Improved Approximations for the k-Hotlink Assignment Problem and for Binary Searching in Trees

Eduardo Laber[1], **Marco Molinaro**[2]

[1]Departamento de Informática - PUC-RIO

[2]Tepper School of Business - Carnegie Mellon University

laber@inf.puc-rio.br, molinaro@cmu.edu

Abstract. *We present a study on two optimization problems in trees: the k-Hotlink Assignment Problem and the problem of Binary Searching in Trees. For the first problem we prove the existence of an FPTAS when $k = 1$, improving upon the constant factor algorithm recently obtained in* [Jacobs, WADS 2007]. *In addition, we develop the first constant factor approximation algorithm for arbitrary k. For the second problem we present a linear time algorithm which is the first to achieve a constant factor approximation. This represents a significant improvement over previous $O(\log n)$-approximation. These results are included in Marco Molinaro's master's dissertation, submitted to and approved by the Departamento de Informática of PUC-RIO in 2008.*

1. Introduction

In this document we describe the problems studied as well as the main results obtained in Marco Molinaro's master's dissertation. In this dissertation, we developed new algorithms and analysis techniques for two optimization problems in trees, namely, the problem of *binary searching in a tree* and the *hotlink assignment problem*. The former is a natural generalization of the problem of searching an element in an ordered set, where the elements have non-uniform probabilities access, and it has applications, among others, in designing asymmetric communication protocols. The latter is a problem that has attracted some attention in the theoretical computer science community in the last decade and consists in reducing the navigation time of hierarchical structures. For both problems we managed to substantially improve upon the state-of-the-art algorithms. We remark that the student played a fundamental role in obtaining all the results presented. The complete dissertation is available at http://bib-di.inf.puc-rio.br/Theses/2008.htm.

2. Searching in Trees

Searching in ordered structures is a fundamental problem in theoretical computer science. In one of its most basic variants, the objective is to find a special element of a totally ordered set by making queries which iteratively narrow the possible locations of the desired element. This can be generalized to searching in more general structures which have only a partial order for their elements instead of a total order [Carmo et al. 2004, Lipman and Abrahams 1995, Ben-Asher et al. 1999, Onak and Parys 2006, Mozes et al. 2008].

In this work, we focus on searching in structures that lay between totally ordered sets and general posets: we wish to efficiently locate a particular node in a tree. More formally, as input we are given a rooted tree $T = (V, E)$ which has a 'hidden' *marked*

node and a function $w : V \to \mathbb{R}^+$ that gives the likelihood of a node being the one marked. For example, T could be modeling a network with one defective unit. In order to discover which node of T is marked, we can perform *edge queries*: after querying the arc (i, j) of T (j being a child of i)[1], we receive an answer stating that either the marked node is a descendant[2] of j (called a **yes** answer) or that the marked node is not a descendant of j (called a **no** answer).

A search strategy is a procedure that decides the next query to be posed based on the outcome of previous queries. As an example, consider the strategy for searching the tree T of Figure 1.a represented by the decision tree D of Figure 1.b. A decision tree can be interpreted as a strategy in the following way: at each step we query the arc indicated by the node of D that we are currently located. In case of a **yes** answer, we move to the right child of the current node and we move to its left child otherwise. We proceed with these operations until the marked node is found. Back to the example in Figure 1, let us assume that 4 is the marked node of T. We start at the root of D and query the arc $(3, 4)$ of T, asking if the marked node is a descendant of node 4 in T. Since the answer is **yes**, we move to the right child of $(3, 4)$ in D and we query the arc $(4, 6)$ of T. In this case, the outcome of the query $(4, 6)$ is **no** and then we move to node $(4, 5)$ of D. By querying this node we conclude that the marked node of T is indeed 4.

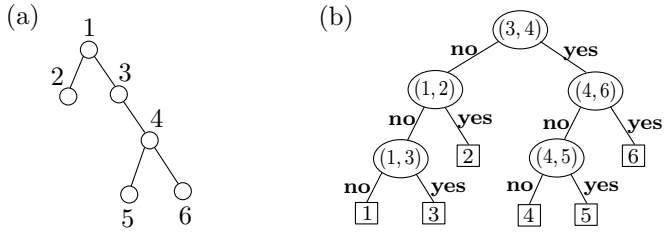

Figure 1. (a) Tree T. (b) Example of a decision tree for T; Internal nodes correspond to arcs of T and leaves to nodes of T.

We define the expected number of queries of a strategy $\mathcal{S}$ as $\sum_{v \in V(T)} s_v w(v)$, where s_v is the number of queries needed by $\mathcal{S}$ to find the marked node when v is the marked node. Therefore, our optimization problem is to find the strategy with minimum expected number of queries.

Besides generalizing a fundamental problem in theoretical computer science, searching in posets (and in particular in trees) also has practical applications in file system synchronization and software testing [Mozes et al. 2008]. We remark that although these applications were considered in the 'worst-case' version of this problem, taking into account the likelihood that the elements are marked (for instance via code complexity measures in the former example) may lead to improved searches.

Apart from the above-mentioned applications, strategies for searching in trees have a potential application in the context of asymmetric communication protocols [Adler et al. 2006, Adler and Maggs 2001]. In this scenario, a client has to send a binary string $x \in \{0, 1\}^t$ to the server, where x is drawn from a probability distribution D that is only available to the server. The asymmetry comes from the fact that the client has

[1]Henceforth, when we refer to the arc (i, j), j is a child of i.

[2]We assume that a node is a descendant of itself.

much larger bandwidth for downloading than for uploading. In order to benefit from this discrepancy, both parties agree on a protocol to exchange bits until the server learns the string x. Such protocol typically aims at minimizing the number of bits sent by the client. In one of the first proposed protocols [Adler and Maggs 2001, Laber and Holanda 2002], at each round the server sends to the client a binary string y and the client replies with a single bit depending on whether y is a prefix of x or not. Based on the client's answer, the server updates his knowledge about x and send another string if he has not learned x yet. It is not difficult to realize that this protocol corresponds to a strategy for searching a marked leaf in a complete binary tree of height t. In fact, the binary strings in $\{0,1\}^t$ can be represented by a complete binary tree of height t where every edge that connects a node to its left(right) child is labeled with 0(1). This generates a 1-1 correspondence between binary strings with length at most t and the edges of the tree and, as a consequence, the message y sent by the server naturally corresponds to an edge query.

Statement of the results. Our main result is a linear-time algorithm that provides the first constant-factor approximation for the problem of searching in trees where the goal is to minimize the expected number of queries.

A crucial observation is that this problem can be efficiently solved when the input tree is a path. This is true because it can be easily reduced to the well-known problem of searching a hidden marked element from a total ordered set U in a sorted list $L \subseteq U$ of elements, where each element of U has a given probability of being the marked one. Motivated by this observation, the algorithm decomposes the input tree into special paths, finds decision trees for each of these paths (with modified weight functions) and combine them into a decision tree for the original tree. In order to design and analyze the algorithm, we use a combination of different techniques which range from classical information theoretical bounds to an approximate sorting procedure.

Related work. Searching in totally ordered sets is a very well-studied problem [Knuth 1998]. In addition, many variants have also been considered, such as when there is information about the likelihood of each element being the one marked [de Prisco and de Santis 1993], or where each query has a different fixed cost and the objective is to find a strategy with least total cost [Knight 1988, Laber et al. 1999, Laber et al. 2001].

The more general version of our problem when the input is a poset instead of a tree was first considered by Lipman and Abrahams [Lipman and Abrahams 1995]. Apart from introducing the problem, they present an optimized exponential time algorithm for solving it. In [Kosaraju et al. 1999], Kosaraju et al. present a greedy $O(\log n)$-approximation algorithm. In fact, their algorithm handles more general searches, see [Laber and Nogueira 2004, Chakaravarthy et al. 2007] for other more general results. To the best of our knowledge, this $O(\log n)$-approximation algorithm is the only available result, with theoretical approximation guarantee, for this 'average-case' version of searching in trees. Therefore, our constant approximation represents a significant advance for this problem[3].

In the 'worst-case' version of this problem, the goal is to find a strategy $\mathcal{S}$ which minimizes $\max_{v \in V(T)} s_v$. This problem was first considered in [Ben-Asher et al. 1999], where the authors devise a dynamic programming based algorithm to solve this problem in $O(n^4 \log^3 n)$ time. Recent advances [Onak and Parys 2006, Mozes et al. 2008] have

[3]It has been recently proved that the problem studied here is NP-hard [Cicalese et al. 2009].

reduced the time complexity to $O(n^3)$ and then $O(n)$. In contrast, the more general version of the worst-case minimization problem where the input is a poset instead of a tree is NP-hard [Carmo et al. 2004].

3. The Hotlink Assignment Problem

The investigation of several algorithmic problems related to search, classification and organization of information that could not be well motivated before WWW age are nowadays central to its good behavior. Among these problems, one that has attracted the attention of the theoretical computer science community is the problem of optimizing user access in web sites. This problem can be addressed in different ways, including increasing the bandwidth of the site, maintaining copies of content in different servers and enhancing the site's navigational structure. Here we are interested in the last of these approaches.

On one hand, the navigational structure of a web site (its pages and its links) is designed to be meaningful and helpful to users. On the other hand, it is not likely that the structure takes into account the fact that some pieces of information are much more sought than others. In fact, it may happen that a very 'popular' piece of information is located much farther from the home page than a 'non-popular' one. Then, a reasonable approach to optimize the access to a web site is enhancing its navigational structure through the addition of a set of shortcuts (hotlinks). Notice, however, that the number of shortcuts added to each page should be small, otherwise they could confuse users and actually disturb the navigation process. This scenario leads to the following algorithmic problem.

Problem Definition. Let $G = (V, E)$ be a DAG with n nodes and a unique root r, and let $w : V \to \mathbb{Q}^+$ be a weight function. A k-assignment A for G is a set of directed arcs that satisfies the following properties: (i) both endpoints of arcs in A belong to V; (ii) for each node $u \in V$ there can be at most k hotlinks of A leaving u.

The cost of a k-assignment A is given by

$$\text{EP}(G, A, w) = \sum_{u \in V} d(r, u, G + A)w(u) \ ,$$

where $d(r, u, G + A)$ is the length (in number of arcs) of the path traversed by a typical user (this will be detailed soon) from r to u in the enhanced graph $G + A = (V, E \cup A)$. An optimal k-assignment A^* is one that minimizes $\text{EP}(G, A, w)$ over all possible k-assignments A. Given a digraph G, with an unique source r, and a weight function $w : V \to \mathbb{Q}^+$, the k-Hotlink Assignment Problem (k-HAP for short) consists of finding an optimal k-assignment for (G, w).

In this paper, we focus on the case where G is a directed tree T and the desired information is always at the leaves of T, that is, $w(u) = 0$ for every node u that is not a leaf of T (the case with positive weights on internal nodes can be modeled via artificial leaves). As for the definition of distance $d(\cdot)$, the cost spent by a typical user to find his (her) target information is directly related to how he (she) navigates on the site. Two models of navigation have been considered in the literature: the clairvoyant user model and the greedy user model. The former is somewhat unrealistic since it assumes that users have a map of the entire web site and follow a shortest path from the root to their target information. The greedy user model assumes that a user follows the link

(original link or hotlink) that leads him (her) closest *in the original tree* T to his (her) target information. Figure 2.b illustrates the choice of links made by greedy users. For instance, users that want to reach the information represented by the node x traverse the path $(r \to b \to c \to x)$. Notice that users following this greedy strategy do not reach x in the shortest way possible; a shorter one corresponds to following the path $(r \to a \to x)$. Like most of the papers in this subject, we consider the greedy user model.

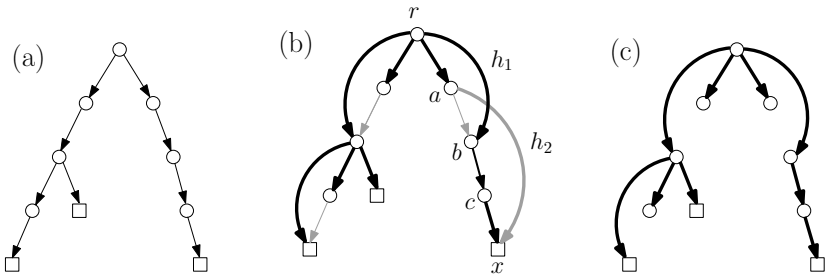

Figure 2. (a) Original tree. (b) Enhanced tree, with greedy paths in bold. Hotlinks h_1 and h_2 are crossing. (c) Tree induced by user paths.

Related Work. The idea of hotlinks was first suggested in [Perkowitz and Etzioni 1997]. In [Czyzowicz et al. 2003], the authors present experimental results showing the validity of the hotlink approach. In addition, they describe a software tool to automatically assign hotlinks to web sites. Experimental results also appear in [Pessoa et al. 2004, Jacobs 2008a].

Turning our attention to theoretical results, in [Bose et al. 2000] Bose et al. prove that the 1-Hotlink Assignment Problem is NP-Complete for DAG's in the clairvoyant user model. Recently, Jacobs [Jacobs 2008b] proved that 1-HAP is NP-Complete for trees in the greedy model.

In [Bose et al. 2000], Shannon's coding theorem is employed to provide the following lower bound: given a tree T and a normalized weight function w, the inequality

$$\text{EP}(T, A, w) \geq \frac{H(w)}{\log(\Delta + 1)}$$

holds for every 1-assignment A for T, where Δ is the maximum outdegree of T and $H(w) = -\sum_{u \in T} w(u) \log w(u)$ is the entropy induced by w.

In [Kranakis et al. 2004], Kranakis et al. present a quadratic time algorithm that produces a 1-assignment A such that $\text{EP}(T, A, w) \leq \frac{H(w)\Delta}{\log \Delta}$ (for large Δ). In [Douïeb and Langerman 2005] and [Douïeb and Langerman 2006], the authors present algorithms that construct 1-assignments whose associated costs are $O(H(w))$. This upper bound together with the above entropy lower bound guarantee that these methods provide a $O(\log n)$ approximation for the 1-HAP. In [Douïeb and Langerman 2006], it is also presented a way to construct a k-assignment with cost $O(H(w)/\log k)$. The first algorithm with constant approximation ratio for the 1-HAP is due to Jacobs [Jacobs 2007] – it runs in $O(n^4)$ and achieves a 2-approximation. In this same paper, Jacobs mentions that it is not clear how to extend his method to guarantee a constant approximation for the k-HAP.

Exact algorithms for the 1-HAP were independently discovered by Gerstel at. al. [Gerstel et al. 2003] and Pessoa et al. [Pessoa et al. 2004] (see also [Kutten et al. 2007]

for a journal version merging both papers). The algorithm of [Gerstel et al. 2003] is exponential in the height of the input tree. Now notice that the paths that users take to reach the desired information induce a tree on $T + A$ (see Figure 2.c). We denote such tree by T^A. The algorithm of [Pessoa et al. 2004], which can be viewed as an optimized version of the one proposed in [Gerstel et al. 2003], has the following property: for each integer D, it calculates in $O(n3^D)$ time the best 1-assignment among the 1-assignments A such that the height of T^A is at most D.

Statement of the Results. First, we show the existence of a FPTAS for the 1-HAP. [4] In order to obtain this results, we first prove that for any tree T with n nodes and for any weight function w, there is an optimal assignment A^* for (T, w) such that the height of T^{A^*} is $O(\log w(T) + \log n)$, where $w(T) = \sum_{u \in T} w(u)$. Once this result is proved, a pseudo-polynomial time algorithm for the 1-HAP can be obtained by executing the algorithm of [Pessoa et al. 2004], mentioned in the previous section, with $D = c(\log w(T) + \log n)$ for a suitable constant c. Then, we scale the weights w in a fairly standard way to obtain the FPTAS. The difficult part in deriving our FPTAS is proving the bound on the height of T^{A^*}– it requires the combination of different kinds of tree decompositions with a non-trivial transformation of an optimal tree.

Our second contribution is a 16-approximation for the k-HAP, which is the first algorithm with constant-factor approximation for this problem. Moreover, it can be implemented to run in $O(n \log n)$ time. It is worth mentioning that our algorithm coincides with the one proposed by Douieb and Langerman [Douïeb and Langerman 2005] for the particular case where $k = 1$. Thus, our analysis here shows that their algorithm provides a constant approximation for the 1-HAP (this was not known before). Although other algorithms with constant approximation do exist for the 1-HAP, the one by Douieb and Langerman has the following advantages: it can be implemented in linear time and it can be dynamized to handle insertions and deletion in logarithmic time.

Our algorithm relies on the decomposition of the input tree into heavy subtrees of maximum degree k. The key idea in the analysis is a novel lower bound on the cost of an optimal k-assignment which is much stronger than the entropy-based one given in [Bose et al. 2000] – roughly speaking, our lower bound is given by a sum of entropy-like functions associated with the trees obtained due to our decomposition.

4. Final Remarks

The results obtained in this dissertation were accepted by first-rate international conferences and journals:

- *An Approximation Algorithm for Binary Searching in Trees.*
  Proceedings ICALP 2008
  Eduardo Laber and Marco Molinaro
- *An Approximation Algorithm for Binary Searching in Trees. (extended version)*
  Algorithmica (to appear)
  Eduardo Laber and Marco Molinaro
- *Improved Approximations for the Hotlink Assignment Problem.*
  ACM Transactions on Algorithms (to appear)
  Eduardo Laber and Marco Molinaro

[4] If a minimization problem admits an FPTAS then for each $\epsilon > 0$ it is possible to obtain a solution costing at most $(1 + \epsilon)$ times the optimal one in $polytime(n, 1/\epsilon)$.

Finally, we would like to mention that after completing the dissertation, we still invested some energy to address some of the problems that remained open. We managed to prove that the problem of searching in trees is NP-Complete and we devised an FPTAS for it [Cicalese et al. 2009].

References

Adler, M., Demaine, E., Harvey, N., and Patrascu, M. (2006). Lower bounds for asymmetric communication channels and distributed source coding. In *SODA*, pages 251–260.

Adler, M. and Maggs, B. (2001). Protocols for asymmetric communication channels. *Journal of Computer and System Sciences*, 63(4):573–596.

Ben-Asher, Y., Farchi, E., and Newman, I. (1999). Optimal search in trees. *SIAM Journal on Computing*, 28(6):2090–2102.

Bose, P., Kranakis, E., Krizanc, D., Martin, M., Czyzowicz, J., Pelc, A., and Gasieniec, L. (2000). Strategies for hotlink assignments. In *ISAAC*, pages 23–34.

Carmo, R., Donadelli, J., Kohayakawa, Y., and Laber, E. (2004). Searching in random partially ordered sets. *Theoretical Computer Science*, 321(1):41–57.

Chakaravarthy, V., Pandit, V., Roy, S., Awasthi, P., and Mohania, M. (2007). Decision trees for entity identification: Approximation algorithms and hardness results. In *PODS*, pages 53–62.

Cicalese, F., Laber, E., and Molinaro, M. (2009). On the complexity of searching in trees: average-case minimization (in preparation).

Czyzowicz, J., Kranakis, E., Krizanc, D., Pelc, A., and Martin, M. (2003). Enhancing hyperlink structure for improving web performance. *Journal of Web Engineering*, 1(2):93–127.

de Prisco, R. and de Santis, A. (1993). On binary search trees. *Information Processing Letters*, 45(5):249–253.

Douïeb, K. and Langerman, S. (2005). Dynamic hotlinks. In *WADS*, pages 182–194.

Douïeb, K. and Langerman, S. (2006). Near-entropy hotlink assignments. In *ESA*, pages 292–303.

Gerstel, O., Kutten, S., Matichin, R., and Peleg, D. (2003). Hotlink enhancement algorithms for web directories: (extended abstract). In *ISAAC*, pages 68–77.

Jacobs, T. (2007). Constant factor approximations for the hotlink assignment problem. In *WADS*, pages 188–200.

Jacobs, T. (2008a). An experimental study of recent hotlink assignment algorithms. In *ALENEX*, pages 142–151.

Jacobs, T. (2008b). On the complexity of optimal hotlink assignment. In *ESA*, pages 540–552.

Knight, W. (1988). Search in an ordered array having variable probe cost. *SIAM Journal on Computing*, 17(6):1203–1214.

Knuth, D. (1998). *The art of computer programming, volume 3: sorting and searching*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA.

Kosaraju, R., Przytycka, T., and Borgstrom, R. (1999). On an optimal split tree problem. In *WADS*, pages 157–168.

Kranakis, E., Krizanc, D., and Shende, S. M. (2004). Approximate hotlink assignment. *Information Processing Letters*, 90(3):121–128.

Kutten, S., Gerstel, O., Laber, E., Matichin, R., Peleg, D., Pessoa, A., and Souza, C. (2007). Reducing human interactions in web directory searches. *ACM Transactions on Information Systems*, (25).

Laber, E. and Holanda, L. (2002). Improved bounds for asymmetric communication protocols. *Information Processing Letters*, 83(4):205–209.

Laber, E., Milidiú, R., and Pessoa, A. (1999). Strategies for searching with different access costs. In *ESA*, pages 236–247.

Laber, E., Milidiú, R., and Pessoa, A. (2001). On binary searching with non-uniform costs. In *SODA*, pages 855–864.

Laber, E. and Nogueira, L. (2004). On the hardness of the minimum height decision tree problem. *Discrete Applied Mathematics*, 144(1-2):209–212.

Lipman, M. and Abrahams, J. (1995). Minimum average cost testing for partially ordered components. *IEEE Transactions on Information Theory*, 41(1):287–291.

Mozes, S., Onak, K., and Weimann, O. (2008). Finding an optimal tree searching strategy in linear time. In *SODA*, pages 1096–1105.

Onak, K. and Parys, P. (2006). Generalization of binary search: Searching in trees and forest-like partial orders. In *FOCS*, pages 379–388.

Perkowitz, M. and Etzioni, O. (1997). Adaptive web sites: an AI challenge. In *IJCAI*, pages 16–23.

Pessoa, A., Laber, E., and Souza, C. (2004). Efficient implementation of hotlink assignment algorithms for web sites. In *ALENEX*, pages 79–87.