Minimizing the number of critical stages for the on-line steiner tree problem

Nicolas Thibault, Christian Laforest
IBISC, Université d'Evry, Tour Evry 2, 523 place des terrasses, 91000 EVRY France

Keywords: on-line algorithm, steiner tree, optimization, rebuilding

Abstract

This paper is devoted to the following *incremental* problem. Initially, a graph and a distinguished subset of vertices, called *initial group*, are given. This group is connected by an initial tree. The incremental part of the input is given by an on-line sequence of vertices of the graph, not yet in the current group, *revealed* on-line one after one. The goal is to connect each new member to the current tree, while satisfying a *quality* constraint: the *weight* of each constructed tree must be at most c times the weight of an optimal steiner tree (with c a given constant). Under this quality constraint, our objective is to minimize the *number of critical stages*. We call "critical" a stage where the inclusion of a new member implies heavy changes in the current tree. Otherwise, the new member is just added by connecting it with a (well chosen) path to the current tree. We propose a strategy leading to at most $\frac{i}{\lfloor 2^{c-\frac{\ln 3}{2}-2}-1\rfloor} \in O(i)$ critical stages (where i is the number of new members and c the constant of the *quality* constraint). We also prove that there exists situations where at least $\frac{i}{2^{\lceil 4c\rceil+1}}-1 \in \Omega(i)$ critical stages are necessary to *any algorithm* to maintain the *quality* constraint. Our strategy is then worst case optimal in order of magnitude for the number of critical stages.

The *Steiner tree* problem, where the goal is to span a set (called *group*) of distinguished vertices (called *members*) with a minimum weight tree, has been extensively studied. As the problem is NP-complete (see [6]), numerous *approximation algorithms* have been designed (see [2, 8] for example). In [13], Waxman was the first to present the on-line version of this problem in which new members are *revealed* one by one (see [4, 5] references on on-line problems). In this first paper, he divides the problem into two categories: the model in which changes in the current tree are not allowed and the model in which changes are allowed. Imaze and Waxman propose in [9] two different strategies corresponding to the two models above. In the first one the tree is just incremented and the degradation of the weight is evaluated, whereas in the second one they allow changes in the current tree to maintain a certain guaranty on the weight. They prove that they construct with the first strategy a tree whose weight is at a logarithmic ratio compared to the optimal one (i.e. the weight of a Steiner tree of the current group), and that they construct with the second strategy a tree whose weight is at a constant ratio compared to the optimal one. They give for the second strategy an upper bound of $O(\sqrt{i})$ on the average number of elementary changes per stage (where i is the number of new members). However, the tree can potentially be changed at each stage; this means that each adding stage is potentially what we call later a *critical stage*. Then, we can divide (as Waxman did in [13]) the other works that have been made since [9] concerning on-line steiner trees. In [1, 3, 14], the model in which no changes are allowed is considered. In [1], the authors give a lower bound of $\Omega(\frac{\log i}{\log\log i})$ for the competitive ratio (i is the number of added members) for the on-line steiner tree problem in the Euclidean plane. In [3], the authors consider the on-line generalized steiner tree problem and they propose an algorithm with a competitive ratio of $O(\log^2 i)$. In [14], linear upper bounds and lower bounds are obtained for the on-line generalized steiner tree and the on-line steiner tree problem on a directed graph. In [7], the model with allowed changes is considered. The aim is to minimize simultaneously the weight of the current tree and the length from a particular node to all the other ones of the tree. The authors propose a method with a competitive ratio of $O(\log i)$ for the weight and constant for the length from the particular node. Note that in [7, 9], only the number of elementary changes is taken into account to measure the level of damage due to the allowed changes in the current tree (i.e. each stage is potentially a *critical stage*).

In our paper we are also concerned by an incremental group problem where the members of the group are revealed on-line one by one. We fix a "relative budget" on the weight of each successive tree, called *quality constraint*, and we propose an algorithm minimizing the number of critical stages necessary to guarantee this budget constraint at each stage. Our work is the first which focus on minimizing the number of critical stages instead of the number of elementary changes (we already consider this parameter in [11, 12], but for with

a different *quality* constraint, offering guarantees on the maximum and average distance between members in the tree instead of guarantees on the weight of the tree). We distinguish critical stages from other stages since they generate a lot of perturbations. Indeed, the communication routes between members already in the current group have to be changed. All the routing tables of the nodes may be modified. This generates a heavy traffic to update them. Moreover the current communications between members initiated before the changes can be interrupted. For these reasons, the number of critical stages must be minimized. Note that it is proved in [9] that *any* on-line algorithm without critical stage *cannot* guaranty a constant quality constraint. That is why we consider here the model in which changes are allowed.

In Section 1, we describe and motivate the constraints (namely the *tree* and *quality* constraints) that must be satisfied at each stage of addition and we give the definition of a *critical stage*. We propose our strategy called OWM (for On-line Weight Minimization) and prove that it satisfies the construction constraints in Section 2. We also prove that our algorithm leads to at most $\frac{i}{\lfloor 2^{c-\frac{\ln 3}{2}-2} \rfloor} \in O(i)$ critical stages (where i is the number of new members and c the constant of the *quality* constraint). In Section 3, we prove that there exists a situation in which at least $\frac{i}{2^{\lceil 4c \rceil+1}} - 1 \in \Omega(i)$ critical stages are necessary for *any* on-line algorithm to satisfy the *quality* constraint. These results show that Algorithm OWM is worst case optimal in order of magnitude for the number of critical stages.

1. Definitions and notations

Let $G = (V, E, w)$ be any connected weighted graph representing a network. V is the set of vertices (modeling the nodes of the network), E the set of edges (modeling the set of physical links) and w a positive weight function of the edges (modeling the length of the edges).

Definition 1 (Optimal Steiner Tree) *Let M be a group of members and let $T = (V_T, E_T, w)$ be a tree spanning M. We denote the* weight *of T by $w(T) = \sum_{e \in E_T} w(e)$ and we denote by $T_{opt}(M)$ an* optimal steiner tree *spanning the group M, i.e. a tree satisfying $w(T_{opt}(M)) = \min\{w(T) : T \text{ spanning } M\}$.*

Construction constraints. In our problem, the graph $G = (V, E, w)$ and an *initial group* $M_0 \subseteq V$ are given (with $M_0 \neq \emptyset$). For example, in a meeting on network (called net-meeting) this initial group M_0 represents the set of participants present from the beginning of the meeting. A structure, denoted as T_0, must be created to connect the *members* of M_0. However, in the case of an open net-meeting for example, new participants can join the meeting. These new participants must be integrated to the current group by connecting them to the current connection structure. We suppose here that these new participants are *not* known in advance and arrive in an on-line way: a new participant, which is any vertex of the graph, is *revealed* when it decides to integrate the group, at any moment.

The *incremental* part of the problem consists in integrating a new member when it is revealed. We call that a *stage of addition*. If S is a sequence of new members revealed, $S = u_1, u_2, \ldots, u_i$, for every k, $1 \leq k \leq i$, we denote as $M_k = M_{k-1} \bigcup \{u_k\}$ the k^{th} group. Thus, starting from the initial connection structure T_0 for M_0, we must integrate, at each addition step k, the new member u_k by updating the current structure T_{k-1} (spanning M_{k-1}) to obtain T_k spanning M_k. Note that, as the members are revealed one by one, we are in an *on-line* model. It means that we do not know the future: neither in which order the members arrive, nor what is the set of new members that will be revealed. Hence, each stage can potentially be the last one; this explains why we are interested by giving guarantees *at each stage*. We are now ready to give the two constraints that each current structure T_k must satisfy.

The *tree* constraint: for every $k \geq 0$, T_k must be a *tree* with all leaves in M_k (we call that a *pruned* tree).

The *quality* constraint: let $c \geq 1$ be any fixed constant representing the *required level of quality*. Then, for every k, we must have $w(T_k) \leq c \cdot w(T_{opt}(M_k))$.

As in a net-meeting the current structure T_k is used to support the communications between members of M_k, the *tree constraint* is set in order to simplify the mechanisms of routing and duplication of information in T_k. Indeed, there is only one route between any pair of members in a tree; moreover as there is no cycle, a simple flooding process can be used to broadcast information from any member. This flooding naturally ends at the leaves that are members (because trees are pruned); there is no need of costly process to control it. The

quality constraint of level c is set to guaranty that the weight of each tree T_k is not too far (up to at most a constant factor of c) from the optimal one.

In the rest of the paper we say that an algorithm solves our problem if, for any on-line sequence $M_0, \ldots, M_i$, it returns a sequence of trees $T_0, \ldots, T_i$ (T_i spanning M_i) satisfying the *tree* and *quality* constraints.

Definition 2 (Critical stage) *Let* A *be any algorithm solving our problem by returning a sequence of trees* $T_0 = (V_0, E_0), \ldots, T_i = (V_i, E_i)$ *satisfying the* tree *and* quality *constraints.*
Stage k *(*$1 \leq k \leq i$*) is a* critical stage *if*

$$E_{k-1} \not\subseteq E_k.$$

We denote by $\sharp CS(T_0, \ldots, T_i)$ *the total* number of critical stages *after* i *added members.*

We recall that we distinguish critical stages from other stages since they generate a lot of perturbations. Indeed, the communication routes between members already in the current group M_{k-1} have to be changed. Potentially all the routing tables of the connecting nodes must be modified. This generates a heavy traffic to update them. Moreover the current communications between members of M_{k-1} initiated before the changes can be interrupted. The number of critical stage must then be minimized. On the other hand, a simple (non critical) connection of the new member by just adding a path in the tree (instead of breaking partially or completely the current tree) generates only local changes. The update of the routing can just be done by broadcasting the identity of the new member in the new tree T_k. This does not create any re-routing between the other members.

2. Our Algorithm OWM

The main idea of Algorithm OWM (for On-line Weight Minimization) is to define particular stage numbers, called *rebuilding stages* (determined in function of the given level of quality c) during which we (totally) reconstruct the current structure by using any algorithm solving the off-line steiner tree problem with approximation ratio a. Then, we choose a vertex v of this a-approximate steiner tree (any vertex), and between two successive *rebuilding stages*, we add each new member by using Imaze and Waxman on-line Algorithm (see [9]), starting from vertex v. Imaze and Waxman Algorithm adds each new member by a shortest path to the current tree. As Imaze and Waxman prove in [9] that their algorithm build a tree whose weight is at most $O(\log j)$ times the weight of an optimal steiner tree (where j is the number of new members added in an on-line way), the resulting spanning structure built by our Algorithm OWM is basically the union of a a-approximate steiner tree spanning a part of the current group and a $O(\log j)$-approximate steiner tree spanning the rest of the current group. By using this two properties, we prove in Section 3 (see Theorem 1) that if we rebuild completely the structure each $\lfloor 2^{c-a-1} - 1 \rfloor$ new added members, then the *quality* constraint is satisfied (with $c \geq a + 1$ the given constant level of the quality constraint and a the approximation ratio of the algorithm used to rebuild the tree off-line at each rebuilding stage). We give our Algorithm OWM in Table 1.

Important remarks: the rebuilding stages correspond to the critical stages of OWM (because the current structure is broken and rebuilt). The other stages are non critical because the algorithm only adds a path to the current structure to connect a new member. Note that OWM is polynomial. Note also that as defined here, OWM does not necessarily build a tree at each stage. Indeed, when a new path is added to the current structure by using Imaze and Waxman Algorithm on a subtree of the current structure, a cycle may be created. This algorithm can be refined in order to obtain at each stage a tree. Nevertheless, due to space limitation, we choose here to avoid this refinement to simplify the presentation. Note that even if the built structure is not a tree (i.e. if it is a structure with heavier weight, by definition of a tree), we prove that it respects the *quality* constraint (see Section 3). As in the refinement, the tree is included in the structure, it also respects the *quality* constraint.

Numerical illustration: if we use the $(1 + \frac{\ln 3}{2})$-approximate algorithm proposed by Robins and Zelikovsky in [10] to rebuild the tree at each rebuilding stage and we set $c = 10$, then we have $\lfloor 2^{c-a-1} - 1 \rfloor = 173$. This means that to satisfy the *quality* constraint with a level $c = 10$, we have to rebuild the current tree each 173 new added members.

Let $G = (V, E, w)$ be a graph and $M_0 \subseteq V$ be the initial group.
At stage 0 :
  Build a tree T_0 spanning M_0 with any a-approximate polynomial time off-line algorithm for the steiner tree problem.
After the last rebuilding stage k :
  Let M_{k+j} be the current group and v_k be any vertex in M_k.
  Let u_{k+j} be the j^{th} member to add since the last rebuilding stage k.
IF $j < \lfloor 2^{c-a-1} - 1 \rfloor$.
  THEN Build T_{k+j} spanning $M_{k+j} = M_{k+j-1} \cup \{u_{k+j}\}$ by using Imaze and Waxman Algorithm, i.e. by adding a shortest path between u_{k+j} and its closest vertex in the subtree of T_{k+j-1} spanning $M_{k+j-1} \backslash (M_k \backslash \{v_k\})$
  ELSE, we have $j = \lfloor 2^{c-a-1} - 1 \rfloor$ (rebuilding stage).
    Break the current tree and build a new tree T_{k+j}, spanning M_{k+j} with any a-approximate polynomial time off-line algorithm for the steiner tree problem. $k + j$ is the new last rebuilding stage.

Table 1: On-line Weight Minimization - OWM

OWM respects the *quality* constraint

The following Theorem shows that OWM respects the *quality* constraint, if the required level of quality is a constant $c \geq a + 1$ (where a is the approximation ratio of the algorithm used to rebuild the tree off-line at each rebuilding stage).

Theorem 1 *For any constant $c \geq a + 1$, for any adding sequence of $i \geq 0$ new members,* OWM *respects the* quality *constraint with level c.*

Proof. Let k be the last rebuilding stage. After stage k, there exists j, $0 \leq j \leq \lfloor 2^{c-a-1} - 1 \rfloor$ such that $i = k + j$. Let $T_i = (V_i, E_i)$ be the subgraph built by algorithm OWM spanning M_i. By definition of OWM, T_i is the result of the union between T_k spanning M_k, built off-line at the last rebuilding stage (see definition of OWM) and the subtree T_i^+ of T_i spanning $M_i^+ = M_i \backslash (M_k \backslash \{v_k\})$, built on-line with the Algorithm of Imaze and Waxman (see definition of OWM). Thus, we have:

$$\begin{aligned} w(T_i) &\leq w(T_k) + w(T_i^+) \leq a \cdot w(T_{opt}(M_k)) + w(T_i^+) \\ &\quad \text{(because, by definition of Algorithm OWM, } T_k \text{ is a } a\text{-approximated tree spanning } M_k\text{)} \\ &\leq a \cdot w(T_{opt}(M_k)) + \lceil \log_2(j+1) \rceil \cdot w(T_{opt}(M_i^+)) \\ &\quad \text{(as } T_i^+ \text{ is a tree spanning } M_i^+ \text{ built by Imaze and Waxman Algorithm, Lemma 3 in [9] holds)} \\ &\leq (a + \lceil \log_2(j+1) \rceil) \cdot w(T_{opt}(M_i)) \leq (a + \lceil c - a - 1 \rceil) \cdot w(T_{opt}(M_i)) \leq c \cdot w(T_{opt}(M_i)) \\ &\quad \text{(because } M_k \subseteq M_i, M_i^+ \subseteq M_i \text{ and } j \leq \lfloor 2^{c-a-1} - 1 \rfloor\text{)} \end{aligned}$$

□

OWM leads to at most $\frac{i}{\lfloor 2^{c-a-1}-1 \rfloor}$ critical stages

Theorem 2 *For any constant $c \geq a + 1$ (representing the required level of quality), for any sequence $M_0 \subset \cdots \subset M_i$ of additions, let $T_0, \ldots, T_i$ be the sequence of trees constructed by* OWM*. We have*

$$\sharp CS(T_0, \ldots, T_i) \leq \frac{i}{\lfloor 2^{c-a-1} - 1 \rfloor}.$$

Proof. By definition of Algorithm OWM, if there are p rebuildings (that are critical stages), we have: $p \cdot (\lfloor 2^{c-a-1} - 1 \rfloor) \leq i < (p+1) \cdot (\lfloor 2^{c-a-1} - 1 \rfloor) \quad \Rightarrow \quad p \leq \frac{i}{\lfloor 2^{c-a-1}-1 \rfloor}$. □

3. Lower bound for the number of critical stages of any algorithm

In this section, we prove that, for *any* on-line algorithm, if the *tree* and *quality* constraints are satisfied, then, for any sufficiently large i, there exists a particular adding sequence leading to $\frac{i}{2^{\lceil 4c \rceil}+1} - 1 \in \Omega(i)$ critical stages. We first define the graph $G_{p,n}$ and the particular sequence of additions.

Definition of Graph $G_{p,n}$. For every $p \geq 1$, we define Graph $G_{p,n}$ made of $n \geq 1$ subgraphs $G_p^0, \ldots, G_p^k, \ldots, G_p^n$. Each subgraph G_p^k is the graph introduced by M. Imase and B. Waxman in Section 3.1 of [9] with every edge of weight 2^k. The subgraphs G_p^k are connected to form $G_{p,n}$ in the following way. For all k, $0 \leq k \leq n$, let u_1^k, u_2^k be the two first members of G_p^k reveled. For every k, $1 \leq k \leq n$, there is an edge of weight $\frac{1}{n}$ between u_2^{k-1} and u_1^k.

Preliminary results. We use an *adaptive adversary*, which chooses new members to add in order to trap *any* on-line algorithm. The adaptive adversary add new members in $G_{p,n}$ in the following order. For all k_1, k_2, $0 \leq k_1 < k_2 \leq n$, the adversary add new members in subgraph $G_p^{k_1}$ before adding new members in subgraph $G_p^{k_2}$. The way members are added in each subgraph G_p^k ($0 \leq k \leq n$) is described in [9]. We first prove that to maintain at each stage a constant level of quality c, *any* algorithm needs to rebuild the current tree (i.e. leads to a *critical stage*) after the adversary has added chosen members of Subgraph G_p^k ($0 \leq k \leq n$). To prove this result (Lemma 2), we use the following result, coming from [9], given here with our notations.

Lemma 1 [9] *For every $p \geq 1$, let N^k be the last group built by the adaptive adversary (described in [9]) in Subgraph G_p^k ($0 \leq k \leq n$) of Graph $G_{p,n}$ described above. Let $T_{opt}(N^k)$ be any optimal steiner tree spanning N^k and let T^k be the tree spanning N^k given by any on-line algorithm. We have:*

$$w(T^k) \geq \left(1 + \frac{1}{2}\left\lfloor \log_2(|N^k| - 1) \right\rfloor\right) \cdot w(T_{opt}(N^k))$$

Lemma 2 *Let $c > 1$ be any constant. Let $p = \lceil 4c \rceil$ and let k, $1 \leq k \leq n$. Let N (resp. N') be the group after all members in subgraphs $G_p^0, \ldots, G_p^k$ (resp. $G_p^0, \ldots, G_p^{k-1}$) to be add have been added by our adaptive adversary. Let $T_{opt}(N)$ be any optimal steiner tree spanning N and let T (resp. T') be the tree spanning N (resp. N') given by any on-line algorithm. If we have $W(T) \leq c \cdot W(T_{opt}(N))$, then*

$$\sharp CS(T', \ldots, T) \geq 1.$$

Proof. We prove Lemma 2 by contradiction. Suppose that there exists an on-line algorithm such that there exists k, $1 \leq k \leq n$ satisfying $W(T) \leq c \cdot W(T_{opt}(N))$ with $\sharp CS(T', \ldots, T) = 0$ (i.e. without critical stage). We first upper bound $W(T_{opt}(N))$ and lower bound $W(T)$. Let $N^0, \ldots, N^k$ respectively be the groups of members added by the adaptive adversary in subgraphs $G_p^0, \ldots, G_p^k$. By structure of graph $G_{p,n}$:

$$\begin{aligned}
w(T_{opt}(N)) &= w(T_{opt}(N^0)) + \tfrac{1}{n} + w(T_{opt}(N^1)) + \tfrac{1}{n} + \cdots + w(T_{opt}(N^{k-1})) + \tfrac{1}{n} + w(T_{opt}(N^k)) \\
&= k \cdot \tfrac{1}{n} + \tfrac{w(T_{opt}(N^k))}{2^k} + \tfrac{w(T_{opt}(N^k))}{2^{k-1}} + \cdots + \tfrac{w(T_{opt}(N^k))}{2^1} + \tfrac{w(T_{opt}(N^k))}{2^0} \\
&\quad \text{(because for every } k, 0 \leq k \leq n, \text{ all edges of subgraph } G_p^k \text{ has weight } 2^k) \\
&= \tfrac{k}{n} + w(T_{opt}(N^k)) \cdot \textstyle\sum_{l=0}^{k} \tfrac{1}{2^k} \;=\; \tfrac{k}{n} + \left(2 - \tfrac{1}{2^k}\right) w(T_{opt}(N^k)) \;\leq\; 2w(T_{opt}(N^k))
\end{aligned}$$

(because as for every k, $0 \leq k \leq n$, all edges of subgraph G_p^k has weight 2^k, $\frac{w(T_{opt}(N^k))}{2^k} \geq 1 \geq \frac{k}{n}$)

Let T^k be the subtree of T spanning N^k. As we also have $w(T) \geq w(T^k)$, we obtain:

$$\begin{aligned}
\tfrac{W(T)}{W(T_{opt}(N))} &\geq \tfrac{w(T^k)}{2w(T_{opt}(N^k))} \geq \tfrac{\left(1 + \frac{1}{2}\left\lfloor \log_2(|N^k|-1) \right\rfloor\right) w(T_{opt}(N^k))}{2w(T_{opt}(N^k))} \quad \text{(by Lemma 1)} \\
&= \tfrac{1}{2} + \tfrac{1}{4}\left\lfloor \log_2(|N^k| - 1) \right\rfloor \;=\; \tfrac{1}{2} + \tfrac{1}{4}\left\lfloor \log_2(2^p + 1 - 1) \right\rfloor \;=\; \tfrac{1}{2} + \tfrac{\lfloor p \rfloor}{4} \;\geq\; \tfrac{1}{2} + c \;>\; c \\
&\quad \text{(because by [9], the adversary adds } 2^p + 1 \text{ members in each subgraph } G_p^k \text{ and because } p = \lceil 4c \rceil)
\end{aligned}$$

This result contradicts $W(T) \leq c \cdot W(T_{opt}(N))$, thus, Lemma 2 is proved by contradiction. □

Main result of the section. The following Theorem shows that if we want a constant level of *quality*, *any* on-line algorithm leads to $\frac{i}{2^{\lceil 4c \rceil}+1} - 1 \in \Omega(i)$ critical stages (where i is the number of added members).

Theorem 3 *Let $c \geq 1$ be any constant. For any on-line algorithm, for every $i \geq 2^{\lceil 4c \rceil} + 1$, there exists a graph G, there exists $M_0 \subset \cdots \subset M_i$, such that if the algorithm returns a sequence of trees $T_0, \ldots, T_i$ (such that for every l, with $0 \leq l \leq i$, T_l spans M_l) respecting the* quality *constraint with level c, we have*

$$\sharp CS(T_0, \ldots, T_i) \ \geq \ \frac{i}{2^{\lceil 4c \rceil}+1} - 1 \ \in \ \Omega(i).$$

Proof. Let c be any constant $c \geq 1$. We set $p = \lceil 4c \rceil$. Let $i \geq 2^p + 1 = 2^{\lceil 4c \rceil} + 1$. Let G be the graph $G_{p,n}$ and $M_0 \subset \cdots \subset M_i$ be the sequence of additions of the adversary defined above. Thus, there exists k, $1 \leq k \leq n$ such that $k \cdot (2^p + 1) \leq i \leq (k+1) \cdot (2^{\lceil 4c \rceil} + 1)$. Thus, we have: $\quad k \geq \frac{i}{2^{\lceil 4c \rceil}+1} - 1 \quad (1)$

For every k', $0 \leq k' \leq k$, let $T^{k'}$ be the subtree of T_i spanning all the members added by the adaptive adversary in Subgraph $G_p^{k'}$.

Then, by Lemma 2, we have: $\begin{cases} \sharp CS(T^0, \ldots, T^1) \ \geq \ 1 \\ \sharp CS(T^1, \ldots, T^2) \ \geq \ 1 \\ \qquad \vdots \\ \sharp CS(T^{k-1}, \ldots, T^k) \ \geq \ 1 \end{cases}$

$\Rightarrow \quad \sharp CS(T^0, \ldots, T^k) \ \geq \ k \quad \Rightarrow \quad \sharp CS(T_0, \ldots, T_i) \ \geq \ k \qquad$ (because $i \geq k \cdot 2^p + 1$)

$\Rightarrow \quad \sharp CS(T_0, \ldots, T_i) \ \geq \ \frac{i}{2^{\lceil 4c \rceil}+1} - 1 \ \in \ \Omega(i) \qquad$ (by (1) and because c is constant)

□

References

[1] N. Alon and Y. Azar. On-line steiner trees in the euclidean plane. In *Symposium on Computational Geometry*, pages 337–343. ACM Press, 1992.

[2] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and approximation*. Springer, 1999.

[3] B. Awerbuch, Y. Azar, and Y. Bartal. On-line generalized steiner problem. In *SODA: ACM-SIAM Symposium on Discrete Algorithms*, pages 68–74, 1996.

[4] A. Borodin and R. El-Yaniv. *Online computation and competitive analysis*. Camb. Univ. press, 1998.

[5] A. Fiat and G. J. Woeginger. *Online algorithmes: The state of the art*. LNCS no. 1442, Springer, 1998.

[6] M. Garey and D. Johnson. Computers and intractability. In *Freeman and compagny, 1979*.

[7] A. Goel and K. Munagala. Extending greedy multicast routing to delay sensitive applications. *Algorithmica*, 33(3):335–352, 2002.

[8] D. Hochbaum. *Approximation algorithms for NP-hard problems*. PWS publishing compagny, 1997.

[9] M. Imase and B.M. Waxman. Dynamic steiner tree problem. *SIAM J. Discr. Math.*, 4(3):369–384, 1991.

[10] G. Robins and A. Zelikovsky. Improved steiner tree approximation in graphs. In *SODA*, pages 770–779. Society for Industrial and Applied Mathematics, 2000.

[11] N. Thibault and C. Laforest. An optimal rebuilding strategy for a decremental tree problem. In *SIROCCO*, LNCS 4056, pages 157–170. Springer, 2006.

[12] N. Thibault and C. Laforest. An optimal rebuilding strategy for an incremental tree problem. In *Journal of Interconnexion Networks*, accepted, 2006.

[13] B. Waxman. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications*, 6(9):1617–1622, 1988.

[14] J. Westbrook and D. Yan. Linear bounds for on-line steiner problems. *Information Processing Letters*, 55(2):59–63, 1995.