AUT Journal of Mathematics and Computing

AUT J. Math. Com., 1(2) (2020) 145-152
DOI: 10.22060/ajmc.2018.15109.1014

Approximation algorithms for multi-multiway cut and multicut problems on directed graphs

Ramin Yarinezhad[a], Seyed Naser Hashemi[*a]

[a]*Department of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran*

ABSTRACT: In this paper, we study the directed multicut and directed multi-multiway cut problems. The input to the directed multi-multiway cut problem is a weighted directed graph $G = (V, E)$ and k sets $S_1, S_2, ..., S_k$ of vertices. The goal is to find a subset of edges of minimum total weight whose removal will disconnect all the connections between the vertices in each set S_i, for $1 \leq i \leq k$. A special case of this problem is the directed multicut problem whose input consists of a weighted directed graph $G = (V, E)$ and a set of ordered pairs of vertices $(s_1, t_1), ..., (s_k, t_k)$. The goal is to find a subset of edges of minimum total weight whose removal will make for any $i, 1 \leq i \leq k$, there is no directed path from s_i to t_i. In this paper, we present two approximation algorithms for these problems. The so called region growing paradigm is modified and used for these two cut problems on directed graphs. using this paradigm, we give an approximation algorithm for each problem such that both algorithms have the approximation factor of $O(k)$ the same as the previous works done on these problems. However, the previous works need to solve k linear programming, whereas our algorithms require only one linear programming. Therefore, our algorithms improve the running time of the previous algorithms.

Review History:
Received:08 October 2018
Revised:21 July 2019
Accepted:26 December 2019
Available Online:01 September 2020

Keywords:
Approximation algorithm
Complexity
NP-hard problems
Directed multi-multiway cut
Directed multicut cut

1. Introduction

In the following, we first review some of the important cut problems which serve as a background for the problems considered in this paper. The undirected multiway cut problem is defined on an undirected graph $G = (V, E)$ with a given set $S = \{s_1, ..., s_k\} \subseteq V$ of vertices called terminals and a weight function c_e, $e \in E$. Here, the goal is to find the minimum weight subset of edges so that by deleting them, all terminals in S are disconnected. In other words, there is no path between any two vertices in S. It is proved that this problem, for $k \geq 3$, is NP-hard and MAX SNP-hard, for which a $2 - 2/k$ factor approximation algorithm is given [1]. In [2], using a geometric relaxation, an algorithm with an approximation factor of $1.5 - 1/k$ is introduced and it is improved to $1.3438 - \epsilon_k$ in [3].

For directed graphs, the version of the directed multiway cut problem is defined. Likewise, given a set of terminals same as above, we look for a minimum weight subset of edges whose deletions disconnect all directed paths between each pair of terminals. Vazirani and Yannakakis [4] showed that a directed multiway cut problem is also NP-hard and MAX SNP-hard. They introduced an algorithm with an approximation factor of $2 \log k$. The best known approximation algorithm, presented by Noar and Zosin [5], used a novel relaxation multiway flow to have an approximation algorithm within a factor of 2.

The problem of undirected multicut is another well-known problem defined on undirected graphs with a non-negative edge cost c_e, $e \in E$, and a set of ordered pairs of vertices, namely; $(s_1, t_1), ..., (s_k, t_k)$, which are called source-terminal vertices. In this case, the goal is to achieve a minimum cost subset of edges so that after removing

*Corresponding author.
E-mail addresses: yarinezhad@aut.ac.ir, nhashemi@aut.ac.ir

them, all sources become inaccessible from their corresponding terminals. For $k \geq 3$, it is shown that the problem is NP-hard and MAX SNP-hard [1]. Garg, Vazirani, and Yannakakis [6] proposed, by the region growing technique, an approximation algorithm with the approximation factor of $O(\log k)$. In [7] for this problem with this requirement that there are at most $(k-1)$ edge-disjoint paths connecting s_i to t_i in in the given graph, after removing the minimum cost subset of edges, an approximation algorithm has been proposed with an approximation factor of $O(r \log^{3/2} k)$, where r is a part of the input instance.

The directed multicut problem is defined as follows: given a directed graph $G = (V, E), |V| = n$ with a non-negative function $c_e > 0, e \in E$, and a set of ordered pairs of vertices $(s_1, t_1), ..., (s_k, t_k)$, we find a subset $F \subseteq E$ with minimum cost so that their removal from the graph makes each pair disconnected. That is, for any $i, 1 \leq i \leq k$, there is no directed path from s_i to t_i in the graph $G(V, E - F)$.

Furthermore, if the desire is also to disconnect the paths from t_i to s_i, for any $i, 1 \leq i \leq k$, we have an alternative version of the multicut problem called directed symmetric multicut problem.

As shown in [4], for $k \geq 2$, the directed multicut problem is NP-hard and MAX SNP-hard. In some papers it was shown that another version of this problem is NP-hard [8]. In literature, most of the works on directed multicut have been focused on the directed symmetric multicut problem [9, 10, 11, 12]. Even, Noar, Schieber and Sudan [11] presented an approximation algorithm with a factor of $O(min\{\log \tau^* \log\log \tau^*, \log n \log\log n)\}$, where τ^* is the value of the optimum fractional solution of the problem at hand, and n is the number of vertices in the graph. In general, for a non-symmetric version, using the technique of region growing, an algorithm with the approximation factor of $O(\sqrt{n \log k})$ is given [13]. For the general case, Gupta [14] introduced a simpler algorithm and improved the approximation factor to $O(\sqrt{n})$. Both above problems, studied by [13, 14], use a linear programming relaxation to approximate the solution. In the work of Saks, Samorodnitsky, Zosin [15], it is shown that the integrality gap for the linear programming relaxation is $O(k)$.

A more general problem on undirected graphs is the multi-multiway cut problem [16] in which the weight function $w : E \to \Re^+$ and k sets $S_1, S_2, ..., S_k$ are given. Here, our aim is to obtain a minimum weight subset of edges whose removal from the graph will disconnect all connections between the vertices in each set S_i, for $1 \leq i \leq k$. For $k = 1$, this problem is an undirected multiway cut problem, and if $|S_i| = 2, (1 \leq i \leq k)$, an undirected multicut problem is obtained. Avidor and Langberg [16] showed that the undirected multi-multiway cut problem is NP-hard and MAX SNP-hard, and using the region growing technique they could present an approximation algorithm within the factor of $O(\log k)$. When the input graph is a tree, in [17] has been shown that this problem is solvable in polynomial time, if the number of terminal sets is fixed and in [18] has been presented an approximation algorithm with a factor $O(\sqrt{k})$.

A directed version of the above problem is also defined namely as a directed multi-multiway cut problem. Similarly, for this problem, a weight function $w : E \to \Re^+$ on edges and k sets $S_1, S_2, ..., S_k$ are given. We seek to find a minimum weight subset of edges whose removal from the graph will disconnect all paths between the vertices in each set S_i, for $1 \leq i \leq k$. This problem generalizes the problems of directed multiway cut and directed symmetric multicut (when $k = 1$ and $|S_i| = 2$, respectively).

Note that the problem of directed multi-multiway cut cannot be viewed as a generalization of the undirected multi-multiway cut problem only by replacing each undirected edge by two unparalleled directed edges. For example, consider a tree with a root r, containing three leaves a, b and c and assuming that the weight of each edge is equal to one. In this case, we get the optimal value, $OPT = 2$, whereas substituting each edge by two directed edges gives $OPT = 3$, and this proves that two problems above are not equivalent.

From the results stated above, it is a direct result that the problems of directed multicut and directed multi-multiway cut can be approximated by a factor of $O(k)$. But for each of these problems, we require k linear programming to be solved in order to obtain the desired approximation solution. In this paper, we show that we can achieve the same result, i.e. an approximation with a factor of $O(k)$, by solving only one linear programming. To achieve this goal, the so called paradigm of region growing, introduced in [6] for undirected cut problems, is modified so that it can be useful to produce an approximate solution of the multicut and multi-multiway cut problems on directed graphs. In this paper, we solve the linear programmings, which are defined in the approximation algorithms, using the ellipsoid algorithm. Although the ellipsoid algorithm runs in polynomial-time, it is very slow practically. Nevertheless, it is a very important theoretical tool for developing polynomial time algorithms for solving the linear programmings. In fact, the ellipsoid algorithm runs in polynomial time but it is very slow in practice. Therefore, reducing the number of linear programmings in the proposed algorithm from k to 1 leads the proposed algorithms to be more practical than the previous algorithms for these problems.

1.1. Organization

The rest of this paper is organized as follows: In section 2, we present a linear programming relaxation for the directed multi-multiway cut problem which is used in [16] and [4]. Section 3 contains necessary definitions and

lemmas for the algorithm directed multi-multiway cut which proposed in section 4. Directed multicut Algorithm presented in section 5 and conclusion is brought in section 6.

2. Linear Programming Relaxation for the Directed Multi-Multiway Cut

Let a characteristic function $x(e)$, for any edge e in E, be defined as follows: if e belongs to the directed multi-multiway cut, put $x(e) = 1$, otherwise $x(e) = 0$. By using this function, we can find a directed multi-multiway cut with the minimum cost for every directed path between two vertices in a group. We call the set of all directed paths between any two vertices, which belong to one group, P. An integer program for the problem can be defined as follows:

$$\begin{aligned} \textit{minimize} \quad & \sum_{e\in E} w(e)x(e) \\ \textit{subject to} \quad & \sum_{e\in p} x(e) \geq 1, \qquad \forall p \in P \\ & x(e) \in \{0,1\}, \quad \forall e \in E. \end{aligned}$$

By relaxing this IP, we obtain the following linear programming relaxation:

$$\begin{aligned} \textit{minimize} \quad & \sum_{e\in E} w(e)x(e) \\ \textit{subject to} \quad & \sum_{e\in p} x(e) \geq 1, \quad \forall p \in P \\ & x(e) \geq 0, \quad \forall e \in E. \end{aligned}$$

In this LP, there is a constraint for each path. On the other hand, we may have an exponential number of paths with respect to the input size and as a result, exponential number of constraints. Nevertheless, we can solve this LP in polynomial time, using the ellipsoid algorithm [19]. For this LP, the separation oracle operates as follows: we get a solution x and assume that the length of each edge e is equal to $x(e)$. Then, we find the shortest directed path between two vertices which are needed to be disconnected from each other. For example for the pair (u, v), if the shortest path between u and v (either $v \to u$ or $u \to v$) is more than 1, then this constraint $\sum_{e\in p} x(e) \geq 1$ is true for all paths between u and v. Therefore, this LP can be solved in polynomial time.

To express the approximation algorithm for directed multi-multiway cut, we need several definitions and lemmas which are presented in the next section.

3. Definitions and Lemmas

To round the solution of the mentioned LP and obtain a directed multi-multiway cut, we use the region growing technique [6, 16]. Note that definitions in [6, 16] are related to undirected graphs while definitions presented here, are related to directed graphs.

We define a distance on edges and assume x is an optimal solution for the LP. Let $x(e)$ be the length of the edge e. The distance between two vertices u and v (either $v \to u$ or $u \to v$), which is defined based on $x(e)$, is the length of the shortest path between them. We represent this shortest path with $dist(u, v)$. We define:

$$B_x(s_{ij}, r) = \{v \in V : dist(s_{ij}, v) \leq r\},$$

where $1 \leq i \leq k$, $1 \leq j \leq |S_i|$ and $r \in \mathbb{R}^+$. $B_x(s_{ij}, r)$ is an area like a ball with center s_{ij} and radius r. Assume that $\delta(S)$ is the set of all edges which only one of their endpoints is in the set S. For a given radius r, let $wt(\delta(B_x(s_{ij}, r)))$ be the sum of weights of all edges which one of their endpoints is in $B_x(s_{ij}, r)$. $wt(\delta(B_x(s_{ij}, r)))$ is defined more precisely,

$$wt(\delta(B_x(s_{ij}, r))) = \sum_{e\in\delta(B_x(s_{ij},r))} w(e).$$

Let $c_i(r)$ be the sum of weights of directed edges whose one head only is inside these balls, where $1 \leq i \leq k$. $c_i(r)$ is defined as follows:

$$c_i(r) = \sum_{j=1}^{|S_i|} wt(\delta(B_x(s_{ij}, r))).$$

Assume that each edge e in the graph as being a pipe with cross-sectional area $w(e)$ and length $x(e)$. Then, the product $w(e)x(e)$ is equal to the volume of edge e. Thus, the solution of LP is the minimum volume of edges such that $dist(u, v) \geq 1$, where u and v are in the same group, and there is a path between them (either $v \to u$ or $u \to v$). Let x be an optimal solution for the LP, and $V^* = \sum_{e \in E} w(e)x(e)$ be the volume of all edges. We know that $V^* \leq OPT$ such that OPT is the optimal value for the IP. $v_i(r)$ is defined as follows:

$$v_i(r) = \beta V^* + \sum_{j=1}^{|S_i|} \Big(\sum_{\substack{e=(u,v)\in E \\ u,v \in B_x(s_{ij},r)}} w(e)x(e) + \sum_{\substack{e=(u,v)\in E \\ u \in B_x(s_{ij},r) \\ v \notin B_x(s_{ij},r)}} w(e)(r - dist(s_{ij}, u)) \Big),$$

where $\beta > 0$ and it is independent from r. We notice that an edge may appear in $c_i(r)$ more than once. That means we may have $\delta(B_x(s_{ij}, r)) \cap \delta(B_x(s_{ij'}, r)) \neq \emptyset$, for $1 \leq j \neq j' \leq |S_i|$. Thus, $c_i(r)$ is an upper bound on the cut. According to these definitions, we can express the Lemma 3.1, which is used in the proof of Lemma 3.2.

Lemma 3.1. *The function $v_i(r)$ is differentiable in $(0, \infty)$ except in some finite number of points. The derivative of this function is $c_i(r)$.*

Proof. The function $v_i(r)$ is not differentiable in points which the value of function $B_x(s_{ij}, r)$ changes. The function $B_x(s_{ij}, r)$, changes for the values of r in which there is a vertex v such that $dist(s_{ij}, v) = r$. Thus the number of points in which the function $v_i(r)$ is not differentiable, is finite. Beside this, according to the definition done for the function $v_i(r)$, the derivative of this function is $c_i(r)$.

□

Lemma 3.2 says in directed graphs, we can always find a radius $r < \frac{1}{2}$, such that the cost $v_i(r)$ is an upper bound for $c_i(r)$.

Lemma 3.2. *Let x be a feasible solution for the LP. Then for every s_{ij} there is a $r < \frac{1}{2}$ and at least an α ($\alpha > 0$) such that $c_i(r) \leq \alpha v_i(r)$.*

The proof of this lemma is given in Section 4.1. We first present the algorithm using this lemma.

4. Approximation Algorithm for Directed Multi-Multiway Cut

Our polynomial time approximation algorithm for directed multi-multiway cut is described in Algorithm 1. The algorithm solves the LP first and finds the optimal solution x. Then, the algorithm enters to a repetition loop and till there exists a path between two vertices in a group, the algorithm works as follows: assume that the set S_i is chosen in this iteration. In the beginning, it finds an r which satisfies the inequality of Lemma 3.2, and then it finds the set of balls with the center of vertices inside the S_i with the radius of r. Then it puts the edges, which have been cut by these balls, in the answer set F. Then, all of the vertices in balls and incident edges with them will be deleted from the graph.

Lemma 4.1. *Algorithm 1 returns a Directed Multi-Multiway Cut.*

Proof. For each ball like $B_x(s_{ij}, r)$, where $1 \leq i \leq k, 1 \leq j \leq |S_i|$, there is no vertex with the same group with s_{ij} in $B_x(s_{ij}, r)$ because the radius of each ball is smaller than $\frac{1}{2}$. The only case may lead to problems is that there are two vertices u and v in one ball, which are the members of another group and there is a path between them. In this case, only the central vertices and their incident edges will be deleted from the graph. Therefore, the path between the two vertices u and v will not be deleted from the graph. In the next iterations, at least one of the edges of the path between u and v will be put in the answer set.

□

Let α and β are two constants whose values are determined later on.

Algorithm 1 Approximation Algorithm for Directed Multi-Multiway Cut

Result: A Directed Multi-Multiway Cut

$F \leftarrow \emptyset$

Solve the LP and get the optimal solution x

while *there is a path between* $s_{ij} \in S_i$ *and* $s_{ij'} \in S_i$, *where* $1 \leq i \leq k$ *and* $1 \leq j \neq j' \leq |S_i|$ **do**

Find r_i such that $c_i(r_i) \leq \alpha v_i(r_i)$

Add $\bigcup_{j=1}^{|S_i|} \delta(B_x(s_{ij}, r_i))$ to F

Remove $\bigcup_{j=1}^{|S_i|} B_x(s_{ij}, r_i)$ and incident edges with it form the graph

$\forall l \in \{1, ..., k\},\ S_l \leftarrow S_l \cap V$

end

Return F

Theorem 4.2. *Algorithm 1 is a* $(\alpha(1+\beta)k)$*-approximation algorithm for Directed Multi-Multiway Cut.*

Proof.

According to Lemma 3.2, we have $c_i(r) \leq \alpha v_i(r)$, for every $1 \leq i \leq k$. Thus, $\sum_{i=1}^{k} c_i(r) \leq \alpha \sum_{i=1}^{k} v_i(r)$. Besides, according to the definition of $v_i(r)$ and algorithm, we have $\sum_{i=1}^{k} v_i(r) \leq (kV^* + k\beta V^*)$ .Thus:

$$F \leq \sum_{e \in F} w(e) = \sum_{i=1}^{k} c_i(r) \leq \alpha \sum_{i=1}^{k} v_i(r) \leq \alpha(1+\beta)kV^* \leq \alpha(1+\beta)kOPT.$$

The reason behind $\sum_{e \in F} w(e) = \sum_{i=1}^{k} c_i(r)$ is that F is a subset of the edges, which have been cut by the balls and $\sum_{i=1}^{k} c_i(r)$ is all of the edges, which have been cut by the balls.

□

4.1. The proof of Lemma 3.2 and finding the best values for α and β

Proof. (The proof of Lemma 3.2) We use the contradiction method. Assume that for every value of $r < \frac{1}{2}$ and every α $(\alpha > 0)$ we have $c_i(r) > \alpha v_i(r)$. Thus we have:

$$\int_0^{\frac{1}{2}} \frac{c_i(r)}{v_i(r)} dr > \alpha \int_0^{\frac{1}{2}} dr$$

According to Lemma 3.1, the function $v_i(r)$ is not differentiable at only a finite number of points. We call these points $r_0 = 0 \leq r_1 \leq ... \leq r_l \leq r_{l+1} = \frac{1}{2}$. Thus we have:

$$\begin{aligned} \int_0^{\frac{1}{2}} \frac{1}{v_i(r)} (\frac{dv_i(r)}{dr}) dr &= \sum_{j=0}^{l} \int_{r_j}^{r_{j+1}} \frac{1}{v_i(r)} (\frac{dv_i(r)}{dr}) dr \\ &= \sum_{j=0}^{l} (\ln(v_i(r_{j+1}^-)) - \ln(v_i(r_j))). \end{aligned}$$

Since $v_i(r)$ is an increasing function, this last sum is at most

$$\sum_{j=0}^{l}(\ln(v_i(r_{j+1})) - \ln(v_i(r_j))) = \ln v_i(\frac{1}{2}) - \ln v_i(0),$$

and we have $v_i(0) = \beta V^*$ because there are no balls when $r = 0$. In addition, we have $v_i(\frac{1}{2}) \leq \beta V^* + V^*$ because V^* is an upper bound on the edges in the balls. Thus, we have:

$$\ln(\frac{\beta V^* + V^*}{\beta V^*}) \geq \ln(\frac{v_i(\frac{1}{2})}{v_i(0)}) > \frac{\alpha}{2}$$
$$\ln(\frac{\beta+1}{\beta}) > \frac{\alpha}{2}. \qquad (*)$$

In order to reach a contradiction, we have to choose values for α and β such that the inequality (*) will not be true. On the other hand, the approximation factor of the algorithm is dependent on these two parameters directly. So we have to choose the appropriate value for α and β. Indeed, to find the best value for α and β, we should solve the following nonlinear program:

$$\begin{aligned} &\textit{minimize} && \alpha(1+\beta) \\ &\textit{subject to} && \ln(\frac{\beta+1}{\beta}) \leq \frac{\alpha}{2} \\ & && \alpha, \beta > 0. \end{aligned}$$

We have solved this nonlinear program using Matlab software and found the optimal value of α and β. These values are as follows $\alpha = 0.1$ and $\beta = 20.504$. If we put these values in the inequality (*), the contradiction is reached and Lemma 3.2 is proved. Using these values for α and β, the algorithm is an approximation algorithm with factor $(2.1504)k$ for the Directed Multi-Multiway cut problem.

□

5. Approximation Algorithm for Directed Multicut

Similar to the LP, presented in the previous section, we can provide an LP for the directed multicut problem. Let the characteristic function $x(e)$ be defined as follows: if e belongs to the directed multicut, set $x(e) = 1$, for each edge e, otherwise $x(e) = 0$. By using this function, we are able to find the directed multicut with the minimum weight for each directed path from s_i to t_i for $1 \leq i \leq k$. We call P the set of all directed paths from s_i to t_i for $1 \leq i \leq k$. A linear programming for solving this problem is given as follows:

$$\begin{aligned} &\textit{minimize} && \sum_{e\in E} w(e)x(e) \\ &\textit{subject to} && \sum_{e\in p} x(e) \geq 1, \quad \forall p \in P \\ & && x(e) \geq 0, \quad \forall e \in E. \end{aligned}$$

We provide a direct version of definitions like those used in region growth technique in [6]. We define a distance on edges, assume that x is an optimal solution for LP, let $x(e)$ be the length of edge e. We denote the shortest path from u to v, which is based on $x(e)$, with $dist(u, v)$. If there is no directed path from u to v, the value of $dist(u, v)$ is the shortest path between u and v in the graph, without noticing the direction of edges. Now we define:

$$B_x(s_i, r) = \{v \in V : dist(s_i, v) \leq r\}.$$

$B_x(s_i, r)$ is an area like a ball with center s_i, where $1 \leq i \leq k$, and radius $r \in \mathbb{R}^+$. Assume that the product of $w(e)x(e)$ is equal to the volume of edge e. Thus, the solution of the LP is the minimum volume of edges such that $dist(s_i, t_i) \geq 1$, for $1 \leq i \leq k$. Assume that x is an optimal solution for the LP. Let $V^* = \sum_{e\in E} w(e)x(e)$ be the volume of all edges, indeed V^* is the optimal value of LP. We know that $V^* \leq OPT$ such that OPT is the optimal value for the LP. $v_x(s_i, r)$ is defined as follows:

$$v_x(s_i,r) = \beta V^* + \sum_{\substack{e=(u,v)\in E \\ u,v\in B_x(s_i,r)}} w(e)x(e) + \sum_{\substack{e=(u,v)\in E \\ u\in B_x(s_i,r) \\ v\notin B_x(s_i,r)}} w(e)(r - dist(s_i,u)).$$

Let $\delta(s)$ be the set of all edges which only one of their endpoints is in the set s. For a given radius r, we define:

$$wt(\delta(B_x(s_i,r))) = \sum_{e\in\delta(B_x(s_i,r))} w(e).$$

According to these definitions, we can express Lemma 5.1, which is used in the proof of Lemma 5.2.

Lemma 5.1. *The function $v_x(s_i,r)$ is differentiable in $(0,\infty)$ except some finite numbers of points. The derivative of this function is $wt(\delta(B_x(s_i,r)))$.*

Lemma 5.2 demonstrates that in directed graphs, we can always find a radius $r < \frac{1}{2}$, such that the cost $v_x(s_i,r)$ is an upper bound for $wt(\delta(B_x(s_i,r)))$.

Lemma 5.2. *Assume that x is a feasible solution for LP. For every s_i there is a $r < \frac{1}{2}$ and at least an α ($\alpha > 0$) such that the following inequality is true:*

$$wt(\delta(B_x(s_i,r))) \le \alpha v_x(s_i,r).$$

The proof of Lemma 5.1 and Lemma 5.2 is similar to the proof of Lemma 3.1 and Lemma 3.2, respectively. In the rest of the paper, for simplicity we assume that $wt(r) = wt(\delta(B_x(s_i,r)))$ and $v(r) = v_x(s_i,r)$.

Our polynomial time approximation algorithm for directed multicut, which is called Algorithm 2, is similar to Algorithm 1. It first solves the LP and finds the optimal solution x. In every iteration, the algorithm finds a pair which there is a path between them and finds an area with a radius that satisfies the condition in Lemma 5.2. Then, the algorithm puts the edges, which have been cut by the area, in the answer set.

Theorem 5.3. *Algorithm 2 is an $O(k)$-approximation algorithm for the Directed Multicut problem.*

Proof.

We denote the set of vertices in the ball $B_x(s_i,r)$ with B_i. We assume that $B_i = \emptyset$ when no ball is selected for vertex s_i. We also assume F_i is the set of cut edges for B_i. It means F_i is equal to $\delta(B_i)$. Thus, we have $F = \bigcup_{i=1}^{k} F_i$. Assume that V_i is equal to the volume of all edges which are in the ball B_i and also the volume of edges which have one head in B_i. According to this definition, we have $V_i \ge v_x(s_i,r) - \beta V^*$ because V_i includes the volume of all edges in F_i. But $v_x(s_i,r)$ is contained only some part of these edges and an addition value βV^*. According to Lemma 5.2 and the value chosen for r in the algorithm, we have $wt(F_i) \le \alpha v_x(s_i,r) \le \alpha(V_i + \beta V^*)$. We know that the algorithm may not remove the edges which are incident with vertices in B_i in this iteration. Thus, an edge may belong to more than one area, on the other hand, there are at most k areas. Therefore, $\sum_{i=1}^{k} V_i \le kV^*$. So we have the following inequalities:

$$\sum_{e\in F} w(e) = \sum_{i=1}^{k} wt(F_i) \le \alpha \sum_{i=1}^{k} (V_i + \beta V^*) \le \alpha(1+\beta)kV^* \le \alpha(1+\beta)kOPT.$$

□

Similar to Section 4, the optimal value for α is 0.1 and β is 20.504. Thus, Algorithm 2 is an $(2.1504)k$-approximation algorithm for the Directed Multicut problem.

6. Conclusions

In this paper, we design approximation algorithms for the directed multi-multiway cut and directed multicut problems using the region growing technique [6, 16]. By this paradigm, we give an $O(k)$-approximation algorithm. The works previously done on these problems need to solve k linear programs, whereas our algorithms require to solve only one linear programming. Both algorithms use the same linear programming relaxation. A question of interest is to find the integrality gap of the linear programming relaxation for these problems.

References

[1] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, M. Yannakakis, The complexity of multiway cuts, In Proceedings of the twenty-fourth annual ACM symposium on Theory of computing, (1992) 241-251.

[2] G. Călinescu, H. Karloff, Y. Rabani, An improved approximation algorithm for multiway cut, In Proceedings of the thirtieth annual ACM symposium on Theory of computing, (1998) pages 48-52.

[3] D. R. Karger, P. Klein, C. Stein, M. Thorup, N. E. Young, Rounding algorithms for a geometric embedding of minimum multiway cut, Mathematics of Operations Research, 29(3) (2004) 436-461.

[4] N. Garg, V. V. Vazirani, M. Yannakakis, Multiway cuts in directed and node weighted graphs, In International Colloquium on Automata, Languages, and Programming, 487-498, Springer, 1994.

[5] J. Naor, L. Zosin, A 2-approximation algorithm for the directed multiway cut problem, SIAM Journal on Computing, 31(2) (2001) 477-482.

[6] N. Garg, V. V. Vazirani, M. Yannakakis, Approximate max-flow min-(multi) cut theorems and their applications, SIAM Journal on Computing, 25(2) (1996) 235-251.

[7] J. Chuzhoy, Y. Makarychev, A. Vijayaraghavan, Y. Zhou, Approximation algorithms and hardness of the k-route cut problem, ACM Transactions on Algorithms (TALG), 12(1) (2015) 1-40.

[8] J. Bang-Jensen, A. Yeo, The complexity of multicut and mixed multicut problems in (di) graphs, Theoretical Computer Science, 520 (2014) 87-96.

[9] G. Even, J. S. Naor, S. Rao, B. Schieber, Divide-and-conquer approximation algorithms via spreading metrics, *Journal of the ACM (JACM)*, 47(4) (2000) 585-616.

[10] T. Leighton, S. Rao, An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms, Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE MICROSYSTEMS RESEARCH CENTER, 1989.

[11] G. Even, J. S. Naor, B. Schieber, M. Sudan, Approximating minimum feedback sets and multicuts in directed graphs, Algorithmica, 20(2) (198) 151-174.

[12] P. N. Klein, S. A. Plotkin, S. Rao, E. Tardos, Approximation algorithms for steiner and directed multicuts, Journal of Algorithms, 22(2) (1997) 241-269.

[13] J. Cheriyan, H. Karloff, Y. Rabani, Approximating directed multicuts, Combinatorica, 25(3) (2005) 251-269.

[14] A. Agarwal, N. Alon, M. S. Charikar, Improved approximation for directed cut problems, In Proceedings of the thirty-ninth annual ACM symposium on Theory of computing, (2007) 671-680.

[15] M. Saks, A. Samorodnitsky, L. Zosin, A lower bound on the integrality gap for minimum multicut in directed networks, Combinatorica, 24(3) (2004) 525-530.

[16] A. Avidor, M. Langberg, The multi-multiway cut problem, Theoretical Computer Science, 377(1-3) (2007) 35-42.

[17] I. Kanj, G. Lin, T. Liu, W. Tong, G. Xia, J. Xu, B. Yang, F. Zhang, P. Zhang, B. Zhu, Improved parameterized and exact algorithms for cut problems on trees, Theoretical Computer Science, 607 (2015) 455-470.

[18] P. Zhang, D. Zhu, J. Luan, An approximation algorithm for the generalized k-multicut problem, *Discrete Applied Mathematics*, 160(7-8) (2012) 1240-1247.

[19] M. Grötschel, L. Lovász, A. Schrijver, The ellipsoid method and its consequences in combinatorial optimization, Combinatorica, 1(2) (1981) 169-197.

Please cite this article using:

Ramin Yarinezhad, Seyed Naser Hashemi, Approximation algorithms for multi-multiway cut and multicut problems on directed graphs, AUT J. Math. Com., 1(2) (2020) 145-152
DOI: 10.22060/ajmc.2018.15109.1014