Approximability of the k-Server Disconnection Problem

Sung-Pil Hong
Department of Industrial Engineering, Seoul National University, Seoul, Korea

Byung-Cheon Choi
Semiconductor Business, Samsung Electronics Co. Ltd., Gyeonggi-Do, Korea

Consider a network of k servers and their users. Each server provides a unique service that has a certain utility for each user. Now comes an attacker who wishes to destroy a set of network edges to maximize his net gain, namely the total disconnected utilities of the users minus the total edge-destruction cost. This *k-server disconnection problem* is NP-hard and, furthermore, cannot be approximated within a polynomially computable factor of the optimum when k is part of the input. Even for any fixed $k \geq 2$, there is a constant $\epsilon > 0$ such that approximation of the problem within a factor $1/(1+\epsilon)$ of the optimum is NP-hard. However, a $(\frac{1}{2} + \frac{1}{2^{k+1}-2})$-approximation can be created in the time of $O(2^k)$ applications of a min-cut algorithm. The main idea is to approximate the optimum with special solutions computable in polynomial time due to supermodularity. Therefore, when the the network has, as is usual in most cases, only a few servers, a 0.5-approximation can be carried out in polynomial time.
 NETWORKS, Vol. 50(4), 273–282 2007

Keywords: server–network disconnection; inapproximability; approximation algorithm; fixed-parameter cases

1. INTRODUCTION

Consider a network of k servers and their users. Each server provides its own unique service. For instance, the first and second servers can be, respectively, the mail- and web-server of an intranet. In our model, each server is also allowed to be the user of services other than its own. Each service offers a certain utility to each user. Now, we consider an attacker who wants to destroy a set of network edges with an objective that optimally trades off the disconnected utilities of users and the edge-destruction cost.

Received April 2006; accepted March 2007
Correspondence to: B. C. Choi; e-mail: cbc@optimn.anu.ac.kr
Contract grant sponsor: KOSEF; Contract grant number: R01-2005-000-10271-0
DOI 10.1002/net.20203
Published online in Wiley InterScience (www.interscience.wiley.com).

Such a model, to the authors' best knowledge, was first proposed by Cunningham [3], who considered a directed network with a single "headquarters" and the importance, expressed in a certain weight, of each node being reachable from the headquarters. Then the problem is to find a set of edges whose removal maximizes the ratio of the total weight of nodes disconnected from the headquarters to the edge-destruction cost. Therefore, one is concerned only with the one-way communication ability of the headquarters to send messages to the nodes. Also, Martel et al. [9] considered the single-server problem of maximizing the total disconnected utilities under an edge-destruction budget constraint, which has application in a distributive data storage system. The problem is then shown to be NP-hard by a reduction from the maximum clique problem. Martel et al. also proposed an exact method based on enumeration of "maximal cuts" and the cut-cost submodularity. Among the cut problems with a budget constraint studied by Engelberg et al. [5], both the weighted and unweighted version of the budgeted separating multiway cut problem (BSMC), can be considered as special cases of the multiserver extension of the model of Martel et al. [9]. They showed that the weighted BSMC is at least as difficult to approximate as the sparsest cut problem, but is approximable within a constant factor on trees. Hong and Choi [7], alternatively, in an undirected extension of the single-server model proposed by Cunningham [3], considered the problem of maximizing the ratio of the total disconnected utilities of users to the incurred edge-destruction cost.

In this paper, we consider the problem of maximizing the attacker's net gain, namely, the total disconnected utilities of users minus the edge-destruction cost. (Hence, we assume that the utilities and the edge-destruction cost can be converted to a common unit.) We first show that the problem is NP-hard and does not admit any approximation: for any polynomially computable number $\alpha(n)$, $\frac{1}{\alpha(n)}$-approximation is impossible unless P $=$ NP. The well-known multiway cut problem with $(k+1)$-terminals, as we will show, is easily reduced to the k-server disconnection problem. Hence, the k-server disconnection problem is NP-hard even for

$k = 2$. Furthermore, the polynomial reduction, combined with the inapproximability results for the multiway cut problem, enables us to show that for any fixed $k \geq 2$ there is some constant $\epsilon > 0$, such that an approximation of the problem within a factor $1/(1+\epsilon)$ of the optimum is impossible unless P = NP.

The second half of this paper treats the approximability of the problem with a fixed k. First we observe that a feasible solution of the problem determines a node partition $N = N_0 \cup N_1 \cup \cdots \cup N_{p-1}$, with p distinct sets for $p = 1, 2, \ldots, k+1$, or a *p-partition* (where p is determined by an optimality structure discussed later in Section 3). Accordingly then, the optimum will be the maximum of the values z_p^*, $p = 1, 2, \ldots, k+1$, where z_p^* denotes the optimum among the p-partitions. However, the larger the p value becomes, the more difficult it is to compute z_p^*: computing z_3^* is already NP-hard. The main idea of the approximation algorithm is to approximate z_p^*s of large ps with z_2^*, which is polynomially computable due to its supermodularity. More specifically, we first show that z_2^* can be computed within the time of $O(2^k)$ applications of a min-cut algorithm (which is polynomial for a fixed k). Then we prove that $z_2^* \geq \left(\frac{1}{2} + \frac{1}{2^{k+1}-2}\right) z_p^*$, for every $p = 3, 4, \ldots, k+1$, and z_2^*. This implies that when the network has, as is usual in most cases, only a few servers, a 0.5-approximation can be performed efficiently.

The rest of this section is devoted to a formal definition of the problem. In Section 2, we study how the k-server disconnection problem is related to other models of network node partitioning. By doing so, we establish inapproximability results for the problem. Also, we discuss some polynomial cases that will be used to develop the approximation algorithm in Section 3. Finally, Section 4 offers concluding remarks and future research.

1.1. Problem Formulation

Problem 1.1 (The k-server disconnection problem). *Consider a connected undirected graph $G = (N, E)$ with the node set $N = \{1, 2, \ldots, n\}$. Each edge $e \in E$ will be denoted by its end points $i < j$: $e = (i, j)$.*

We assume that the server set $K = \{s_1, \ldots, s_k\} \subseteq N$. For notational convenience, we also assume that $s_1 = 1, \ldots, s_k = k$. The service of s_l offers utility d_i^l to each user $i \in N$. Therefore, the utility of user i can be expressed as the k-dimensional vector

$$d_i = \left[d_i^1, \ldots, d_i^k\right]^T \in \mathbb{Z}_+^k.$$

Each server $s_l \in K$ is also the user of services other than its own, and hence the utility vector is also well defined for the server node s_l. (In this case, it is natural to assume $d_l^l = 0$, but it is inessential to the discussion of this paper.) The nonnegative edge-destruction cost will be denoted by $c_e \in \mathbb{Z}_+$, $e \in E$. If an edge set $F \subseteq E$ is removed from G, some users will be disconnected from their servers. Then the *k-server disconnection problem* is to find F that maximizes the total disconnected utilities $\text{dscn}(F)$ minus the total edge-destruction cost $c(F)$: $\max_F\{\text{dscn}(F) - c(F)\}$. In Figure 1a, a 2-server problem and its solution are illustrated. The utility vector d_i is attached to each node i. For instance, the first (second) server offers a utility of 2 (8, respectively) units to the user 3 and hence $d_3 = [2, 8]^T$.

If $F = \{(3, 5), (s_2, 4)\}$, then its removal creates a partition of N: $N_0 = \{4, 5\}$, $N_1 = \{s_1, 3\}$, $N_2 = \{s_2\}$. Then we have $\text{dscn}(F) = 36$. As $c(F) = 9$, the objective value is 27. We will assume, unless stated otherwise, that G is complete. Notice that we can do so without loss of generality by introducing a zero edge-destruction cost for nonadjacent nodes (Fig. 1b). Clearly, an optimal solution of the original graph is also optimal for the complete graph and vice versa. This completeness assumption simplifies the discussion. For instance, for any node partition $N = N_0 \cup N_1 \cup \cdots \cup N_{p-1}$, each set N_l, $l = 0, 1, \ldots, p-1$, induces a connected subgraph.

1.2. Polynomiality of the Single-Server Case

If there is a single server, say node 1, the problem is to find a cut $(N_1; N \setminus N_1)$ (with $1 \in N_1$) that maximizes the objective $\sum_{i \in N \setminus N_1} d_i^1 - \sum_{e \in (N_1; N \setminus N_1)} c_e$. It is easy to confirm that the first term $\delta(N_1) \equiv \sum_{i \in N \setminus N_1} d_i^1$ is a modular function: $\delta(A) + \delta(B) = \delta(A \cap B) + \delta(A \cup B)$ for all $A, B \subseteq N$. However, the cut cost $\gamma(N_1) \equiv \sum_{e \in (N_1; N \setminus N_1)} c_e$ is a well-known submodular function: $\gamma(A) + \gamma(B) \geq \gamma(A \cap B) + \gamma(A \cup B)$ for all $A, B \subseteq N$. Hence the objective function is a supermodular function of N_1, and the problem, as a supermodular maximization, is solvable in (strongly) polynomial time (see e.g. [11]). As seen later, the single-server problem can be formulated as a min-cut problem.

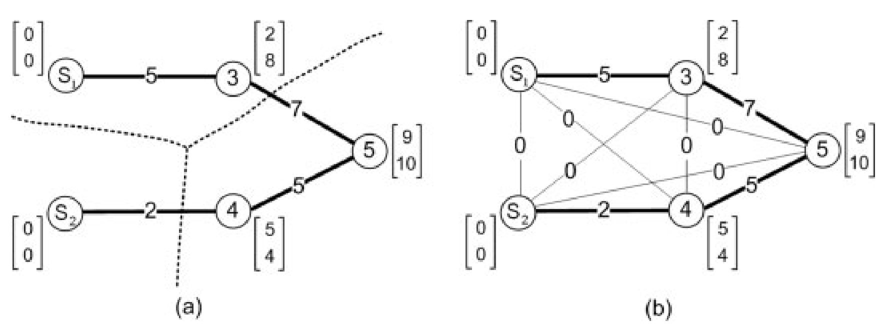

FIG. 1. (a) A 2-server disconnection problem and a solution whose value is 27. (b) Equivalent problem on a complete graph.

1.3. An Integer Programming Formulation

The k-server disconnection problem can be formulated as an integer program:

$$\max \sum_{l \in K} \sum_{i \in N} d_i^l y_i^l - \sum_{e \in E} c_e x_e \tag{1}$$

$$\text{s.t. } \sum_{e \in P} x_e \geq y_i^l, \quad \forall\, P \in \mathcal{P}_l(i),\ \ \forall i \in N,\ \ \forall\, l \in K, \tag{2}$$

$$x_e \in \{0,1\}, \quad \forall\, e \in E, \tag{3}$$

$$y_i^l \in \{0,1\}, \quad \forall i \in N,\ \ \forall l \in K, \tag{4}$$

where $\mathcal{P}_l(i)$ is the set of $l-i$ paths. Here y_i^l is a binary variable indicating whether node i is disconnected from server l. If so, that is, if $y_i^l = 1$, every $l - i$ path has at least one edge e destroyed ($x_e = 1$), which is enforced by (2). This integer program will be used to show that the k-server disconnection problem is polynomially reducible to the multicut problem.

2. RELATED PROBLEMS AND INAPPROXIMABILITY

Problem 2.1 (k-server-net vulnerability problem)**.** *The k-server-net vulnerability problem is a ratio version of the k-server disconnection problem: we want to find a set $F \subseteq E$ that, if removed, maximizes the ratio of the total disconnected utility to the total edge-destruction cost:* $\max_{F \subseteq E} \left\{ \frac{\text{dscn}(F)}{c(F)} \right\}$.

Notice that the optimal value, the maximum disconnected utility per unit cost, is a natural measure of the vulnerability of a multiserver network. For arbitrary k, the problem was proven NP-hard by a reduction from the sparsest cut problem [7].

Now, we show that any approximation algorithm for the k-server disconnection problem would yield a polynomial algorithm for the k-server-net vulnerability problem. Consider the binary search query to solve the k-server-net vulnerability problem. For $\lambda \in \left[\frac{1}{|E|c_{\max}}, knd_{\max}\right]$, is $\max_F \left\{ \frac{\text{dscn}(F)}{c(F)} \right\} > \lambda$? Here, $c_{\max}$ ($d_{\max}$) is the maximum value of an edge cost (a service utility, respectively). By standard arguments, it is not difficult to see that using $\log(k|E|nc_{\max}d_{\max})$ queries, we can find the optimum of the k-server-net vulnerability problem. The answer to the query is "yes" if and only if $\max_F\{\text{dscn}(F) - \lambda c(F)\} > 0$. The latter problem is yes if and only if the k-server disconnection problem with edge costs λc_e has a nontrivial solution $F \neq \emptyset$ whose value is positive. Thus, if we have an approximation algorithm with any polynomially computable factor, it can be used to answer the query correctly and hence to solve the k-server-net vulnerability in polynomial time.

Theorem 2.2. *For arbitrary k, the k-server disconnection problem is not approximable within any polynomially computable factor within the optimum.*

Problem 2.3 (Minimum k-terminal (or Multiway) cut problem)**.** *Given an undirected graph $G = (N, E)$ with nonnegative edge costs and a set of k specified nodes or* terminals*, $\{t_1, t_2, \ldots, t_k\}$, find a minimum cost set of edges $F \subseteq E$ such that the removal of F from G disconnects each terminal from all of the others.*

The minimum k-terminal cut problem is Max-SNP-hard for $k \geq 3$ [4]. Chopra and Rao [2] proposed an integer programming formulation and studied the associated polyhedron. Calinescu et al. [1] developed an algorithm with the approximation factor of $\left(\frac{3}{2} - \frac{1}{k}\right)$. Karger et al. [8] also proposed an 1.3438-approximation algorithm, which is an improvement for large ks.

Proposition 2.4. *The minimum 3-terminal cut problem (M3C) is polynomially reducible to the 2-server disconnection problem.*

Proof. On the same graph $G = (N, E)$, designate the first two terminals t_1 and t_2 of M3C as the two servers, and designate t_3 as a nonserver node in an instance of the 2-server disconnection problem. Assign utility vectors to the nodes as follows: set $d_{t_1} \equiv \left[d_{t_1}^{t_1}, d_{t_1}^{t_2}\right]^T = [0, M]^T$, $d_{t_2} \equiv \left[d_{t_2}^{t_1}, d_{t_2}^{t_2}\right]^T = [M, 0]^T$, and $d_{t_3} \equiv \left[d_{t_3}^{t_1}, d_{t_3}^{t_2}\right]^T = [M, M]^T$. The remaining nodes are assigned the zero utility vector $[0, 0]^T$. If M is sufficiently large, for instance, larger than z_{M3C}, the optimal value of M3C, then clearly we are better off by separating all of the nodes, t_1, t_2, and t_3, in the 2-server disconnection problem. Then an optimal solution of the 2-server instance is a min-cost edge set separating all three terminals. Hence, the 2-server disconnection problem instance is equivalent to the 3-terminal problem. Therefore, we can choose any M that has both polynomial encoding length and is greater than z_{M3C}, such as the sum of all edge costs plus 1, for a valid reduction, and this completes the proof.

However, we will show that setting M exactly to z_{M3C} is sufficient for the reduction to be valid, which will be used in the proof of Lemma 2.5. More specifically, we will show that when $M = z_{\text{M3C}}$, there is an optimal solution of the 2-server disconnection problem that separates all three nodes t_1, t_2, and t_3, and we can guarantee such an output with an additional but polynomial time computation.

Let F_{M3C} and F^*, respectively, be any optimal solution of M3C and the corresponding 2-server disconnection problem. First, notice that F^* separates t_1 and t_2; for, otherwise, adding the edges of F_{M3C} to F^* will increase the objective value of the 2-server disconnection problem by at least z_{M3C}, which contradicts the optimality of F^*.

We can assume further that F^* disconnects t_3 from both t_1 and t_2. Suppose, on the contrary, that t_3 is connected to, say, t_2. Then we introduce the additional step of computing the minimum $t_3 - t_2$ cut in the connected component containing t_2 and t_3 of the subgraph $G \setminus F^*$. Since $E(G \setminus F^*) \cap F_{\text{M3C}}$ contains a $t_3 - t_2$ cut of the component with cost no greater than z_{M3C}, the minimum $t_3 - t_2$ cut value should be no greater than z_{M3C} (where $E(G\backslash F^*)$ denotes the edges of the subgraph $G \setminus F^*$). Thus, the net change in the objective value of the 2-server disconnection problem, due to the addition of the edges of the minimum $t_3 - t_2$ cut to F^*, cannot be negative. Therefore, we can guarantee that, with an extra but polynomial time computation, all three nodes will be disconnected from each other in the optimal solution of the corresponding 2-server problem, and hence the reduction is also valid when $M = z_{\text{M3C}}$. ∎

Thereby we have established the NP-hardness of the k-server disconnection problem for any fixed $k \geq 2$. However, by combining it with the approximability results on the k-terminal cut problem, we can obtain stronger results:

Lemma 2.5. *If there exists an α-approximation of the 2-server disconnection problem, there also exists a $\frac{14-11\alpha}{3}$-approximation of the 3-terminal cut problem M3C.*

Proof. We first reduce the given instance of M3C to a 2-server disconnection problem, as in the proof of Proposition 2.4. In particular, we set $M = w^H$, where w^H is the objective value obtained from applying the $\left(\frac{3}{2} - \frac{1}{k}\right)$-approximation of Calinescu et al. [1] to the given instance of M3C. Then $z_{\text{M3C}} \leq w^H \leq \frac{7}{6} z_{\text{M3C}}$ (where z_{M3C} is the optimal value of M3C).

Subsequently, we apply the α-approximation to the 2-server disconnection problem. Since $M = w^H \geq z_{\text{M3C}}$ according to the arguments of the latter half of the proof of Proposition 2.4, by performing extra but polynomial time computation, if necessary, we can guarantee that all three nodes t_1, t_2, and t_3 will be separated without decreasing the objective value of the α-approximate solution. Therefore, we may assume that the α-approximate solution of the 2-server disconnection problem is also feasible for M3C.

Let z^α and z^*, respectively, be the objective values of the α-approximate solution and an optimal solution of the 2-server disconnection problem. Also by w^α we denote the edge-destruction cost of the α-approximate solution. Notice that w^α is then the objective value of the α-approximate solution as a feasible solution of M3C. From the correspondence between the solutions of the two problems, we have

$$z^* = 4M - z_{\text{M3C}} = 4w^H - z_{\text{M3C}} \quad \text{and} \quad z^\alpha = 4M - w^\alpha = 4w^H - w^\alpha,$$

and hence, from the assumption of α-approximability, we obtain

$$\frac{z^\alpha}{z^*} = \frac{4w^H - w^\alpha}{4w^H - z_{\text{M3C}}} \geq \alpha. \tag{5}$$

Combining (5) with $w^H \leq \frac{7}{6} z_{\text{M3C}}$, we obtain $w^\alpha \leq 4(1 - \alpha)w^H + \alpha z_{\text{M3C}} \leq \frac{14-11\alpha}{3} z_{\text{M3C}}$. ∎

Lemma 2.5 implies if α approaches 1, so also does the approximation factor of w^α. However, since M3C is Max-SNP-hard, the approximation factor α of the 2-server disconnection problem cannot be arbitrarily close to 1. Also, it is not difficult to see that if the $(k+1)$-server disconnection problem has an α-approximation, so does the k-server disconnection problem. Therefore, we have established the following inapproximability result for the k-server disconnection problem.

Theorem 2.6. *For any fixed $k \geq 2$, there is some constant $\epsilon > 0$ such that approximation of the k-server disconnection problem within a factor $1/(1+\epsilon)$ of the optimum is impossible unless* P = NP.

Another model related to the k-server disconnection problem is the multicut problem.

Problem 2.7 (Multicut problem). *Given an undirected graph $G = (N, E)$ with nonnegative edge costs and a set $S(\subseteq N \times N)$ of k pairs of nodes, $\{(s_1, t_1), \ldots, (s_k, t_k)\}$, find a minimum cost set of edges $F \subseteq E$ such that the removal of F from G disconnects s_i from t_i for all $i = 1, 2, \ldots, k$.*

The multicut problem is a generalization of the minimum k-terminal cut problem. For instance, given a 3-terminal cut problem, we can construct an equivalent multicut instance preserving the objective value simply by constructing all of the $\binom{3}{2}$ pairs of terminals. Hence, the multicut problem is also Max-SNP-hard for $k \geq 3$. Currently, $O(\log k)$-approximation is available [6].

Using a similar idea, the multicut problem with k pairs is also polynomially reducible to the k'-server disconnection problem with $k' \leq k$. Consider, for simplicity, the case $k = 3$. Assume, without loss of generality, that the s_i are all distinct. On the same topology and edge costs, designate s_1, s_2, and s_3, in order, as the three servers in the 3-server disconnection problem instance. The zero utility vectors are assigned to every node other than t_1, t_2, and t_3 whose utility vectors are, respectively, $[M, 0, 0]^T$, $[0, M, 0]^T$, and $[0, 0, M]^T$. If M is sufficiently large, all of the three pairs are necessarily separated at optimality. However, this reduction seems useless with respect to approximation.

Perhaps more interestingly, the converse is also true. The k-server disconnection problem is polynomially reducible to the multicut problem. To see this, we consider the integer programming formulation (1)–(4) of the k-server disconnection problem in Section 1. If we replace the y-variables with $z_i^l = 1 - y_i^l$, change the objective into minimization form by multiplying by -1, and remove the constant term $\sum_{l \in K} \sum_{i \in N} d_i^l$:

$$\max \sum_{l \in K} \sum_{i \in N} d_i^l y_i^l - \sum_{e \in E} c_e x_e$$
$$\Leftrightarrow \max \sum_{l \in K} \sum_{i \in N} d_i^l - \sum_{l \in K} \sum_{i \in N} d_i^l z_i^l - \sum_{e \in E} c_e x_e$$
$$\Leftrightarrow \min \sum_{l \in K} \sum_{i \in N} d_i^l z_i^l + \sum_{e \in E} c_e x_e,$$

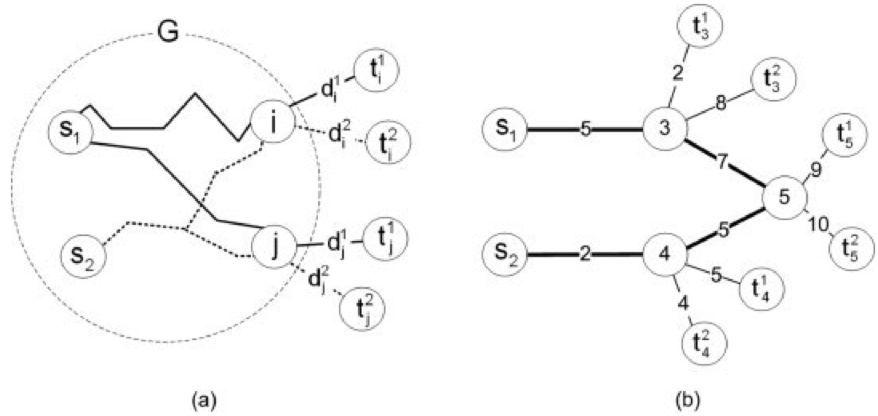

FIG. 2. (a) Extended path interpretation of (7) for $k = 2$. (b) Reduction to multicut applied to the example in Figure 1.

then we get the following integer program:

$$\min \sum_{l\in K}\sum_{i\in N} d_i^l z_i^l + \sum_{e\in E} c_e x_e \tag{6}$$

$$\text{s.t.} \sum_{e\in P} x_e + z_i^l \geq 1, \quad \forall\, P \in \mathcal{P}_l(i), \quad \forall\, i \in N,\ \forall\, l \in K, \tag{7}$$

$$x_e \in \{0, 1\}, \quad \forall\, e \in E, \tag{8}$$

$$z_i^l \in \{0, 1\}, \quad \forall i \in N,\ \forall l \in K, \tag{9}$$

The constraint (7) has an interesting interpretation. It requires that for every path in $\mathcal{P}_l(i)$, either at least one edge of the path is removed or z_i^l is 1. Thus, if we extend each path in $\mathcal{P}_l(i)$ to a new node t_i^l by adding an edge (i, t_i^l) of cost d_i^l, the constraint (7) is equivalent to requiring that every $l - t_i^l$ path is disconnected (Fig. 2.) There is a one-to-one correspondence between the $l - i$ paths and the $i - t_i^l$ paths. Therefore, by adding an edge (i, t_i^l) for all $i \in N$ and $l \in K$, the problem becomes the multicut problem with the pairs $\{(l, t_i^l) : l \in K,\ i \in N\}$.

Proposition 2.8. *The k-server disconnection problem is polynomially reducible to a multicut problem with the pairs between k distinct s-nodes and 2kn distinct t-nodes.*

In the earlier transformation, one might consider merging all t_i^l with the same l into a single node, say t^l, to obtain a multicut problem with k pairs $\{(s_1, t^1), \ldots, (s_k, t^k)\}$. That would reduce the 2-server disconnection problem to a muticut problem with two pairs. However, we can easily construct an example in which a feasible solution of the k-server disconnection problem maps to an infeasible solution of the multicut problem in the case that the t-nodes are merged. (Fig. 3a). The thick solid lines are the original edges with the positive destruction cost of the 2-server disconnection instance from Figure 1. (As we assigned zero utility vectors to the two servers, the added edges to the servers have zero cost and are omitted from the illustration for simplicity.) If we choose $F = \{(s_2, 4), (3, 5)\}$, then nodes 4 and 5 are disconnected from server s_1 and every node is disconnected from server s_2. (Note that as node 3 remains connected after deletion of F, the augmented edge $(3, t_3^1)$ is disconnected, in the augmented graph, for the multicut problem defined by

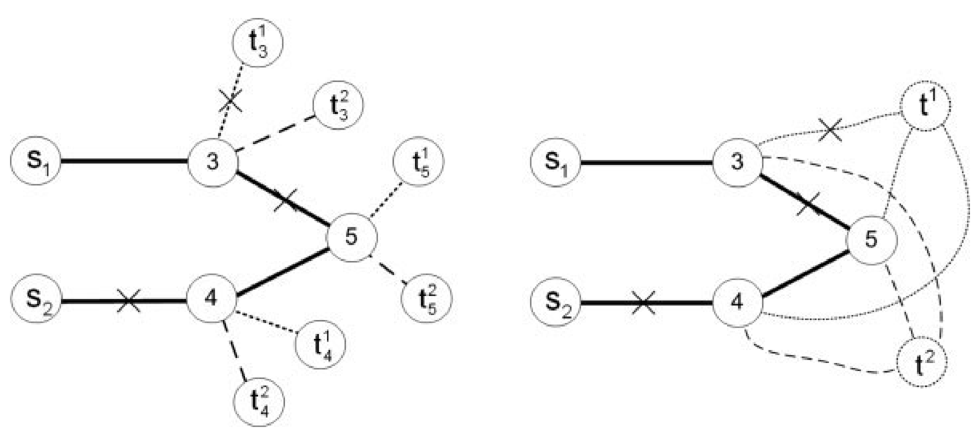

FIG. 3. (a) The removed edges in the multicut problem corresponding to the 2-server solution $F = \{(s_2, 4), (3, 5)\}$. (b) The multicut solution becomes infeasible if the t_i^l are merged into t^l.

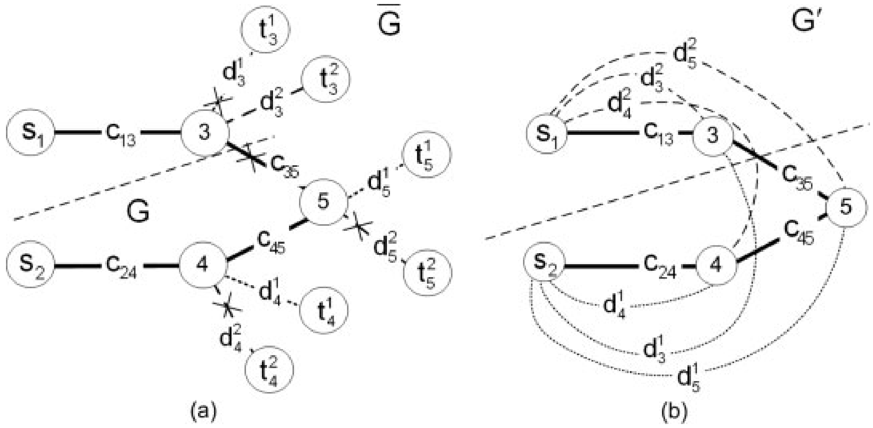

FIG. 4. (a) Edges $\{(3,5),(3,t_3^1),(5,t_5^2),(4,t_4^2)\}$ of the multicut problem corresponding to the $s_1 - s_2$ cut $(\{s_1,3\};\{s_2,4,5\})$ in the graph from Figure 1a. (b) Corresponding edges $\{(3,5),(3,s_2),(5,s_1),(4,s_1)\}$ of the equivalent min-cut problem.

(6)–(9).) However, if the t_i^l are merged into a single node t^l for $l = 1, 2$, then s_1 and 5 are connected to each other as in Figure 3b. Thus, the $O(kn)$ t-nodes are essential in the transformation.

However, there are some special cases in which a simplification is possible. First, if $k = 1$, it is easy to see that merging t-nodes into a single node maintains the validity of the transformation. Then the problem becomes the multicut problem with a single pair, which is, in turn, the well-known min-cut problem. Hence, we recapture the polynomiality of the single-server case.

Corollary 2.9. *The single-server disconnection problem is polynomially solvable via a min-cut problem.*

Another case is the 2-server disconnection problem which has an optimal solution inducing a cut $(N_1; N\backslash N_1)$ separating two servers: $s_1 \in N_1$ and $s_2 \in N\backslash N_1$. There are several ways to prove the polynomiality of this case. First, similar to the arguments for the single-server case in Section 1, we can identify the modularity of the disconnected utility $\mathrm{dscn}(F)$ with respect to the cuts separating the two servers. Second, we can show that the special case of the 2-server disconnection problem can be formulated as an unconstrained $0-1$ quadratic maximization with a nonnegative off-diagonal matrix which is known to be polynomially solvable [10]. In this paper, we establish the polynomiality by showing that the special condition reduces the multicut formulation of the problem further to a min-cut problem.

Corollary 2.10. *Suppose that there is an optimal solution of the* 2*-server disconnection problem that induces a cut* $(N_1; N\backslash N_1)$ *with* $s_1 \in N_1$ *and* $s_2 \in N\backslash N_1$*. Then the problem is polynomially solvable via the min-cut problem.*

Proof. Given such a 2-server disconnection problem, construct a min-cut problem as follows. In the graph of the 2-server disconnection problem (with all of the utility vectors removed), connect every node i to s_2 (instead of creating a new node t_i^1 as in the proof of Proposition 2.8) by an edge with cost d_i^1. Similarly, connect every node i to s_1 by an edge with cost d_i^2. For instance, if we apply the procedure to the original graph G of the 2-server disconnection problem from Figure 1a, we obtain a min-cut problem G' as shown in Figure 4b. The graph $\bar{G}$ in Figure 4a, on the other hand, illustrates the multicut reformulation based on the integer programming formulation (6)–(9) and Proposition 2.8.

Therefore, to prove the corollary, it suffices to show that there is an objective value preserving the one-to-one correspondence between the multicuts in $\bar{G}$ induced by the 2-server disconnection problem solutions disconnecting s_1 from s_2, and the $s_1 - s_2$ cuts of G'. To do so, consider an $s_1 - s_2$ cut $(N_1; N \setminus N_1)$ and the cut edges F of G. The corresponding edges of $\bar{G}$ are the edges from F and the edges (i, t_i^l) for all is and ls such that i remains connected to s_l after the deletion of F due to (7). For example, in Figure 4a, the edges of the $s_1 - s_2$ cut $(\{s_1, 3\}; \{s_2, 4, 5\})$ of G determined by the solution $F = \{(3, 5)\}$ correspond to the multicut edges $(3, 5)$, $(3, t_3^1)$, $(4, t_4^2)$, $(5, t_5^2)$, which are marked "×". We will show that these precisely correspond to the edges of the $s_1 - s_2$ cut $(N_1; N \setminus N_1)$ of G'. But this is clear from the construction of the min-cut problem. If node i is on the same side as s_1, (i, t_i^1) should be a multicut edge. Through the construction, an edge (i, t_i^1) of the multicut problem is replaced by (i, s_2) in the min-cut problem. Since $i \in N_1$, (i, s_2) is in the $s_1 - s_2$ cut $(N_1; N \setminus N_1)$ of G'. For instance, in Figure 4, $(3, t_3^1)$ corresponds to $(3, s_2)$. Similarly, if node i is on the s_2-side, the multicut edge (i, t_i^2) corresponds to the edge (i, s_1) of the $s_1 - s_2$ cut $(N_1; N \setminus N_1)$ of G'. For instance, $(4, t_4^2)$ corresponds to $(4, s_1)$ in Figure 4. Clearly, this correspondence preserves the objective value, as the corresponding edges have the same cost.

In light of the "extended path interpretation" of (7), the proof of the converse correspondence is almost identical and hence will be omitted here. ■

3. APPROXIMATION OF *K*-SERVER DISCONNECTION PROBLEM FOR FIXED *K*

In this section, we show that for a fixed k, the k-server disconnection problem is approximable within a factor of $\left(\frac{1}{2}+\frac{1}{2^{k+1}-2}\right)$ of the optimum.

Given a solution to the k-server disconnection problem, we denote by N_l the set of nodes connected to server $l \in K$. Then, it is easy to see that for any $l \neq l'$, the two sets N_l and $N_{l'}$ are either identical or mutually exclusive: $N_l = N_{l'}$ if l and l' are connected; $N_l \cap N_{l'} = \emptyset$, otherwise. Hence, if we denote by N_0 the set of nodes disconnected from all of the servers, the family $\mathcal{N} = \{N_0, N_1, \ldots, N_k\}$ is a partition of N. If $N_{l_1}, N_{l_2}, \ldots, N_{l_p}$ are all of the distinct sets of $\mathcal{N}$ we will also write

$$\mathcal{N} \equiv (N_{l_1}; N_{l_2}; \ldots; N_{l_p}) \equiv (N_{l_q} : q = 1, 2, \ldots, p),$$

and $\mathcal{N}$ will be referred to as a *p set partition solution*, or simply, a *p-partition*. Also its objective value will be denoted by $z(\mathcal{N})$. If F is the set of edges determining $\mathcal{N} = (N_0; N_1; \ldots; N_k)$, then the objective value is written as

$$z(\mathcal{N}) = \sum_{l\in K}\sum_{i\in N_l}\left(d_i(K) - d_i^l\right) + \sum_{i\in N_0} d_i(K) - \sum_{e\in F} c_e, \quad (10)$$

where, for each $i \in N$ and $Q \subseteq K$, $d_i(Q) = \sum_{l\in Q} d_i^l$. For $p = 1, 2, \ldots, k+1$, we denote by $\mathcal{N}_p^*$ and z_p^*, respectively, an optimal solution over the p-partitions and its objective value. Then, obviously, the optimum value z^* of the problem is given by

$$z^* = \max\left\{z_p^* : p = 1, 2, \ldots, k+1\right\}. \quad (11)$$

(Trivially, $z_1^* = 0$.)

The rest of this section is organized as follows. In Section 3.1, we prove that for a fixed k, z_2^* can be computed polynomially using $O(2^k)$ applications of the min-cut problem. Then, in Section 3.2, using counting arguments, we establish the inequality

$$z_2^* \geq \left(\frac{1}{2} + \frac{1}{2^{k+1}-2}\right) z_p^*, \text{ for } p = 3, 4, \ldots, k+1. \quad (12)$$

Thus, combining (11) and (12), we obtain the following theorem:

Theorem 3.1. *For any fixed k, the k-server disconnection problem is approximable within a factor* $\left(\frac{1}{2}+\frac{1}{2^{k+1}-2}\right)$ *of the optimum using* $O(2^k)$ *applications of a min-cut algorithm.*

Notice that if $z_2^* \leq 0$, (12) implies that $z_p^* \leq 0$ for $p = 2, 3, \ldots, k+1$ and hence the trivial solution $F = \emptyset$ corresponding to $z_1^* = 0$ is optimal. Conversely, if $z^* > 0$, again from (11) and (12) we have $z_2^* > 0$. Thus, the k-server disconnection problem has a positive optimal value if and only if $z_2^* > 0$.

3.1. *Polynomiality of z_2^* for Fixed k*

To demonstrate the polynomiality of z_2^* for a fixed k, consider the following problem.

Problem 3.2. *Given $Q \subseteq K$, compute an optimal 2-partition, namely, a partition $\mathcal{N} = (M; N\backslash M)$ with $Q \subseteq M$ and $Q^c \subseteq N\backslash M$ that maximizes $z(M; N \setminus M)$.*

First, notice that Problem 3.2 can be reduced to the 2-server disconnection problem. Merge all the servers of Q into a single server, say, s_Q, and those of Q^c into s_{Q^c}. If parallel edges occur, replace them with a single edge that is assigned the sum of the costs of the merged edges. Each node $i \in N$ is assigned two utility values $d_i(Q) \equiv \sum_{l\in Q} d_i^l$ and $d_i(Q^c) \equiv \sum_{l\in Q^c} d_i^l$. See for example Figure 5.

Then, it is easy to see that solving Problem 3.2 is equivalent to finding an optimal 2-partition separating s_Q and s_{Q^c} in the 2-server disconnection problem. Hence, by Corollary 2.10, Problem 3.2 is solvable in polynomial time.

The set Q in Problem 3.2 may be empty when we want an optimal partition $\mathcal{N} = (M; N\backslash M)$ such that M is the set of nodes disconnected from all of the servers. In this case, the transformation above will result in a single-server problem. Therefore, in this case, Corollary 2.9 implies the polynomiality of Problem 3.2.

The value z_2^* can be computed simply by solving Problem 3.2 for all $Q \subseteq K$ and taking the maximum of the optimal

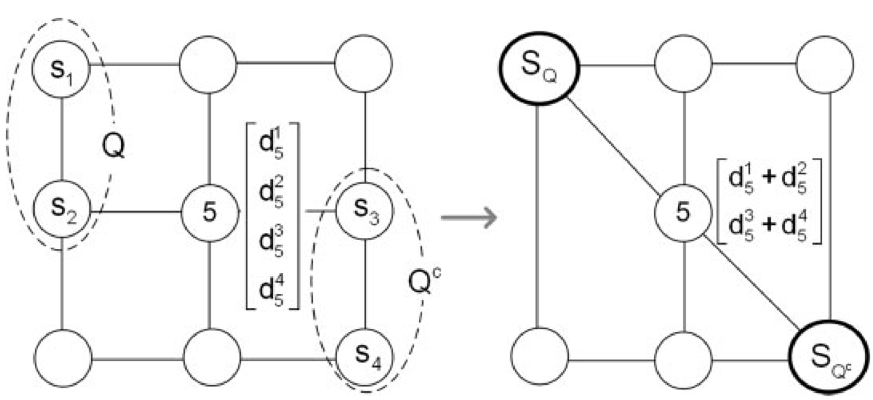

FIG. 5. Transformation to a 2-server problem to compute an optimal 2-partition with $Q = \{s_1, s_2\}$. (For simplicity, only the positive cost edges are illustrated; thus the graph is not complete.)

values for the Qs. It requires solving $O(2^k)$ min-cut problems whose sizes are essentially the same as that of the original problem.

Proposition 3.3. *z_2^* can be computed within the time for $O(2^k)$ applications of a min-cut algorithm.*

3.2. Approximation of z_p^*s with z_2^*

In Section 3.1, we saw that z_2^* is polynomially computable for a fixed k. As p increases, z_p^* becomes more difficult to compute. For instance, when $k = p = 3$, if $d_i^l = 0$ for $l = 1, 2, 3$ and for every $i \in N$, to compute z_3^* is equivalent to the minimum 3-terminal cut problem. Hence, even for a fixed k, $p = 2$ is the maximum for which z_p^* is polynomially computable.

This observation motivates us to consider z_2^* as an approximation of z_p^* for $p \geq 3$.

Proposition 3.4.

$$z_2^* \geq \left(\frac{1}{2} + \frac{1}{2^{k+1} - 2}\right) z_{k+1}^*.$$

Proof. Let $\mathcal{N}^*$ be a $(k+1)$-partition whose objective value is z_{k+1}^*. We write $\mathcal{N}^* = (N_0^*; N_1^*; \ldots; N_k^*)$, where N_l^*, $l \in K$, is the set of nodes connected to the server l, and N_0^* is the set of nodes disconnected from all of the servers. Accordingly, the set of edges F^* whose removal results in $\mathcal{N}^*$ can be also decomposed: for $l, l' \in K$ with $l < l'$, let

$$F_{ll'}^* = \{e = (i, j) : i \in N_l^*,\ j \in N_{l'}^*, i < j\}.$$

Then $F^* = F_0^* \cup F_1^* \cup \cdots \cup F_k^*$, where, for $l = 0, 1, \ldots, k$,

$$F_l^* = \bigcup_{l' < l} F_{l'l}^* \cup \bigcup_{l < l'} F_{ll'}^*.$$

In Figure 6a, the notation is illustrated for $k = 3$.

From now on, for notational convenience, we introduce a dummy server 0 connected to every node $i \in N$ via edges $(0, i)$ with zero cost. Every node $i \in N$ has utility $d_i^0 = 0$ for this dummy server. It is easy to confirm that the problem is equivalent to the original one. With this setting, we can consider N_0^* as the set of nodes connected to the server 0, and we can treat the sets $N_0^*, N_1^*, \ldots, N_k^*$ symmetrically. For instance, writing

$$K' := K \cup \{0\},$$

the objective value (10) for $\mathcal{N}^*$ is given as

$$z_{k+1}^* = z(\mathcal{N}^*) = \sum_{l \in K'} \sum_{i \in N_l^*} \left(d_i(K') - d_i^l\right) - \sum_{e \in F^*} c_e. \quad (13)$$

For simplicity, let us write

$$N_Q = \bigcup_{l \in Q} N_l^*, \quad \text{and} \quad F_Q = \bigcup_{l \in Q} F_l^*, \text{ for } Q \subseteq K'.$$

Given $Q \subseteq K'$, consider the 2-partition $(N_Q; N_{Q^c})$ obtained from the $(k+1)$-partition $\mathcal{N}^*$. See Figure 6b for the reduction of the 4-partition $\mathcal{N}$ in Figure 6a to the 2-partition $(N_Q; N_{Q^c})$ with $Q = \{0, 1\}$. Let Δ_Q be the decrease of the objective value due to this reduction. (Recall that G is complete and hence the subgraphs induced by N_Q and N_{Q^c} are connected.) Then Δ_Q is the sum of two values. The first one is the decrease due to the merging of the subpartition $(N_l^* : l \in Q)$ into the single set N_Q. This is exactly the objective value $z(N_l^* : l \in Q)$ of the partition $(N_l^* : l \in Q)$ of the $|Q|$-server disconnection problem defined on the subgraph of G induced by the node set N_Q. We have

$$z\left(N_l^* : l \in Q\right) = \sum_{l \in Q} \sum_{i \in N_l^*} \left(d_i(Q) - d_i^l\right) - \sum_{e \in F^* \setminus F_{Q^c}} c_e. \quad (14)$$

(Compare (13) and (14).) Similarly, the second value is the objective value $z(N_l^* : l \in Q^c)$ of the partition $(N_l^* : l \in Q^c)$ of the $|Q^c|$-server disconnection problem defined on the subgraph induced by the node set N_{Q^c}. To summarize,

$$\Delta_Q = z\left(N_l^* : l \in Q\right) + z\left(N_l^* : l \in Q^c\right) \quad (15)$$

$$= \left(\sum_{l \in Q} \sum_{i \in N_l^*} \left(d_i(Q) - d_i^l\right) - \sum_{e \in F^* \setminus F_{Q^c}} c_e\right) + \left(\sum_{l \in Q^c} \sum_{i \in N_l^*} \left(d_i(Q^c) - d_i^l\right) - \sum_{e \in F^* \setminus F_Q} c_e\right), \quad \text{and}$$

$$z(N_Q; N_{Q^c}) = z(\mathcal{N}^*) - \left(z\left(N_l^* : l \in Q\right) + z\left(N_l^* : l \in Q^c\right)\right). \quad (16)$$

See Figure 7 for an illustration of (16). Now, we take the summation of (16) over all Qs such that $|Q| = q$ for fixed $1 \leq q \leq k$ to obtain

$$\sum_{Q \subseteq K' : |Q| = q} z(N_Q; N_{Q^c}) = \binom{k+1}{q} z_{k+1}^* - \sum_{Q \subseteq K' : |Q| = q} \left(z\left(N_l^* : l \in Q\right) + z\left(N_l^* : l \in Q^c\right)\right).$$

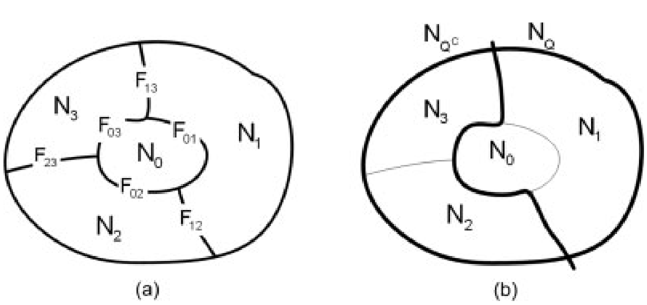

FIG. 6. (a) N_l's and $F_{ll'}$'s of a 4-partition. (b) Reduction of a 4-partition $\mathcal{N}$ to a 2-partition $(N_Q; N_{Q^c})$ with $Q = \{0, 1\}$.

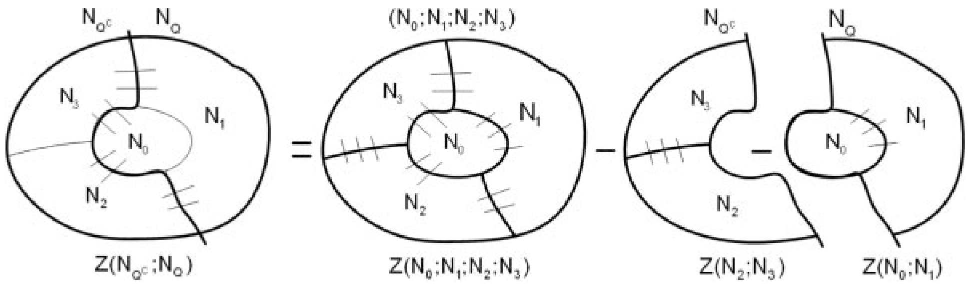

FIG. 7. Decomposition of $z(\mathcal{N}^*)$ into $z(N_Q;N_{Q^c})$, $z(N_l^*: l \in Q)$, and $z(N_l^*: l \in Q^c)$.

Therefore, by the definition of z_2^*, we get

$$\binom{k+1}{q} z_2^* \geq \binom{k+1}{q} z_{k+1}^* - \sum_{Q \subseteq K':|Q|=q} \left(z\left(N_l^*: l \in Q\right) + z\left(N_l^*: l \in Q^c\right)\right). \quad (17)$$

Taking the summation of (17) over $q = 1, 2, \ldots, k$, we have

$$\sum_{q=1}^{k} \binom{k+1}{q} z_2^* \geq \sum_{q=1}^{k} \binom{k+1}{q} z_{k+1}^* - 2\sum_{q=1}^{k} \sum_{Q \subseteq K':|Q|=q} z\left(N_l^*: l \in Q\right). \quad (18)$$

The last term is from the symmetry of (15) with respect to Q and Q^c. ∎

Lemma 3.5.

$$\sum_{Q \subseteq K':|Q|=q} z\left(N_l^*: l \in Q\right) = \begin{cases} 0, & \text{if } q = 1, \\ \binom{k-1}{q-2} z_{k+1}^*, & \text{if } q \geq 2. \end{cases}$$

Proof of Lemma 3.5. The case for $q = 1$ is trivial. Hence, suppose $|Q| = q \geq 2$ and compare $z(\mathcal{N}^*)$ of (13) and $z(N_l: l \in Q)$ of (14). (See for instance, $z(N_0;N_1;N_2;N_3)$ and $z(N_0;N_1)$ in Fig. 7.)

Consider any c_e for $e = (i,j) \in F^*$ and suppose $i \in N_s$ and $j \in N_t$. It appears also in $z(N_l: l \in Q)$ if and only if $s,\ t \in Q$. Clearly, there are $\binom{k-1}{q-2}$ combinations of such Qs. Similarly, for each $i \in N_s$ $(s \neq t)$, the term d_i^t in $z(\mathcal{N}^*)$ appears also in $z(N_l: l \in Q)$ if and only if $s, t \in Q$. Again, there are $\binom{k-1}{q-2}$ combinations of such Qs.

From these observations, we see that each term of $z(\mathcal{N}^*)$ appears in $z(N_l: l \in Q)$ exactly for $\binom{k-1}{q-2}$ distinct Qs. Therefore, the lemma follows. ∎

Because of Lemma 3.5, (18) implies

$$\sum_{q=1}^{k} \binom{k+1}{q} z_2^* \geq \sum_{q=1}^{k} \binom{k+1}{q} z_{k+1}^* - 2\sum_{q=2}^{k} \binom{k-1}{q-2} z_{k+1}^*. \quad (19)$$

From (19),

$$(2^{k+1}-2)z_2^* \geq ((2^{k+1}-2) - 2(2^{k-1}-1))z_{k+1}^*, \quad \text{or}$$

$$z_2^* \geq \left(\frac{1}{2} + \frac{1}{2^{k+1}-2}\right) z_{k+1}^*.$$

This completes the proof of Proposition 3.4. ∎

It is not difficult to see that the proof of Proposition 3.4 easily extends to $p \leq k+1$, thus giving $z_2^* \geq \left(\frac{1}{2} + \frac{1}{2^p-2}\right)z_p^*$. Also notice that the proof does not depend on the sign of z_p^* for any $p = 2, 3, \ldots, k+1$. Therefore we have the following:

Corollary 3.6. *If $z_2^* \leq 0$, then $z_p^* \leq 0$ for $p = 3, 4, \ldots, k+1$ and the trivial solution $F = \emptyset$ is optimal. If, by contrast, $z_2^* > 0$, then*

$$z_2^* \geq \left(\frac{1}{2} + \frac{1}{2^{k+1}-2}\right) z_p^*, \quad \text{for } p = 3, 4, \ldots, k+1.$$

Now, we formally describe the approximation algorithm.

Algorithm 3.7. ***Approximation Algorithm***

STEP 1. Compute z_2^*.

STEP 2. If $z_2^* > 0$, return the corresponding 2-partition.

STEP 3. Otherwise, return $F = \emptyset$ as an optimal solution.

The validity of Algorithm 3.7 immediately follows from Corollary 3.6. Combining this with Proposition 3.3, we obtain Theorem 3.1.

4. FURTHER RESEARCH

We proposed a network attack model and established its inapproximability. We also developed an approximation algorithm with a constant approximation guarantee for the problem involving only a few servers. Improving the approximation factor seems a rather challenging problem requiring a different approach as only z_2^* among the z_p^*s is computable in polynomial time (even for fixed $k \geq 3$).

It will also be interesting to explore the approximability of a directed version of the problem modeling cases when the communication between node pairs is not symmetric.

Acknowledgments

The authors are indebted to Young-Soo Myung for the fruitful discussions that eventually motivated the authors to perform a tighter analysis of the approximation algorithm. They also thank two anonymous referees for their helpful comments and the editor-in-chief, Douglas Shier, for his careful editorial corrections.

REFERENCES

[1] G. Calinescu, H. Karloff, and Y. Rabani, An improved approximation algorithm for muliway cut, J Comp Syst Sci 60 (2000), 564–574.

[2] S. Chopra and M.R. Rao, On the multiway cut polyhedron, Networks 21 (1991), 51–89.

[3] W.H. Cunningham, Optimal attack and reinforcement of a network, J ACM 32 (1985), 549–561.

[4] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, and M. Yannakakis, The complexity of multiterminal cuts, SIAM J Comput 23 (1994), 864–894.

[5] R. Engelberg, J. Konemann, S. Leonardi, and J. Naor, Cut problems in graphs with a budget constraint, Proceedings of the 7th Latin American Symposium on Theoretical Informatics, Valdivia, Chile, 2006, pp. 435–446.

[6] N. Garg, V.V. Vazirani, and M. Yannakakis, Approximating max-flow min-(multi)cut theorems and their applications, SIAM J Comput 25 (1996), 235–251.

[7] S.-P. Hong and B.-C. Choi, Polynomiality of sparsest cuts with fixed number of sources, Oper Res Lett, Article in Press.

[8] D.R. Karger, P. Klein, C. Stein, M. Thorup, and N.E. Young, Rounding algorithms for a geometric embedding of minimum multiway cut, Math Oper Res 29 (2004), 436–461.

[9] C. Martel, G. Nuckolls, D. Sniegowski, and M. Haungs, Computing the disconnectivity of a graph, Technical Report CSE-2002-38, University of California, Davis, 2002. Available at http://www.cs.ucdavis.edu/research/tech-reports/.

[10] J.C. Picard and H.D. Ratliff, Minimum cuts and related problems, Networks 5 (1974), 357–370.

[11] A. Schrijver, A combinatorial algorithm minimizing submodular functions in strongly polynomial time, Journal of Combinatorial Theory Series B 80 (1999), 346–355.