A DEFERRED PROOFS OF SUBSECTION 3.1

Proof of Lemma 1. We first note that the removal of any edge creates two binary trees. Next we show how to find an edge satisfying the rest of the properties.

Given the rooted tree T, we travel down the tree from the root such that we always pick the child that contains more data points in its subtree (compared to the other child, if another child exists). We denote the i'th node along this path that contains exactly two children, by u_i for $i \in \{1, 2, \ldots\}$. Furthermore, we denote the sets of data points contained by its two children by A_i and B_i such that, $|A_i| \geq |B_i|$.

Let $k^* := \arg\min_i\{|B_1| + \cdots + |B_i| \geq \frac{n}{3}\}$. Since $|A_{k^*}| + |B_1| + \cdots |B_{k^*}| = n$, we are guaranteed that $|A_{k^*}| \leq \frac{2n}{3}$. On the other hand, since $|A_{k^*}| \geq |B_{k^*}|$ and $|A_{k^*}| + |B_{k^*}| = n - (|B_1| + \cdots |B_{k^*-1}|)$ we are also guaranteed that, $|A_{k^*}| \geq \frac{n}{3}$.

Therefore, removing the edge between u_{k^*} and its child associated with A_{k^*} guarantees that the resulting trees each have at most at least $n/3$ data points thereby completing the proof. □

Proof of Lemma 2. Let n denote the number of data points in T. We define the following recursive algorithm: for any binary tree instance T find the edge given by Lemma 1. Remove said edge and continue recursively on both resulting trees. Stop once the input tree has less than $3\epsilon n$ data points.

The algorithm is clearly polynomial. Let F denote the set of resulting edges. Due to our stopping condition, every tree in $T - F$ contains between ϵn and $3\epsilon n$ data points. Therefore, $\frac{1}{4\epsilon} + 1 \leq |F| \leq \frac{1}{\epsilon}$ for $\epsilon < 1/12$. □

Proof of Lemma 3. Let T be some tree on n nodes and let ℓ denote some leaf. We prove by induction on n. If $n = 1$ or $n = 2$ clearly we are done. Otherwise, traverse T starting at ℓ (i.e., hopping from a node to one of its untravelled neighbours). If during this traversal we arrive at a leaf before we arrive at a node with degree ≥ 3, then $|V_3| = 0$ and we are done. Otherwise let u denote the first node we traverse with degree ≥ 3. Remove all nodes in the traversal upto but not including u, denote the new tree as T'.

Thus, $|V_3| \leq |V_3'| + 1$ and $|L| - 1 = |L'|$. Furthermore, since T' has at most $n - 1$ nodes we may use our induction hypothesis. Therefore,

$$|V_3| \leq |V_3'| + 1 \leq |L'| = |L| - 1.$$

□

Proof of Lemma 4. We first note that a node is a leaf in T^R if and only if it was a leaf in T (since every contracted connected component either contained data points or will have a child following the contraction). Next, we categorize the internal nodes of T^R. These nodes are either colored (green or blue), or they are a contracted node or an auxiliary node. We denote the set of each such nodes by G,B,C and A respectively.

It is not hard to see that the second part of our lemma holds. This is due to the fact that by Remark 1 every node in G,B and C has at most 2 immediate children. For nodes in A, by Lemma 2 and by A's definition, we are guaranteed that any such node has at most $3\epsilon n$ children.

In order to show the first part of the lemma we bound each of the four sets of nodes. By the definition of B, $|B| \leq 2/\epsilon$. By definition of A, $|A| \leq |C|$. Furthermore, every node in C has a parent that is colored green or blue and thus due to Remark 1, $|C| \leq 2(|G| + |B|)$. Therefore, $|A| + |C| \leq 4(|G| + |B|)$.

Next we bound $|G|$. In order to do so, we first simplify T^R in a way that does not affect $|G|$. Since no auxiliary node contains green nodes in their subtree, we may detach them without affecting any green or blue nodes. Furthermore, this removal upholds the fact that any green node's degree is at least 3 (since we did not remove any blue nodes). We then also remove any contracted node which now happens to be a leaf (since they too, do not affect the green or blue nodes).

Therefore, in the resulting tree, any leaf must be blue and any green node must have degree at least 3. Thus, if we denote by V_3 the set of vertices with degree ≥ 3 and by L the set of leaves, then,

$$|G| \leq |V_3| \leq |L| - 1 \leq |B| - 1,$$

where the second inequality is due to Lemma 3. Thus,

$$|A| + |C| + |G| + |B| \leq 5(|G| + |B|) \leq 10|B| \leq 20/\epsilon.$$

Now, in order to show the complement (i.e., T^R contains $\Omega(1/\epsilon)$ internal nodes) it is enough to consider Lemma 2 thereby concluding the proof. $\square$

B DEFERRED PROOFS AND DEFINITIONS OF SUBSECTION 3.2

Observation 3. *Due to Fact 1 if we denote by T^O our optimal solution, then since our instance is ρ, τ-weighted we get,*

$$rev(T^O) \geq \frac{\rho \tau n^3}{3},$$

for some smaller, yet still constants ρ and τ.

Proof of Lemma 6. Let T_{alg} denote the tree returned by Algorithm 3. Furthermore denote by α_ℓ and β_{ij} the real values of T^R_ϵ. Therefore,

$$\begin{aligned}
rev(T_{alg}) &\geq \sum_{i \leq j} \sum_{\ell \in S} \big((\alpha_\ell - n\epsilon^2 - n\epsilon_{err}) \\
&\quad \cdot (\beta_{ij} - n^2\epsilon^3 - n^2\epsilon_{err})\big) \\
&\geq \sum_{i \leq j} \sum_{\ell \in S} (\alpha_\ell \beta_{ij}) - \sum_{i \leq j} \sum_{\ell \in S} (\beta_{ij} n\epsilon^2) \\
&\quad - \sum_{i \leq j} \sum_{\ell \in S} (\beta_{ij} n\epsilon_{err}) - \sum_{i \leq j} \sum_{\ell \in S} (\alpha_\ell n^2 \epsilon^3) \\
&\quad - \sum_{i \leq j} \sum_{\ell \in S} (\alpha_\ell n^2 \epsilon_{err}) \\
&\geq \big(\sum_{i \leq j} \sum_{\ell \in S} \alpha_\ell \beta_{ij}\big) - n^3\epsilon^2 20k - n^3\epsilon_{err} 20k \\
&\quad - n^3\epsilon^3(20k)^2 - n^3\epsilon_{err}(20k)^2 \\
&= \big(\sum_{i \leq j} \sum_{\ell \in S} \alpha_\ell \beta_{ij}\big) - n^3\big(\epsilon^2 20k \\
&\quad + \epsilon^3(20k)^2 + \epsilon_{err} 20k + \epsilon_{err}(20k)^2\big) \\
&\geq rev(T^R_\epsilon) - n^3(421\epsilon + 20k\epsilon_{err} + 400k^2\epsilon_{err}),
\end{aligned}$$

where the first inequality follows from the property tester's guarantees and the fact that we did not guess α_ℓ and β_{ij} to their exact values. The third inequality follows since there are at most k sets in the partition, $\sum \beta_{ij} \leq n^2$ and $\sum \alpha_\ell \leq n$. The last inequality is due to the fact that $k \leq 1/\epsilon + 1$ and ϵ is chosen to be small enough.

Due to Observation 3, Theorem 1 and by choosing $\epsilon_{err} = \frac{\epsilon^3}{400}$, we get,

$$\begin{aligned}
rev(T_{alg}) &\geq rev(T^R_\epsilon) - n^3(O(\epsilon)) \\
&\geq rev(T^R_\epsilon) - \frac{O(\epsilon)}{\rho\tau} rev(T^O) \\
&\geq (1 - O(\epsilon) - \frac{O(\epsilon)}{\rho\tau}) rev(T^O).
\end{aligned}$$

Thus by choosing ϵ small enough, we get the desired result. $\square$

C DEFERRED PROOFS AND DEFINITIONS OF SUBSECTION 3.3

Proof of Theorem 3. Let $w_{ij} = g(d_{ij})$ and let $w'_{ij} = w_{ij} + \epsilon$. Denote by O and O' the trees which generate the maximal revenue with respect to w_{ij} and w'_{ij} respectively. Finally, given an HC tree T, let $Rev(T)$ and $Rev'(T)$ denote the revenue generated by T with respect to w_{ij} and w'_{ij} respectively.

By Theorem 2 we are guarnateed that for any constant $\delta > 0$, $Rev'(A) \geq (1-\delta)Rev'(O')$. Furthermore, by the definitions of O and O' we have that $Rev'(O') \geq Rev'(O)$. Therefore,

$$Rev'(A) \geq (1-\delta)Rev'(O') \geq (1-\delta)Rev'(O). \tag{3}$$

By Fact 3 and since $w_{ij} + \epsilon = w'_{ij}$ we are guaranteed that for any tree T, $Rev(T) = Rev'(T) - \epsilon \frac{n}{3}\binom{n}{2}$. Combining this with equation 3 we get that,

$$\begin{aligned} Rev(A) &= Rev'(A) - \epsilon \frac{n}{3}\binom{n}{2} \\ &\geq (1-\delta)Rev'(O) - \epsilon \frac{n}{3}\binom{n}{2} \\ &= (1-\delta)Rev(O) - \delta\epsilon \frac{n}{3}\binom{n}{2}. \end{aligned}$$

Let α denote the diameter of the metric. Since the metric is scale invariant we may assume w.l.o.g. that $\alpha = 1$. By the definition of the doubling dimension, $D(M) = D$, there are $2^{D(\ell+1)}$ balls of radius $\frac{1}{2^{\ell+1}}$ that cover the entirety of the data. Let x_i denote the number of data points that belong to the i'th ball but not to balls $1, \ldots, i-1$. Therefore, $\sum_{i=1}^{2^{D(\ell+1)}} x_i = n$. On the other hand by Cauchy-Schwarz inequality, $\sum_{i=1}^{2^{D(\ell+1)}} x_i^2 \geq \frac{n^2}{2^{D(\ell+1)}}$. Therefore, the number of pairs of data points within the same ball is $\sum_{i=1}^{2^{D(\ell+1)}} \binom{x_i}{2} \geq \frac{n^2}{2^{D(\ell+1)+1}} - \frac{n}{2}$. Due to the fact that pairs of points that belong to the same ball are at distance of at most $\frac{1}{2^\ell}$ and since similarity function g is defined an non-increasing, we get that,

$$\begin{aligned} \sum_{i,j} w_{ij} &\geq g(\frac{1}{2^\ell}) \sum_{i=1}^{2^{D(\ell+1)}} \binom{x_i}{2} \\ &\geq g(\frac{1}{2^\ell})(\frac{n^2}{2^{D(\ell+1)+1}} - \frac{n}{2}). \end{aligned} \tag{4}$$

By Fact 1 and equation 4 we are guaranteed that for $c = \frac{2^{D(\ell+1)}}{g(\frac{1}{2^\ell})}$, $c\delta\epsilon Rev(O) \geq \delta\epsilon\frac{n}{3}\binom{n}{2}$. Combining the above,

$$Rev(A) \geq (1 - \delta - c\delta\epsilon)Rev(O).$$

Due to the fact that $g(0) = 1$ and that g is ℓ-Lipschitz continuous, $g(\frac{1}{2^\ell}) = \Omega(1)$. On the other hand since $D = O(1)$ and $\ell = O(1)$ we may choose ϵ and δ small enough in order to guarantee an EPRAS. ☐

D DEFERRED PROOFS OF SUBSECTION 4.1

Proof of Lemma 7. Consider the proof of Lemma 4. The only difference between T^R and T^D (with respect to the number of their internal nodes) is the fact that in T^D the contracted nodes are multiplied by $1/\epsilon$ (and therefore the auxiliary nodes as well). Thus, clearly the lemma holds. ☐

Proof of Lemma 8. In order to prove the lemma we consider the following observations. The first of which is Observation 1 which holds here as well. The second is the following.

Observation 4. *Consider any two data points, i and j, that are contained in the same contracted node in $K(T^O)$. Further assume that they end up under different auxiliary nodes. Therefore, any descendant of the corresponding contracted node (in $K(T^O)$) is contained in T^D_{ij}.*

Consider two data points in T^O, i and j and consider some $k \in T^O_{ij}$. As before, we denote their lca's by v_{ik}, v_{jk} and v_{ij} and assume without loss of generality that i is clustered first with k and therefore, $v_{ij} = v_{kj}$.

We would like to bound the number of k's for which $k \notin T^D_{ij}$. As before, let $\{T^{B\cup G}_\ell\}$ denote the set of trees defined by $T^O - (B \cup G)$ and let $T^{B\cup G}_i$ (resp. $T^{B\cup G}_j$ and $T^{B\cup G}_k$) denote the tree in $T^O - (B \cup G)$ containing i (resp. j and k). If $k \in T^{B\cup G}_i$ or $k \in T^{B\cup G}_j$ then since the number of data points contained in these trees is at most $6\epsilon n$, we may disregard such k's and incur an additive loss of $6\epsilon n$. Therefore, we assume, $k \notin T^{B\cup G}_i$ and $k \notin T^{B\cup G}_j$.

Thus, we split into the following cases. The first is the case where v_{jk} is green/blue. Otherwise, this means that v_{jk} has at most one child with a blue descendant. It can not be the child containing j since that would mean that $k \in T^{B\cup G}_i$. Thus, we may only consider the following final cases: either exists a green/blue node on the path $v_{ik} \to v_{ij}$ or there must exist a green/blue node both on the path $k \to v_{ik}$ and on the path $i \to v_{ik}$ (since $k \notin T^{B\cup G}_i$). Otherwise, exists a green/blue node on the path $k \to v_{ik}$ and not on the path $i \to j$.

We prove our lemma for each of these cases.

1. v_{jk} is green/blue: Due to Observation 1 we are guaranteed that $k \in T^D_{ij}$.
2. There exists a green/blue node on the path $v_{ik} \to v_{ij}$: Due to Observation 1 we are guaranteed that $k \in T^D_{ij}$.
3. There exists a green/blue node both on the path $k \to v_{ik}$ and on the path $i \to v_{ik}$: In this case v_{ik} is green/blue and therefore, again due to Observation 1 we are guaranteed that $k \in T^D_{ij}$.
4. There exists a green/blue node on the path $k \to v_{ik}$ and not on the path $i \to j$: In this case i and j are in the same contracted node in $K(T^O)$. If they end up under different auxiliary nodes, then by Observation 2 $k \in T^D_{ij}$. Since we partitioned the data points in the contracted nodes randomly (under restriction that the sets are of the same size), the probability that i and j will end up under different auxiliary nodes is $\geq (1-\epsilon)$.

Thus, in any case, $E[|T^D_{ij}|] \geq (1-\epsilon)|T^O_{ij}| - 6\epsilon n$. □

Proof of Theorem 4. Lemma 7 guarantees the first bullet. For the second bullet, denote by T^O the optimal solution. We note that T^O is binary. Furthermore, due to Lemma 8 and Fact 2, we get,

$$\begin{aligned} E[dis(T^D)] &= \sum_{i<j} w_{ij} E[|T^D_{ij}|] \\ &\geq \sum_{i<j} w_{ij}((1-\epsilon)|T^O_{ij}| - 12\epsilon n) \\ &= (1-\epsilon)dis(T^O) - 12\epsilon n \sum_{i<j} w_{ij} \\ &\geq (1-38\epsilon)dis(T^O). \end{aligned}$$

Since the expectation is over trees with our desired characteristics (i.e., constant number of internal nodes and each node contains a small number of children), we deterministically take T^D to be the tree maximizing the expectation. Thus, by choosing $\epsilon' = \epsilon/38$ we get the desired result. □

E DEFERRED ALGORITHMS OF SUBSECTION 4.2

Algorithm 5 EPRAS for the dense dissimilarity case.

```
Enumerate over all trees, $T$, with $k$ internal leaves.
for each such $T$ do
  for $\{\alpha_i\}_{i\leq k} \subset \{i\epsilon^2 n : i \in \mathbb{N} \wedge i \leq \frac{3}{\epsilon}\}$ do
    for $\{\beta_{ij}\}_{i\leq k, j\leq k} \subset \{i\epsilon^3 n^2 : i \in \mathbb{N} \wedge i \leq \frac{9}{\epsilon}\}$ do
      Run $PT(\{\alpha_i\}, \{\beta_{ij}\}, \epsilon_{err} = \epsilon^3, \delta)$.
  Compute the dissimilarity based on $T$ and $PT$'s output.
Return the maximal dissimilarity tree encountered.
```

F DEFERRED PROOFS OF SECTION 5

Proof of Proposition 1. For each vertex $v \in V$, our algorithm maintains scores $s(v)$ which are initially set to zero. The algorithm will actually remove the node of largest score at each step and recurse on the remaining vertices, hence producing a caterpillar tree (a tree whose every internal node has at least one leaf). A similar greedy strategy to the one described below can also produce a tree (not necessarily caterpillar) in a bottom-up fashion by repeatedly merging node pairs. Notice that the algorithm is deterministic.

For every edge (i,j) of similarity weight w^s_{ij}, decrease $s(i)$ and $s(j)$ by $\frac{n-2}{2}w^s_{ij}$, and increase every other score $s(k)$ by w^s_{ij}, where $k \in V \setminus \{i,j\}$. The intuition behind such assignments, is that for a pair i,j of similarity w^s_{ij}, whenever we remove another node k first, k's contribution to the *hcc* objective increases by w^s_{ij}, as k lies outside of the lowest common ancestor between i,j. Similarly, for every edge (i,j) of dissimilarity w^d_{ij}, we increase $s(i)$ and $s(j)$ by $\frac{n}{2}w^d_{ij}$, and decrease every other score $s(k)$ by w^d_{ij}, where $k \in V \setminus \{i,j\}$.

Next, let $u \in V$ have the largest score and $V' = V \setminus \{u\}$. Remove u and any adjacent edges from the graph, then recursively construct a tree T'_1 restricted on V' for its leaves (if $|V'| = 2$, just output the unique binary tree on the two nodes). The final output of the algorithm is a new tree T_1 with one child being u and the other child being the root of T'_1.

We now prove correctness: Let u as above and let $w^s_u = \sum_{(u,v)} w^s_{uv}, w^d_u = \sum_{(u,v)} w^d_{uv}, W^s = \sum_{(i,j)} w^s_{ij}, W^d = \sum_{(i,j)} w^d_{ij}$. Notice that according to the scoring rule of our algorithm:

$$s(u) = (W^s - w^s_u) - \tfrac{n-2}{2}w^s_u - (W^d - w^d_u) + \tfrac{n}{2}w^d_u$$

Note that by induction, tree T'_1 that has $n-1$ leaves, satisfies the conclusion of the proposition:

$$hcc(T'_1) \geq \tfrac{1}{3}(n-3)(W^s - w^s_u) + \tfrac{2}{3}(n-1)(W^d - w^d_u) \tag{5}$$

Since u had the largest score, it follows that $s(u) \geq 0$. Therefore:

$$(W^s - w^s_u) - (W^d - w^d_u) \geq \tfrac{n-2}{2}w^s_u + \tfrac{n}{2}w^d_u$$

We add $\frac{1}{2}[(W^s - w^s_u) - (W^d - w^d_u)]$ to both sides:

$$(W^s - w^s_u) - (W^d - w^d_u) \geq \tfrac{1}{3}(hcc^s_u - (W^d - w^d_u) - nw^d_u)$$

where $hcc^s_u = (n-2)w^s_u + (W^s - w^s_u)$ is the total contribution u can have due to similarity weights in any tree. By rearranging terms:

$$(W^s - w^s_u) + nw^d_u \geq \tfrac{1}{3}hcc^s_u + \tfrac{2}{3}hcc^d_u \tag{6}$$

where $hcc^d_u = (W^d - w^d_u) + nw^d_u$ is the total contribution u can have due to dissimilarity edges in any tree.

Let $hcc_u(T_1)$ be the contribution towards the *hcc* objective of node u in T_1 and observe we can easily compute this quantity as u got removed first. In other words, $hcc_u(T_1) = (W^s - w^s_u) + nw^d_u$, as any dissimilarity edge $(u,\cdot)$ has a lowest common ancestor of size n and for every similarity edge $(i,j), i,j \neq u$, u is a non-leaf of T_{ij}. Summing up eq. (5) and (6), and noting that $hcc(T_1) = hcc_u(T_1) + hcc(T'_1)$ concludes the proof. □

Proof of Theorem 6. A simple calculation suggests that the expected value for `HCC` is at least:

$$\min_p \left\{p \cdot \tfrac{1}{3} + 0.585 \cdot (1-p), p \cdot \tfrac{2}{3} + (1-p) \cdot \tfrac{1}{3}\right\}$$

By balancing the two terms, the minimum is achieved when the parameter $p = 1 - \frac{\frac{1}{3}}{0.585}$ and the final approximation factor becomes 0.4767. □

Proof of Theorem 7. There are two cases to consider: either $\sum_e w^d_e \geq \sum_e w^s_e$ or $\sum_e w^d_e \leq \sum_e w^s_e$. We first consider the case that $\sum_e w^d_e \geq \sum_e w^s_e$ (the second is handled symmetrically). We rewrite the objective function for some HC tree T.

$$\begin{aligned} hcc^{\pm}(T) &= \sum_e w_e^d(T_e) + \sum_e w_e^s(n - T_e) \\ &= \sum_e w_e^d(T_e) + \sum_e (1 - w_e^d)(n - T_e) \\ &= 2\sum_e w_e^d(T_e) + \sum_e (n - T_e) - n\sum_e w_e^d \\ &= 2\sum_e w_e^d(T_e) + \frac{1}{3}n\binom{n}{2} - n\sum_e w_e^d, \end{aligned}$$

where the last equality follows from Fact 3. We first observe that a tree that maximizes the dissimilarity instance defined by w_e^d is a tree that maximizes the original $\texttt{HCC}^{\pm}$ objective. Let O^d denote the tree maximizing the dissimilarity objective and let O denote the tree maximizing the $\texttt{HCC}^{\pm}$ objective. By Theorem 5 we know that for any constant $\epsilon > 0$ algorithm 7 (denoted henceforth as ALG) generates dissimilarity of at least $(1 - \epsilon)\sum_e w_e^d(O_e^d) = (1 - \epsilon)\sum_e w_e^d(O_e)$. Therefore, for any $\epsilon > 0$,

$$\begin{aligned} hcc^{\pm}(ALG) &= 2\sum_e w_e^d(ALG_e) + \frac{1}{3}n\binom{n}{2} - n\sum_e w_e^d \\ &\geq 2(1-\epsilon)\sum_e w_e^d(O_e) \\ &+ \frac{1}{3}n\binom{n}{2} - n\sum_e w_e^d \\ &= (1-2\epsilon)\sum_e w_e^d(O_e) \\ &+ \sum_e w_e^d(O_e) + \frac{1}{3}n\binom{n}{2} - n\sum_e w_e^d \\ &\geq (1-2\epsilon) \\ &\cdot \Big(\sum_e w_e^d(O_e) + \frac{1}{3}n\binom{n}{2} - n\sum_e w_e^d\Big) \\ &= hcc^{\pm}(O), \end{aligned}$$

where the last inequality follows from Fact 2.

The case that $\sum_e w_e^d \leq \sum_e w_e^s$ is solved symmetrically (using Theorem 2 and Fact 1) which concludes the proof. $\square$

G DEFERRED PROOFS OF SECTION 6

Proof of Theorem 8. Note that clearly the problem is in NP (since given a tree its revenue may be checked efficiently), therefore we only need to show that it is NP-hard.

Ahmadian et al. (2019) showed that the unweighted revenue case is APX-hard under the Small Set Expansion hypothesis. This in turn guarantees that the unweighted revenue problem is NP-hard assuming the Small Set Expansion. Next we show how to reduce an unweighted revenue instance to a dense unweighted revenue instance (in polynomial time).

Roughly speaking we will simply add a disconnect clique of size n to the general graph. Formally, let $G = (D, E_D, w)$ denote a general revenue instance such that, $D = \{d_1, \ldots, d_n\}$. We convert G to a dense instance $G' = (V, E_V, w')$ simply by adding a clique of size n (disconnected from V) with similarities of size 1. We denote this clique's set of nodes by $L = \{\ell_1, \ldots, \ell_n\}$. Therefore, $w'(\ell_i, \ell_j) = 1, w'(d_i, d_j) = w(d_i, d_j)$ and $w'(\ell_i, d_j) = 0$.

Clearly G' is dense. Let T' denote the optimal solution to G'. It is known that the optimal tree first cuts the disconnected components of G'. Therefore, there exists a node u in T' such that the subtree rooted at u contains the entirety of L and no data points from D. Since D is disconnected from L and due to the definition of the revenue goal function, taking u and moving it to the top of T' (formally, if r' is the root of T', then we create a new root, r and attach u and r' as its immediate children), can only increase T''s revenue. Thus, we may assume w.l.o.g. that in T' the root already disconnects L and D.

Let v_D and v_L denote T''s root's immediate children containing D and L respectively. Let T'_D denote the subtree rooted at u_D. T'_D is clearly optimal for instance G (since otherwise, we could have replaced T'_D with the optimal tree for G, thereby increasing T''s revenue, contradicting the fact that it is optimal).

Thus, we converted, in polynomial time, the optimal tree for G' to the optimal tree for G, proving that the dense revenue problem is NP-hard. □

Definition 6. *We say that an unweighted graph is complement-dense if its complement graph (i.e., the graph we get by removing all existing edges and adding all missing edges) is dense.*

Lemma 10. *The problem of finding a maximal revenue tree for revenue instances which are complement-dense is NP-complete (assuming the Small Set Expansion hypothesis).*

Proof. Note that clearly the problem is in NP (since given a tree its revenue may be checked efficiently),therefore we only need to show that it is NP-hard.

As in Theorem 8, we reduce an unweighted revenue instance to a complement-dense unweighted revenue instance. Specifically we do this by adding a disconnected path of length n^2 to the original graph. Formally, let $G = (D, E_D, w)$ denote a general revenue instance such that, $D = \{d_1, \ldots, d_n\}$. We convert G to a complement-dense instance $G' = (V, E_V, w')$ simply by adding a path of size n^2 (disconnected from V) with similarities of size 1. We denote this path's set of nodes by $L = \{\ell_1, \ldots, \ell_{n^2}\}$. Therefore, $w'(\ell_i, \ell_{i+1}) = 1, w'(d_i, d_j) = w(d_i, d_j)$ and $w'(\ell_i, d_j) = 0$. Note that G' is clearly complement-dense.

As in the proof of Theorem 8 exists a node u in the optimal solution of G', T', such that u contains the entirety of L and no data points from D. Again, we may move u and its subtree to the root of T' thereby only increasing the revenue. Thus, given T' we may take its child that contains D as our optimal tree for G. □

Observation 5. *Since the problem of finding a minimal (Dasgupta) cost tree is the dual problem of the revenue problem, the unweighted, complement-dense Dasgupta cost problem is NP-complete (assuming the Small Set Expansion hypothesis).*

Proof of Theorem 9. Note that clearly the problem is in NP (since given a tree its dissimilarity may be checked efficiently), therefore we only need to show that it is NP-hard. We do this by reducing the unweighted, complement-dense Dasgutpa cost problem to this problem.

Roughly speaking we simply consider the complement graph of the HC instance. Formally, given a complement-dense HC instance $G = (V, E, w)$ we define its complement as $G_c = (V_c, E_c, w_c)$. Therefore, for any edge e, $w_c(e) = 1 - w(e)$. Thus,

$$\begin{aligned}\min_T cost_G(T) &= \min_T \sum w(e)|T_e| \\ &= \min_T \sum (1 - w_c(e))|T_e|.\end{aligned}$$

Dasgupta (2016) proved that for any binary tree T and for any HC instance which is a clique H its cost is fixed and $cost_H(T) = \frac{1}{3}(|V(H)|^3 - |V(H)|)$. Since the optimal tree for this cost function is in fact binary we get,

$$\min_T \sum (1 - w_c(e))|T_e| =$$
$$\frac{1}{3}(|V(G)|^3 - |V(G)|) - \max_T \sum w_c(e)|T_e|.$$

Since w defines a complement-dense instance, w_c defines a dense instance. Thus, we reduced our original problem to $\max_T \sum w_c(e)|T_e|$ such that w_c is dense, thereby completing the proof. □

Proof of Theorem 10. The theorem is proven simply by rewriting the $\texttt{HCC}^{\pm}$ objective in terms of either revenue or dissimilarity (choosing that which contributes more to the total weight) as in the proof of Theorem 7 and then using Theorems 8 and 9. □