Minimum Robinson-Foulds Distance Supertree

Xilin Yu, Thien Le, Sarah Christensen, Erin Molloy, Tandy Warnow

June 4, 2019

Throughout the paper, we consider only unrooted trees. For any tree T, let $V(T)$, $E(T)$, and $L(T)$ denote the vertex set, the edge set, and the leaf set of T, respectively. For any $v \in V(T)$, let $N_T(v)$ A tree is *fully resolved* if every non-leaf node has degree 3. Let $\mathcal{T}_S$ denote the set of all fully resolved trees on leaf set S. In any tree T, each edge e induces a bipartition $\pi_e := A|B$ of the leaf set, where A and B are the leaves in the two components of $T - e$, respectively. A bipartition $A|B$ is non-trivial if both sides have size at least 2. For a tree T, $C(T) := \{\pi_e \mid e \in E(T)\}$ denotes the set of all bipartitions of T. For a fully resolved tree with n leaves, $C(T)$ contains $2n - 3$ bipartitions, exactly $n - 3$ of which are non-trivial. A tree T' is a *refinement* of T if T can be obtained from T' by contracting a set of edges. Equivalently, T' is a refinement of T if and only if $C(T) \subseteq C(T')$.

Two bipartitions π_1 and π_2 of the same leaf set are *compatible* if and only if there exists a tree T such that $\pi_1, \pi_2 \in C(T)$. The following theorem and corollary give other categorizations of compatibility.

Theorem 1 (Theorem 2.20 of [1])**.** *A pair of bipartitions $A|B$ and $A'|B'$ of the same set is compatible if and only if at least one of the four pairwise intersections $A \cap A'$, $A \cap B'$, $B \cap A'$, $B \cap B'$ is empty.*

Corollary 1. *A pair of bipartitions $A|B$ and $A'|B'$ of the same set is compatible if and only if one side of $A|B$ is a subset of one side of $A'|B'$.*

A tree T restricted to a subset R of its leaf set, denoted $T|_R$, is the minimal subtree of T spanning R with nodes of degree two suppressed. A bipartition $\pi = A|B$ restricted to a subset $R \subseteq A \cup B$ is $\pi|_R = A \cap R|B \cap R$. We have the following intuitive lemma with its proof in the appendix.

Lemma 1. *Let T be a tree with leaf set S and let $\pi = A|B \in C(T)$ be a bipartition induced by $e \in E(T)$. Let $R \subseteq S$.*

1. *If $R \cap A = \emptyset$ or $R \cap B = \emptyset$, then $e \notin E(T|_R)$.*
2. *If $R \cap A \neq \emptyset$ and $R \cap B \neq \emptyset$, then for any $\pi' \in C(T|_R)$ induced by $e' \in E(T|_R)$, $\pi|_R = \pi'$ if and only if $e \in P(e')$.*

Corollary 2. *Let T be a tree with leaf set S and let $\pi = A|B \in C(T)$ be a bipartition induced by $e \in E(T)$. Let $R \subseteq S$ such that $R \cap A \neq \emptyset$ and $R \cap B \neq \emptyset$. Then $\pi|_R \in C(T|_R)$.*

Definition 1. *For two trees T, T' with the same leaf set, the bipartition support of them is $bisup(T, T') := |C(T) \cap C(T')|$.*

Let T_1 and T_2 be two fully resolved trees on leaf sets S_1 and S_2, respectively, such that $X := S_1 \cap S_2 \neq \emptyset$. Let $S := S_1 \cup S_2$. The maximum bipartition support supertree problem on two input trees, abbreviated MAX-BISUP-SUPERTREE-2, finds a fully resolved supertree T^* on leaf set S that maximizes the sum of the bipartition support of T^* with respect to T_1 and T_2. That is,

$$
\begin{aligned}
T^* &= \operatorname*{argmax}_{T \in \mathcal{T}_S} bisup(T|_{S_1}, T_1) + bisup(T|_{S_2}, T_2) \\
&= \operatorname*{argmax}_{T \in \mathcal{T}_S} |C(T|_{S_1}) \cap C(T_1)| + |C(T|_{S_2}) \cap C(T_2)|.
\end{aligned}
$$

We call $bisup(T|_{S_1}, T_1) + bisup(T|_{S_2}, T_2)$ the support score of T when T_1 and T_2 are clear from context.

The more general maximum bipartition support supertree problem on a set of N input trees, abbreviated MAX-BISUP-SUPERTREE-N, takes in a set of input trees $T_1, T_2, \dots, T_N$ with leaf sets $S_1, S_2, \dots, S_N$, respectively. MAX-BISUP-SUPERTREE-N finds a fully resolved supertree T^* on leaf set S that maximizes the sum of the bipartition support of T^* with respect to every input tree. That is,

$$
T^* = \operatorname*{argmax}_{T \in \mathcal{T}_S} \sum_{i \in [N]} bisup(T|_{S_i}, T_i)
$$

We first set up the notations for the algorithm and the analysis. Let T_1, T_2, S_1, S_2, and X be defined as from the problem statement. Let $T_1|_X$ and $T_2|_X$ be the backbone trees of T_1 and T_2, respectively. Let Π be the set of bipartitions of X. Let Triv and NonTriv denotes the set of trivial and non-trivial bipartitions in $C(T_1|_X) \cup C(T_2|_X)$. For each $e \in E(T_i|_X)$, $i \in \{1, 2\}$, let $P(e)$ denote the path in T_i from which e is obtained by suppressing all degree-two nodes. Let $w(e)$ be the number of edges on $P(e)$.

We define a weight function $w : \Pi \to \mathbb{N}_{\geq 0}$ such that for any bipartition π of X, $w(\pi) = w(e_1) + w(e_2)$, where e_i induces π in $T_i|_X$ for $i \in \{1, 2\}$. If for any $i \in \{1, 2\}$, no e_i exists that induces π in $T_i|_X$, then we use $w(e_i) = 0$.

For each $i \in \{1, 2\}$ and each $e \in E(T_i|_X)$, let $\mathrm{In}(e)$ be the set of internal nodes of $P(e)$. For each $v \in \mathrm{In}(e)$, let $L(v)$ be the set of leaves in $S_i \backslash X$ whose connecting path to the backbone tree $T_i|_X$ goes through v and let $T(v)$ be the minimal subtree spanning $L(v)$ in T_i. We say $T(v)$ is an extra subtree attached to v.

We let the node u which is the neighbor of v in $T(v)$ be the root of $T(v)$. Let $\mathcal{T}(e) := \{T(v) \mid v \in \mathrm{In}(e)\}$. Then $\mathcal{T}(e)$ is the set of extra subtrees attached to internal nodes of $P(e)$ in T_i. We note that $|\mathcal{T}(e)| = |\mathrm{In}(e)| = w(e) - 1$. For any bipartition $\pi \in C(T_1|_X) \cup C(T_2|_X)$, we denote $\mathcal{T}(\pi) := \mathcal{T}(e_1) \cup \mathcal{T}(e_2)$, where e_i is the edge that induces π in $T_i|_X$ for $i \in \{1, 2\}$ if $\pi \in C(T_i|_X)$. Let $\mathrm{Extra}(T_i) := \bigcup_{e \in E(T_i|_X)} \mathcal{T}(e)$. Then $\mathrm{Extra} := \mathrm{Extra}(T_1) \cup \mathrm{Extra}(T_2)$ denotes the set of all extra subtrees in T_1 and T_2.

Algorithm 1 Max-BiSup Supertree

Input: two fully resolved trees T_1, T_2 with leaf sets S_1 and S_2 where $S_1 \cap S_2 = X \neq \emptyset$
Output: a fully resolved supertree T on leaf set $S = S_1 \cup S_2$ that maximizes the support score

compute $C(T_1|_X)$ and $C(T_2|_X)$
for each $\pi \in C(T_1|_X) \cup C(T_2|_X)$ **do**
    compute $\mathcal{T}(\pi)$ and $w(\pi)$
construct T by having a star of leaf set X with center vertex $\hat{v}$ and connecting the root of each $t \in \mathrm{Extra}$ to $\hat{v}$, let $\hat{T} = T$
for each $\pi \in \mathrm{Triv}$ **do**
    $T \leftarrow \text{Refine-Triv}(T_1, T_2, T, \pi, \hat{v}, \mathcal{T})$
let $\tilde{T} = T$
construct the incompatibility graph $G = (V_1 \cup V_2, E)$, where $V_1 = C(T_1|_X) - C(T_2|_X)$ and $V_2 = C(T_2|_X) - C(T_1|_X)$, and $E = \{(\pi, \pi') \mid \pi \in V_1, \pi' \in V_2$, π is not compatible with $\pi'\}$
compute the maximum weight independent set I in G with weight w
let $H(\hat{v}) = \mathrm{NonTriv} \cap (C(T_1|_X) \cup C(T_2|_X))$
for each $\pi \in \mathrm{NonTriv} \cap (C(T_1|_X) \cup C(T_2|_X))$ **do**
    $sv(\pi) = \hat{v}$
for each $\pi \in \mathrm{NonTriv} \cap (I \cup (C(T_1|_X) \cap C(T_2|_X)))$ **do**
    $T \leftarrow \text{Refine}(T_1, T_2, T, \pi, H, sv, \mathcal{T})$
let $T^* = T$
refine T arbitrarily at polytomies until it is fully resolved
return T

For the analysis of the algorithm, we differentiate between two kinds of bipartitions in $C(T_1) \cup C(T_2)$. Let $\Pi_Y = \{\pi = A|B \in C(T_1) \cup C(T_2) \mid$ either $A \cap X = \emptyset$, or $B \cap X = \emptyset\}$. Let $\Pi_X = \{\pi = A|B \in C(T_1) \cup C(T_2) \mid A \cap X \neq \emptyset$ and $B \cap X \neq \emptyset\}$. Intuitively, Π_X is the set of bipartitions in $C(T_1) \cup C(T_2)$ that are induced by edges in the backbone trees $T_1|_X$ and $T_2|_X$ while Π_Y is the set of bipartitions in $C(T_1) \cup C(T_2)$ that are induced by edges inside or connecting extra subtrees of T_1 and T_2. It follows by definition that Π_X and Π_Y is a disjoint decomposition of $C(T_1) \cup C(T_2)$.

Let $p_X(T)$ and $p_Y(T)$ (we omit the parameters T_1 and T_2 for brevity) be the contributions to the support score of T from bipartitions of Π_X and Π_Y for any $T \in \mathcal{T}_S$, respectively. Formally, we have

$$p_X(T) = |C(T|_{S_1}) \cap C(T_1) \cap \Pi_X| + |C(T|_{S_2}) \cap C(T_2) \cap \Pi_X|,$$
$$p_Y(T) = |C(T|_{S_1}) \cap C(T_1) \cap \Pi_Y| + |C(T|_{S_2}) \cap C(T_2) \cap \Pi_Y|.$$

By definition of support score, any bipartition can only contribute to the support score if it is in $C(T_1) \cup C(T_2)$. Thus, the support score of T equals $p_X(T)+p_Y(T)$ for any tree T on leaf set S. Therefore, it is enough for us to show that Algorithm 1 finds a tree T that maximizes both $p_X(T)$ and $p_Y(T)$ at the same time.

Claim 1. *If Algorithm 1 returns a tree T such that $p_X(T) \geq p_X(T')$ and $p_Y(T) \geq p_Y(T')$ for any tree T' with leaf set S, then Algorithm 1 solves* Max-Bisup-Supertree-2 *correctly.*

Lemma 2. *For any tree T of leaf set S and any refinement T' of T, $p_X(T') \geq p_X(T)$ and $p_Y(T') \geq p_Y(T)$.*

Lemma 3. *For any tree T of leaf set S, $p_Y(T) \leq |\Pi_Y|$. In particular, let $\hat{T}$ be the tree constructed in Algorithm 1. Then, $p_Y(\hat{T}) = |\Pi_Y|$.*

Claim 2. *Let $\hat{T}$ be the tree constructed in Algorithm 1, then $p_X(\hat{T}) = 2|X|$.*

Lemma 4. *Let $\pi = A|B$ be a bipartition of X. Let T be a tree of leaf set S such that $\pi \notin C(T|_X)$ and all bipartitions in $C(T|_X)$ are compatible with π. Let T' be a refinement of T such that for all $\pi' \in C(T'|_{S_i}) \backslash C(T|_{S_i})$ for some $i \in \{1, 2\}$, $\pi'|_X = \pi$. Then, $p_X(T') - p_X(T) \leq w(\pi)$.*

Lemma 5. *For any compatible set F of bipartitions of X, let T be a tree of leaf set S such that $C(T|_X) = F$. Then $p_X(T) \leq \sum_{\pi \in F} w(\pi)$.*

Claim 3. *Let $\tilde{T}$ be the tree constructed in Algorithm 1, then $p_X(\tilde{T}) = \sum_{\pi \in \mathrm{Triv}} w(\pi)$.*

Lemma 6. *Let T be a tree from Algorithm 1 before a refinement step. Let $\pi = A|B \in \mathrm{NonTriv} \cap (I \cup (C(T_1|_X) \cap C(T_2|_X)))$. Let T' be a refinement of T obtained from running Algorithm Refine on T and π, with the auxiliary data structures H, sv, and $\mathcal{T}$. Then, $p_X(T') - p_X(T) = w(\pi)$.*

Let G be the incompatibility graph defined in Algorithm 1 and I be the maximum weight independent set in G with weight function w. Let $G' = (V_1' \cup V_2', E')$ be another incompatibility graph such that $V_1' = C(T_1|_X)$ and $V_2' = C(T_2|_X)$, and $E' = \{(\pi, \pi') \mid \pi \in V_1', \pi' \in V_2',\ \pi \text{ is not compatible with } \pi'\}$. Let $I' := I \cup (C(T_1|_X) \cap C(T_2|_X))$.

Claim 4. *I' is a maximum weight independent set in G' with weight function w.*

Claim 5. *Let T^* be the tree defined in Algorithm 1, $p_X(T^*) \geq p_X(T)$ for any tree T of leafset S.*

Theorem 2. *Algorithm 1 correctly solves* Max-Bisup-Supertree-2 *in $O(n^3)$ (Checking this) time.*

Theorem 3. Max-Bisup-Supertree-3 *is NP-hard.*

References

[1] Tandy Warnow. *Computational phylogenetics: an introduction to designing methods for phylogeny estimation.* Cambridge University Press, 2017.