# Strategies for Hotlink Assignments
(Extended Abstract)

Prosenjit Bose[*‡] Jurek Czyzowicz[§‡] Leszek Gąsieniec[‖]
Evangelos Kranakis[*‡] Danny Krizanc[*‡] Andrzej Pelc[§‡]
Miguel Vargas Martin[*¶]

**Abstract**

Consider a DAG (directed acyclic graph) $G = (V, E)$ representing a collection $V$ of web pages connected via links $E$. All web pages can be reached from the home page, represented by a certain source node $s$ of $G$. Each web page carries a weight representative of the frequency with which it is visited. By adding hotlinks, at most one per page, we are interested in minimizing the expected number of steps needed to visit from the home page a selected set of web pages. For arbitrary DAGs we show that the problem is NP-complete. We also give algorithms for assigning hotlinks, as well as upper and lower bounds on the expected number of steps to reach the leaves from the home page $s$ located at the root of the tree. Depending on the probability distribution (arbitrary, uniform, Zipf) the expected number of steps is at most $c \cdot n$, where $c$ is a constant less than 1. For the geometric distribution we show how to obtain a constant average number of steps.

[§]Dept. Informatique, Univ. du Québec à Hull, Hull, Québ ec J8X 3X7, Canada. e-mail: {czyzowic,pelc}@uqah.uquebec.ca

[*]Carleton University, School of Computer Science, Ottawa, ON, K1S 5B6, Canada. e-mail: {jit,kranakis,krizanc,mvargas}@scs.carleton.ca

[‡]Research supported in part by NSERC (Natural Sciences and Engineering Research Council of Canada) grants.

[‖]Department of Computer Science, University of Liverpool, Peach Street, L69 7ZF, Liverpool, UK. e-mail: leszek@csc.liv.ac.uk

[¶]Research supported in part by MITACS (Mathematics of Information Technology and Complex Systems) and a CONACYT fellowship from the Mexican government.

# 1 Introduction

Two important ways to improve internet performance include "understanding its topological structure" and "creating adaptive designs".

In the first approach, by understanding the network structure we hope to improve network performance, but in general this is a very complex task [6, 2, 9, 3]. The required mathematical analysis is usually related to the behavior of some type of random graphs [4].

In the second approach we ignore topological considerations for the sake of improving the design of the web page. This usually involves the creation of adaptive web sites that continuously simplify web design and improve performance [12, 13]. Thus, for example, learning statistical patterns (like frequency of visits) is especially important for improving the organization of web pages. There are several experimental adaptive systems, e.g., [7].

In this paper we consider an intermediate approach and show how to improve web design by taking advantage of the knowledge of the network topology. Consider a DAG (directed acyclic graph) $G = (V, E)$ representing a collection $V$ of web pages connected via links $E$. All web pages can be reached from the home page, represented by a certain source node $s$ of $G$. Each web page carries a weight representative of the frequency with which it is visited. By adding hotlinks, at most one per page, we are interested in minimizing the expected number of steps needed to visit from the home page a selected set of web pages. In the case of trees, this webpage is found by searching through a "tree" of nodes each including a set of possible choices. Thus the leaves of the tree are "sought web sites" that we want to link to the root of the tree with hotlinks. A naive way to do this is to add a hotlink from the root for each leaf. This is clearly unreasonable, because it will exceed the space capacity of the web page at the root, if the tree is large. An alternative approach is for a given integer $k$ to connect the source node $s$ with $k$ hotlinks to $k$ specially selected nodes so as to maximize the gain. This approach is considered in [5] and is certainly limited by the desired size $k$ of the web page at the source node.

A more practical approach is to distribute hotlinks efficiently over the nodes of the tree so as to minimize the average number of hops required to visit the web sites. The question arising is

> How do we distribute the hotlinks so as to minimize the expected number of steps from the root needed to reach a web page located at a leaf?

We consider collections of web pages organized as full binary trees and allow at most one hotlink going from a node $v$ to some other node of the subtree rooted at $v$. Web sites situated at leaves of the tree are visited with a certain frequency depending on their "popularity". This yields a probability distribution on the leaves of the tree. The hotlink assignment chosen depends on the frequency with which web pages are visited, i.e.,on this probability distribution.*

*We note that a similar problem occurs with Integrated Voice Registers (also called IVRs) used by

## 1.1 Results of the paper

For arbitrary DAGs we show that the problem is NP-complete. The table below gives upper and lower bounds on the expected number of steps to reach a web page located at a leaf of the full binary tree from the root, which are derived from the results of our paper.

| Ditribution | Upper Bound | Lower Bound |
|---|---|---|
| **Uniform** | $(3n+1)/4$ | $(3n+1)/4$ |
| **Geometric** | $O(1)$ | $\Omega(1)$ |
| **Arbitrary** | $(3n+1)/4$ | $H(p)/\log 3$ |
| **Zipf** | $(3n+1)/4$ | $n/(2\log 3)$ |
| **Zipf** (Sorted) | $n/3+O(\sqrt{n}\log n)$ | $n/(2\log 3)$ |

Thus, for the sorted Zipf distribution (i.e. the $i$-th leaf of the tree is assigned probability $\frac{1}{iH_N}$) we can prove the stronger upper bound $n/3+O(\sqrt{n}\log n)$. Here $H(p)$ is the Entropy (see [1]) of the probability distribution $p =< p_i : i = 1,\ldots,N >$, which is defined by the formula†

$$H(p) = \sum_{i=1}^{N} p_i \log(1/p_i). \tag{1}$$

Note that the lower bounds are valid under any permutation of the distribution on the leaves of the tree.

## 1.2 Notation and preliminaries

In this paper we consider full binary trees with $n$ levels (not counting the root), $N = 2^n$ leaves, and a total of $2^{n+1}-1$ nodes. Without hotlinks a leaf can be reached from the root in $n$ steps, which is also the height of the tree. In the sequel we study the following distributions.

- **Arbitrary:** $p_1 + p_2 + \cdots + p_N = 1$.
- **Uniform:** $p_1 = p_2 = \cdots = p_N = 1/N$,
- **Geometric:** $p_i = a^{-i}$, for $i = 1,\ldots,N-1$, and $p_N = 2-(1-a^{-N})/(1-a^{-1})$, where $a > 1$.
- **Zipf:** $p_i = \frac{1}{iH_N}$, where $H_N = \sum_{i=1}^{N} 1/i$ is the harmonic number.‡

service providers, phone companies, government agencies, etc. Here the desired piece of information is located at a leaf of a tree. We want to find an assignment of "hyperlinks" that will improve the overall expected number of steps.

†Unless otherwise indicated, throughout this paper all logarithms are in base 2.

‡This distribution was popularized by Zipf [14] and is related to the "80-20" rule of thumb stating that 80 percent of transactions are dealing with the most active 20 percent of a list of items [11]

We give algorithms for assigning hotlinks to the nodes of the full binary tree. A hotlink is an additional directed edge from any node of the tree to a descendant node. We are restricting ourselves to the case where at most one hotlink can be added per node.

Let $r$ be the root (i.e., home page) of the tree. For any node $u$ let $d(u)$ be the distance of $u$ from the root. Let $T^A$ be the graph resulting from an assignment $A$ of hotlinks. The expected number of steps from the root to find a web page is defined by

$$E[T^A, p] = \sum_{x \text{ is a leaf}} d_A(x) p_x,$$

where $d_A(x)$ is the distance of the node $x$ from the root in $T^A$, and $p =< p_x : x = 1, \ldots, N >$ is the probability distribution on the leaves of the original tree $T$. In order to estimate the value of a given set $A$ of hotlinks we associate with $A$ a gain function $g(A)$ such that

$$g(A) = E[T, p] - E[T^A, p] = \sum_{x \text{ is a leaf}} (d(x) - d_A(x)) p_x, \tag{2}$$

i.e., $g(A)$ is the difference between the expected number of steps to find a web page in the original tree $T$ and in the graph $T^A$. A set of hotlinks is optimal if it offers the greatest gain over all sets of hotlinks, where at most one hotlink can be added per node. We will sometimes refer to $g(\{h\})$ as the gain of a single hotlink $h$. Observe that for a hotlink $h$ joinning vertex $u$ to vertex $v$ the gain $g(\{h\})$ comes from the descendants of $v$ and in fact $g(\{h\}) = (d(u,v) - 1)q_v$, where $d(u,v)$ is the distance from $u$ to $v$ and $q_v$ the weight at $v$, i.e., the sum of the probabilities of the leaves descendant to $v$.

The reader may note that for balanced distributions short hotlinks offer better gain than long ones, while for unbalanced distributions long hyperlinks are better. For example, for the uniform distribution and a hotlink $h$ joinning vertex $v$ to its descendant at distance $d$ we have that $g(\{h\})$, obtains its maximum value for $d = 2$ or $d = 3$ (see Section 3.3). On the other hand for the degenerate distribution $p_1 = 1, p_2 = p_3 = \cdots = p_N = 0$ the hotlink from the root to the leaf of weight $p_1$ leads to the greatest gain.

It will often be necessary to choose the hotlinks in such a way that the gain provided by one of them does not interfere with the gain obtained from another one. Two hotlinks $h_1, h_2$ are called independent if $g(\{h_1, h_2\}) = g(\{h_1\}) + g(\{h_2\})$. The proof of the following lemma is easy.

**Lemma 1.1** *Let $h_1, h_2$ be two hotlinks joinning vertices $v_1$ to $w_1$ and $v_2$ to $w_2$, respectively. Then $h_1$ and $h_2$ are not independent if and only if either "$w_1$ is a descendant of $w_2$ and $w_2$ a descendant of $v_1$" or "$w_2$ is a descendant of $w_1$ and $w_1$ a descendant of $v_2$".* ■

It is useful to assume that the probability distribution $p$ is extended to the internal nodes of the tree so that the weight at a node equals the sum of the probabilities of its descendant leaves. Moreover, it is easy to see that these weights can be computed in $O(N)$ time.

In the sequel we assume the leaves of the tree are numbered $1, 2, \ldots, N$ from left to right and for some permutation $\pi$ of the set $1, 2, \ldots, N$ the $i$-th node is assigned probability $p_{\pi(i)}$, for each $i = 1, 2, \ldots, N$. If $\pi$ is the identity permutation and $p_1 \geq p_2 \geq \cdots \geq p_N$ then we say that the distribution is sorted.

# 2 Arbitrary Graphs

In this section we prove that the problem of optimal assignment of hotlinks in arbitrary directed graphs is NP-hard, even for uniform distributions. More precisely, consider the following optimization problem.

**HOTLINK ASSIGNMENT**

**Instance:** Directed graph $G = (V, E)$, a source node $s \in V$ which can reach every node of the graph, and positive integer $g$.

**Question:** Does there exist a hotlink assignment from the source node $s$ for which the average length of a path from $s$ to the nodes of $G$ is reduced by at least $g$?

**Theorem 1** *The problem HOTLINK ASSIGNMENT is NP-hard.*

Proof The transformation is from the following NP-complete decision problem [8]:

**EXACT COVER BY 3-SETS (X3C)**

**Instance:** Set $S$ with $|S| = 3k$ and a collection $C$ of 3-element subsets of $S$.

**Question:** Does $C$ contain an exact cover for $S$, i.e., a subset $C' \subseteq C$ such that every element of $S$ belongs to exactly one member of $C'$?

Given an instance $C, S$ of X3C we construct an instance $G, g$ of HOTLINK ASSIGNMENT as follows. Let the set $S = \{s_1, s_2, \ldots, s_m\}$ be of size $m = 3k$, and the set $C$ of size $t$. The graph $G = (V, E)$ is defined as follows.

- The vertex set $V$ consists of the vertices below:
  1. $A_0$ is the source node of the graph,
  2. $A_1, \ldots, A_{k-1}$. $B_1, B_2, \ldots, B_{k-1}$, and $C_1, C_2, \ldots, C_t$,
  3. $s_{i,1}, s_{i,2}, \ldots, s_{i,k}$, for $i = 1, 2, \ldots, m$, are $m$ rows of vertices each row of size $k$.
- The edge set $E$ consists of the directed edges bekow:
  1. $(A_0, A_i)$, and $(A_i, B_i)$, for all $i = 1, \ldots, k-1$,
  2. $(B_i, C_j)$, for all $i = 1, \ldots, k-1$, and $j = 1, 2, \ldots, t$,
  3. if $s_i \in C_j$ then there exist directed edges $(C_j, s_{i,1}), (C_j, s_{i,2}), \ldots, (C_j, s_{i,k})$.

The directed graph $G$ is depicted in Figure 1. We can prove the following lemma.

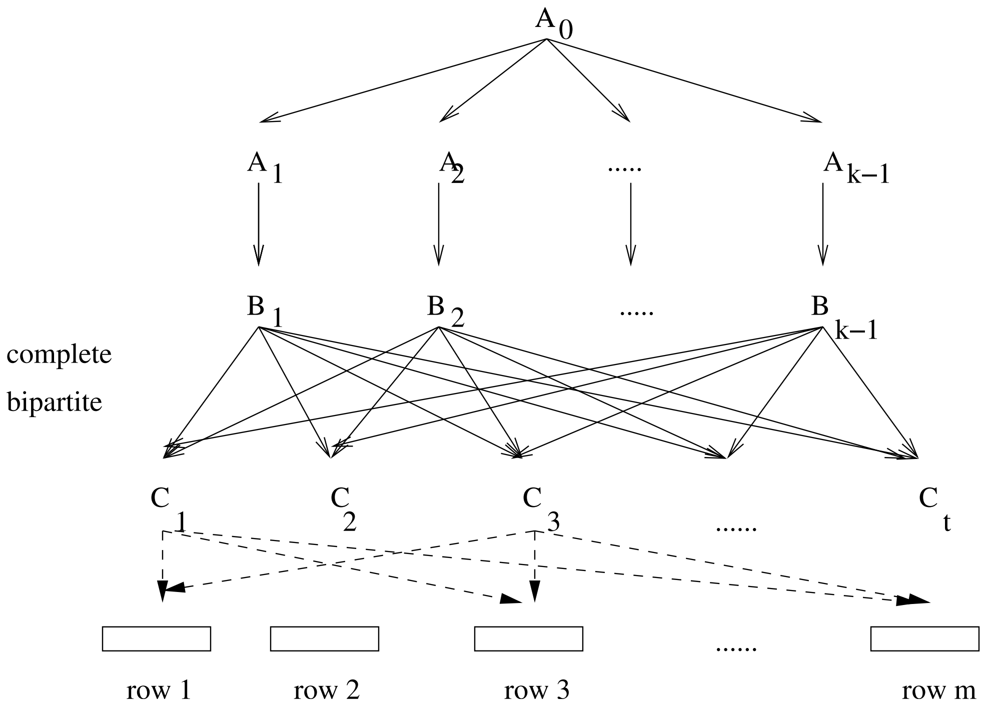


Figure 1: An instance of HOTLINK ASSIGNMENT. The $i$-th rectangular box at the bottom row represents the row $s_{i,1}, s_{i,2}, \ldots, s_{i.k}$, for $i = 1.2, \ldots, m$. If $s_i \in C_j$ then there is a directed edge from $C_j$ to each of the verices of the $i$-th row.

**Lemma 2.1** *There is a set cover of size at most $k$ if and only if the directed graph $G$ has a hotlink assignment which attains a gain of at least $mk + 3k$.*

Clearly the lemma implies the theorem. To prove the lemma it is enough to prove the following two claims.
**Claim 1:** If there is an exact cover of $S$ by 3-sets then there is a hotlink assignment for the graph $G$ whose gain is at least $mk + 3k$.
Indeed, let $C_{i_0}, C_{i_1}, \ldots, C_{i_{k-1}}$ be such a set cover of the set $S$. Consider the hotlinks:

$$A_0 \to C_{i_0}, A_1 \to C_{i_1}, \ldots, A_{k-1} \to C_{i_{k-1}}.$$

The gain resulting from the hotlink $A_0 \to C_{i_0}$ is $2(3k)$ and the gain resulting from the remaing $k-1$ hotlinks is $(m-1)3k$. Hence the total gain is $mk + 3k$.
**Claim 2:** If there is no set cover of size $k$ then the gain from any hotlink assignment on the graph $G$ is less than $mk + 3k$.
Indeed, consider any hotlink assignment of $G$. On the one hand, if this hotlink assignment is using a hotlink from $A_0$ to some vertex $B_i$, for some $i = 1, 2, \ldots, k-1$, then the maximum gain that can be attained is less than $mk + k - 1 + 1 \leq ml + 3k$. On the other hand, if this hotlink assignment is never using a hotlink from $A_0$ to any vertex $B_i$, for any $i = 1, 2, \ldots, k-1$, then again the maximum gain that can be attained is less than $(m-1)k + 3k + k - 1 + 1 = mk + 3k$. This completes the proof of the lemma and hence also of the theorem. ∎

We conclude this section with a useful observation that shows that the hotlink assignment problem on arbitrary graphs is reducible to the hotlink assignment problem on trees. Observe that for a given hotlink assignment from a source node to several destination nodes in a general graph, we can "selectively drop edges" without altering the average length of a shortest path from the source node to the destinations in such a way that the resulting undelying graph is a tree rooted at the source. More precisely we have the following result.

**Lemma 2.2** *Let $G = (V, E)$ be a digraph with real weights associated to its vertices and some designated source vertex $s$ such that every vertex of $G$ is reachable from $s$. Then there exists a subgraph $G' = (V, E')$ such that $E' \subseteq E$ and $G'$ is a tree and the average shortest path from $s$ is the same for $G$ and $G'$.*

Proof We prove by contradiction that a subgraph $G'$ of $G$ with minimal number of edges and the same expected length must be a tree. Note that no incoming edge of $s$ can be in $G'$. If $g'$ is not a tree and the indegree of $s$ is 0, some vertex $v$ must have at least two incoming edges $e_1, e_2 \in E'$. We can discard at least one of these, thus contradicting the minimality of $G'$. Indeed, if the shortest path tree from $s$ to $v$ uses, say, edge $e_1$, then edge $e_2$ is never used in the shortest path from $s$ to $v$ or any of its descendants. ∎

# 3 Full Binary Trees

In this section we consider the hotlink assignment problem on full binary trees.

## 3.1 Geometric distribution

We can assign hotlinks by a simple bottom-up heuristic. The idea is to sort the nodes by weight of the probability and assign to a leaf a source node not assigned before which is the furthest from it so that the node is in the subtree rooted at the source node. This simple heuristic will assign at most one leaf per node and can be efficient if the probability distribution is heavily weighted towards a few nodes. The algorithm we consider is the following.

**Bottom-up Strategy:**
1. Sort the $N$ leaves by the size of the probability distribution, i.e., if $u$ precedes $v$ then $p_u \leq p_v$.
2. Assign the highest probability leaf to the root.
3. Assign the remaining leaves recursively as follows. Consider an unassigned leaf, say $l$, of highest probability. Find an internal yet unassigned node $u$ such that $l$ is a leaf to the subtree rooted at $u$ and the distance from $u$ to $l$ is maximized. Assign $l$ to $u$.

A simple analysis of the performance of the algorithm is as follows. Let $u_1, u_2, \ldots, u_{n-1}$ be the $n-1$ leaves of the 1st, 2nd,..., $(n-1)$st highest probability in the given probability distribution $p$. It is easy to see that the resulting gain

$$\begin{aligned} &\geq (n-1)p_{u_1} + (n-2)p_{u_2} + \cdots + (n-(n-1))p_{u_{n-1}} \\ &= n\sum_{i=1}^{n-1} p_{u_i} - \sum_{i=1}^{n-1} ip_{u_i}. \end{aligned}$$

In particular, the expected number of steps satisfies

$$\begin{aligned} E[T^A, p] &\leq n - n\sum_{i=1}^{n-1} p_{u_i} + \sum_{i=1}^{n-1} ip_{u_i} \\ &= n\left(1 - \sum_{i=1}^{n-1} p_{u_i}\right) + \sum_{i=1}^{n-1} ip_{u_i} \end{aligned} \tag{3}$$

Inequality (3) can already provide good bounds for probability distributions which are heavily weighted on "a few" nodes. In particular, we have the following result.

**Theorem 2** *Consider a rooted full binary tree on $n$ levels. There is an algorithm linear in the number of vertices of the tree which for the geometric probability distribution on the leaves of the tree assigns at most one hotlink per node in such way that the expected number of steps to reach a leaf of the tree is $O(1)$.*

Proof This follows easily from Inequality (3). In particular, we have that

$$\begin{aligned} E[T^A, p] &\leq n\left(1 - \sum_{i=1}^{n-1} p_{u_i}\right) + \sum_{i=1}^{n-1} ip_{u_i} \\ &= n\left(1 - \sum_{i=1}^{n-1} 2^{-i}\right) + \sum_{i=1}^{n-1} i2^{-i} \\ &= O(1) \end{aligned}$$

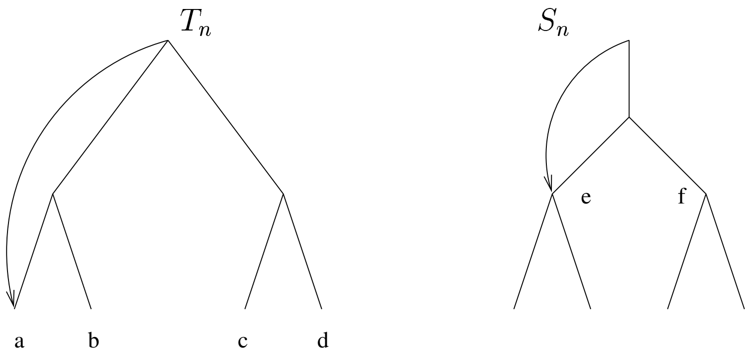


Figure 2: Assignment of hotlinks to subtrees. The leftmost tree is $T_n$ and the rightmost tree is $S_n$.

This completes the proof of the theorem. ■

A similar argument works for the general geometric distribution. Details are left to the reader.

## 3.2 Arbitrary distributions

In this section we give an algorithm for arbitrary distributions.

**Theorem 3** *Consider a rooted full binary tree on $n$ levels. There is an algorithm linear in the number of vertices of the tree which for any probability distribution on the leaves of the tree assigns a hotlink per node in such a way that the expected number of steps to reach a leaf of the tree is* $(3n + 1)/4$.

PROOF As mentioned before in $O(N)$ time we can propagate the original weights on the leaves of the tree through the entire tree using a bottom-up process. Once all these internal node weights are assigned we use a top-down method to assign hotlinks. Each hotlink is of length two, i.e., each time from level $i \geq n - 2$ to level $i + 2$. The root is assigned the hotlink to the level two node of heighest weight. By symmetry we can suppose that this hotlink is to the leftmost descendant at distance two from the root (see tree $T_n$ in Figure 2).

The assignment of hotlinks is done recursively. The recursive process assigns hotlinks to the two subtrees $T_n$ and $S_n$ as depicted in Figure 2. Let $a, b, c, d$ be the probabilities at the second level of the tree $T_n$ and $e, f$ the probabilities at the second level of the tree $S_n$. Without loss of generality assume that

$$a \geq b, c, d \text{ and } e \geq f,$$

We select the hotlinks from the root to the node with highest probability at the second level of $T_n$ and $S_n$, respectively. This leads to the hotlink assignment depicted in Figure 2.

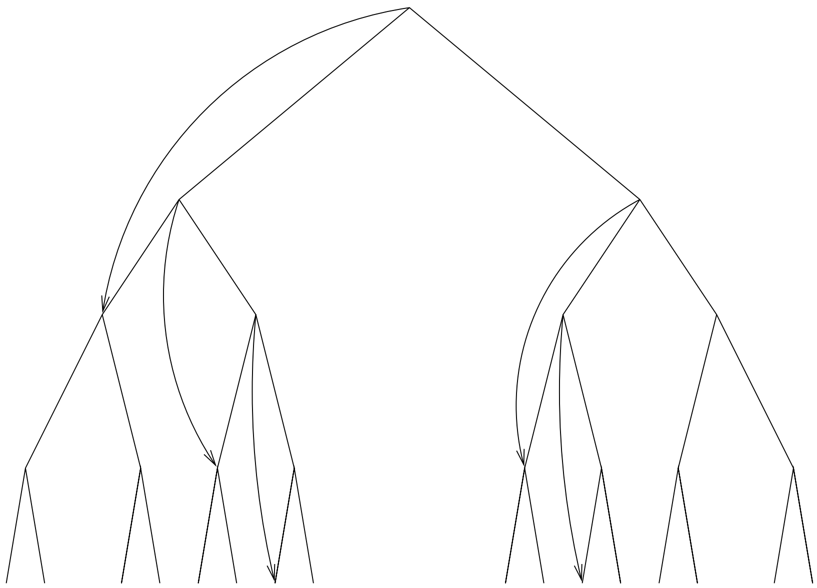

Figure 3: The iteration of the assignment of hotlinks

The overall algorithm is illustrated in Figure 3. Notice that by assumption we have that

$$\begin{array}{l} a+b+c+d=1 \\ e+f=1 \end{array}$$

Let $s_n$ (resp., $t_n$) be the expected number of steps to reach the leaves of the tree $T_n$ (resp. $S_n$), We now have the recurrences

$$\begin{array}{ll} t_n & = 1+(c+d)t_{n-1}+at_{n-2}+bs_{n-1} \\ s_n & = 1+et_{n-2}+fs_{n-2}, \end{array}$$

where $t_n$ and $s_n$ are the weights of the subtrees $T_n$ and $S_n$, respectively. We can prove the following claim.
**Claim:**

$$\begin{array}{ll} t_n & \leq 3n/4+1/4 \\ s_n & \leq 3n/4 \end{array}$$

**Proof of the claim:** The proof is by induction on $n$. Cases $n=2,3,4$ can be proved by inspection. First we consider $s_n$.

$$\begin{array}{ll} s_n & = 1+et_{n-2}+fs_{n-2} \\ & \leq 1+e(3(n-2)/4+1/4)+f3(n-1)/4 \\ & = 1+(e+f)3n/4-5e/4-3f/4 \\ & \leq 1+3n/4-(e+f)3/4-e/2 \\ & \leq 1+3n/4-1 \\ & = 3n/4, \end{array}$$

the last inequality being true because by assumption $e \geq 1/2$.

Next we consider $t_n$.

$$\begin{aligned} t_n &= 1 + at_{n-2} + (c+d)t_{n-1} + bs_{n-1} \\ &\leq 1 + a(3(n-2)/4 + 1/4) + (c+d)(3(n-1)/4 + 1/4) + 3(n-1)/4 \\ &= 1 + (a+b+c+d)3n/4 - 5a/4 - 3b/4 - (c+d)/2 \\ &= 1 + 3n/4 - 3a/4 - b/4 - (a+b+c+d)/2 \\ &= 1/2 + 3n/4 - 3a/4 - b/4 \end{aligned}$$

Now we have two cases depending on the size of $a$.

**Case 1:** $a \geq 1/3$.

In this case, the claim follows from the inequalities

$$\begin{aligned} t_n &\leq 1/2 + 3n/4 - 3a/4 - b/4 \\ &\leq 1/2 + 3n/4 - 1/4 \\ &\leq 1/4 + 3n/4 \end{aligned}$$

**Case 2:** $a < 1/3$.

In this case, write $a = 1/3 - x$, where $x > 0$, and notice that

$$b + c + d = 1 - a = 2/3 + x$$

and

$$\begin{aligned} c &\leq a = 1/3 - x \\ d &\leq a = 1/3 - x \end{aligned}$$

Consequently,

$$b = 2/3 + x - c - d \geq 3x.$$

It follows that

$$\begin{aligned} t_n &\leq 1/2 + 3n/4 - 3a/4 - b/4 \\ &\leq 1/2 + 3n/4 - 3/4(1/3 - x) - 3x/4 \\ &= 1/4 + 3n/4. \end{aligned}$$

This completes the proof of Case 2 and hence also of the claim. The proof of Theorem 3 is now complete. ■

## 3.3 Uniform distribution

Consider the uniform distribution in a full binary tree. It is easy to see that adding a link from the root to a node at distance $k$ will save exactly a fraction of $(k-1)2^{-k}$ of the total weight of the tree. Moreover this saving is maximized at $k = 2$ or $k = 3$ and the maximum value is $1/4$. Similarly, any hotlink we add to a node can save at most $1/4$ of the total weight of the subtree rooted at this node. This indicates that the maximal gain attained by a hotlink on a given node is at most $1/4$ of the weight of the subtree rooted at this node. Therefore it is not difficult to see that by adding one hotlink per node on a tree with the uniform distribution we can never attain a gain higher than $(n-1)/4$. The previous discussion and Theorem 3 imply the following result.

**Theorem 4** *Consider a rooted full binary tree on $n$ levels. There is an algorithm linear in the number of vertices of the tree which for the uniform distribution on the leaves of the tree assigns at most one hotlink per node in such a way that the expected number of steps to reach a leaf of the tree is at most $(3n+1)/4$. Moreover, $(3n+1)/4$ is a lower bound on any such algorithm.* ■

## 3.4 Zipf distribution

In this section we consider the sorted Zipf distribution, i.e. the $i$-th leaf of the tree is assigned probability $\frac{1}{iH_N}$, and prove an $n/3 + O(\sqrt{n}\log n)$ upper bound on the expected number of steps to reach a leaf.

Recall the following estimate for the harmonic number $H_m = \sum_{i=1}^{m} 1/i$ from [10][page 75]:

$$H_m = \ln m + \gamma + \frac{1}{2m} - \frac{1}{12m^2} + \frac{1}{120m^4} - \epsilon,$$

where $0 < \epsilon < \frac{1}{252m^6}$ and $\gamma = 0.5772\ldots$ is Euler's constant. Consequently we have that

$$\gamma + \ln m < H_m < 1 + \ln m.$$

It follows that for $l \le n$ we have the Inequalities

$$\frac{l}{n} + \frac{1}{n} \ge \sum_{i=1}^{2^l} p_i = \sum_{i=1}^{2^l} \frac{1}{iH_N} = \frac{1}{H_N}\sum_{i=1}^{2^l}\frac{1}{i} = \frac{H_{2^l}}{H_{2^n}} \ge \frac{l}{n}\cdot\left(\frac{n}{n+2}\right). \tag{4}$$

Let $r$ be the root of the tree. Define the sequence $r_0, r_1, \ldots, r_i, \ldots$ of tree nodes, where $i \le n$, as follows: $r := r_0$ and for each $i$, $r_i$ is the left child of $r_{i-1}$. Next we assign hotlinks rooted to these nodes in the following manner. Assign to $r_i$ a hotlink to the node $r_{\lceil n/2^{i+1}\rceil}$, provided that $i < \lceil n/2^{i+1}\rceil$. The weight of the subtree rooted at $r_{\lceil n/2^{i+1}\rceil}$ is equal to $\sum_{i=1}^{2^{l_{i+1}}} p_i$, where $l_{i+1} = n - \lceil n/2^{i+1}\rceil$. Since $n/2^{i+1} \le \lceil n/2^{i+1}\rceil \le 1 + n/2^{i+1}$ we can use Inequality (4) to obtain that

$$\left(1 - \frac{1}{2^{i+1}}\right)\frac{n}{n+2} - \frac{1}{n+2} \le \sum_{i=1}^{2^{l_{i+1}}} p_i \le \left(1 - \frac{1}{2^{i+1}}\right) + \frac{2}{n}. \tag{5}$$

In view of Inequality (5) it is not difficult to see that the gain resulting from adding a single hotlink from the node $r_i$ to the node $r_{\lceil n/2^{i+1}\rceil}$ is at least

$$\left(\frac{n}{2^{i+1}} - i - 1\right)\sum_{i=1+2^{l_i}}^{2^{l_{i+1}}} p_i \ge \left(\frac{n}{2^{i+1}} - i - 1\right)\cdot\frac{1}{2^{i+1}}\cdot\left(\frac{1}{2^{i+1}} - \frac{5}{n}\right). \tag{6}$$

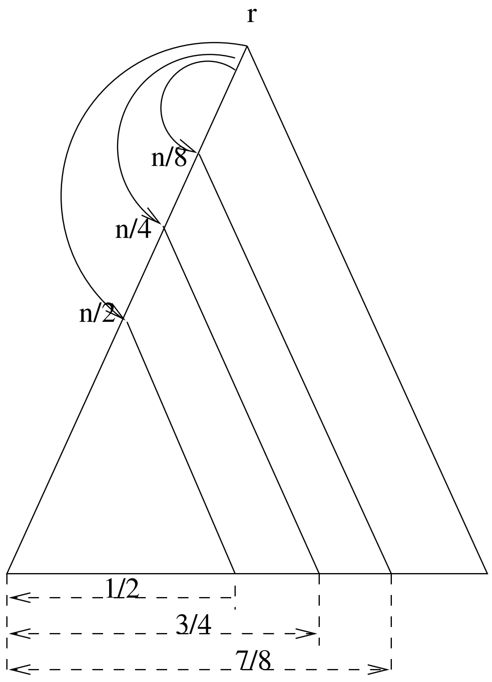


Figure 4: The iteration of the assignment of hotlinks

The hotlink assignment is illustrated in Figure 4. By summing the terms in Formula (6) for $i$ such that $i < \lceil n/2^{i+1} \rceil$ we obtain that the gain resulting from all these hyperlinks is at least

$$\left(\frac{n}{2} - 1\right) \cdot \left(\frac{1}{2} - \frac{5}{n}\right) + \left(\frac{n}{4} - 2\right) \cdot \left(\frac{1}{4} - \frac{5}{n}\right) + \left(\frac{n}{8} - 3\right) \cdot \left(\frac{1}{8} - \frac{5}{n}\right) + \cdots. \tag{7}$$

Let $k := \lceil (\log n)/3 \rceil - 1$. Clearly, $k$ satisfies $k2^{k+1} \leq \sqrt{n} < n$. Summing Formula (7) we obtain that the total gain achieved from all these hotlinks is at least

$$n \sum_{i=1}^{k} \frac{1}{2^{2i}} - \sum_{i=1}^{k} \left(\frac{i}{2^i} - \frac{5}{n}\right) - \frac{5k}{2} \geq \frac{n}{3} - O(\sqrt{n}). \tag{8}$$

Let $c > 0$ be the constant in Inequality (8). We now apply the previous hotlink assignment recursively to each subtree rooted at the node $r_{\lceil n/2^i \rceil}$, respectively, for $i \leq l = \lfloor \log n \rfloor$. Using Inequality (8) it is easy to see that the total gain achieved must be at least

$$\sum_{i=0}^{l-1} \left(\frac{n2^{-i}}{3} - c\sqrt{n/2^i}\right) \geq \frac{n}{3} \sum_{i=0}^{k-1} 2^{-i} - c\sqrt{n} \log n \geq \frac{2n}{3} - O(\sqrt{n} \log n).$$

It follows that after adding these hotlinks to the tree the expected length of a path to reach a leaf is at most

$$n - \frac{2n}{3} + O(\sqrt{n} \log n)) = \frac{n}{3} + O(\sqrt{n} \log n).$$

Therefore we have proved the following result.

**Theorem 5** *Consider a rooted full binary tree on $n$ levels. There is an algorithm linear in the number of vertices of the tree which for the sorted Zipf distribution on the leaves of the tree assigns at most one hotlink per node in such a way that the expected number of steps to reach a leaf of the tree is at most $n/3 + O(\sqrt{n}\log n)$.* ■

# 4 Lower Bounds

A "code-alphabet" is any nonempty set. A "code-word" in the code-alphabet is a word (i.e., any concatenation) formed from letters of the code-alphabet; the number of letters in the code-word is the length of the code-word. A "code" is any set of code-words. The code is called a "prefix-code" if no code-word in the code is a prefix of any other code-word in the same code.

To prove lower bounds we use Shannon Theory and the Entropy [1] $H(p)$ of the probability distribution $p$, which is defined by Equation (1). Shannon's theorem states the following:

**Theorem 6** *Let $p_1, p_2, \ldots, p_N$ be a probability distribution. Given a prefix code $w_1, w_2, \ldots, w_N$ of respective lengths $l_1, l_2, \ldots, l_N$ in an alphabet of size $k$, the expected length $\sum_{i=1}^{N} l_i p_i$ of the code is at least $H(p)/\log k$.* ■

A binary tree can be thought of as the encoding of the leaves with the two symbol alphabet $0, 1$. Adding a hotlink increments the alphabet by a single symbol to form a three symbol alphabet.

For a given hotlink assignment $A$, the distance of the $i$-th leaf from the root, previously denoted by $d_A(i)$, is the length of the encoding of the $i$-th leaf in this new alphabet. Notice that if two hotlinks are targeting the same node then the shortest one can be omitted without changing the value of $E[T^A, p]$. As a consequence, $E[T^A, p]$ is also the expected length of the "encoding" of the leaves of the tree $T^A$ represented as code words in a three letter alphabet. Moreover, the resulting encoding is a prefix code. In particular, Shannon's theorem applies and we have that

$$E[T^A, p] \geq \frac{1}{\log 3} \cdot H(p) = \frac{1}{\log 3} \cdot \sum_{i=1}^{N} p_i \log(1/p_i). \tag{9}$$

We have proved the following theorem.

**Theorem 7** *For any probability distribution $p$ on the leaves of a full binary tree and any assignment of at most one hotlink per source node the expected number of steps to reach a webpage located at a leaf from the root of the tree is at least $H(p)/\log 3$, where $H(p)$ is the entropy of $p$.* ■

Theorem 7 is directly applicable. Straightforward calculations give lower bounds for several distributions. For the uniform distribution it is easily seen that $H(p) = n$. For the Zipf distribution,

$$\begin{aligned} H(p) &= \sum_{i=1}^{N} p_i \log(1/p_i) \\ &= \tfrac{1}{H_N} \sum_{i=1}^{N} \tfrac{\log i}{i} + \log(H_N) \\ &\geq \tfrac{1}{2} \cdot \log N + \Omega(\log \log N) \\ &= \tfrac{n}{2} + \Omega(\log n). \end{aligned}$$

In particular, this implies the $n/(2 \log 3)$ lower bound on the expected number of steps for the Zipf distribution.

# 5 Conclusions

In this paper we have considered the problem of assigning hotlinks to the nodes of a full binary tree so as to minimize the expected number of steps from the root to the leaves under various probability distributions. In addition to improving upper and lower bounds for various probability distributions, interesting open problems include: considering other topologies, like arbitrary trees, DAGs, etc.

# References


[1] N. Abramson, "Information Theory and Coding", McGraw Hill, 1963.

[2] J. Abello, P. M. Pardalos and M. G. C. Resende. 1999. "On maximum clique problems in very large graphs". In External Memory Algorithms (J. Abello and J. Vitter, eds.), AMS-DIMACS Series on Discrete Mathematics and Theoretical Computer Science, Vol. 50.

[3] W. Aiello, F. Chung, L. Lu, "A random graph model for massive graphs". To appear in Proceedings of the 32nd Annual ACM Symposium on the Theory of Computing.

[4] B. Bollobas, "Random Graphs", Academic Press, London, 1985.

[5] J. Czyzowicz, E. Kranakis, A. Pelc, M. Vargas Martin, "Optimal Assignment of Bookmarks to Web Pages", to appear.

[6] M. Faloutsos, P. Faloutsos, C. Faloutsos, "On Power-Law Relationships of the Internet Topology", In proceedings of SIGCOM: IEEE Conference on Communication, 1999.

[7] J. Fink, A. Kobsa, A. Nill, "User Oriented Adaptivity and Adaptability in the AVANTI Project", in Designing for the Web: Empirical Studies, Microsoft Usability Group, Rednont WA, 1996.

[8] M.R. Garey and D.S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness", Freeman and Co., New York, 1979.

[9] J. M. Kleinberg, R. Kumar, P. Raghavan, S. Rajagopalan, A. S. Tomkins, "The Web as a graph: measurements, models, and methods. In Proceedings of the Fifth International Conference on Computing and Combinatorics", Tokyo July 26-28, 1999 (COCOON 99). Berlin: Springer-Verlag, pages 1-17, 1999.

[10] D. Knuth, "The Art of Computer Programming: Volume 1, Fundamental Algorithms", Addison Wesley, 2nd ed., 1997.

[11] D. Knuth, "The Art of Computer Programming: Volume 3, Sorting and Searching", Addison Wesley, 3rd ed., 1998.

[12] M. Perkowitz, O. Etzioni, "Towards Adaptive Web Sites: Conceptual Cluster Mining", in Proceedings of IJCAI99.

[13] M. Perkowitz and O. Etzioni. "Adaptive Web Sites: an AI Challenge", in Proceedings of IJCAI99.

[14] C. K. Zipf. "Human Behavior and the Principle of Least Effort", Reading Mass., Addison Wesley, 1949.