5
Constant Factor Approximation for the k-Hotlink Assignment Problem

In this section we present the first algorithm for the k-Hotlink Assignment Problem that attains a constant factor approximation. This algorithm is motivated by the observation that the entropy of the weight function w provides a good lower bound on the cost of optimal k-assignments for trees of 'low' degree (maximum degree $\approx k$), and that there is an algorithm that works well for these bounded degree trees [DL06]. Based on these facts, our algorithm greedily decomposes T into subtrees of low degree and then determines a good assignment for these subtrees using a modified version of the algorithm from [DL06]. In our analysis, we obtain a lower bound on the optimal solution and an upper bound on the returned solution in terms of the costs of the assignment for the subtrees. Thus, the approximation guarantee of the algorithm is basically a constant times the guarantee of the approximation used to compute the assignment for the subtrees.

Consider a tree T rooted at node r and a weight function w. For every node $u \in T$, we define the *cumulative weight* of u as the weight of its descendants, namely $w(T_u)$. A *heavy k-tree* Q of T is defined recursively as follows: r belong to Q; for every node u in Q, the k non-leaf children of u with greatest cumulative weight also belong to Q (if there are less than k non-leaf children then all of them belong to Q).

For $q \in Q$, if j is a child of q that does not belong to Q, then we define T_q^j as the subtree T_j. Also define $\overline{T}_q$ as the union of T_q^j over all meaningful j's (Figure 5.1.a). In order to simplify the notation during the analysis, we often omit the range of variables indexing the trees T_q^j's.

The algorithm proceeds as follows:

(i) Find a heavy k-tree Q of T and for each $q \in Q$ define $w'(q) = w(\overline{T}_q)/w(T)$.

(ii) Calculate a non-crossing k-assignment A_Q for the instance (Q, w') based on the modification of the algorithm proposed in [DL06].

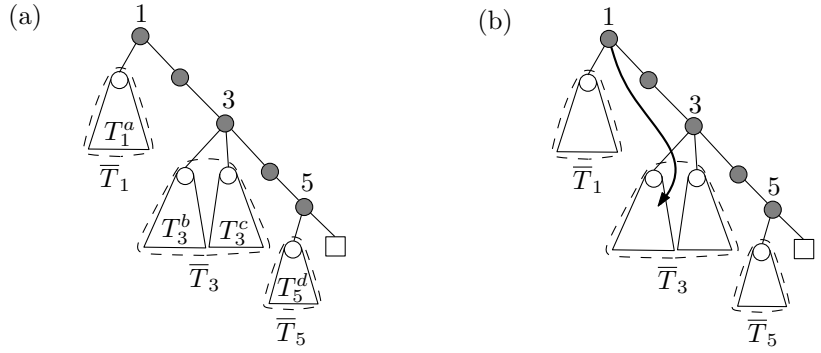

Figure 5.1: (a) Structures T_q^j and $\overline{T}_q$, with the nodes of the heavy 1-tree in gray. (b) Node 1 captures $\overline{T}_3$, so $c_3 = 1$. Also nodes 1 and 5 capture their own forests $\overline{T}_1$ and $\overline{T}_5$.

(iii) Calculate recursively a k-assignment A_q^j for each instance (T_q^j, w).

(iv) Output the assignment $A = A_Q \cup \bigcup_{q,j} A_q^j$.

We remark that w' may be positive for some internal nodes of Q, so the algorithm from [DL06] cannot be employed directly in Step (ii). However, in Section A.4(a) of the appendix we present a modification of this algorithm that can be used instead. Finally, it should be clear from the definition of the algorithm that it outputs a non-crossing k-assignment for T.

5.1 Upper Bound

In this section we devise an upper bound for the expected path of the enhanced tree returned by the presented algorithm.

Now notice that the trees $\{T_q^j\}$ define a partition of $T - Q$ and that Q does not contain any leaf of T. Therefore, the trees $\{T_q^j\}$ contain all leaves of T and by hypothesis all nodes of T with nonzero weights. Thus it suffices to analyze the paths in $T + A$ from r to nodes of the trees $\{T_q^j\}$ in order to provide a bound for the cost of A.

Consider a tree T_q^j and a node u of it. Informally, as there are no hotlinks in A going straight inside this subtree, a user walking from r to u in $T + A$ has to go to node q by using hotlinks assigned to the heavy k-tree Q, then go to node j and finally continue his path to u. That is, the path from r to u in $T + A$ has the form $(r \rightsquigarrow q \to j \rightsquigarrow u)$[1]. Weighting over all nodes in the trees

[1] A more formal argument is that because there are no hotlinks in A from proper ancestors of j to descendants of j and because A is a non-crossing assignment, Definition 1-(ii) implies that $T_q^j \subseteq \mathbf{T}_q(A)$ and $T_q^j \subseteq \mathbf{T}_j(A)$.

$\{T_q^j\}$ we have:

$$\begin{aligned}\mathrm{EP}(T,A) &= \sum_{q\in Q}\sum_{j}\sum_{u\in T_q^j}\left((d(r,q,T+A)+1+d(j,u,T+A)\right)w(u)\\ &= w(T)+\sum_{q\in Q} d(r,q,T+A)w(\overline{T}_q)+\sum_{q\in Q}\sum_{j}\sum_{u\in T_q^j} d(j,u,T+A)w(u)\\ &= w(T)+w(T)\sum_{q\in Q} d(r,q,Q+A_Q)w'(q)+\sum_{q\in Q}\sum_{j}\mathrm{EP}(T_q^j,A_q^j)\end{aligned}$$

where the last equality follow from Proposition 1.

Notice that the summation $\sum_{q\in Q} d(r,q,Q+A_Q)w'(q)$ that appears in the right-hand side of the above equality is exactly the cost of the assignment for (Q,w') constructed at Step (ii) of the algorithm. For any multiset W we define the entropy of W as $H(W) = -\sum_{w\in W} w\log\frac{w}{\sum_{w'\in W} w'}$. Using the modified version of the algorithm from [DL06], the summation $\sum_{q\in Q} d(r,q,Q+A_Q)w'(q)$ can be bounded by $4H(\{w'(q)\})/\log(k+1) + 4w(T)$. Therefore, substituting this bound on inequality (1) and noticing that $w(T)\cdot H(\{w'(q)\}) = H(\{w(\overline{T}_q)\})$, we have our upper bound:

$$\mathrm{EP}(T,A) \le 5w(T) + \frac{4H(\{w(\overline{T}_q)\})}{\log(k+1)} + \sum_{q\in Q}\sum_{j}\mathrm{EP}(T_q^j,A_q^j) \qquad (1)$$

5.2 Entropy Lower Bound

In this section we devise a lower bound on the cost of an optimal assignment for (T,w) in terms of the entropy of the multiset $\{w(\overline{T}_q)\}$.

Consider a non-crossing optimal k-assignment A^* for T. We say that a node $q \in Q$ *captures* a forest $\overline{T}_u$ if it satisfies the following conditions simultaneously: (i) q has either a hotlink or an arc pointing to a node in $\overline{T}_u$; (ii) no proper ancestor of q in Q satisfies (i). Then, we use c_u to denote the node of Q that captures the forest $\overline{T}_u$ (Figure 5.1.b).

A crucial observation is that every user path in $T + A^*$ from r to nodes in $\overline{T}_q$ must contain c_q – otherwise users would have to use a hotlink to 'jump' over c_q and there would be a crossing between this hotlink and the one from c_q pointing to $\overline{T}_q$, which contradicts the fact that A^* is non-crossing. The following lemma, which is proved in the appendix, states this important property.

Lemma 9 *Consider some node q in Q and let u be a node in $\overline{T}_q$. Then, the user path from r to u in $T + A^*$ contains the node c_q.*

Therefore, for each node $u \in T_q^j$ we can decompose the user path $(r \rightsquigarrow u)$ into $(r \rightsquigarrow c_q)$ and $(c_q \rightsquigarrow u)$, hence $d(r,u,T+A^*) = d(r,c_q,T+A^*)+d(c_q,u,T+$

A^*). Weighting this distance over all nodes u in T_q^j, we have:

$$\mathrm{EP}(T,A)_{T_q^j} \;=\; d(r,c_q,T+A^*)w(T_q^j) + \sum_{u\in T_q^j} d(c_q,u,T+A^*)w(u)$$

Using Lemma 2 with $T' = T_q^j$ and $v = c_q$ to lower bound the last term of the expression, we have that $\mathrm{EP}(T,A^*)_{T_q^j} \geq d(r,c_q,T+A^*)w(T_q^j) + \mathrm{OPT}_k(T_q^j)$. It follows again by the fact that the trees $\{T_q^j\}$ contain all nodes of T with nonzero weight that $\mathrm{OPT}_k(T) = \sum_{q\in Q}\sum_j \mathrm{EP}(T,A^*)_{T_q^j}$ and hence:

$$\begin{aligned}\mathrm{OPT}_k(T) &\geq \sum_{q\in Q}\sum_j d(r,c_q,T+A^*)w(T_q^j) + \sum_{q\in Q}\sum_j \mathrm{OPT}_k(T_q^j)\\ &= \sum_{q\in Q} d(r,c_q,T+A^*)w(\overline{T}_q) + \sum_{q\in Q}\sum_j \mathrm{OPT}_k(T_q^j) \qquad (2)\end{aligned}$$

Before proceeding with the lower bounding of $\mathrm{OPT}_k(T)$ we need to rearrange the previous inequality. The first observation is that a node $q \in Q$ can capture at most $k+1$ different forests $\{\overline{T}_u\}$, because all of its arcs point to the same $\overline{T}_u$. Therefore, a node q can appear at most $k+1$ times as the second parameter of the distance function on the left-hand side of inequality (2). Let w_q^i be the weight $w(\overline{T}_{q'})$ on the ith term that q appears (we zero some of these weights accordingly if q appears in less than $k+1$ of the terms). Then we can write (2) as:

$$\mathrm{OPT}_k(T) \geq \sum_{q\in Q}\left[d(r,q,T+A^*)\cdot\sum_i w_q^i\right] + \sum_{q\in Q}\sum_j \mathrm{OPT}_k(T_q^j) \qquad (3)$$

The idea now is to see the first summation of the previous inequality as the cost of reaching nodes in Q when they are endowed with weights $\{\sum_i w_q^i\}_{q\in Q}$. Because Q has low degree, we can employ Shannon's Coding Theorem [Gallager68] as done in [BKK+00] to obtain a good lower bound for this term.

Lemma 10 $\sum_{q\in Q}\left[d(r,q,T+A^*)\cdot\sum_i w_q^i\right] \geq \frac{H(\{w(\overline{T}_q)\})}{2\log(k+1)} - w(T)$

Proof: Before being able to use Shannon's Coding Theorem as in [BKK+00], we need to normalize the weights $\{w_i^q\}$ and then 'propagate' them to (new) leaves of Q.

Notice that $\sum_{q\in Q}\sum_i w_q^i = \sum_{q\in Q} w(\overline{T}_q) = w(T)$. Thus we define the normalized weights as $w_q'^i = w_q^i/w(T)$.

Now we construct a tree Q' by adding $k+1$ leaves $\{q^1,\ldots,q^{k+1}\}$ to each node $q \in Q$. Also, we define the following weight function w': for each

new leaf q^i, let $w'(q^i) = w'^i_q$ and let $w'(q) = 0$ for every other node q of Q'. This construction guarantees that only the leaves of Q', which are $\{q^i\}$, have nonzero weight with respect to w'.

Notice that A^* do not have hotlinks pointing to nodes $\{q^i\}$. Then it is easy to see that the path from r to q^i in $Q' + A^*$ is exactly the path from r to q in $Q + A^*$ plus one hop from q to q^i. Therefore:

$$\begin{aligned}
\mathrm{EP}(Q', A^*, w') &= \sum_{q \in Q} \sum_{i=1}^{k+1} (d(r, q, T + A^*) + 1) w'(q^i) \\
&= \sum_{q \in Q} \left[d(r, q, T + A^*) \cdot \sum_{i=1}^{k+1} w'(q^i) \right] + \sum_{q \in Q} \sum_{i=1}^{k+1} w'(q^i) \\
&= \frac{1}{w(T)} \sum_{q \in Q} \left[d(r, q, T + A^*) \cdot \sum_{i=1}^{k+1} w^i_q \right] + 1 \qquad (4)
\end{aligned}$$

The next step is to bound the cost $\mathrm{EP}(Q', A^*, w')$. Recall that the user paths of $Q' + A^*$ define a tree Q'^{A^*} (see Figure 1.1). We argue that this tree has degree at most $3k + 1$. Because Q has degree at most k, even with the addition of the new leaves the tree Q' has degree at most $2k + 1$. Moreover, each node of Q' has at most k hotlinks in A^*, and consequently each node of Q' has at most $3k + 1$ arcs or hotlinks in $Q' + A^*$. This in turn implies that Q'^{A^*} has degree at most $3k + 1$.

Let $\mathbb{E}$ denote the expected path (with respect to w') from the root of Q'^{A^*} to its leaves. Clearly a leaf in Q' must also be a leaf in Q'^{A^*}. Therefore, w' is a probability function on the leaves of Q'^{A^*} and we can employ Shannon's Coding Theorem to lower bound $\mathbb{E}$ by $H(\{w'(q)\})/\log(3k + 1)$. In addition, as internal nodes of Q'^{A^*} are also internal nodes of Q' and therefore have zero weight with respect to w', it follows that $\mathbb{E}$ is also the expected path from the root of Q'^{A^*} to all of its nodes. Finally, it is easy to see that this expected path to all nodes of Q'^{A^*} is exactly $\mathrm{EP}(Q', A^*, w')$. Combining the previous observations, we have that:

$$\mathrm{EP}(Q', A^*, w') = \mathbb{E} \geq \frac{H(\{w'(q)\})}{\log(3k + 1)} \qquad (5)$$

Because $k \geq 1$, it follows that $k^2 \geq k \Rightarrow k^2 + 2k + 1 \geq 3k + 1 \Rightarrow (k + 1)^2 \geq 3k + 1$. Therefore, $\log(3k + 1)) \leq 2\log(k + 1)$. Employing this bound on inequality (5) and noticing that $\{w'(q)\} = \{w'^i_q\} = \{w^i_q / w(T)\} =$

$\{w(\overline{T}_q)/w(T)\}$, we have:

$$\mathrm{EP}(Q', A^*, w') \geq \frac{H(\{w'(q)\})}{2\log(k+1)} = \frac{H(\{w(\overline{T}_q)/w(T)\})}{2\log(k+1)} = \frac{H(\{w(\overline{T}_q)\})}{w(T)\cdot 2\log(k+1)} \quad (6)$$

where the last equality holds due to the definition of $H(.)$.

The result then follows by chaining inequalities (4) and (6) and multiplying the resulting inequality by $w(T)$. ■

Applying Lemma 10 to inequality (3) leads to the completion of the lower bound for the cost of an optimal k-assignment for T:

$$\mathrm{OPT}_k(T) \geq \frac{H(\{w(\overline{T}_q)\})}{2\log(k+1)} - w(T) + \sum_{q\in Q}\sum_{j} \mathrm{OPT}_k(T_q^j) \quad (7)$$

5.3 Alternative Lower Lound

When the value of the entropy $H(\{w(\overline{T}_q)\})$ is large enough, it dominates the term $w(T)$ in inequality (7) and we have a sufficiently strong lower bound. However, when this entropy assumes a small value we need to adopt a different strategy to devise an effective bound.

First, we need to refine the definition of capturing forests of T. A node $q \in Q$ *captures* a tree T_u^j with respect to $T + A^*$ if it satisfies the following conditions simultaneously: (i) q has either a hotlink in A^* or an arc in T pointing to a node in T_u^j; (ii) no proper ancestor of q in Q satisfies (i). Then, we use c_u^j to denote the node of Q that captures the tree T_u^j with respect to $T + A^*$. The following lemma can be proved exactly as Lemma 9:

Lemma 11 *Consider some node q in Q and let u be a node in a tree T_q^j. Then, the user path from r to u in $T + A^*$ contains the node c_q^j.*

For each q in the heavy k-tree Q, we define HL_q as roots of the trees $\{T_q^j\}$ that have a node adopted by r, that is $HL_q = \{j : (r, u) \in A^* \text{ for some } u \in T_q^j\}$.

Consider a tree T_r^j such that $j \notin HL_r$. As there are no hotlinks from r to nodes in T_r^j, the user path in $T + A^*$ from r to a node $u \in T_r^j$ is $(r \to j \rightsquigarrow u)$. This implies that $d(r, u, T + A^*) = 1 + d(j, u, T + A^*)$. Weighting this equality over all $u \in T_r^j$ and employing Lemma 2 with $T' = T_r^j$ and $v = j$, we have that:

$$\mathrm{EP}(T, A^*)_{T_r^j} = \sum_{u\in T_r^j} (1 + d(j, u, T + A^*))w(u) \geq w(T_r^j) + \mathrm{OPT}_k(T_r^j)$$

Now for $q \neq r$, consider a tree T_q^j such that $j \notin HL_q$. By the definition of HL_q and the hypothesis on q, there cannot be a hotlink or an arc in $T + A^*$ from r to a node in T_q^j. Therefore, the node c_q^j that captures T_q^j must be different than r. Since Lemma 11 states that the path from r to a node $u \in T_q^j$ is $(r \rightsquigarrow c_q^j \rightsquigarrow u)$, the previous observation implies that $d(r, u, T + A^*) \geq 1 + d(c_q^j, u, T + A^*)$. Again, weighting this inequality over all $u \in T_q^j$ and then applying Lemma 2, we have that $\mathrm{EP}(T, A^*)_{T_q^j} \geq w(T_q^j) + \mathrm{OPT}_k(T_q^j)$.

Finally, for any $q \in Q$ and for any node $j \in HL_q$, we can use Lemma 2 with $T' = T_q^j$ and $v = r$ to obtain the lower bound $\mathrm{EP}(T, A^*)_{T_q^j} \geq \mathrm{OPT}_k(T_q^j)$.

Recalling that all leaves of T belong to the trees $\{T_q^j\}$, we can combine the previous lower bounds to attain the following bound for $\mathrm{EP}(T, A^*)$:

$$\mathrm{EP}(T, A^*) \geq \sum_{q \in Q} \sum_{j \notin HL_q} w(T_q^j) + \sum_{q \in Q} \sum_{j} \mathrm{OPT}_k(T_q^j) \tag{8}$$

Now we argue that the first term of the right-hand side is at least a significant fraction of the total weight $w(T)$, more specifically $w(T)/2$. The reasoning used is roughly the following: we can associate (uniquely) to each node $j \in HL_q$ a sibling j' of j which belongs to the heavy k-tree Q. Therefore, we can associate to each tree T_q^j with $j \in HL_q$ a different tree $T_{j'}$ such that $w(T_{j'}) \geq w(T_q^j)$. This in turn will imply that the sum of the weight of the trees $\{T_q^j\}_{q, j \in HL_q}$ is at most $w(T)/2$. Because the trees $\{T_q^j\}$ contain all nodes of T with nonzero weight, this implies that the weight of the trees $\{T_q^j\}_{q, j \notin HL_q}$ is at least $w(T)/2$.

In order accomplish the crucial association among nodes in HL_q and their siblings, we define the set NHL_q in the following way: a node u belongs to NHL_q if u is a child of q in Q and if no node of T_u is pointed by a hotlink from r in A^*. Furthermore, we focus on the nodes $q \in Q$ for which HL_q is not empty, hence we define $C = \{q \in Q : HL_q \neq \emptyset\}$. The argument is then divided in the following claims:

Claim 1. For any node $q \in Q$, $\sum_{u \in NHL_q} w(T_u) \geq \sum_{j \in HL_q} w(T_q^j)$.

Claim 2. For any node $q \in Q$, consider a tree T_q^j such that $j \in HL_q$. Then for any u in $\bigcup_{q \in Q} NHL_q$ the trees T_u and T_q^j are disjoint.

Claim 3. For any two distinct nodes u and u' in $\bigcup_{q \in C} NHL_q$, the trees T_u and $T_{u'}$ are disjoint.

Proof of Claim 1. Consider a node $q \in Q$. If HL_q is empty then the claim follows trivially, hence assume that HL_q is nonempty. This implies that some tree T_q^j exists and consequently that exactly k children of q in T belong to Q. Let $\{q_1, \ldots, q_k\}$ be these children of q that belong to Q. Because $j \in HL_q$ implies that $j \notin Q$, it follows that the trees $\{T_{q_i}\}_{i=1}^k$ and the trees $\{T_q^j\}_{j \in HL_q} = \{T_j\}_{j \in HL_q}$ are pairwise disjoint. Therefore, at least $|HL_q|$ of the hotlinks of r in A^* point to nodes in the trees $\{T_q^j\}_{j \in HL_q}$ and consequently at most $k - |HL_q|$ hotlinks of r in A^* point to nodes in the tress $\{T_{q_i}\}_{i=1}^k$. Moreover, at least $|HL_q|$ of the trees $\{T_{q_i}\}_{i=1}^k$ do not have a node pointed by a hotlink from r. This implies that at least $|HL_q|$ of the nodes $\{q_1, \ldots, q_k\}$ belong to NHL_q, namely $|NHL_q| \geq |HL_q|$. For each $u \in NHL_q$ and $j \in HL_q$, the fact that u belongs to the heavy k-tree Q and that $j \notin Q$ implies that $w(T_u) \geq w(T_j) = w(T_q^j)$. As a consequence of the last two sentences, we have:

$$\sum_{u \in NHL_q} w(T_u) \geq \sum_{j \in HL_q} w(T_q^j)$$

and the claim follows.

Proof of Claim 2. Consider a tree T_q^j with $j \in HL_q$ and a node $u \in \bigcup_{q \in Q} NHL_q$. Because both u and the root of T belong to the tree Q, it follows that all ancestors of u in T must also belong to Q. As $j \notin Q$, j cannot be an ancestor of u. By means of contradiction suppose that j is a descendant of u. As $T_q^j \in HL$, there is a node x in $T_q^j = T_j$ which is pointed by a hotlink from r in A^*. However, the hypothesis implies that x is also descendant of u and contradicts the fact that $u \in \bigcup_{q \in Q} NHL_q$. Because j is neither an ancestor nor a descendant of u, together with the fact that both T_u and $T_q^j = T_j$ are maximal subtrees of T, we have that these trees must be disjoint.

Proof of Claim 3. The argument is similar to the one used in the previous claim. First, fix a node $q \in C$; as any two nodes $u \neq u'$ in NHL_q are children of q in T, it follows that the trees T_u and $T_{u'}$ are disjoint. Therefore, consider two different nodes q and q' in C and let u be a node in NHL_q and u' a node in $NHL_{q'}$. By means of contradiction suppose that u is an ancestor of u'. The fact that $q \neq q'$ implies that $u \neq u'$ as well, which in turn implies that u is a proper ancestor of u'. Consequently, u is an ancestor of the father of u', that is, $q' \in T_u$. Because q' belongs to C there must be a node x in $T_{q'}$ (more specifically in some tree $T_{q'}^j$) that is pointed by a hotlink from r in A^*. Therefore $x \in T_{q'} \subseteq T_u$, which contradicts the fact that u belongs to NHL_q. A symmetric argument proves that u' also cannot be an ancestor of u. Again,

as T_u and $T_{u'}$ are maximal subtrees, this implies that T_u and $T_{u'}$ are disjoint.

Adding the inequality of Claim 1 for all $q \in C$, we have:

$$\sum_{q \in C} \sum_{u \in NHL_q} w(T_u) \geq \sum_{q \in C} \sum_{j \in HL_q} w(T_q^j) \tag{9}$$

From Claims 2 and 3 we know that the trees that appear in the previous inequality are pairwise disjoint. Therefore, the sum of their weights cannot exceed the weight of the tree T:

$$\sum_{q \in C} \sum_{u \in NHL_q} w(T_u) + \sum_{q \in C} \sum_{j \in HL_q} w(T_q^j) \leq w(T)$$

This fact combined with inequality (9) gives:

$$\sum_{q \in Q} \sum_{j \in HL_q} w(T_q^j) = \sum_{q \in C} \sum_{j \in HL_q} w(T_q^j) \leq \frac{w(T)}{2}$$

Recalling that the trees $\{T_q^j\}$ contain all nodes of T with nonzero weights, we can express the last inequality as $\sum_{q \in Q} \sum_{j \notin HL_q} w(T_q^j) \geq w(T)/2$. Applying this bound on inequality (8) we complete the lower bound:

$$\mathrm{OPT}_k(T) \geq \frac{w(T)}{2} + \sum_{q \in Q} \sum_{j} \mathrm{OPT}_k(T_q^j) \tag{10}$$

5.4 Approximation Guarantee

We finally compare the lower bounds for the cost of an optimal assignment and the upper bound on the cost of the assignment returned by the algorithm to obtain its approximation guarantee.

The proof goes by induction on the number of nodes of the input tree. Assume that for any tree T' with fewer nodes than T the algorithm outputs an α-approximate hotlink assignment for (T', w), for some constant α that we make explicit later. For the base case of the induction, it should be clear that if T' is an empty tree or a single leaf the hypothesis holds.

We argue that the result also holds for T. As each tree T_q^j is properly contained in T, we can employ the inductive hypothesis in the upper bound given by (1):

$$\mathrm{EP}(T, A) \leq 5w(T) + \frac{4H(\{w(\overline{T}_q)\})}{\log(k+1)} + \sum_{q \in Q} \sum_{j} \alpha \mathrm{OPT}_k(T_q^j) \tag{11}$$

There are two cases that should be considered separately depending whether the value of $H(\{w(\overline{T}_q)\})/\log(k+1)$ dominates $w(T)$ or not. In order to simplify the notation, we henceforth use H as a shorthand for $H(\{w(\overline{T}_q)\})$.

High entropy case: $H/\log(k+1) > 3w(T)$**.** From this hypothesis on the entropy, it follows that $-w(T) > -H/3\log(k+1)$. Substituting this inequality in the lower bound of inequality (7), we have:

$$\mathrm{OPT}_k(T) > \frac{H}{6\log(k+1)} + \sum_{q\in Q}\sum_{j}\mathrm{OPT}_k(T_q^j) \tag{12}$$

Also from the entropy hypothesis $w(T) < H/3\log(k+1)$, and substituting in the upper bound (11) we have:

$$\mathrm{EP}(T,A) < \frac{17H}{3\log(k+1)} + \sum_{q\in Q}\sum_{j}\alpha\mathrm{OPT}_k(T_q^j) \tag{13}$$

By choosing $\alpha \geq 34$, inequalities (12) and (13) guarantee that $\mathrm{EP}(T,A) \leq \alpha \cdot \mathrm{OPT}_k(T)$, thus concluding the inductive step for this case.

Low entropy case: $H/\log(k+1) \leq 3w(T)$**.** Combining this entropy hypothesis with the upper bound given by inequality (11), we have:

$$\mathrm{EP}(T,A) \leq 17w(T) + \sum_{q\in Q}\sum_{j}\alpha\mathrm{OPT}_k(T_q^j) \tag{14}$$

Again by choosing $\alpha \geq 34$, inequalities (10) and (14) guarantee that $\mathrm{EP}(T,A) \leq \alpha\mathrm{OPT}_k(T)$. This completes the proof of the approximation guarantee of the algorithm.

Theorem 5 *The presented algorithm provides a constant factor approximation for the k-Hotlink Assignment Problem.*