Eurographics Conference on Visualization (EuroVis) 2013
B. Preim, P. Rheingans, and H. Theisel
(Guest Editors)

Volume 32 (2013), Number 3

Appendix: Maximum Entropy Summary Trees

Howard Karloff[1] and Kenneth E. Shirley[1]

[1]AT&T Labs Research, Florham Park, NJ, USA

This is an appendix to "Maximum Entropy Summary Trees." Section 9 provides details of the proof of the approximation algorithm that is described in Section 6 of the main paper; Section 10 includes a proof of Lemma 2 from Section 5.1 of the main paper (the inductive step of computing maximum entropy summary forests across a given node's children); and Section 11 provides an interesting example of a tree where the greedy heuristic described in Section 8 of the main paper does poorly in comparison to the exact algorithm.

9. A polynomial-time additive approximation algorithm

9.1. High-level description

The additive approximation algorithm takes a tree T, weighted with nonnegative real weights (w_i), a positive integer K, and an $\varepsilon > 0$, and produces, for each $k \leq K$, a k-node summary tree whose entropy is at most ε less than that of the optimal k-node summary tree.

The idea underlying the algorithm is simple—(1) scale the weights uniformly so that they sum to an integer W, whose value will be determined later; (2) carefully round real weight w_i to $w'_i \in \{\lfloor w_i \rfloor, 1 + \lfloor w_i \rfloor\}$; and (3) run the dynamic-programming algorithm of Section 5 of the main paper on the results. Doing so, however, in such a way as to guarantee small enough error relative to the optimum for $\langle w_1, w_2, ..., w_n \rangle$ while simultaneously keeping the running time down is quite nontrivial. Larger W gives better accuracy at the cost of a larger running time. (We will always ensure that $\sum w'_i = \sum w_i = W$ to keep the normalization simple.) Specifically, we have to address two questions: (1) how does the entropy of a maximum entropy summary tree change if weights are rounded, and (2) how should one round the weights?

Our two-part answer to question (1) is given in Lemmas 5 and 6. Any k-node summary tree corresponds to a partition of the vertex set $V = V(T)$ into k parts. For a single fixed but *unknown* partition $\langle S_1, S_2, ..., S_k \rangle$, we are interested in two probability distributions. One, with W_j denoting $\sum_{i \in S_j} w_i$, is the probability distribution $\langle W_1/W, W_2/W, ..., W_k/W \rangle$. The second is the same with w'_i in place of w_i; specifically, it is $\langle W'_1/W, W'_2/W, ..., W'_k/W \rangle / W$, with $W'_j = \sum_{i \in S_j} w'_i$. We are interested in how much the entropies of the distributions $\langle W'_1, W'_2, ..., W'_k \rangle / W$ and $\langle W'_1, W'_2, ..., W'_k \rangle / W$ differ. Fortunately Lemma 6 [Nau04] bounds the entropy difference in terms of the L_1 distance between the distributions. Therefore, we ask, how can we round the weights to keep the L_1 distance $(\sum_{j=1}^{k} |W_j - W'_j|)/W$ small, *without knowing the partition* $\langle S_1, S_2, ..., S_k \rangle$ *in advance*?

The surprising result is that if one ensures that for any node v in the known input tree T, the sums of w_i/W and w'_i/W over descendants of v are almost the same, then for any *unknown* partition $\langle S_1, S_2, ..., S_k \rangle$ derivable from a k-node summary tree, the two induced probability distributions will have small L_1 distance $\sum_{i=1}^{k} |(W'_i - W_i)/W|$. This fact is proven in Lemma 5. The beauty here is that T is known in advance, whereas $\langle S_1, S_2, ..., S_k \rangle$ is not. We define *subtree absolute discrepancy* M to be the maximum, over nodes v, of the absolute difference between the sums of w_i and w'_i over descendants i of v.

This argument motivates the answer to (2): we should round the weights so that the subtree absolute discrepancy is small. How to do so is an interesting question in itself. There is much work on discrepancy theory for general set systems [Spe94, Cha00]. The surprising fact, known before but rediscovered together with an associated algorithm by the authors, is that one can round the w_i's while giving subtree discrepancy M bounded by 1, for any T.

Having said all that, the real argument is more complicated. There is no *single* partition $\langle S_1, S_2, ..., S_k \rangle$ with which one works. One has to argue that rounding weights from w_i to w'_i gives a new solution which is neither too large nor too small. To do this properly, one has to start the argument from the optimal partitions for both weights (w_i) and weights (w'_i). This argument is given in Lemma 7.

9.2. Details

Recall that we denote by T^w an n-node tree on $\{1,2,...,n\}$ whose ith node has real weight w_i.

We give an algorithm which takes a tree T^w on $\{1,2,...,n\}$, whose node i has nonnegative *real* weight w_i, positive integer K, and a positive real ε, and returns, for each $k \le K$, a k-node summary tree whose entropy is at least $OPT_k(T^w) - \varepsilon$, where $\varepsilon > 0$ is a parameter. The running time of the algorithm (to generate all K trees) is $O((K^3/\varepsilon)n\log(\max\{K, 1/\varepsilon\}))$, though this is just a tree-independent worst-case upper bound.

In a rooted tree T, $x \in V(T)$, let T_x denote the subtree of T rooted at x.

Definition 6. *Suppose (w_i), (w'_i) are both real-valued weight functions defined on $\{1,2,...,n\}$.*

1. *The* (signed) discrepancy *$disc(S)$ of a set $S \subseteq \{1,2,...,n\}$ is $disc(S) = \sum_{i\in S}(w'_i - w_i)$.*
2. *The* absolute discrepancy *of a set $S \subseteq \{1,2,...,n\}$ is $|disc(S)|$.*
3. *Relative to tree T, the* subtree absolute discrepancy *M is $\max_i |disc(V(T_i))|$.*
4. *Given an ordered partition $P = \langle S_1, S_2, ..., S_k\rangle$ of $\{1,2,...,n\}$, the* absolute discrepancy *of P is $\sum_{i=1}^k |disc(S_i)|$.*

Definition 7. *Say the pair (w, w') of weight functions is* nearby *if $|w'_i - w_i| \le 1$ for all i.*

We start with our discrepancy lemma.

Lemma 4. *There is a $O(n)$-time algorithm that takes n and an n-node rooted tree T on $\{1,2,...,n\}$ rooted at node 1, and a sequence $\langle w_1, w_2, ..., w_n\rangle$ of nonnegative reals, and produces a sequence $w'_1, w'_2, ..., w'_n$ with $w'_i \in \{\lfloor w_i\rfloor, 1+\lfloor w_i\rfloor\}$ such that the subtree absolute discrepancy M is at most 1. Furthermore, if the w_i's sum to an integer, the w'_i's will have the same sum.*

The existence of a rounding with subtree absolute discrepancy strictly less than 1 follows from a much more general result [Doe04], which itself follows on similar earlier results. The existence of the algorithm also probably follows from earlier results and will not be included here for lack of space.

Here is the algorithm.

1. Choose an integer W, as described later, and scale $\langle w_1, w_2, ..., w_n\rangle$ to have sum W.
2. Using Lemma 4, produce a sequence $\langle w'_1, w'_2, ..., w'_n\rangle$ with $w'_i \in \{\lfloor w_i\rfloor, 1+\lfloor w_i\rfloor\}$ having subtree absolute discrepancy $M \le 1$ and with $\sum w'_i = W$.
3. Run the exact dynamic-programming algorithm of Section 5.2 of the main paper on tree $T_{w'}$, to get an optimal k-node summary tree T' for $T_{w'}$.
4. Output tree Z, which is T' except with weights (w_i) instead. (In other words, output the same summary tree, but with the weight of a cluster containing nodes $S \subseteq \{1,2,...,n\}$ being $\sum_{i\in S} w_i$, instead of $\sum_{i\in S} w'_i$.)

Definition 8. *Say a node v in a summary tree T' is a* singleton *node if its cluster has size 1, is a* tree node *if its cluster has size exceeding 1 and it represents $V(T_x)$ for some node x, and otherwise is an "other" node.*

Note that any "other" cluster which corresponds to the descendants of exactly one child of a node v is being renamed a tree node or a singleton node for the purpose of this definition. Also note that every node in a summary tree is exactly one of singleton, tree, and "other."

Definition 9. *Say a node in a summary tree T' is* active *if it is not an "other" node. Let A_v be the set of active children of v in the summary tree T' and let $a_v = |A_v|$.*

It is obvious that $\sum_{v\in V(T')} a(v) \le k$ if T' is a k-node summary tree, since $A_u \cap A_v = \emptyset$ for $u \ne v$ implies that $\sum_v |A_v| \le k$.

Now we show that keeping small the subtree absolute discrepancy relative to T ensures that the absolute discrepancy of the partition associated with every k-node summary tree of T will be small.

Lemma 5. *Let T be a rooted tree on $V = \{1,2,...,n\}$ and let (w, w') be a nearby pair of weight functions on V. Let M be the subtree absolute discrepancy (relative to tree T) of that pair. Let $D = k + 2kM$. Let $P = \langle S_1, S_2, ..., S_k\rangle$ be the partition of V defined by any k-node summary tree T' for T. Then the absolute discrepancy of P is at most D.*

It is important for this lemma that P be derived from a k-node summary tree for T (and not be an arbitrary partition into k parts).

Proof. We need to prove that $\sum_{i=1}^k |disc(S_i)| \le k + 2kM$, where $M = \max_{v\in V(T)} |disc(V(T_v))|$. Each set S_i corresponds to either a singleton node in T', a tree node in T', or an "other" node in T'.

If S_i corresponds to a singleton node in T', then $|S_i| = 1$ and, say, $S_i = \{u\}$. Then $|disc(S_i)| = |w_u - w'_u| \le 1$, because (w, w') is nearby.

If S_i corresponds to a tree node in T', then there is a node $x \in V(T)$ such that $S_i = V(T_x)$ and $|disc(S_i)| = |\sum_{y\in V(T_x)}(w'_y - w_y)| \le M$.

Now if u is an "other" node in T', which is cluster C in T, whose parent in T' is v, then $disc(V(T_v)) = (w'_v - w_v) + \sum_{a\in A_v} disc(V(T_a)) + disc(C)$, and therefore $disc(C) = disc(V(T_v)) - (w'_v - w_v) - \sum_{a\in A_v} disc(V(T_a))$. Hence $|disc(C)| \le |disc(V(T_v))| + 1 + \sum_{a\in A_v} |disc(V(T_a))| \le M + 1 + a_v M$. Clearly this can be bounded by $1 + (k+1)M$, proving that $\sum_i |disc(S_i)| \le k(1 + (k+1)M)$, but we can do better.

Let k_s be the number of singleton nodes in T', let k_t be the number of tree nodes in T', and let k_o be the number of "other" nodes in T'. Clearly $k_s + k_t + k_o = k$.

Any "other" cluster S has a parent node u in the summary tree. Let $parent(S)$ denote the parent of S, which is a singleton node. Hence $a_{parent(S)}$ denotes the number of active children of the parent of S in the original n-node tree.

We now have $\sum_{i:S_i}$ is an "other" cluster $|disc(S_i)| \leq k_oM + k_o + M\sum_{i:S_i}$ is an "other" cluster $a_{parent(S_i)} \leq k_oM + k_o + Mk$, since $\sum_v a_v \leq k$ and no node has two "other" children. Now $\sum_{\text{all } i} |disc(S_i)| \leq (k_s \cdot 1) + (k_t M) + (k_o M + k_o + Mk) \leq k + 2kM$. □

Note. Via Lemma 4, we can guarantee that $M = 1$ and hence, by Lemma 5, that $D = 3K$.

Definition 10. *Let* $W_0 = \lceil 10D \ln(\max\{K, 1/\varepsilon, 10\})/\varepsilon \rceil$.

In the rest of this section we will prove the following theorem.

Theorem 2. *The tree Z produced by the algorithm is a k-node summary tree for T^w having (binary) entropy at least $OPT_k(T^w) - \varepsilon$, provided that W is chosen large enough that $W \geq D/K$ and that for $\eta = D/W$,*

$$\left(\frac{2}{\ln 2}\right)\eta\,(1 + \ln K - \ln \eta) \leq \varepsilon.$$

The least such W is at most W_0.

Let us first analyze the running time.

Theorem 3. *The running time of the algorithm is* $O((K^3/\varepsilon) n \log(\max\{K, 1/\varepsilon\}))$.

Proof. $\sum_{i=1}^n w_i' = W \leq W_0$. The running time of the exact algorithm is $O(K^2 nW)$ and W_0 is $O((K/\varepsilon)\log(\max\{K, 1/\varepsilon\}))$. □

Definition 11. *For sequences $\langle p_1, p_2, ..., p_k \rangle$ and $\langle q_1, q_2, ..., q_k \rangle$ of the same length, k, of nonnegative reals summing to 1, let $H^e(p) = -\sum_{i=1}^k p_i \ln p_i$, where "$0 \ln 0$" is taken to be 0, and let $||p - q||_1 = \sum_{i=1}^k |p_i - q_i|$.*

To prove Theorem 2, we need a lemma, equation (55) in [Nau04], which is a quantitative version of the statement that almost identical probability distributions on $\{1, 2, ..., k\}$ have almost identical entropy.

Lemma 6. *[Nau04, equation (55)] For $2 \leq k \leq K$, sequences $\langle p_1, p_2, ..., p_k \rangle$, $\langle q_1, q_2, ..., q_k \rangle$ of the same length of nonnegative reals summing to 1, and $\gamma \leq k$ such that $||p - q||_1 \leq \gamma$,*

$$|H^e(p) - H^e(q)| \leq \gamma(1 + \ln K - \ln \gamma).$$

We need a simple lemma whose proof uses Lemma 6. First we need a few definitions, which deal with two different k-node summary trees. (We will apply this lemma to the optimal k-node summary trees for $\langle w_1, w_2, ..., w_n \rangle$ and $\langle w_1', w_2', ..., w_n' \rangle$.) Definition 12 and Lemma 7 essentially state that for a fixed partition of $\{1, 2, ..., n\}$ into k parts, the associated probability distributions defined by w and w' will have almost the same entropy, provided that $\eta = D/W$ is small. Quantitatively the lemma tells us how large W must be, in order to guarantee error less than ε.

Definition 12.

1. *Let $\langle S_1', S_2', ..., S_k' \rangle$ be a partition of $\{1, 2, ..., n\}$ given by a k-node summary tree. Let $\hat{s}_j = \sum_{i \in S_j'} w_i$ and let $\hat{p}_j = \hat{s}_j / W$. Analogously, for the w''s, let $s_j' = \sum_{i \in S_j'} w_i'$ and $p_j' = s_j'/W$.*
2. *Let $\langle S_1, S_2, ..., S_k \rangle$ be a second partition of $\{1, 2, ..., n\}$ given by a k-node summary tree. Just as above, let $s_j = \sum_{i \in S_j} w_i$ and let $p_j = s_j/W$, and analogously let $\bar{s}_j = \sum_{i \in S_j} w_i'$ and $\bar{p}_j = \bar{s}_j / W$.*
3. *Last, let $\Delta = \eta(1 + \ln K - \ln \eta)$, where recall from Theorem 2 that $\eta = D/W$.*

Now we are ready for our lemma.

Lemma 7.

$$|H^e(\hat{p}) - H^e(p')| \leq \Delta \tag{3}$$

and

$$|H^e(p) - H^e(\bar{p})| \leq \Delta. \tag{4}$$

Proof. We have

$$||\hat{p} - p'||_1 = \sum_{j=1}^k |\hat{p}_j - p_j'||$$

$$= \frac{1}{W} \sum_{j=1}^k \left| \left(\sum_{i \in S_j'} w_i \right) - \left(\sum_{i \in S_j'} w_i' \right) \right|$$

$$= \frac{1}{W} |disc(\langle S_1', S_2', ..., S_k' \rangle)| \leq \frac{D}{W} = \eta.$$

By Lemma 6,

$$|H^e(\hat{p}) - H^e(p')| \leq \eta\,(1 + \ln K - \ln \eta) = \Delta.$$

Similarly,

$$||p - \bar{p}||_1 = \sum_{j=1}^k |p_j - \bar{p}_j||$$

$$= \frac{1}{W} \sum_{j=1}^k \left| \left(\sum_{i \in S_j} w_i \right) - \left(\sum_{i \in S_j} w_i' \right) \right|$$

$$\leq \frac{1}{W} |disc(\langle S_1, S_2, ..., S_k \rangle)| \leq \frac{D}{W} = \eta.$$

By Lemma 6,

$$|H^e(p) - H^e(\bar{p})| \leq \eta\,(1 + \ln K - \ln \eta) = \Delta. \quad \square$$

Here is the proof of Theorem 2.

Proof. Let $\langle S_1', S_2', ..., S_k' \rangle$ be the partition of $\{1, 2, ..., n\}$ defined by a k-node summary tree of maximum entropy for weights $\langle w_1', w_2', ..., w_n' \rangle$. Similarly, let $\langle S_1, S_2, ..., S_k \rangle$ be the partition of $\{1, 2, ..., n\}$ defined by a k-node summary tree of maximum entropy for weights $\langle w_1, w_2, ..., w_n \rangle$. Equation

(4) shows that there is a k-node summary tree for $T^{w'}$ (using $\langle S_1, S_2, ..., S_k \rangle$) of entropy at least $H^e(\bar{p}) \geq H^e(p) - \Delta = OPT^e_k(T^w) - \Delta$, where $OPT^e_k(T^w) = (\ln 2) OPT_k(T^w)$ is the optimal value of H^e over k-node summary trees of T^w. Therefore

$$OPT^e_k(T^{w'}) \geq H^e(\bar{p}) \geq OPT^e_k(T^w) - \Delta. \tag{5}$$

It follows that the entropy $H^e(T')$ of the output tree (which has weights derived from w, not w') satisfies $H^e(T) = H^e(\hat{p}) \geq H^e(p') - \Delta$ (by (3)), which equals $OPT^e_k(T^{w'}) - \Delta \geq (OPT^e_k(T^w) - \Delta) - \Delta$ (by (5)), which equals $OPT^e_k(T^w) - 2\Delta$. Converting now from natural to binary entropy, we have $H_w(T') \geq OPT_k(T^w) - \left(\frac{2}{\ln 2}\right)\Delta$. Now it is a simple matter to choose W to be the least positive integer at least D/k (so that $\eta = D/W \leq k$) such that

$$\left(\frac{2}{\ln 2}\right)\frac{D}{W}\left(1 + \ln K + \ln\frac{W}{D}\right) \leq \varepsilon.$$

The reader can verify that the optimal W satisfies $W \leq W_0$. □

Since $g(x) = (D/x)(1 + \ln K + \ln(x/D))$ is decreasing on $(D/K, \infty)$, one can use binary search on $[\lceil D/K \rceil, W_0]$ to find the smallest integer W in that interval with $g(W) \leq \varepsilon$.

10. Proof of Lemma 2 (Section 5.1)

Here we prove the inductive step of computing $g_v(l,k,w)$ for $k = 1, ..., K-1$ and $w = -1, 0, 1, ..., W$ given the following quantities:

1. the values of $g_v(l-1,k,w)$ for $k = 1, ..., K-1$ and $w = -1, 0, 1, ..., W$.

2. the values of $F_{v_l}(j)$ for $j = 1, ..., K-1$ (the entropies of the maximum entropy j-node summary trees rooted at v_l for $j = 1, ..., K-1$).

3. s_{v_l}, the size of node v_l.

Proof. It is clear that for $l \geq 2$, $g_v(l, 1, s_{v_1} + s_{v_2} + \cdots + s_{v_l}) = 0$ and $g_v(l, 1, w) = -\infty$ for all other w, $-1 \leq w \leq W$.

Now suppose $l, k \geq 2$. For part 2 of Lemma 2, a maximum entropy k-node summary forest for the the union of the first l children and having no "other" node must consist of a maximum entropy summary forest for the first $l-1$ children, which has no "other" node, and having some number k_1 of nodes, together with a maximum entropy $(k-k_1)$-node summary tree for the subtree rooted at the lth child.

For part 3 of Lemma 2, consider a maximum entropy summary forest for the union of the subtrees rooted at the first l children, having an "other" node of weight $w \geq 0$. Let Z be the set represented by the "other" node. By the definition of a summary tree, either $Z \cap V(T_{v_l}) = \emptyset$ (which is case (a)), or $Z = V(T_{v_l})$ (which is case (b)), or $Z \underset{\neq}{\supset} V(T_{v_l})$ (which is the most complicated case, (c)).

In case (a), we must have a summary forest for the union of the first $l-1$ children having some number k_1, $1 \leq k_1 \leq k-1$, of nodes (all such possible values for k_1 being valid), together with a summary tree on $k - k_1$ nodes having no "other" node for T_{v_l}. (If $k - k_1 = 1$, the summary tree for T_{v_l} may or may not contain an "other" node; it doesn't matter.) The summary forest and the summary tree must both be of maximum entropy, as otherwise, by equation (2), the final tree would not have maximum entropy. That the formula given in part (a) is correct follows from the computation of the entropy of the resulting summary forest.

In case (b), we must have a summary forest for the first $l-1$ children, which has no "other" node, which is combined with a 1-node summary tree for T_{v_l} which has an "other" node of weight $w = s_{v_l}$. The formula is correct since it simply gives the entropy of the resulting summary forest.

Case (c) is tricky. We had a summary forest of the union of the subtrees rooted at the first $l-1$ children, one node of which was an "other" node, plus one tree, T_{v_l}, all of whose nodes form one "other" node. We have to "merge" the "other" node of the summary forest for the first $l-1$ children with the set $V(T_{v_l})$, to get an enlarged "other" node (and a k-node summary forest for the first l children). It follows that the summary forest of the union of the first $l-1$ children must have had k nodes.

Computing the entropy of the new summary forest is not trivial. Specifically, let $M = s_{v_1} + \cdots + s_{v_{l-1}}$ be the sum of the weights of all nodes in the k-node summary forest for the union of the first $l-1$ children, including the one "other" node. Let H be the entropy of that k-node summary forest.

The surprising lemma that makes dynamic programming feasible is that the entropy of the summary forest in which the set $V(T_{v_l})$ is merged with the "other" node of the first $l-1$ children, to get an enlarged "other" node, is given by a function of H, M, w and s_{v_l} alone (and doesn't depend, for example, on the individual weights of the nodes of the previous k-node summary forest). The result is similar to Equation (2). The calculation is as follows.

$$\begin{aligned} H &= g_v(l-1, k, w - s_{v_l}) \\ &= -\sum_{i=1}^{k} \frac{u_i}{M} \lg\left(\frac{u_i}{M}\right), \end{aligned}$$

by the definition of entropy, where we use u_i to denote the individual weights of the k nodes in the maximum entropy k-node summary forest for the union of the subtrees of the first $l-1$ children. Next, we rewrite the entropy by separating the "other" node from the sum:

$$H = -\sum_{i=1}^{k-1} \frac{u_i}{M} \lg\left(\frac{u_i}{M}\right) - (w - s_{v_l}) \lg\left(\frac{w - s_{v_l}}{M}\right),$$

where the weight of the "other" node is $w - s_{v_l}$.

Next, we write the entropy of the (new) summary forest for the union of the first l children, where the subtree rooted at the lth child has been merged into the "other" node under v:

$$H' = \frac{-1}{M + s_{v_l}} \left[\sum_{i=1}^{k-1} u_i \lg\left(\frac{u_i}{M + s_{v_l}}\right) + w \lg\left(\frac{w}{M + s_{v_l}}\right) \right].$$

Note that this summary forest still has k nodes, the weight of the new, merged "other" node is w, and the total weight of the summary forest is $M + s_{v_l}$.

We focus on the summation in the expression of the entropy of the new summary forest. The key to getting a dynamic program to work is to be able to compute this summation without knowing the individual u_i's. To do this, we write

$$\sum_{i=1}^{k-1} u_i \lg\left(\frac{u_i}{M + s_{v_l}}\right) = M \sum_{i=1}^{k-1} \frac{u_i}{M} \lg\left(\frac{u_i}{M} \frac{M}{M + s_{v_l}}\right),$$

which can be simplified to

$$M \sum_{i=1}^{k-1} \frac{u_i}{M} \lg\left(\frac{u_i}{M}\right) + M \sum_{i=1}^{k-1} \frac{u_i}{M} \lg\left(\frac{M}{M + s_{v_l}}\right).$$

The left term can be written in terms of H, the entropy of the old summary forest:

$$M \sum_{i=1}^{k-1} \frac{u_i}{M} \lg\left(\frac{u_i}{M}\right) = M \left[-H - (w - s_{v_l}) \lg\left(\frac{w - s_{v_l}}{M}\right) \right].$$

The right term can also be simplified:

$$M \sum_{i=1}^{k-1} \frac{u_i}{M} \lg\left(\frac{M}{M + s_{v_l}}\right) = \lg\left(\frac{M}{M + s_{v_l}}\right)(M - w + s_{v_l}),$$

because the sum of the $k-1$ non-"other" weights is simply $M - w + s_{v_l}$.

Now the expression for the entropy of the new k-node summary forest for the union of the first l children can be expressed as a function of H, M, w, and s_{v_l}, and further simplification leads to the formula in part 3(c) of Lemma 2 of the main paper. □

11. A bad example for greedy

An interesting question is, how much smaller than the optimal entropy can the entropy obtained from the greedy heuristic be? Here we give an example for which the heuristic returns a 4-node summary tree of entropy only $2/3$ that of the optimal 4-node summary tree.

Let T be a tree on $\{1, 2, ..., 7\}$, with node 1 as the root, having edges $\{1,2\}$, $\{1,3\}$, $\{1,4\}$, $\{2,5\}$, $\{3,6\}$, and $\{4,7\}$. Nodes 1, 2, 4, and 5 have weight 0. Nodes 3 and 6 have weight 1, and node 7 has weight 2. Sorting the children of node 1 into nondecreasing order by size gives $\langle 2, 3, 4 \rangle$. However, there is a 4-node summary tree of entropy 1.5 which has clusters $\{1\}$, $\{3\}$, $\{6\}$, $\{2,4,5,7\}$. The entropy associated with this tree is $H(1/4, 1/4, 2/4) = 2 \cdot (1/4) \lg 4 + (1/2) \lg 2 = 1.5$. The greedy algorithm produces the following vectors for $K = 4$:

1. $\text{elist}_2 = \langle 0, 0, -\infty, -\infty \rangle$.
2. $\text{elist}_3 = \langle 0, 1, -\infty, -\infty \rangle$.
3. $L_3 = \langle 0, 0, 1, 1 \rangle$.
4. $\text{elist}_4 = \langle 0, 0, -\infty, -\infty \rangle$.
5. $L_4 = \langle 0, 1, 1, 1.5 \rangle$.
6. Final output entropy vector, after attaching the root: $\langle 0, 0, 1, 1 \rangle$.

Hence, for $k = 4$, the optimal algorithm obtains 1.5 bits of entropy, as contrasted with the 1 bit obtained by the heuristic, thereby obtaining $2/3$ of the available entropy.

However, we have no example for which greedy obtains only $2/3$ of the optimal entropy, *when the optimal entropy goes to infinity*. Nor do we know if there is any fixed positive lower bound on the ratio between the entropy obtained by greedy and the optimal entropy, the so-called *performance ratio*.

References

[Cha00] CHAZELLE B.: *The Discrepancy Method: Randomness and Complexity*. Cambridge University Press, New York, NY, USA, 2000. 1

[Doe04] DOERR B.: Linear discrepancy of totally unimodular matrices. *Combinatorica 24*, 1 (January 2004), 117–125. 2

[Nau04] NAUDTS J.: Continuity of a class of entropies and relative entropies. *Reviews in Mathematical Physics 16*, 6 (2004), 809–822. 1, 3

[Spe94] SPENCER J.: *Ten Lectures on the Probabilistic Method*, 2 ed. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1994. 1