Algorithmic Operations Research

On-line Network Synthesis

S. N. Kabadi and D. Du

Volume 2, Number 1, Summer 2007

URI: https://id.erudit.org/iderudit/aor2_1art07

See table of contents

Publisher(s)

Preeminent Academic Facets Inc.

ISSN

1718-3235 (digital)

Explore this journal

Cite this article

Kabadi, S. N. & Du, D. (2007). On-line Network Synthesis. *Algorithmic Operations Research*, *2*(1), 65–74.

Article abstract

We consider on-line network synthesis problems. Let N = {1, , n} be a set of n sites. Traffic flow requirements between pairs of sites are revealed one by one. Whenever a new request rij = rji (i < j) between sites i and j is revealed, an on-line algorithm must install the additional necessary capacity without decreasing the existing network capacity such that all the traffic requirements are met. The objective is to minimize the total capacity installed by the algorithm. The performance of an on-line algorithm is measured by the competitive ratio, defined to be the worst-case ratio between the total capacity by the on-line algorithm and the total optimal (off-line) capacity assuming we have priori information on all the requirements initially. We distinguish between two on-line versions of the problem depending on whether the entire set of sites is known a prior or not. For the first version where the entire set of site is unknown, we present a best possible algorithm along with a matching lower bound. For the second version where the entire set of sites is known a priori, we present a best possible algorithm for n $\leqslant$ 6.

Algorithmic Operations Research Vol.2 (2007) 65–74

On-line Network Synthesis

S. N. Kabadi and D. Du

Faculty of Business Administration, University of New Brunswick, Fredericton, New Brunswick, Canada E3B 5A3

Abstract

We consider on-line network synthesis problems. Let $N = \{1, \cdots, n\}$ be a set of n sites. Traffic flow requirements between pairs of sites are revealed one by one. Whenever a new request $r_{ij} = r_{ji}$ $(i < j)$ between sites i and j is revealed, an on-line algorithm must install the additional necessary capacity without decreasing the existing network capacity such that all the traffic requirements are met. The objective is to minimize the total capacity installed by the algorithm. The performance of an on-line algorithm is measured by the competitive ratio, defined to be the worst-case ratio between the total capacity by the on-line algorithm and the total optimal (off-line) capacity assuming we have priori information on all the requirements initially. We distinguish between two on-line versions of the problem depending on whether the entire set of sites is known a prior or not. For the first version where the entire set of site is unknown, we present a best possible algorithm along with a matching lower bound. For the second version where the entire set of sites is known a priori, we present a best possible algorithm for $n \leq 6$.

Key words: On-line algorithm; Approximation algorithm; Network synthesis; Competitive analysis

1. Introduction

In the traditional NETWORK SYNTHESIS PROBLEM (NSP), one is given an $n \times n$, symmetric, non-negative matrix R (with $r_{ii} = 0\ \forall i = 1, 2, \ldots, n$), of minimum flow requirements between all pairs of distinct sites in the set $N = \{1, 2, \ldots, n\}$. The goal is to construct an undirected network $G = [N, E, c]$ on site set N, with edge set E and non-negative, real-valued edge capacities $\{c(e) : e \in E\}$, such that (i) all the minimum flow requirements are met one at a time, (that is, for any $i, j \in N,\ i \neq j$, the maximum flow value in G between i and j is at least r_{ij}), and (ii) $\sum_{e \in E} c(e)$ is minimum. Without loss of generality, we assume the constructed network is simple, i.e., no loops and parallel edges. Otherwise we can delete any loop and merge any parallel edges without affecting the results in this paper.

Gomory and Hu [10] and Mayeda [13] present efficient combinatorial algorithms for the problem NSP. Gomory-Hu algorithm is strongly polynomial and produces an optimal network with $O(n)$ edges. Also, when all the elements of the matrix R are integers, the edge capacities in the final network are multiples of half. Alternate, combinatorial algorithms for the problem are presented in [9] and [20]. The algorithm in [9] is a modification of the Gomory-Hu algorithm [10]. It has a time complexity $O(n^2)$ and produces a network with at most $2n$ edges.

Very often, in practical network designing, the source and destination and the flow requirements only become known and/or are updated one by one in sequence and after all the previous requirements in the sequence have been served by installing necessary capacity. Any installed capacity cannot be decreased, but can only be increased in future.

In this paper, we consider two on-line versions of the network synthesis problem. The quality of an on-line algorithm will be measured by its *competitive ratio*, which is defined to be the worst-case ratio between the total capacity of the on-line algorithm and the corresponding optimal (off-line) total capacity over all instances.

Network optimization problems, such as matching, assignment and transportation problems [12,11,15,17,18], facility location [14], network design [2], Steiner tree [3,7], set cover [1], traveling salesman [4–6], etc., in an on-line setting have been actively investigated in the literature. The reader is referred to the survey paper by Kalyanasundaram and Pruhs [16] for further information and references up to 1996.

After defining the problems formally in Section 2., we present the main results and analysis for both versions of the on-line network synthesis problem in Sections 3.

⋆ Work supported by NSERC discovery grants awarded to D. Du and S N Kabadi

Email: S. N. Kabadi [kabadi@unb.ca], D. Du [ddu@unb.ca].

and 4., respectively.

2. Problem Description and Preliminaries

In this section, we formally define our problems. For any positive integer n, let $N = \{1, 2, \ldots, n\}$. Given an $n \times n$ symmetric, non-negative matrix R (with $r_{i,i} = 0$ for all $i \in N$), let us define the *potential* of site i to be $\pi_i = \max_{j \in N} r_{ij}$, for all $i \in N$. It is easy to see that the largest two potentials must be equal. For any non-empty $X \subset N$, and any non-empty $A \subseteq X$ and $B \subseteq \overline{X}$, we call the cut $(X, \overline{X})$ an A-B cut. In particular for any $i \in X$ and $j \in \overline{X}$, we call $(X, \overline{X})$ an i-j cut. We denote by $c(X) = \sum_{(i,j) \in (X, \overline{X})} c_{ij}$ capacity of the cut $(X, \overline{X})$.

We will need the following important existing results for the (off-line) NSP.

Proposition 1 *[10,13]: The optimal objective function value of the NSP, with $n \times n$ symmetric, non-negative matrix R as input, is $\frac{1}{2} \sum_{i \in N} \pi_i$.*

Proposition 2 *Suppose the potentials $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ are sorted such that $\pi_1 = \pi_2 \geq \cdots \geq \pi_n$. If we can send π_u units of flow from site 1 to site u in G for any $u \in N - \{1, n\}$, then we can send $\min\{\pi_i, \pi_j\}$ units of flow from site i to site j in G for any $i, j \in N$, $i \neq j$.*

Proof. This follows from the well-known *triple inequality* [8,10]: For any $i, j, k \in N$, the minimum capacity of i-k-cut $\geq \min\{$minimum capacity of i-j cut, minimum capacity of j-k cut$\}$. ■

Proposition 3 *Let $G = [N, E, c]$ be an edge-capacitated undirected network and $N_0 = \{i_1, \cdots, i_p\}$ be a subset of $N - \{1\}$. For each $\ell \in \{1, \cdots, p\}$, let $F(i_\ell)$ be the units of flow from site i_ℓ to site 1. Then we can simultaneously send $F(i_\ell)$ units of flow from site i_ℓ to site 1 for all $\ell \in \{1, \cdots, p\}$ if and only if*

(i) *for any 1-i_ℓ cut, the cut capacity is at least $F(i_\ell)$ for any $\ell \in \{1, \cdots, p\}$; and*

(ii) *for any 1-N_0 cut, the cut capacity is at least $\sum_{\ell=1}^{p} F(i_\ell)$.*

Proof. This follows from the well-known max-flow min-cut Theorem [8]. ■

We consider two on-line versions of the network synthesis problem.

Version 1: In this version, at any point in time, a certain set $\{r_{ij} : (i,j) \in S\}$ of requirements between some set S of pairs of sites, and through it the set $\overline{N} = \{i : (i,j) \in S \text{ for some } j\}$ of sites are known to us. The on-line algorithm is required to have designed a network $\overline{G}$ on site set $\overline{N}$ that meets the revealed set of requirements one at a time. The next piece of information revealed is some requirement r_{xy}, where if some requirement between sites x and y was revealed before then the new value is greater than the previous and replaces the previous; else, r_{xy} is a new revealed requirement and in that case, the new revealed set of sites is $N = \overline{N} \cup \{x, y\}$. The on-line algorithm is required to update $\overline{G}$ to a network G on site set N (that includes at least two more sites) such that none of the previous edges capacities in $\overline{G}$ are decreased and the new requirement r_{xy} is also satisfied.

Version 2: In this version, the entire set N of potential sites is known a priori, but the requirements between pairs of sites are revealed or updated one-by-one in a sequence. At any point in time, the network designed by the on-line algorithm contains the entire site set N and satisfies the revealed set of requirements one at a time. Upon revelation of a new requirement or update of a previous revealed requirement, the on-line algorithm must update the current network by increasing some of the edge capacities so that the new requirement is also satisfied.

The difference between these two versions is that in Version 1, we can only use the currently revealed set of sites to satisfy the requirements; in Version 2, however, we can benefit by taking advantage of all the sites which are given in the first place. Obviously at each state there is more information available in Version 2 than in Version 1, and hence the competitive ratio for Version 2 cannot exceed that for Version 1. We actually show in this paper that the competitive ratio for Version 2 is *strictly* smaller than that for Version 1.

Define two parameters

$$\alpha_n = 2 - \frac{2}{n};$$

$$\beta_n = 2 - \frac{2^{k^*+1}}{n + k^* 2^{k^*}}.$$

In the above $k^* = \lfloor \log_2 n \rfloor$. It is easy to show $\beta_n < \alpha_n$ for any fixed n.

Some values of the α_n and β_n are shown below.

n	3	4	5	6	$\cdots$	∞
α_n	$\frac{4}{3}$	$\frac{3}{2}$	$\frac{8}{5}$	$\frac{5}{3}$	$\cdots$	2
β_n	$\frac{6}{5}$	$\frac{4}{3}$	$\frac{18}{13}$	$\frac{10}{7}$	$\cdots$	2

The main result of this paper is summarized in the following, which follows from Lemmas 5, 6, 7, and 9.

Theorem 4 *For any number of sites n,*

(1) the best possible competitive ratio is α_n for Version 1;
(2) the best possible competitive ratio is β_n for Version 2 when $n \leq 6$.

An obvious open question therefore is to solve Version 2 for $n > 6$. (Our algorithm for Version 2 (Algorithm POTENTIAL) can actually be shown to produce an upper bound of β_n for any $n \leq 11$. But the arguments are more complex and we omit the details here. Version 2 is thus open for $n > 11$.)

3. A Best Possible Algorithm for Version 1

In this section, we establish the lower and upper bounds for Version 1, respectively.

3.1. *Lower Bound*

Lemma 5 *No on-line algorithm for Version 1 of the network synthesis problem can have a competitive ratio less than α_n, for any number of revealed sites n.*

Proof. Suppose we have an on-line algorithm which has a competitive ratio less than α_n for some value of n. Consider a problem instance where $(n-1)$ r_{ij}'s each of value 1 are revealed in the following order: $r_{12}, r_{23}, \cdots, r_{n-1,n}$. Let c_{ij} $(1 \leq i < j \leq n)$ be the capacity assigned on each edge (i,j) by this algorithm after $n-1$ steps. Then

$$\sum_{j=1}^{k-1} c_{jk} \geq r_{k-1,k} = 1 \ \forall k = 2, \cdots, n. \qquad (1)$$

Note that the total sum of capacities assigned by this algorithm is $\sum_{1 \leq i < j < n} c_{ij}$. So, by (1), we have $\sum_{1 \leq i < j < n} c_{ij} = \sum_{k=2}^{n} \sum_{j=1}^{k-1} c_{jk} \geq n-1$. The optimal value is $n/2$ by Lemma 1. By comparing these two, we get $\frac{\sum_{1 \leq i < j < n} c_{ij}}{n/2} \geq \frac{n-1}{n/2} = \alpha_n$, a contradiction. ■

3.2. *Upper Bound*

We shall now present an algorithm that achieves the lower bound claimed above, and hence it is a best possible one for Version 1.

We introduce some notations first. Let $\widetilde{N} = \{1, \cdots, n-1\}$ be the currently revealed site set with requirement matrix $R \in R_+^{(n-1)\times(n-1)}$, where $r_{ii} = 0$ for any $i \in \widetilde{N}$, and $r_{ij} = 0$ for all $i, j \in \widetilde{N}$ such that the requirement between i and j is not revealed. Let $\pi_i = \max_{j \in \widetilde{N}} r_{ij}$ for each site $i \in \widetilde{N}$. Let $\widetilde{G} = [\widetilde{N}, \widetilde{E}, \widetilde{c}]$ be the network designed by our on-line algorithm that meets the currently revealed requirements.

Suppose r_{xy} is the next revealed/updated requirement. Then $N = \widetilde{N} \cup \{x, y\}$ is the updated set of revealed sites. Let $G = [N, E, c]$ be obtained from $\widetilde{G}$ by adding isolated sites $\{x, y\} - \widetilde{N}$ to it. For each $i \in \{x, y\} - \widetilde{N}$, set $\pi_i = 0$.

Algorithm TRIANGULAR: Sort the set $N = \{1, \cdots, n\}$ of revealed sites in a non-increasing potential order, i.e., $\pi_1 \geq \cdots \geq \pi_n$. Without loss of generality, let $x > y$. Let $\pi'_x = \max\{\pi_x, r_{xy}\} = \pi_x + \delta_x$; and $\pi'_y = \max\{\pi_y, r_{xy}\} = \pi_y + \delta_y$. If $y = 1$ then increase the edge capacity c_{x1} by δ_x, else let Δ_{xy1} denote the set of three edges (x,y),$(x,1)$ and $(y,1)$. For any $(i,j) \in \Delta_{xy1}$, increase the edge capacity c_{ij} by θ_{ij} in the following way:

- If $\pi'_y < \pi'_1 = \pi_1$, then

$$\theta_{xy} = \frac{1}{2}\delta_y; \qquad (2)$$

$$\theta_{x1} = \delta_x - \frac{1}{2}\delta_y; \qquad (3)$$

$$\theta_{y1} = \frac{1}{2}\delta_y. \qquad (4)$$

- Othewise, if $\pi'_y \geq \pi'_1 = \pi_1$, then

$$\theta_{xy} = \delta_x - \frac{1}{2}(\pi_1 - \pi_y); \qquad (5)$$

$$\theta_{x1} = \frac{1}{2}(\pi_1 - \pi_y); \qquad (6)$$

$$\theta_{y1} = \frac{1}{2}(\pi_1 - \pi_y). \qquad (7)$$

Lemma 6 *Let $G' = [N', E', c']$ be the current network, and $\pi_1 = \pi_2 \geq \cdots \geq \pi_{n-1} \geq \pi_n$ be the current potentials. Algorithm* TRIANGULAR *maintains the following conditions at any state for an appropriate ordering of sites in non-increasing values of potentials.*

(1) For any $i \in N - \{1\}$, at least π_i units of flow can be sent from i to 1 in G.
(2) $\sum_{e \in E} c(e) \leq \sum_{i \in N - \{1\}} \pi_i$. Hence,

$$\frac{\sum_{e \in E} c(e)}{\frac{1}{2}\sum_{i=1}^{n} \pi_i} \leq \alpha_n.$$

Proof. The result is obviously correct when there are only two sites revealed. Suppose the result is currently correct. Thus we have a network $\widetilde{G} = [\widetilde{N}, \widetilde{E}, \widetilde{c}]$ on currently revealed set $\widetilde{N}$ of sites that for some site order such that $\pi_1 = \pi_2 \geq \cdots \geq \pi_{|\widetilde{N}|}$ (where the π's are the

current site potentials) satisfies conditions 1-2 above. We want to show it is still true after the processing of the next requirement.

Let G be obtained by adding to $\widetilde{G}$ isolated sites in $\{x,y\}-\widetilde{N}$. For each site $i \in \{x,y\}-\widetilde{N}$, set $\pi_i = 0$ and arrange such sites last in the ordering. For convenience, let us assume that with this site ordering, r_{xy} is the next requirement revealed/updated with $x > y$. Denote $N = \widetilde{N} \cup \{x,y\}$.

Then, the new potentials $\pi'_i = \pi_i \ \forall i \in N - \{x,y\}$; $\pi'_x = \max\{\pi_x, r_{xy}\} = \pi_x + \delta_x$; and $\pi'_y = \max\{\pi_y, r_{xy}\} = \pi_y + \delta_y$. Evidently $\pi'_x = \max\{\pi_x, r_{xy}\} \leq \max\{\pi_y, r_{xy}\} = \pi'_y$ and $\delta_x = \max\{0, r_{xy} - \pi_x\} \geq \max\{0, r_{xy} - \pi_y\} = \delta_y$ since $\pi_x \leq \pi_y$.

Obviously there is nothing to prove if $\delta_x = 0$ (and therefore $\delta_y = 0$ also), since the network G satisfies all the requirements. So we assume $\delta_x > 0$ in the rest of the proof.

We consider two cases.

Case 1 *$y = 1$; then the only site in $N - \{1\}$ with changed potential is site x with $\pi'_x = \pi_x + \delta_x$.*

By inductive hypothesis, we can send π_x units of flow from x to 1 in G and using the additional capacity, we can send an additional δ_x units of flow. Thus condition 1 is satisfied. Also $\sum_{e\in E'} c'(e) = \sum_{e\in E} c(e) + \delta_x \leq \sum_{i\in N-\{1\}} \pi_i + \delta_x = \sum_{i\in N-\{1\}} \pi'_i$. Thus condition 2 is also satisfied.

Case 2 *$\pi'_y < \pi'_1 = \pi_1$; sort the sites in non-decreasing order of new potentials and keeping site 1 as the first site.*

To prove Condition 1 of the theorem, we only need to consider x and y since no other potentials have changed.

For x, at least π_x units of flow can be sent from x to 1 along G by induction hypothesis. Using the additional capacities assigned to edges in the set Δ_{xy1}, an extra δ_x units of flow can be sent from x to 1. Thus a total of at least $\pi_x + \delta_x = \pi'_x$ units of flow is guaranteed from x to 1 in G'.

Similarly, a total of at least $\pi_y + \delta_y = \pi'_y$ units of flow is guaranteed from y to 1 in G'.

Also,

$$\begin{aligned}\sum_{e\in E'} c'(e) &= \sum_{e\in E} c(e) + \theta_{xy} + \theta_{x1} + \theta_{y1} \\ &\leq \sum_{i\in N-\{1\}} \pi_i + \delta_x + \tfrac{1}{2}\delta_y \\ &\leq \sum_{i\in N-\{1\}} \pi_i + \delta_x + \delta_y \\ &= \sum_{i\in N-\{1\}} \pi'_i,\end{aligned}$$

Where the first inequality follows from the inductive hypothesis and (2)-(4). Hence Condition 2 of the lemma follows.

Case 3 *$y > 1$ and $\pi'_y \geq \pi'_1 = \pi_1$; obtain a new site ordering with non-decreasing values of potentials such that y now occupies the first position.*

To prove that Condition 1 is satisfied, consider any $i \in N - \{y\}$ and any cut $(S, \overline{S})$ in G' with $i \in S$ and $y \in \overline{S}$.

First suppose the previously ordered site 1 is in $\overline{S}$. Then from the max-flow min-cut Theorem [8] and the inductive hypothesis, the cut capacity of $(S, \overline{S})$ is at least π_i. If $i \neq x$, then the capacity of the cut in G' is at least $\pi_i = \pi'_i$. If, otherwise, $i = x$, then the sum of additional capacities, assigned to the edges of the set Δ_{xy1} that are in cut $(S, \overline{S})$ is δ_x; the total capacity of cut $(S, \overline{S})$ is therefore at least $\pi_x + \delta_x = \pi'_x$.

Next suppose the previously ordered site 1 is in S. Similarly the capacity of cut $(S, \overline{S})$ is at least π_y. If $i \neq x$, then the sum of additional capacities assigned to the edges of the set Δ_{xy1} that are in the cut is at least $(\pi_1 - \pi_y)$; the total capacity of cut $(S, \overline{S})$ is therefore at least $\pi_1 \geq \pi_i = \pi'_i$. If, otherwise, $i = x$ then the sum of additional capacities assigned to the edges of the set Δ_{xy1} in the cut is δ_x; the total capacity of cut $(S, \overline{S})$ is therefore at least $\pi_y + \delta_x \geq \pi'_x$.

Using the max-flow min-cut theorem [8], it now follows that for any $i \in N - \{y\}$ we can send π'_i units of flow from i to y in G'.

Also,

$$\begin{aligned}\sum_{e\in E'} c'(e) &= \sum_{e\in E} c(e) + \theta_{xy} + \theta_{x1} + \theta_{y1} \\ &\leq \sum_{i\in N-\{1\}} \pi_i + \delta_x + \tfrac{1}{2}(\pi_1 - \pi_y) \\ &\leq \sum_{i\in N-\{1\}} \pi_i + \delta_x + (\pi_1 - \pi_y) \\ &= \sum_{i\in N-\{y\}} \pi'_i,\end{aligned}$$

Where the first inequality follows from the inductive hypothesis and (5)-(7). Hence Condition 2 of the lemma follows. This proves the lemma. ■

4. A Best Possible Algorithm for Version 2 when $n \leq 6$

In this section, we consider Version 2 of the problem where n is known a priori. We first establish a lower bound for any n. Then we present an algorithm which matches this lower bound for $n \leq 6$.

4.1. *Lower Bound*

Lemma 7 *No on-line algorithm for Version 2 of the network synthesis problem can have a competitive ratio less than β_n, for any number of sites n.*

Proof. For any n, suppose, on the contrary, there exists an on-line algorithm A with a competitive ratio less than β_n. Consider the following sequence of k^*+1 unit requirements revealed one by one. Initially, at stage 1, algorithm A receives the first unit requirement $r_{12}=1$ and processes it by installing necessary edge capacities to meet this requirement; let $1-\eta$ be the capacity installed on edge $(1,2)$. Consider stage $k-1$ ($2 \le k \le k^*+1$) when $k-1$ unit requirements are revealed and processed by algorithm A. Let c^{k-1} be the installed edge capacity vector up to stage $k-1$. For any two disjoint site subsets X and Y of N, denote $C^{k-1}(X,Y)=\sum_{i\in X, j\in Y} c_{ij}^{k-1}$. By relabeling the sites if necessary, without loss of generality, we assume that site $k+1$ satisfies

$$C^{k-1}(\{k+1\},\{1,\cdots,k\}) = \min_{\ell=1,\cdots,n-k} C^{k-1}(\{k+\ell\},\{1,\cdots,k\}). \quad (8)$$

Then let the next incoming requirement be $r_{k,k+1}=1$. Note that $C^{k-1}(\{k+1\},\{1,\cdots,k\}) < 1$; for else, $\sum_{1\le i<j\le n} c_{ij}^{k-1} \ge n/2$, which implies $(\sum_{1\le i<j\le n} c_{ij}^{k-1})/(k/2) \ge n/k \ge \beta_n$, a contradiction.

We show the following fact, which implies the desired result.

$$\frac{k+1}{2}\beta_n > k+\frac{n-2^k}{n-2}\eta, \text{ for } k=1,\cdots,k^*+1. \quad (9)$$

Indeed, if (9) were correct, consider the two inequalities corresponding to $k=k^*$ and k^*+1:

$$\frac{k^*+1}{2}\beta_n > k^*+\frac{n-2^{k^*}}{n-2}\eta; \quad (10)$$

$$\frac{k^*+2}{2}\beta_n > k^*+1+\frac{n-2^{k^*+1}}{n-2}\eta. \quad (11)$$

Note that the coefficient of η in (10) is non-negative, and that in (11) is negative. If the coefficient of η in (10) is zero, then $n=2^{k^*}$ and $\beta_n=(2\ln n)/(1+\ln n)$. But inequality (10) reduces to $\beta_n > (2k^*)/(k^*+1)=(2\ln n)/(1+\ln n)=\beta_n$, a contradiction. If the coefficient of η in (10) is positive, then multiplying (10) and (11) by appropriate positive numbers and adding the two inequalities, we get

$$\beta_n > 2-\frac{2^{k^*+1}}{n+k^*2^{k^*}}=\beta_n,$$

an obvious contradiction, and therefore the lemma is proved.

We are left only to prove (9). Consider stage k. Denote

$$\begin{aligned}
a_0 &= \frac{k+1}{2}\beta_n;\\
a_\ell &= \ell+1-\eta+\frac{n-k-2^\ell+\ell}{n-k+\ell-1}C^{k-\ell}(\{1,\cdots,k-\ell+1\},\{k-\ell+2,\cdots,n\})+\sum_{j=2}^{k-\ell}C^{k-\ell}(\{1,\cdots,j\},\{j+1\}) \text{ for } \ell=1,\cdots,k-1.
\end{aligned}$$

We prove inductively that algorithm A satisfies the following conditions (12)-(13)

$$a_0 > a_1; \quad (12)$$

$$a_\ell \ge a_{\ell+1},\ \ell=1,\cdots,k-1. \quad (13)$$

Conditions (12)-(13) imply (9) by noting that

$$\begin{aligned}
a_0 &= (k+1)\beta_n/2;\\
a_{k-1} &= k-\eta+\frac{n-2^{k-1}-1}{n-2}C^1(\{1,2\},\{3,\cdots,n\})\\
&\ge k-\eta+\frac{n-2^{k-1}-1}{n-2}2\eta\\
&= k+\frac{n-2^k}{n-2}\eta.
\end{aligned}$$

Basis Step: For $\ell=0$,

$$\begin{aligned}
a_0 &= \frac{k+1}{2}\beta_n > \sum_{1\le i<j\le n} c_{ij}^k\\
&\ge c_{12}^{k-1}+\sum_{j=2}^{n-1}C^{k-1}(\{1,\cdots,j\},\{j+1\})+(1-C^{k-1}(N-\{k+1\},\{k+1\})\\
&= (c_{12}^{k-1}+1)+\sum_{j=2}^{k-1}C^{k-1}(\{1,\cdots,j\},\{j+1\})\\
&\quad +(\sum_{j=k}^{n-1}C^{k-1}(\{1,\cdots,j\},\{j+1\})\\
&\quad -C^{k-1}(N-\{k+1\},\{k+1\}))\\
&\ge 2-\eta+\sum_{j=2}^{k-1}C^{k-1}(\{1,\cdots,j\},\{j+1\})\\
&\quad +C^{k-1}(\{1,\cdots,k\},\{k+2,\cdots,n\})
\end{aligned}$$

$$\geq 2-\eta+\sum_{j=2}^{k-1} C^{k-1}(\{1,\cdots,j\},\{j+1\})+ \frac{n-k-1}{n-k} C^{k-1}(\{1,\cdots,k\},\{k+1,\cdots,n\}) = a_1,$$

where the first inequality follows from the assumption of algorithm A being less than β_n-competitive; the second inequality is true because at least $1-C^{k-1}(N-\{k+1\},\{k+1\}$ new capacity is needed to satisfy the requirement $r_{k,k+1}=1$; the third inequality follows from $c_{12}^{k-1}\geq 1-\eta$; the fourth inequality follows from assumption (8).

Inductive Step: Assuming (13) were true for $\ell-1$, we want to prove it is still true for ℓ.

$$\begin{aligned} a_\ell &= \ell+1-\eta+\frac{n-k-2^\ell+\ell}{n-k+\ell-1} C^{k-\ell}(\{1,\cdots, k-\ell+1\},\{k-\ell+2,\cdots,n\})+\sum_{j=2}^{k-\ell} C^{k-\ell}(\{1,\cdots,j\},\{j+1\}) \\ &= \ell+1-\eta+\frac{n-k-2^\ell+\ell}{n-k+\ell-1} C^{k-\ell}(\{1,\cdots, k-\ell\},\{k-\ell+2,\cdots,n\})+\sum_{j=2}^{k-\ell-1} C^{k-\ell}(\{1,\cdots,j\},\{j+1\})+\frac{n-k-2^\ell+\ell}{n-k+\ell-1} \\ &\quad C^{k-\ell}(\{k-\ell+1\},\{k-\ell+2,\cdots,n\}) \\ &\quad +C^{k-\ell}(\{1,\cdots,k-\ell\},\{k-\ell+1\}) \end{aligned}$$

Note that

$$\begin{aligned} &\frac{n-k-2^\ell+\ell}{n-k+\ell-1} C^{k-\ell}(\{k-\ell+1\},\{k-\ell+2,\cdots,n\})+C^{k-\ell}(\{1,\cdots,k-\ell\},\{k-\ell+1\}) \\ &= \frac{n-k-2^\ell+\ell}{n-k+\ell-1}(C^{k-\ell}(\{k-\ell+1\},N-\{k-\ell+1\}))+\left(1-\frac{n-k-2^\ell+\ell}{n-k+\ell-1}\right) C^{k-\ell}(\{1,\cdots,k-\ell\},\{k-\ell+1\}) \\ &\geq \frac{n-k-2^\ell+\ell}{n-k+\ell-1}+\left(\frac{2^\ell-1}{n-k+\ell-1}\right) C^{k-\ell}(\{1,\cdots,k-\ell\},\{k-\ell+1\}) \\ &\geq \frac{n-k-2^\ell+\ell}{n-k+\ell-1}+\left(\frac{2^\ell-1}{n-k+\ell-1}\right)(1-C^{k-\ell}(\{1,\cdots,k-\ell\},\{k-\ell+2,\cdots,n\})) \\ &= 1-\left(\frac{2^\ell-1}{n-k+\ell-1}\right) C^{k-\ell}(\{1,\cdots,k-\ell\},\{k-\ell+2,\cdots,n\}), \end{aligned}$$

where the first inequality follows from $C^{k-\ell}(\{k-\ell+1\},N-\{k-\ell+1\})\geq 1$; the second inequality follows from $C^{k-\ell}(\{1,\cdots,k-\ell\},\{k-\ell+1,\cdots,n\})\geq 1$.

So

$$\begin{aligned} a_\ell &\geq \ell+1-\eta+\frac{n-k-2^\ell+\ell}{n-k+\ell-1} C^{k-\ell}(\{1,\cdots,k-\ell\},\{k-\ell+2,\cdots,n\})+\sum_{j=2}^{k-\ell-1} C^{k-\ell}(\{1,\cdots,j\},\{j+1\})+1-\left(\frac{2^\ell-1}{n-k+\ell-1}\right) C^{k-\ell}(\{1,\cdots,k-\ell\},\{k-\ell+2,\cdots,n\}) \\ &= \ell+2-\eta+\frac{n-k-2^{\ell+1}+\ell+1}{n-k+\ell-1} C^{k-\ell}(\{1,\cdots,k-\ell\},\{k-\ell+2,\cdots,n\})+\sum_{j=2}^{k-\ell-1} C^{k-\ell}(\{1,\cdots,j\},\{j+1\}) \\ &\geq \ell+2-\eta+\sum_{j=2}^{k-\ell-1} C^{k-\ell}(\{1,\cdots,j\},\{j+1\})+\frac{n-k-2^{\ell+1}+\ell+1}{n-k+\ell-1} C^{k-\ell-1}(\{1,\cdots,k-\ell\},\{k-\ell+2,\cdots,n\}) \\ &\geq \ell+2-\eta+\sum_{j=2}^{k-\ell-1} C^{k-\ell}(\{1,\cdots,j\},\{j+1\})+\left(\frac{n-k-2^{\ell+1}+\ell+1}{n-k+\ell-1}\right)\left(\frac{n-k+\ell-1}{n-k+\ell}\right) C^{k-\ell-1}(\{1,\cdots,k-\ell\},\{k-\ell+1,\cdots,n\}) \\ &\geq \ell+2-\eta+\frac{n-k-2^{\ell+1}+\ell+1}{n-k+\ell} C^{k-\ell-1}(\{1,\cdots,k-\ell\},\{k-\ell+1,\cdots,n\})+\sum_{j=2}^{k-\ell-1} C^{k-\ell-1}(\{1,\cdots,j\},\{j+1\}) \\ &= a_{\ell+1}, \end{aligned}$$

where the thrid inequality follows from assumption (8) and the last inequality is true because no capacity ever decreases. ■

4.2. *Upper Bound*

Throughout this section, we assume $n \leq 6$. We introduce some notations and explain the main idea of the algorithm first. Consider any state with a network $G = [N, E, c]$ with site set $N = \{1, \cdots, n\}$, and edge capacities $\{c_{ij} : (i,j) \in E\}$. Let $\pi = (\pi_1, \cdots, \pi_n)$ be the site potential vector. Without loss of generality, let the sites be sorted in nondecreasing potential order, i.e., $\pi_1 = \pi_2 \geq \cdots \geq \pi_n$. Let

$$\eta^* = \frac{n-2}{n-2^{k^*}} \left(\frac{k^*+1}{2} \beta_n - k^* \right) = \frac{n-2}{n+k^* 2^{k^*}}.$$

Let r_{xy} ($x, y \in N$ and $x < y$) be the next requirement that is revealed/updated and ready for processing. Either $\pi_x \geq r_{xy} > \pi_y$, in which case only the site potential π_y increases to $\pi'_y = r_{xy} = \pi_y + \delta_y$; or $r_{xy} > \pi_x$, in which case π_x and π_y both increase to $\pi'_x = \pi'_y = r_{xy}$. All other sites potentials remain unchanged.

The processing of this new requirement is described in the algorithm POTENTIAL below. The algorithm at every stage maintains a network G such that maximum flow value between every pair of sites i and j in N is at least $\min\{\pi_i, \pi_j\} \geq r_{ij}$. Thus, it does not explicitly use r_{xy} as input, but instead uses π'_x and π'_y. In the following any notation just introduced with a prime attached will denote the corresponding meaning after requirement r_{ij} is revealed/updated and processed; for example $\pi' = (\pi'_1, \cdots, \pi'_n)$ denotes the new potential vector.

Algorithm POTENTIAL:

Input: A site set $N = \{1, 2, \cdots, n\}$; a network $G = [N, E, c]$ after the processing of some requirements. The sites are sorted such that $\pi_1 = \pi_2 \geq \cdots \geq \pi_n$, where π is the vector of current site potentials. Either, for some $y \in N$, π_y is increased to some $\pi'_y = \pi_y + \delta_y \leq \pi_1$ or, for some $\{x, y\} \subseteq N$, $x < y$, π_x and π_y are both increased to $\pi'_x = \pi'_y = u$. For every other site $i \in N$, $\pi'_i = \pi_i$.

Output: An updated network $G = [N, E', c']$ that satisfies the flow requirement $\min\{\pi'_i, \pi'_j\}$ between every pair of sites $i, j \in N$.

Case 1. If $y = 2$ (and hence $x = 1$, therefore both π_1 and π_2 increase by the same amount δ_y), then update the capacities as follows:

$$c'_{12} = c_{12} + (1 - \eta^*)\delta_y; \tag{14}$$

$$c'_{k\ell} = c_{k\ell} + \frac{\eta^*}{n-2}\delta_y, \ k = 1, 2; \ \ell = 3, \cdots, n. \tag{15}$$

Case 2. If $y \in \{3, \cdots, k^*+1\}$ and only π_y is increased to $\pi'_y = \pi_y + \delta_y = u$, then update the capacities as follows:

2.1. If $\delta_y \leq \pi_{y-1} - \pi_y$

$$c'_{y1} = c_{y1} + \left(1 - \frac{2^{y-2}(n-y+1)}{n-2}\eta^*\right)\delta_y; \tag{16}$$

$$c'_{y\ell} = c_{y\ell} + \frac{2^{y-2}\eta^*}{n-2}\delta_y, \ \ell = y+1, \cdots, n. \tag{17}$$

2.2. If $\delta_y > \pi_{y-1} - \pi_y$, then let $\overline{\delta}_y = \pi_{y-1} - \pi_y$. Update the edge capacities as in Case 2.1 above using $\overline{\delta}_y$ instead of δ_y. Set $\pi'_y = \pi_{y-1}$. Renumber site y as $y-1$ and $y-1$ as y. Set $\pi =$ the updated vector of site potentials, $\pi'_{y-1} = u$ as the only increased site potential and repeat the algorithm.

Case 3. If $y \in \{k^*+2, \cdots, n\}$ and π_y is the only site potential increased to $\pi'_y = \pi_y + \delta_y = u$, then update the capacities as follows:

3.1. If $\delta_y \leq \pi_{k^*+1} - \pi_y$, then

$$c'_{y1} = c_{y1} + \left(1 - \frac{2^{k^*}}{n-2}\eta^*\right)\delta_y. \tag{18}$$

3.2. If $\delta_y > \pi_{k^*+1} - \pi_y$, then let $\overline{\delta}_y = \pi_{k^*+1} - \pi_y$. Update the edge capacities as in Case 3.1 above using $\overline{\delta}_y$ instead of δ_y. Set $\pi'_y = \pi_{k^*+1}$. Renumber site y as k^*+1 and ℓ as $\ell+1$ for $\ell = k^*+1, \cdots, y-1$. Set $\pi =$ the updated vector of site potentials, $\pi'_{k^*+1} = u$ as the only increased site potential and repeat the algorithm.

Case 4. If, for some $x, y \in N$, $x < y$, then both π_x and π_y are increased to some common value u, then break down this increment into the following sequence of increments: (i) increase only π_y to $\min\{u, \pi_1\}$; (ii) increase only π_x to $\min\{u, \pi_1\}$; (iii) if $u > \pi_1$, then both π_x and π_y increase to u. For (i), perform Case 2 or 3 of the algorithm; for (ii), perform Case 2 or 3 of the algorithm using x instead of y; for (iii), perform Case 1 of the algorithm.

The following is easy to verify:

Proposition 8 *For $n \leq 6$, the updated edge capacities by the algorithm satisfy $c'_{ij} \geq c_{ij}$, for all $(i,j) \in E'$.*

For any $j \in \{3, \cdots, n\}$ and sorted potential vector $\pi = (\pi_1, \cdots, \pi_n)$, where $\pi_1 = \pi_2 \geq \cdots \geq \pi_n$, we define

$$f_j(\pi) = \frac{2\eta^*}{n-2}(\pi_2 - \pi_j) + \sum_{\ell=3}^{\min\{j-1, k^*+1\}} \frac{2^{\ell-2}\eta^*}{n-2}(\pi_\ell - \pi_j);$$

$$g(\pi) = \beta_n \left(\frac{1}{2} \sum_{\ell=1}^{n} \pi_\ell \right) - \sum_{\ell=1}^{k^*-1} \left(\frac{\ell+1}{2}\beta_n - \ell - \frac{n-2^\ell}{n-2}\eta^* \right) (\pi_{\ell+1} - \pi_{\ell+2}).$$

Lemma 9 *Suppose $n \leq 6$. Let $G = [N, E, c]$ be the network produced by algorithm* POTENTIAL *after processing some requirements. Let $\pi_1 = \pi_2 \geq \cdots \geq \pi_n$ be the current potentials. Then G satisfies the following conditions.*

(1) We can send π_2 units of flow from site 2 to site 1.

(2) For any $j \in \{3, \cdots, n\}$, we can send simultaneously $\pi_j + f_j(\pi)$ units of flow from site j to site 1, and $f_j(\pi)$ units of flow from site p to site 1, for every $p \in \{j+1, \cdots, n\}$.

(3) The total capacity at the current state $\sum_{e\in E} c_e \leq g(\pi)$. Hence,

$$\frac{\sum_{e\in E} c_e}{\frac{1}{2}\sum_{\ell=1}^{n} \pi_\ell} \leq \beta_n.$$

Proof. We show inductively that Conditions 1, 2 and 3 are satisfied. Initially, when there is no requirement revealed, all the site potentials are zero and these conditions are obviously satisfied. Suppose they are satisfied currently at an arbitrary stage by the current network $G = [N, E, c]$. We show that they are still true after the processing of the next requirement r_{xy} ($x < y$).

Suppose the potentials before r_{xy} is revealed are sorted in a non-increasing order, that is, $\pi_1 = \pi_2 \geq \cdots \geq \pi_n$. Note that each of the Cases 2.2, 3.2 and 4 reduces to a sequence of Cases 1, 2.1 and 3.1. Hence, it suffices to show that the lemma is true for each of Cases 1, 2.1, and 3.1.

Case 1. *Case 1 of algorithm* POTENTIAL *occurs, and hence $x = 1$, $y = 2$.*

First, for Condition 1, consider any cut $(S, \overline{S})$ with $1 \in S$ and $2 \in \overline{S}$. By the max-flow min-cut Theorem [8] and the inductive hypothesis, the capacity of the cut $(S, \overline{S})$ in G is at least $\pi_2 (= \pi_1)$. By (14)-(15), the total extra capacity added to edges of this cut is

$$(1-\eta^*)\delta_2 + (n-2)\frac{\eta^*}{n-2}\delta_2 = \delta_2$$

Therefore at least $\pi_2 + \delta_2 = \pi_2'$ units of flow can now be sent from site 2 to site 1.

Second, for Condition 2, consider any $j \in \{3, \cdots, n\}$. Let $(S, \overline{S})$ be a 1-j cut, that is, $1 \in S$ and $j \in \overline{S}$. Suppose among the sites $\{j+1, \cdots, n\}$, $m (\in [0, n-j])$ of them belong to $\overline{S}$. By (14)-(15), the total extra capacity added to edges of this cut is either

$$(1-\eta^*)\delta_2 + (n-2)\frac{\eta^*}{n-2}\delta_2 = \delta_2, \text{ if } 2 \in \overline{S}.$$

or

$$(m+1)\frac{2\eta^*}{n-2}\delta_2, \text{ if } 2 \notin \overline{S}.$$

It is easy to verify, by the definition of η^*, that, for any $0 \leq m \leq n-j$

$$(m+1)\frac{2\eta^*}{n-2}\delta_2 \leq \delta_2.$$

Therefore the total extra capacity added to edges of this cut is at least

$$(m+1)\frac{2\eta^*}{n-2}\delta_2, \text{ for any } m \in \{0, \cdots, n-j\}. \quad (19)$$

(i) Choosing $m = 0$ in (19) implies that the minimum cut capacity among all 1-j cuts is at least

$$\frac{2\eta^*}{n-2}\delta_2 = \pi_j' + f_j(\pi') - \pi_j - f_j(\pi).$$

where the equality follows from $\pi_\ell' = \pi_\ell$ for all $\ell \in \{3, \cdots, n\}$, and $\pi_2' = \pi_2 + \delta_2$.

(ii) Because of the symmetry of j and any site $p \in \{j+1, \cdots, n\}$, analogously, the minimum cut capacity among all 1-p cuts is also at least

$$\frac{2\eta^*}{n-2}\delta_2, \text{ for any } p \in \{j+1, \cdots, n\}.$$

(iii) choosing $m = n - j$ in (19) implies that the minimum cut capacity among all cuts that separate 1 from all sites in $\{j+1, \cdots, n\}$, is at least

$$(n-j+1)\frac{2\eta^*}{n-2}\delta_2.$$

Now Condition 2 follows from the inductive hypothesis and (i), (ii) and (iii), based on Proposition 3.

For Condition 3, by (14)-(15), the total capacity increases by

$$\sum_{e\in E'} c_e' - \sum_{e\in E} c_e = (1-\eta^*)\delta_2 + (n-2)\frac{2\eta^*}{n-2}\delta_2 = (1+\eta^*)\delta_2.$$

On the other hand g increases by the same amount

$$g(\pi') - g(\pi) = \beta_n\delta_2 - (\beta_n - 1 - \eta^*)\delta_2 = (1+\eta^*)\delta_2.$$

Case 2. *Case 2.1 of algorithm* POTENTIAL *occurs, and hence* $y \in \{3, \cdots, k^*+1\}$, and $\delta_y \leq \pi_{y-1} - \pi_y$.

First, Condition 1 is implied directly by the inductive hypothesis because $\pi'_2 = \pi_2$.

Second, for Condition 2, it is easy to see that this condition follows immediately when $j \in \{3, \cdots, y-1\}$ because neither π_j nor $f_j(\pi)$ changes. So we focus on $j \in \{y, \cdots, n\}$.

If $j = y$, then $\pi'_y = \pi_y + \delta_y$ and $f_y(\pi') = f_y(\pi) - \frac{2^{y-2}\eta^*}{n-2}\delta_y$ using $\sum_{\ell=3}^{y-1} 2^{\ell-2} = 2^{y-2} - 2$. In G, we can simultaneously send $\pi_y + f_y(\pi)$ units of flow from y to 1 and $f_y(\pi)$ units of flow from ℓ to 1, for all $\ell > y$. Thus, in G, after sending $\pi_y + f_y(\pi')$ units of flow from y to 1 and $f_y(\pi')$ units of flow from ℓ to 1, for all $\ell > y$, we have additional residual flow of $\frac{2^{y-2}\eta^*}{n-2}\delta_y$ that can be sent from each $\ell \in \{y, \cdots, n\}$ to 1. Using extra capacity of $\frac{2^{y-2}\eta^*}{n-2}\delta_y$ added to each of the edge $\{(y,\ell) : \ell \in \{y+1, \cdots, n\}\}$, this can be converted to an additional $(n-y+1)\frac{2^{y-2}\eta^*}{n-2}\delta_y$ units of flow from y to 1. Also, using extra capacity to edge $(1, y)$ we can send $\left(1 - \frac{2^{y-2}(n-y+1)\eta^*}{n-2}\right)\delta_y$ units of flow from y to 1.

If $j > y$, then $\pi'_j = \pi_j$ and $f_j(\pi') = f_j(\pi) + \frac{2^{y-2}\eta^*}{n-2}\delta_y$. So we need to simultaneously send extra flow of $\frac{2^{y-2}\eta^*}{n-2}\delta_y$ units along each of the sites in $\{j, \cdots, n\}$ to site 1. we can send the flow along the set of paths $\{(p-y-1) : p \in \{j, \cdots, n\}\}$ using the extra capacities added to the edges $\{(p, y) : p \in \{1, j, \cdots, n\}\}$. By direct verification, this is feasible for $n \leq 6$ since total extra flow on edge $(1, y)$ does not exceed the extra capacity added to the edge $(1, y)$:

$$(n-j+1)\frac{2^{y-2}\eta^*}{n-2}\delta_y \leq \left(1 - \frac{2^{y-2}(n-y+1)\eta^*}{n-2}\right)\delta_y.$$

Finally, for Condition 3, by (16)-(17), the total capacity increases by

$$\sum_{e\in E'} c'_e - \sum_{e\in E} c_e = \left(1 - \frac{2^{y-2}(n-y+1)}{n-2}\eta^*\right)\delta_y + \sum_{\ell=y+1}^{n} \frac{2^{y-2}\eta^*}{n-2}\delta_y = \left(1 - \frac{2^{y-2}}{n-2}\eta^*\right)\delta_y.$$

On the other hand g increases by the same amount

$$g(\pi') - g(\pi) = \frac{\beta_n}{2}\delta_y - \left(\frac{\beta_n}{2} - 1 + \frac{2^{y-2}}{n-2}\eta^*\right)\delta_y = \left(1 - \frac{2^{y-2}}{n-2}\eta^*\right)\delta_y.$$

Case 3. *Case 3.1 of algorithm* POTENTIAL *occurs, and hence* $y \in \{k^*+2, \cdots, n\}$ and $\delta_y \leq \pi_{k^*+1} - \pi_y$.

First, Conditions 1 is implied directly by the inductive hypothesis.

Second, for Condition 2, it is easy to see that this condition follows immediately when $j \neq y$ because neither π_j nor $f_j(\pi)$ changes. So we focus on $j = y$. Let $(S, \overline{S})$ be a 1-y cut, that is, with $1 \in S$ and $y \in \overline{S}$. Suppose among the sites $\{y+1, \cdots, n\}$, $m \in [0, n-y]$ of them belongs to $\overline{S}$. By the max-flow min-cut Theorem [8] and the inductive hypothesis, the cut capacity of $(S, \overline{S})$ currently is at least

$$\pi_y + (m+1)f_y(\pi) \tag{20}$$

By (18), the algorithm increases capacities on edges of cut $(S, \overline{S})$ by

$$\left(1 - \frac{2^{k^*}}{n-2}\eta^*\right)\delta_y = \delta_y + \left(-\frac{2\eta^*}{n-2} - \sum_{\ell=3}^{k^*+1}\frac{2^{\ell-2}\eta^*}{n-2}\right)\delta_y = \delta_y + (f_y(\pi') - f_y(\pi)),$$

where the second equality follows from $\sum_{\ell=3}^{k^*+1} 2^{\ell-2} = 2^{k^*} - 2$. Adding this to (20) implies that cut capacity of $(S, \overline{S})$ at the next stage is at least

$$\pi'_y + mf_y(\pi) + f_y(\pi') \geq \pi'_y + (m+1)f_y(\pi'), \tag{21}$$

where the inequality follows from $f_y(\pi) \geq f_y(\pi')$.

(i) Choosing $m = 0$ in (21) implies that the minimum cut capacity among all 1-y cuts is at least $\pi'_y + f_y(\pi')$.

(ii) Choosing $m = n - y$ in (21) implies that the minimum cut capacity among all cuts that separate 1 from all sites in $\{y+1, \cdots, n\}$, is at least $\pi'_j + (n-y+1)f_j(\pi')$.

(iii) Note that $f_y(\pi') \leq f_y(\pi)$. Therefore by the inductive hypotheses, we can send at least $f_y(\pi')$ units of flow from p to 1, for any $p \in \{y+1, \cdots, n\}$.

Now Condition 2 for $j = y$ follows from (i), (ii) and (iii), based on Proposition 3.

Finally, for Condition 3, by (18), the total capacity increases by

$$\sum_{e\in E'} c'_e - \sum_{e\in E} c_e = \left(1 - \frac{2^{k^*}}{n-2}\eta^*\right)\delta_y = \frac{\beta_n}{2}\delta_y,$$

which is the same amount increased by g

$$g(\pi') - g(\pi) = \frac{\beta_n}{2}\delta_y.$$

■

Acknowledgements

We are thankful to a referee for constructive suggestions which improved the presentation of the paper.

References

[1] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder and J. Naor, The on-line set cover problem, Proceedings of the 35th Annual ACM Symposium on Theory of Computing STOC (2003), 100-105.

[2] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder and J. Naor, A general approach to on-line network optimization problems, Proceedings of the Fifteenth Annual ACM-SIAM SODA (2004), 577-586.

[3] N. Alon and Y. Azar, On line Steiner trees in the Euclidean plane, Proc. 8th ACM Symp. on Computational Geometry, ACM Press (1992), 337-343. Also: Discrete and Computational Geometry 10 (1993), 113-121.

[4] G. Ausiello, V. Bonifaci, L. Laura, On-line algorithms for the Asymmetric Traveling Salesman Problem, WADS 2005, LNCS, Springer, 2005.

[5] G. Ausiello, M. Demange, L. Laura, V. Paschos, Algorithms for the on-line quota traveling salesman problem, Information Processing Letters, 92(2) (2005), 89-94

[6] G. Ausiello, E. Feuerstein, S. Leonardi, L. Stougie, M. Talamo, Algorithms for the on-line traveling salesman, Algorithmica, 29 (2001), 560-581

[7] B. Awerbuch, Y. Azar, Y. Bartal. , On-line generalized steiner problem, Proceedings of 7th SODA, (1996), 68-74.

[8] L. R. Ford and D. R. Fulkerson, Flows in Networks, Princeton University Press, Princeton, New Jersey, 1962.

Received 19 April 2006; revised 22 January 2007; accepted 3 February 2007

[9] D. Gusfield, Simple Constructions for the Multi-Terminal Network Flow Synthesis, SIAM Journal on Computing, 12(1) (1983), 157-165.

[10] R.E. Gomory and T.C. Hu, Multi-terminal Network Flows, Journal of SIAM, 9, (1961), 551-570.

[11] E. Grove, M. Kao, P. Krishnan, and J. Vitter, Online perfect matching and mobile computing, Proceedings of International Workshop on Algorithm and Data Structures, 1995.

[12] R.M. Karp, U.V. Vazirani, and V. V. Vazirani. An optimal algorithm for on-line barpartite matching, Proceedings of the 22nd Annual ACM Symposium on Theory of Computing STOC (1990), 352-358.

[13] W. Mayeda, Terminal and Branch Capacity Matrices of a Communication Net, IRE Transactions on Circuit Theory, CT-7 (1960), 261-269.

[14] A. Meyerson. Online facility location. In Proceedings of the 42nd Annual Symposium on Foundations of Computer Science FOCS, (2001) 426-431.

[15] B. Kalyanasundaram and K. Pruhs: Online Weighted Matching. Journal of Algorithms, 14(3) (1993), 478-488.

[16] B. Kalyanasundaram and K. Pruhs, Online network optimization problems, Online Algorithms: The State of the Art, A. Fiat and G. Woeginger (eds.), Lecture Notes in Computer Science 1442, Springer-Verlag, (1998) 268-278.

[17] B. Kalyanasundaram and K. Pruhs: An optimal deterministic algorithm for on-line b-matching. Theoretical Computer Sciences 233(1-2) (2000), 319-325.

[18] B. Kalyanasundaram and K. Pruhs: The online transportation problem. SIAM Journal on Discrete Mathematics. 13(3) (2000), 370-383.

[19] A. Schrijver, Combinatorial Optimization: Polyhedra and Efficiency, Algorithms and Combinatorics 24, Springer-Verlag, New York, 2003.

[20] K. Talluri, Network synthesis with few edges, Networks, 27 (1996), 109-115.