Lowering Eccentricity of a Tree by Node-Upgrading*

Toshihide Ibaraki[1] Xiao-guang Yang[2]

[1] Department of Informatics, School of Science and Technology
Kwansei Gakuin University, Sanda, Japan 669-1337
[2] Institute of Systems Science, Academy of Mathematics and Systems Science
Chinese Academy of Sciences, Beijing 100080, China

Abstract *The eccentricity lowering problem is to reduce the eccentricity of a communication network within a given bound, by upgrading some nodes (i.e., shrinking the lengths of the edges linking to such nodes), where we want to minimize the required cost. We consider two types of node-upgrading strategies; i.e., the continuous upgrading and the discrete upgrading, where the improvement under the first strategy is a continuous variable, and the improvement under the second strategy is a fixed amount. These problems are NP-hard for the general graph. When the graph is a tree, we present an $O(|V|^2)$ time algorithm to solve the eccentricity lowering problem under the continuous upgrading strategy. However, the problem under the discrete upgrading strategy is still NP-hard even if the graph is a line.*

Keywords eccentricity, communication networks, node upgrading, discrete upgrading strategy, continuous upgrading strategy, tree

1 Introduction

In a communication network, the communication delay on the edges from a node can be reduced by node-upgrading (i.e., to install a faster communication equipment at the node). This strategy is sometimes applied to improve the communication quality of the network.

Given a network, the eccentricity of a node is defined as the largest distance from the designated node to other nodes. Identifying the delay of an edge as its length, we consider the problems of how to reduce the eccentricity of a given network by node-grading strategies in the minimum cost. These problems have

*The research was supported by the Scientific Grant in Aid by the Ministry of Education, Science, Culture and Sports of Japan, by the 21 Century COE Program of Japan "Informatics Research Center for Development of Knowledge Society Infrastracture", and by the National Key Research and Development Program of China (Grant No. 2002CB312004) and NSFC (Grant No. 70425004).

many potential applications in real world such as broadcasting, facility-accessing and so forth.

Let a graph $G = (V, E)$ represent a communication network, where node set V stands for the transmission stations, and edge set E stands for the links between transmission stations. The communication delay $d(u, v)$ associated with an edge (u, v) from u to v, can be decomposed into three parts, $s(u)$ the sending time from node u, $t(u, v)$ the transmission time over link (u, v), and $r(v)$ the receiving time at v; i.e., $d(u, v) = s(u) + t(u, v) + r(v)$. The communication delay between any two nodes (adjacent or nonadjacent) is given by the length of the shortest path between the two nodes, where $d(u, v)$ is regarded as the length of edge (u, v). For any node v, upgrading it will shorten the sending time and the receiving time at v. We consider two types of upgrading strategies, the continuous upgrading strategy and the discrete upgrading strategy.

In the continuous upgrading strategy, each node v is given two positive shrinking coefficients $t_s(v) > 0$, $t_r(v) > 0$ and a cost coefficient $c(v) \geq 0$. Also v is associated with a continuous variable $0 \leq x_v \leq b(v)$ (called improvement made at v), where $b(v)$ is the upper bound on the improvement. The sending time from node v under improvement x_v becomes $\max\{s(v) - t_s(v)x_v, 0\}$ and the receiving time at v becomes $\max\{r(v) - t_r(v)x_v, 0\}$. Without loss of generality, we assume that $b(v) \leq \max\{\frac{s(v)}{t_s(v)}, \frac{r(v)}{t_r(v)}\}$. The eccentricity lowering problem under continuous upgrading strategy is to find a vector $x : V \to R_+^{|V|}$ that minimizes the total cost $\sum_{v \in V} c(v)x_v$ under the constraints that the eccentricity of a designated node v_0 after the upgrading is within U and $x_v \leq b(v)$ holds for all nodes v. This problem is denoted as CELP (continuous version of the eccentricity lowering problem).

In the discrete upgrading strategy, each node v is given three fixed values $0 \leq p_s(v) \leq s(v)$, $0 \leq p_r(v) \leq r(v)$ and $c(v) \geq 0$. If node v is upgraded, the sending time will become $s(v) - p_s(v)$, and the receiving time will become $r(v) - p_r(v)$ at the cost of $c(v)$. The eccentricity lowering problem under discrete upgrading strategy is to find a subset $S \subseteq V$ such that upgrading all nodes in S reduces the eccentricity of the source node v_0 within a given level U at the minimum total cost $\sum_{v \in S} c(v)$. We denote the problem as DELP (discrete version of the eccentricity lowering problem).

The difference between the continuous upgrading strategy and the discrete upgrading strategy is that, the cost and reduction under the continuous upgrading strategy depend linearly on the magnitude of the improvement x_v made at each node v, but the cost and reduction under the discrete upgrading strategy depend only on whether the nodes are upgraded or not.

It easily follows from the known results (as briefly surveyed in the next section) that both DELP and CELP are NP-hard. So our attention focuses on graphs with some simple structures. Our main result in Section 4 shows that CELP can be solved in $O(|V|^2)$ time if the graph is a tree.

2 Literature Review

The node-based upgrading was first studied by Paik and Sahni [10]. The authors considered that the delay of an edge is reduced by a factor α if one endpoint of the edge is upgraded, and by α^2 if both endpoints of the edge are upgraded, where $0 \leq \alpha < 1$ is a constant. The objective is to find a minimum number of nodes to be upgraded such that there exists some subgraph with a specific structure to satisfy delay requirements. For example, problem $ShortPath(x, B)$ seeks the smallest number of nodes to be upgraded such that no pair of nodes has the shortest path of length $> B$. This problem is the same as ours except that the upgrading strategy is different. Paik and Sahni showed that several node upgrading problems are NP-hard. Recently Krumke et al. [6, 7, 8, 9] generalized Paik and Sahni's model of [10]. They associated each edge e with three integers $d_0(e) \geq d_1(e) \geq d_2(e)$, where $d_i(e)$ describes the delay of e when exactly i of its endpoints are upgraded. The cost $c(v)$ of upgrading v is also associated. The node upgrading problem considered by them are the bottleneck tree upgrading problem and the minimum length tree upgrading problem, which are to find a minimum cost set of upgraded nodes so that the resulting network has a spanning tree such that the maximum delay and the total delay of the spanning tree do not exceed given bounds. Krumke et al. [7, 8] presented $O(\log |V|)$ approximating algorithms for these problems.

There is a related class of network improvement problems called the edge-upgrading problem, where the lengths of edges are reduced. This class of problems has been studied in [1, 2, 11, 12]. For a general graph, it is hard to approximate within an $O(\log |V|)$ approximation ratio. If the graph is a tree, however, some polynomial algorithms have been presented [1, 11]. In particular, Zhang et al. [11] contains an $O(|V| \log |V|)$ time algorithm.

3 Summary of results

Concerning problems CELP and DELP as defined above, we obtained the following results.

Theorem 1 *If the underline graph $G = (V, E)$ is a general graph, both CELP and DELP are strongly NP-hard.*

Theorem 2 *Even if the underline graph $G = (V, E)$ is a line, DELP is NP-hard, but admits a PTAS.*

Theorem 3 *If the underline graph $G = (V, E)$ is a tree, CELP can be solved in $O(|V|^2)$ time.*

The proofs of the first two theorems can be found in the accompanying paper [5]. In this reference, it is also stated that CELP on a tree can be solved in $O(|V| \log |V|)$ time, which is shown by reducing CELP to the edge upgrading problem and applying the algorithm of [12] to it. Although the time bound of our

result is inferior, we present the proof of theorem 3, since it is based on a different formulation (i.e., as a convex programming problem), which may turn out to be useful for more general problems.

4 Problem CELP on a Tree

4.1 Problem formulation

Assume that a tree $T = (V, E)$, a designated node $v_0 \in V$ and a bound U on the eccentricity are given. It is clear that the problem is feasible if and only if the eccentricity of the network becomes within U when all nodes are upgraded to their upper bounds. From now on, we assume that the problem is feasible. Also, we regard the given tree T as a directed tree with the orientation from the source node v_0 to all leaves. Let L denote the leaf set of the tree (excluding v_0 if v_0 itself is a leaf node).

Assuming that x_v be the improvement made at an intermediate node v, let us consider the distance reduction y_v of the paths from v_0 to leaf nodes incurred by x_v. By the definition of the continuous upgrading strategy, y_v varies on different intervals; when $0 \le x_v \le \min\{\frac{s(v)}{t_s(v)}, \frac{r(v)}{t_r(v)}, b(v)\}$, then we have $y_v = (t_s(v)+t_r(v))x_v$; when $\frac{s(v)}{t_s(v)} < b(v) \le \frac{r(v)}{t_r(v)}$ and $\frac{s(v)}{t_s(v)} < x_v \le b(v)$, then we have $y_v = s(v) + t_r(v)x_v$; when $\frac{r(v)}{t_r(v)} < b(v) \le \frac{s(v)}{t_s(v)}$ and $\frac{r(v)}{t_r(v)} < x_v \le b(v)$, then we have $y_v = r(v) + t_s(v)x_v$.

For the source node v_0 (and each leaf node $l \in L$), we can define $t_r(v_0) := 0$ (and $t_s(l) := 0$), and define $\frac{r(v_0)}{t_r(v_0)} := 0$ (and $\frac{s(l)}{t_s(l)} := 0$ respectively). Then all nodes can be treated in an unified way.

Note that y_v is an increasing function in x_v, and therefore x_v can be written as the inverse function of y_v. Let $\sigma_v^{(1)} = \frac{1}{t_s(v)+t_r(v)}$, and

$$\sigma_v^{(2)} = \begin{cases} \frac{1}{t_s(v)}, & \text{if } \frac{s(v)}{t_s(v)} < \frac{r(v)}{t_r(v)}, \\ \frac{1}{t_r(v)}, & \text{otherwise.} \end{cases} \tag{1}$$

Let $y_v^{(0)} = 0$; $y_v^{(1)} = (t_s(v) + t_r(v)) \min\{\frac{s(v)}{t_s(v)}, \frac{r(v)}{t_r(v)}, b(v)\}$; $y_v^{(2)} = s(v) + t_r(v)b(v)$ if $\frac{s(v)}{t_s(v)} < b(v) \le \frac{r(v)}{t_r(v)}$, or $y_v^{(2)} = r(v) + t_r(v)b(v)$ if $\frac{r(v)}{t_r(v)} < b(v) \le \frac{s(v)}{t_s(v)}$. Then x_v is written as

$$x_v = \begin{cases} \sigma_v^{(1)} y_v, & \text{if } y_v^{(0)} \le y_v \le y_v^{(1)}, \\ \sigma_v^{(1)} y_v^{(1)} + \sigma_v^{(2)}(y_v - y_v^{(1)}), & \\ & \text{if } y_v^{(1)} < y_v \le y_v^{(2)}. \end{cases} \tag{2}$$

We transform $c(v)x_v$, the cost incurred by x_v, into a cost function in y_v as follows

(see Fig. 1):

$$\tilde{c}_v(y_v) = \begin{cases} +\infty, & y_v < y_v^{(0)}, \\ c(v)\sigma_v^{(1)} y_v, & y_v^{(0)} \le y_v \le y_v^{(1)}, \\ c(v)(\sigma_v^{(1)} y_v^{(1)} + \sigma_v^{(2)} (y_v - y_v^{(1)})), & \\ & y_v^{(1)} < y_v \le y_v^{(2)}, \\ +\infty, & y_v > y_v^{(2)}. \end{cases} \tag{3}$$

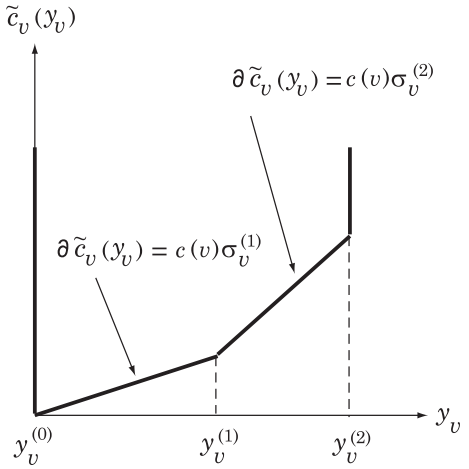

Figure 1: Cost function $\tilde{c}_v(y_v)$

It is straightforward to see that $\tilde{c}_v(y_v)$ is a piecewise linear, convex and increasing function in its proper region. Moreover $\tilde{c}_v(y_v)$ has at most three breaking points $y_v^{(0)}, y_v^{(1)}, y_v^{(2)}$, and $\tilde{c}_v(y_v^{(0)}) = 0$ always holds.

At these breaking points, the subgradients of $\tilde{c}_v(y_v)$ are given as follows.

$$\begin{aligned} \partial\tilde{c}_v(y_v^{(0)}) &= [-\infty,\ c(v)\sigma_v^{(1)}], \\ \partial\tilde{c}_v(y_v^{(1)}) &= [c(v)\sigma_v^{(1)},\ c(v)\sigma_v^{(2)}], \\ \partial\tilde{c}_v(y_v^{(2)}) &= [c(v)\sigma_v^{(2)},\ +\infty]. \end{aligned}$$

We denote the upper subgradient by $\partial^+\tilde{c}_v(y_v)$, and the lower subgradient by $\partial^-\tilde{c}_v(y_v)$. For example, we have $\partial^+\tilde{c}_v(y_v^{(1)}) = c(v)\sigma_v^{(2)}$, and $\partial^-\tilde{c}_v(y_v^{(1)}) = c(v)\sigma_v^{(1)}$. It is easily seen that $\partial^-\tilde{c}_v(y_v) = \partial^+\tilde{c}_v(y_v)$ holds if y_v is not a breaking point, and furthermore the upper subgradient at a breaking point is equal to the lower subgradient at the next breaking point.

Now we are ready to give a convex program formulation for CELP. We introduce some notations. For each pair of nodes u and v, there is the unique path between them, which we denote by the node set $N(u, v)$ in the path. For each $l \in L$, denote

$D(l)$ the length of the path from v_0 to l (called v_0-l path) before any upgrading. Let

$$\Delta_l = \max\{D(l) - U, 0\}, \quad l \in L,$$

i.e., the excess of $D(l)$ over U. Then CELP can be formulated as follows:

$$\begin{array}{ll} \text{minimize} & \sum\limits_{v \in V} \tilde{c}_v(y_v) \\ \text{subject to} & \sum\limits_{v \in N(s,l)} y_v \geq \Delta_l, \quad l \in L. \end{array} \tag{4}$$

4.2 Algorithm for solving CELP

For each $v \in V$, let $L_v := \{l \in L \mid v \in N(v_0, l)\}$. We call L_v the set of leaf nodes covered by v. From the tree structure, we have that $L_v \cap L_u \neq \emptyset$ if and only if either $L_v \subseteq L_u$ or $L_u \subseteq L_v$ holds.

Given a subset $Q \subseteq L$, we call a subset $V' \subseteq V$ a v_0-Q cut if
(1) $L_v \cap L_u = \emptyset$ for any $v, u \in V'$ with $v \neq u$,
(2) $\bigcup\limits_{v \in V'} L_v \supseteq Q$.

Given a current solution $y = (y_v : v \in V)$, define the weight of v at y by

$$w_v(y) = \partial^+ \tilde{c}_v(y_v), \quad v \in V. \tag{5}$$

Then a minimum v_0-Q cut at y is a v_0-Q cut V' that minimizes its weight sum $\sum\limits_{v \in V'} w_v(y)$.

The main idea of our algorithm is to reduce the eccentricity iteratively by using the minimum v_0-Q cuts. It can be described as follows.

Algorithm OPT-CELP:

Step 1: Let $y_v := 0$ for all $v \in V$.

Step 2: For each $l \in L$, let $\Delta'_l(y) := \Delta_l - \sum\limits_{v \in N(s,l)} y_v$. Let

$$\Delta'_{\max}(y) := \max\{\Delta'_l(y) \mid l \in L\},$$

$$Q(y) := \{l \in L \mid \Delta'_l(y) := \Delta'_{\max}(y)\}.$$

Step 3: If $\Delta'_{\max}(y) = 0$, halt; the current y is an optimal solution.

Step 4: Let $V_{\min}(y)$ be a minimum v_0-$Q(y)$ cut with respect to weights $w_v(y)$. Let $\overline{Q}(y) := L \setminus (\bigcup\limits_{v \in V_{\min}(y)} L_v)$, and let

$$\alpha(y) := \begin{cases} \min\{\Delta'_{\max}(y) - \Delta'_l(y) \mid l \in \overline{Q}(y)\}, & \text{if } \overline{Q}(y) \neq \emptyset, \\ +\infty, & \text{otherwise.} \end{cases}$$

For each $v \in V_{\min}(y)$, let $y_v^{(\text{next})}(y)$ be the smallest breaking point of $\tilde{c}_v(y_v)$ which is greater than y_v. Let

$$\begin{aligned} \beta(y) \quad := \quad & \min\{\Delta'_{\max}(y), \alpha(y), \min\{y_v^{(\text{next})}(y) - y_v \mid \\ & v \in V_{\min}(y)\}\}. \end{aligned}$$

Step 5: Upgrade $y_v := y_v + \beta(y)$ for all $v \in V_{\min}(y)$, and return to Step 2.

The implications of the notations introduced in Algorithm OPT-CELP are as follows. At the current solution y, $\Delta'_l(y)$ is the remaining reduction for $l \in L$, $\Delta'_{\max}(y)$ is the largest remaining reduction, $Q(y)$ is the set of leaf nodes with the largest remaining reduction, $\overline{Q}(y)$ is the set of leaf nodes which are not covered by the nodes in $V_{\min}(y)$, and $\alpha(y)$ is the difference between the current eccentricity and the longest path from v_0 to the leaf nodes $l \in \overline{Q}(y)$. Based on these, $\beta(y)$ is the largest possible increment of y_v, $v \in V_{\min}(y)$, before the eccentricity reaches U under the conditions that the remotest leaf nodes remain to be remotest and none of the new y_v, $v \in V_{\min}(y)$, exceed the next breaking points $y_v^{(\mathrm{next})}(y)$.

After obtaining the optimal distance reduction y^* by Algorithm OPT-CELP, we can convert it into the optimal improvement x^* by (2).

4.3 Validity of OPT-CELP

(i) Algorithm OPT-CELP is well-defined.

To this end, we need to show that $\beta(y) > 0$ holds in Step 5. It is straightforward to see that $\Delta'_{\max}(y) > 0$ and $\alpha(y) > 0$ in Step 4. Therefore we prove $\min\{y_v^{(\mathrm{next})}(y) - y_v \mid v \in V_{\min}(y)\} > 0$.

By the definition of $\tilde{c}_v(y_v)$, $w_v(y_v) = +\infty$ holds if the current y_v reaches its largest breaking point $y_v^{(2)}$; otherwise it is possible to increase y_v further. By the feasibility assumption of the instance, and by $\Delta'_{\max}(y) > 0$ in Step 4, we know that any $l \in Q(y)$ must have a node $v \in N(s, l)$ such that $y_v < y_v^{(2)}$. Otherwise this v_0-l path can not be shortened anymore, which is a contradiction to the feasibility assumption. Thus all nodes $v \in V_{\min}(y)$ satisfy $y_v < y_v^{(2)}$ by the minimality of v_0-$Q(y)$ cut, and hence we have $y_v < y_v^{(\mathrm{next})}(y)$ for all $v \in V_{\min}(y)$.

(ii) Algorithm OPT-CELP terminates in finite iterations.

¿From the definitions of $\alpha(y)$ and $\beta(y)$, $\Delta'_{\max}(y)$ is reduced exactly by $\beta(y)$ at each execution of Step 5, but the remotest leaf nodes remain to be remotest at the end of each iteration. As $\beta(y) > 0$ always holds as shown in (i), $\Delta'_{\max}(y)$ strictly decreases in each iteration.

After each iteration, one of the following three cases occurs depending on the value of $\beta(y)$.

(1) We have $\beta(y) = \Delta'_{\max}(y)$ and Algorithm OPT-CELP halts.

(2) We have $\beta(y) = \alpha(y)$, and at least one leaf node in $\overline{Q}(y)$ will enter $Q(y)$ after this iteration. $Q(y)$ strictly increases in the next iteration.

(3) We have $\beta(y) = y_v^{(\mathrm{next})}(y) - y_v$ for some node $v \in V_{\min}(y)$, and y_v proceeds to the next breaking point $y_v^{(\mathrm{next})}(y)$ and its weight $w_v(y)$ changes by (5).

Since either $Q(y)$ strictly increases or some y_v proceeds one breaking point, and the number of breaking points for each node is at most three, the total number of iterations is at most $|L| + 2|V| < 3|V|$.

(iii) Algorithm OPT-CELP converges to an optimal solution.

For this, recall the Karush-Kuhn-Tucker condition for the optimality of convex programming problem (4).

Lemma 1 [4] *A solution $y^* = (y_v^* : v \in V)$ is an optimal solution of* (4) *if and only if*

(a) *y^* is feasible, i.e., $\sum_{v \in N(v_0,l)} y_v^* \geq \Delta_l$ for all $l \in L$;*

(b) *there exist Lagrangian multipliers $\lambda^* = (\lambda_l^* : l \in L)$ such that*

$$\lambda_l^* \geq 0, \quad l \in L, \tag{6}$$

$$\lambda_l^* = 0, \quad \text{if } l \in L \text{ and } \sum_{v \in N(v_0,l)} y_v^* > \Delta_l, \tag{7}$$

$$\sum_{l \in L_v} \lambda_l^* \in \partial \tilde{c}_v(y_v^*), \quad v \in V. \tag{8}$$

Note that the solution y in each iteration of Algorithm OPT-CELP does not satisfy (a) before the algorithm halts, but becomes feasible when the algorithm halts.

To show the optimality of the solution obtained by Algorithm OPT-CELP, we will construct Lagrangian multiplies λ after each iteration such that y and λ satisfy conditions (6)-(8). Thus, when the algorithm halts, both conditions of Lemma 1 are satisfied, showing the optimality.

For the initial solution $y = 0$, i.e., $y_v = y_v^{(0)}$ for $v \in V$, we start with $\lambda_l = 0$ for all $l \in L$. It is trivial to see that conditions (6)-(8) are satisfied because $\sum_{l \in L_v} \lambda_l = 0 \in \partial \tilde{c}_v(y_v^{(0)})$ for all $v \in V$.

Now assume that there exists a vector λ together with the current y satisfying the following conditions:

$$\lambda_l \geq 0, \quad l \in L, \tag{9}$$

$$\lambda_l = 0, \quad \text{if } l \notin Q(y), \tag{10}$$

$$\partial \tilde{c}_v^-(y_v) \leq \sum_{l \in L_v} \lambda_l \leq \partial \tilde{c}_v^+(y_v) = w_v(y), \quad v \in V. \tag{11}$$

Note that if $\sum_{v \in N(v_0,l)} y_v > \Delta_l$ holds for some $l \in L$, then $\Delta_l' < 0$ in Step 2, and hence $l \notin Q(y)$ by the definition of $Q(y)$. Thus condition (10) implies condition (7). Therefore if y and λ satisfy conditions (9)-(11), then they also satisfy conditions (6)-(8). Moreover, it is straightforward to see that the initial solution $y = 0$ and the initial Lagrangian multiplies $\lambda = 0$ satisfy conditions (9)-(11).

For $v \in V$, let $Q_v(y) := L_v \cap Q(y)$, and define

$$\delta_v = w_v(y) - \sum_{l \in L_v} \lambda_l = w_v(y) - \sum_{l \in Q_v(y)} \lambda_l. \tag{12}$$

By (11), we have $\delta_v \geq 0$ for all $v \in V$.

For a node $v \in V_{\min}(y)$, let V' be any v-$Q_v(y)$ cut. We claim that

$$\delta_v \leq \sum_{u \in V'} \delta_u \tag{13}$$

holds. To prove this, note that $v \in V_{\min}(y)$ implies

$$w_v(y) \leq \sum_{u \in V'} w_u(y). \tag{14}$$

For otherwise, we can replace the $v \in V_{\min}$ by V' to obtain a v_0-$Q(y)$ cut with a smaller weight, which is a contradiction. Notice also that we have $\sum_{l \in L_v} \lambda_l = \sum_{l \in Q_v(y)} \lambda_l = \sum_{u \in V'} \sum_{l \in Q_u(y)} \lambda_l = \sum_{u \in V'} \sum_{l \in L_u} \lambda_l$. Then combining this with (14) and $\delta_u \geq 0$ for all $u \in V$, we obtain

$$\delta_v = w_v(y) - \sum_{l \in L_v} \lambda_l \leq \sum_{u \in V'} w_u(y) - \sum_{u \in V'} \sum_{l \in L_u} \lambda_l = \sum_{u \in V'} \delta_u.$$

Now let y' be the new solution updated in Step 5, i.e.,

$$y'_v = \begin{cases} y_v + \beta(y), & v \in V_{\min}(y), \\ y_v, & v \in V \setminus V_{\min}(y). \end{cases}$$

We distribute the margin δ_v to all λ_l, $l \in Q_v(y)$, to construct Lagrangian multipliers λ' for y', by the following procedure.

Procedure DIST:

Step 1: Let $\delta'_u := \delta_u$ for all nodes u in the subtree of v. Arrange the leaf nodes in $Q_v(y)$ in an arbitrary order. Start from the first node in $Q_v(y)$, say l.

Step 2: Let $\gamma_l := \min\{\delta'_u \mid u \in N(v,l)\}$, and let $\lambda'_l := \lambda_l + \gamma_l$. For each $u \in N(v,l)$, update $\delta'_u := \delta'_u - \gamma_l$.

Step 3: If $\delta'_v = 0$ holds, then halts. Otherwise, return to Step 2 with the next node l in $Q_v(y)$.

Inequality (13) tells that $\sum_{u \in V'} \delta_u$ is big enough for any v-$Q_v(y)$ cut V' to cover δ_v. Thus Procedure DIST is well-defined and halts in Step 3.

We now show that the new y' and λ' defined above satisfy (9)-(11). From Procedure DIST, λ'_l never decreases for any $l \in L$, and may increase only for $l \in Q(y)$. Since $Q(y)$ is nondecreasing in each iteration as shown in (ii), it is trivial to see that y' and λ' satisfy (9) and (10). To consider condition (11), note that $y'_v \neq y_v$ is possible only for $v \in V_{\min}(y)$. For $v \in V_{\min}(y)$, if none of y_v and y'_v is a breaking point, we have $\partial^- \tilde{c}_v(y_v) = \partial^+ \tilde{c}_v(y_v) = \partial^- \tilde{c}_v(y'_v) = \partial^+ \tilde{c}_v(y'_v)$, and if at least one of them is a breaking point, we have $\partial^+ \tilde{c}_v(y_v) = \partial^- \tilde{c}_v(y'_v)$. Thus y' satisfies

$$\partial^+ \tilde{c}_v(y_v) = w_v(y) = \partial^- \tilde{c}_v(y'_v), \quad \text{for all } v \in V_{\min}(y), \tag{15}$$

$$\partial \tilde{c}_v(y_v) = \partial \tilde{c}_v(y'_v), \quad \text{for all } v \in V \setminus V_{\min}(y). \tag{16}$$

By Procedure DIST and (12) we have for $v \in V_{\min}$,

$$\sum_{l \in L_v} \lambda'_l = \sum_{l \in L_v} \lambda_l + \delta_v = w_v(y) = \partial^- \tilde{c}_v(y'_v) \in \partial \tilde{c}_v(y'_v).$$

Also for any node $u \neq v$ in the subtree rooted at v, we have

$$\sum_{l \in L_u} \lambda_l \leq \sum_{l \in L_u} \lambda'_l \leq \sum_{l \in L_u} \lambda_l + \delta_u = w_u(y) = \partial^+ \tilde{c}_u(y_u).$$

The first inequality holds because $\lambda'_l \geq \lambda_l$ for all $l \in L$, and the second inequality holds because Step 2 in Procedure DIST guarantees that the total increment of λ'_l from λ_l for $l \in Q_u(y)$ does not exceed δ_u. As $\sum_{l \in L_u} \lambda_l \geq \partial^- \tilde{c}_u(y_u)$ by (11), we obtain $\sum_{l \in L_u} \lambda'_l \in \partial \tilde{c}_u(y_u) = \partial \tilde{c}_u(y'_u)$.

Finally denote the set of the remaining nodes between v_0 and $V_{\min}(y)$ (not including the nodes in $V_{\min}(y)$) by $V^-(y)$. For $u \in V^-(y)$, let $V^u_{\min}(y)$ be the set of nodes in $V_{\min}(y)$ which lie on the paths from u to L. Once again, by the definition of a minimum v_0-$Q(y)$ cut, we have

$$w_u(y) \geq \sum_{v \in V^u_{\min}(y)} w_v(y), \tag{17}$$

since otherwise, we can replace $V^u_{\min}(y)$ by u to obtain a v_0-$Q(y)$ cut with a smaller weight. Then $\sum_{l \in L_u} \lambda_l = \sum_{v \in V^u_{\min}(y)} \sum_{l \in L_v} \lambda_l$, (12), (16) and (17) together imply

$$\delta_u \geq \sum_{v \in V^u_{\min}(y)} \delta_v,$$

and hence

$$\begin{aligned} \sum_{l \in L_u} \lambda'_l &= \sum_{v \in V^u_{\min}(y)} \sum_{l \in L_v} \lambda_l + \sum_{v \in V^u_{\min}(y)} \delta_v \\ &\leq \sum_{l \in L_u} \lambda_l + \delta_u = w_u(y) = \partial^+ \tilde{c}_u(y_u) \\ &= \partial^+ \tilde{c}_u(y'_u). \end{aligned}$$

As $\sum_{l \in L_u} \lambda'_l \geq \sum_{l \in L_u} \lambda_l \geq \partial^- \tilde{c}_u(y_u)$ follows from (11), we have $\sum_{l \in L_u} \lambda'_l \in \partial \tilde{c}_u(y_u) = \partial \tilde{c}_u(y'_u)$.

Combining the above three cases, we conclude that λ' and y' satisfy (9)-(11). Since condition (a) is satisfied when Algorithm OPT-CELP halts, we proved the following theorem by induction and Lemma 1.

Theorem 4 *Algorithm OPT-CELP obtains an optimal solution of* (4) *for problem CELP on a tree.*

4.4 Time complexity of OPT-CELP

Now let us derive the complexity of Algorithm OPT-CELP. It is straightforward to see that $\Delta'_l(y)$, $\Delta'_{\min}(y)$, $Q(y)$, $\alpha(y)$ and $\min\{y_v^{(\mathrm{next})}(y) - y_v \mid v \in V_{\min}(y)\}$ can be computed in $O(|V|)$ time in each iteration of OPT-CELP. We show that the computation of a minimum v_0-$Q(y)$ cut, $V_{\min}(y)$, is also done in $O(|V|)$ time.

For the current solution y, denote the set of nodes lying on the paths from v_0 to the nodes in $Q(y)$ by $V(y)$. For an arc (u, v), we call v a child of u, and denote the set of child nodes of u by $C(u)$. We use the following procedure to find a $V_{\min}(y)$.

Procedure MINCUT:

Step 1: For all $v \in Q(y)$, let $\xi_v := w_v(y)$ and $V_v := \{v\}$, and mark all nodes in $Q(y)$.

Step 2: If v_0 is marked, then halt. $V_{\min}(y) := V_{v_0}$ is a minimum v_0-$Q(y)$ cut with weight ξ_{v_0}.

Step 3: Choose any unmarked node $v \in V(y)$ such that all its child nodes are all marked, and mark this node v. If $w_v(y) \leq \sum_{u \in C(v)} \xi_u$, then let $\xi_v := w_v(y)$ and $V_v := \{v\}$, otherwise let $\xi_v := \sum_{u \in C(v)} \xi_u$, and $V_v := \bigcup_{u \in C(v)} V_u$. Return to Step 2.

In Procedure MINCUT, it is not difficult to see that V_v obtained in Step 3 is a minimum v-$Q_v(y)$ cut with its weight ξ_v. Thus $V_{\min}(y)$ obtained in Step 2 is a minimum v_0-$Q(y)$ cut. Moreover as each node is marked once, and the operations in Step 3 can be done in $O(1)$ time, MINCUT runs in $O(|V|)$ time.

Thus the computation time of each iteration of Algorithm OPT-CELP is $O(|V|)$. Since the number of iterations is less than $3|V|$ as obtained in (ii) of subsection 3.3, we conclude that the complexity of Algorithm OPT-CELP is $O(|V|^2)$.

Theorem 5 *Algorithm OPT-CELP solves problem CELP on a tree in $O(|V|^2)$ time.*

Remark: We note that OPT-CELP can be used to solve problem (4) to minimize any convex piecewise linear function. In this case, however, the number of iterations depends on the number of breaking points of all variables. More precisely if the numbers of breaking points of all variables are bounded by B, the complexity becomes $O(B|V|^2)$.

5 Concluding Remarks

In this paper, we considered a new type of network upgrading model. The model divides the communication delay over an edge into three parts; sending time, receiving time and transmission time. Upgrading a node shortens the sending and receiving time of this node, with possible different effects on them. We considered two upgrading strategies; one assumes that the improvement at a node is variable, while the other assumes that the improvement at a node is fixed. These two problems are denoted by CELP and DELP respectively. We showed that CELP

can be solved in polynomial time if the underlying graph is a tree. Note that the eccentricity lowering problems considered in this paper are to minimize the upgrading cost to reduce the eccentricity by a given amount. There is a related version of the eccentricity lowering problems, i.e., to minimize the eccentricity with a given budget. It is straightforward to see that the results in this paper can be extended to such a version.

References

[1] V. Chepoi, H. Noltemeier, and Y. Vaxès, Upgrading trees under diameter and budget constraints, Networks 41 (2003), 24-35.

[2] K.U. Drangmeister, S.O. Krumke, M.V. Marathe, N. Noltemeier, and S.S. Ravi, Modifying edges of a network to obtain short subgraphs, Theoretical Computer Science 203 (1999), 91-121.

[3] M.R. Garey and D.S. Johnson, Computers and intractability: a guide to the theory of NP-completeness, W.H. Freeman and Company, New York, 1979.

[4] T. Ibaraki and N. Katoh, Resource allocation problems: algorithmic approaches, The MIT Press, Cambridge, Massachusetts, 1988.

[5] T. Ibaraki, Y. Vaxès and X. Yang, Lowering eccentricity of a tree by node-upgrading, Networks, to appear.

[6] S.O. Krumke et al., Approximation algorithms for certain network improvement problems, Journal of Combinatorial Optimization 2 (1998), 257-288.

[7] S.O. Krumke et al., Improving spanning trees by upgrading nodes, Theoretical Computer Science 221 (1999), 139-155.

[8] S.O. Krumke et al., Improving minimum cost spanning trees by upgrading nodes, Journal of Algorithms 33 (1999), 92-111.

[9] S.O. Krumke et al., Upgrading bottleneck constrained forests, Discrete Applied Mathematics 108 (2001), 129-142.

[10] D. Paik and S. Sahni, Network upgrading problems, Networks 26 (1995), 45-58.

[11] J.Z. Zhang, X.G. Yang, and M.C. Cai, Inapproximability and a polynomially solvable special case of a network improvement problem, European Journal of Operational Research 155 (2004), 251-257.

[12] J.Z. Zhang, X.G. Yang, and M.C. Cai, A network improvement problem under different norms, Computational Optimization and Applications 27 (2004), 305-319.