, $(c_{ij} - \lambda^1 d_{ij}) - u_i - v_j = (c_{ij} - 0 \cdot d_{ij}) - 0 - 0 \geq -C = -4(\frac{C}{4}) = -4\varepsilon^1$. //

By setting the initial search interval as described above, any assignment is $4\varepsilon^1$-optimal. Hence, we can start our algorithm with an arbitrary assignment (we can find a perfect matching in bipartite graph $G = (I, J, E)$ in $O(\sqrt{n}\ m)$ time by Hopcroft and Karp's algorithm [5]) and Orlin and Ahuja's procedure finds an ε^1-optimal solution from $4\varepsilon^1$-optimal solution in $O(2\sqrt{n}\ m) = O(\sqrt{n}\ m)$ time at the first iteration.

Let us consider the iterations after the first. Following lemma gives an idea to obtain an ε^{k+1}-optimal solution efficiently at $(k+1)$-th iteration.

Lemma 5 *Let X^k be an ε^k-optimal solution to* **FAP**(λ^k). *Then X^k is a $(D\varepsilon^k)$-optimal solution to* **FAP**(λ^{k+1}).

Proof. From the definition of λ^{k+1}, we have two cases: (i) $\lambda^{k+1} \geq \lambda^k$ (when the lower bound increases) and (ii) $\lambda^{k+1} \leq \lambda^k$ (when the upper bound decreases).

We begin with Case (i). Since X^k is an assignment, the $(D\varepsilon^k)$-primal feasibility condition holds. It can be easily shown that $\lambda^{k+1} = \lambda^k + \varepsilon^k$, since the lower bound increases. For each edge $(i,j) \in X^k$,

$$c_{ij} - \lambda^{k+1} d_{ij} - u_i - v_j \leq c_{ij} - \lambda^k d_{ij} - u_i - v_j \leq \varepsilon^k \leq D\varepsilon^k.$$

It implies that X^k satisfies the $(D\varepsilon^k)$-complementary condition. Furthermore, for every edge $(i,j) \in E$,

$$c_{ij} - \lambda^{k+1} d_{ij} - u_i - v_j = c_{ij} - (\lambda^k + \varepsilon^k) d_{ij} - u_i - v_j \geq -\varepsilon^k - \varepsilon^k d_{ij} = -\varepsilon^k (1 + d_{ij}) \geq -D\varepsilon^k.$$

Thus, the $(D\varepsilon^k)$-dual feasibility condition is kept and completes the proof.

Case (ii) can be proved similarly to Case (i). //

From the definition, ε^k is equal to $(4/3)\varepsilon^{k+1}$. Therefore the above lemma implies that the assignment obtained at the k-th iteration is a $((4/3)D\varepsilon^{k+1})$-optimal solution of **FAP**(λ^{k+1}). When we apply the procedure developed by Orlin and Ahuja p times, an $((4/3)D\varepsilon^{k+1}(1/2^p))$-optimal solution is obtained [11]. So, by setting $p = O(\log_2(4/3)D)$, we can find an ε^{k+1}-optimal solution of **FAP**(λ^{k+1}) in $O(\sqrt{n}\, m \log_2((4/3)D)) = O(\sqrt{n}\, m \log D)$ time.

Similarly to the above, it is easy to show that when an assignment $X^{\ell-1}$ is an $\varepsilon^{\ell-1}$-optimal solution to $\mathbf{FAP}(\lambda^{\ell-1})$, it is $(4D\varepsilon^{\ell})$-optimal to $\mathbf{FAP}(UB^{\ell})$. Thus, when the approximate binary search method terminates, our algorithm finds ε^{ℓ}-optimal solution to $\mathbf{FAP}(UB^{\ell})$ in $O(\sqrt{n}\, m \log D)$ time.

Lastly, we will show the overall time complexity of our algorithm. The number of iterations of the approximate binary search method is bounded by $O(\log(nCD))$. In each iteration, we can find an ε-optimal solution of $\mathbf{FAP}(\lambda)$ in $O(\sqrt{n}\, m \log D)$ time and other steps are executed in constant time. Summarizing the above, the following theorem holds.

Theorem 6 *Our algorithm correctly determines a minimum fractional assignment in* $O(\sqrt{n}\, m \log D \log(nCD))$ *time.*

6 Conclusion

In this paper, we developed a polynomial time algorithm for fractional assignment problems. Same discussion will be done for some optimization problem with fractional objective functions if for the linear version of the problem, there exists an algorithm which is based on approximation optimality.

References

[1] D.P. Bertsekas, A Distributed Algorithm for the Assignment Problem, Working Paper, Laboratory for Information and Decision Systems, M.I.T. (1979).

[2] R. Chandrasekaran, Minimal Ratio Spanning Trees, Networks 7 (1977) 335-342.

[3] W. Dinkelbach, On Nonlinear Fractional Programming, Management Science, 13 (1967) 492-498.

[4] M.L. Fredman and R.E. Tarjan, Fibonacci Heaps and Their Uses in Improved Network Optimization Algorithms, J. ACM 34 (1987) 596-615.

[5] J.E. Hopcroft and R.M. Karp, An $n^{2.5}$ algorithm for maximum matchings in bipartite graphs, SIAM J. Comput. 2 (1973) 225-231.

[6] H. Ishii and T. Ibaraki and H. Mine, Fractional Knapsack Problems, Math. Programming 13 (1976) 255-271.

[7] R. Jagannathan, On Some Properties of Programming Problems in Parametric Form Pertaining to Fractional Programming, Management Science 12 (1966) 609-615.

[8] E.L. Lawler, Combinatorial Optimization; Networks and Matroids (Holt, Rinehart and Winston, New York, 1976).

[9] T. Matsui and Y. Saruwatari and M. Shigeno, An Analysis of Dinkelbach's Algorithm for 0-1 Fractional Programming Problems, Technical Report METR92-14, Department of Mathematical Engineering and Information Physics, Unversity of Tokyo (1992).

[10] N. Megiddo, Combinatorial Optimization with Rational Objective Functions, Math. Oper. Res. 4 (1979) 414-424.

[11] J.B. Orlin and R.K. Ahuja, New Scaling Algorithms for The Assignment and Minimum Mean Cycle Problems, Math. Programming 54 (1992) 41-56.

[12] P. Robillard, (0,1) Hyperbolic Programming Problems, Naval Res. Logist. Quart. 18 (1971) 47-57.

[13] S. Schaible and T.Ibaraki, Fractional Programming, European J. Oper. Res. 12 (1983) 325-338.

[14] E. Tardos, A Strongly Polynomial Minimum Cost Circulation Algorithm, Combinatorica 2 (1983) 247-255.

[15] E. Zemel, A Linear Time Randomizing Algorithm for Searching Ranked Functions, Algorithmica 2 (1987) 81-90.