Computing Manhattan Path-Difference Median Trees: a Practical Local Search Approach

Alexey Markin and Oliver Eulenstein

Abstract—Median tree problems are powerful tools for inferring large-scale phylogenetic trees that hold enormous promise for society at large. Such problems seek a median tree for a given collection of input trees under some problem-specific distance. Here, we introduce a median tree problem under the classic Manhattan path-difference distance. We show that this problem is NP-hard, devise an ILP formulation, and provide an effective local search heuristic that is based on solving a local search problem exactly. Our algorithm for the local search problem improves asymptotically by a factor of n on the best-known (naïve) solution, where n is the overall number of taxa in the input trees. Finally, comparative phylogenetic studies using considerably large empirical data and an accuracy analysis for smaller phylogenetic trees reveal the ability of our novel heuristic.

Index Terms—Phylogenetic trees, median trees, supertrees, path-difference distance, local search, Manhattan norm.

1 Introduction

A basic tenet of all biological disciplines is the common history of all life forms, including all extant species. The evolutionary relationships among such entities are usually represented as a bifurcating tree, which is referred to as a *species tree*. Large-scale species trees for thousands of entities hold enormous promise for society at large. While such trees are fundamental throughout biology, their analysis and predictive power is also benefitting various other disciplines, such as agronomy, biochemistry, epidemiology, environmental sciences, and medical sciences [1], [2], [3], [4]. At the same time inferring large-scale species trees confronts us with some of the most difficult computational challenges raised in the field of evolutionary biology today.

Prior to the genomic era, a species tree for a given set of species was built by constructing an evolutionary tree from a common gene sampled from those species. Such trees are called *gene trees*. The implicit assumption of this approach is that the evolution of the chosen gene mimics the evolution of the species themselves. A major problem, however, is that gene trees for distinct genes can be *discordant* for the same set of species, which introduces ambiguity for the resolution of the actual species tree [5]. Such discordance is frequently caused by erroneous gene trees, but can also be caused by genes that have evolved differently due to complex evolutionary processes that have shaped the species' genomes.

The advent of new sequencing technologies provided an unprecedented wealth of new gene sequences, holding the promise of building more resolved and less biased species trees [6]. In contrast to building a species tree based on only one gene tree, discordance can now be averaged out based on the cumulative evidence from thousands of genes sampled from across the species genomes [7], [8].

Median tree problems have emerged as a powerful tool for assembling large-scale species trees from discordant gene trees, which may represent the evolution of different sets of species [5]. Such problems seek a median tree for a given set of input (gene) trees based on some problem-specific distance; that is, a tree with the minimum overall distance to the input trees. Median tree measures can be categorized into mathematically or biologically informed ones. In the latter case these measures are based on biological models that explain discordance in the input trees using evolutionary processes, such as gene duplication and deep coalescence. In contrast, mathematically oriented measures attempt to optimize the evolutionary information that is common to the input trees based on various elementary representations of the evolutionary relationships in the trees, e.g., triplets, clusters, or path-distances; such measures are typically useful when there are erroneous input trees.

Median tree problems have been studied extensively for many of the well-established distances in comparative phylogenetics. Notable exceptions, however, are some of the oldest metrics in comparative phylogenetics, namely the path-difference distances. Any phylogenetic tree can be encoded as a so-called *path-length vector*, where entries represent path-lengths between two distinct leaves. Given a pair of trees, *path-difference distances* measure the distance between the two corresponding path-length vectors. The distance can be measured under various vector norms, such as the Manhattan norm (also known as the Taxicab norm) and the Euclidean norm; the corresponding metrics are called the *Manhattan path-difference distance* and the *Euclidean path-difference distance* respectively. The appeal of path-difference distances is that, unlike many other distances in comparative phylogenetics [9], they can be used to compare any type of trees, including rooted trees, unrooted trees, and non-binary trees. In fact, while most distances are not defined to compare trees with branch-lengths, path-difference distances allow the comparison of such trees. Further, the path-difference distance is not tied to a biological model, which can be seen as an advantage, as it suggests a broader scope of applications of the path-difference median trees.

Despite the appeal and tradition of path-difference dis-

• *A. Markin and O. Eulenstein are with the Department of Computer Science, Iowa State University, Ames, IA, 50011.*
E-mail: {amarkin, oeulenst}@iastate.edu

Manuscript received ?; revised ?

tances the analysis of median trees under these distances is still in its infancy. Only recently the median tree problem under the Euclidean path-difference distance has been analyzed. While this problem is NP-hard, local search heuristics have been developed that allow the inference of credible species tree estimates [10]. However, median tree problems under any other norm for the path-differences have not been analyzed.

This work focuses on the computation of median trees under the Manhattan path-difference distance. We demonstrate that the respective *Manhattan median tree problem* is NP-hard and we employ the classic approach of designing a local search heuristic to address it. By designing an efficient local search algorithm we made our method applicable to significantly large datasets; which in turn allowed us to present an extensive practice-oriented applicability study of the heuristic approaches to the problem. The software for our local search heuristic is freely available from the webpage http://genome.cs.iastate.edu/ComBio/software.htm.

1.1 Related Work

There has been a large body of work focusing on the biological, mathematical, and algorithmic properties of median trees adopting various definitions of distance measures that have been effectively used in comparative phylogenetics [5]. Path-difference distances describe some of the oldest such distances [9], [11], [12], [13] and are defined by a distance-specific norm of the difference between the path-lengths vectors of the two trees whose distance is measured. Each of these *vectors* represents the pairwise distances between the leaves of the corresponding tree that is the number of edges on the simple path between the leaves. Steel and Penny [9] have studied the distribution of the Euclidean (L_2 norm) path-difference distance for unrooted trees. Later, Mir and Rosello [14] computed the mean value of this distance for fully resolved unrooted trees with n leaves, and showed that this mean value grows in $O(n^3)$. Recently, it was shown that the median tree problem under this distance is NP-hard, and effective local search heuristics have been developed [10]. Variants of path-difference distances have been described, including the Manhattan (L_1 norm) path-difference distance [15], and their correlation [16].

While most median tree problems are NP-hard [5], they have been effectively addressed by local search heuristics [17], [18], [19], [20], [21], which have provided credible estimates of large-scale species trees [17], [18]. Such a search heuristic starts with some initial candidate species tree, and finds a tree with the minimum overall distance to the given input trees in the local neighborhood of the initial tree. This constitutes a *local search step*. The tree found in a local search step becomes the starting point for the following local search step, and so on, until a local minima is reached, which is reported by the heuristic. To find a tree with minimum distance in a local search step an instance of a *(local) neighborhood search problem* is solved exactly. The time complexity of this neighborhood search problem depends on the tree edit operation that defines the local neighborhood, as well as on the computation time of the tree distance measure that is used. The classic *subtree prune and regraft (SPR)* tree edit operation where a subtree is pruned from the edited tree and then regrafted back into this tree at another location has been well-studied [22]. The *SPR neighborhood* of a tree T is the set of all trees into which T can be transformed by at most one SPR edit operation, which consists of $\Theta(n^2)$ trees, where n is the size of T. Now, solving the neighborhood search problem for the Manhattan median tree problem naïvely requires $\Theta(kn^4)$ time, since the best-known algorithm to compute the Manhattan pairwise distance for one candidate tree and k input trees requires $\Theta(kn^2)$ time. However, computing large-scale species trees requires typically solving thousands of instances of the neighborhood search problem, and a neighborhood search runtime of $\Theta(kn^4)$ becomes prohibitive.

The effectiveness of standard local search heuristics is typically highly dependent on the choice of the starting tree. Traditionally, greedy heuristics are employed to efficiently construct well-fitting starting trees; that is, trees that consistently have a significantly smaller distance to the input trees than a randomly chosen tree. An extension of this approach, a *hybrid heuristic*, was introduced as a method to improve the power of the traditional approach by applying the local search heuristic in multiple stages of constructing the starting tree itself [10].

1.2 Contribution

Inspired by the well acknowledged study of the Euclidean median tree problem, we introduce the Manhattan median tree problem. This problem is NP-hard for its rooted and unrooted variants, since, as we prove, each of these variants contains an NP-hard problem as a special case. Consequently, we apply a standard SPR based local search heuristic to the Manhattan median tree problem. Since the naïve complexity of local search iterations does not allow us to apply a Manhattan median tree heuristic in practice, we designed an efficient algorithm that exploits structural properties of the SPR operation. This algorithm is based on the analysis of the alternation of the path-length vector of a candidate median tree introduced by an SPR iteration. While this analysis is independent from the vector norm of the path-difference distance used as an objective, the algorithms that exploit the corresponding properties in order to accelerate the local search step were found to depend heavily on the vector norm used. It turned out that the core precomputation algorithm developed for the Euclidean neighborhood search problem [10] is not applicable if we use the Manhattan norm instead. Hence, we designed a more sophisticated preprocessing approach to address the *Manhattan neighborhood search* problem.

The novel preprocessing algorithm makes extensive use of dynamic programming and allows us to efficiently answer more elaborate queries to path-difference matrices as required by the Manhattan median tree heuristic. This algorithm solves an instance of the local neighborhood search problem in $O(kn^3)$ time, which is a significant improvement in comparison to the best-known naïve approach (namely, a factor of n), especially when dealing with large-scale tree assembly problems. We highlight the asymptotic runtime improvement in practice by presenting a scalability study comparing the naïve algorithm with our improved algorithm as applied to a Manhattan local search heuristic in Section 6.1 of our experimental evaluation .

As mentioned above, the hybrid heuristic was introduced as a powerful framework for practical median tree estimation. However, practitioners are faced with the problematic task of identifying suitable heuristics and their corresponding parameters to infer best-possible phylogenetic estimates for their studies. Consequently, we exhibit a pioneer extensive evaluation of two major types of the hybrid heuristic under different parameters on three baseline phylogenetic datasets. This experimental study analyzes both the accuracy and time-efficiency of these heuristics. While this study compares the performance of the Manhattan median tree heuristics amongst themselves, we also deducted a comparative study involving other popular supertree methods and different cost functions (distance measures) on standard phylogenetic datasets. The study's main purpose is to evaluate the effectiveness of the heuristic approach and compare the Manhattan median tree approach with other popular median tree and supetree methods.

In addition to the heuristics, we introduce an integer linear programming (ILP) formulation of the Manhattan median tree problem that allows us to solve small instances of the problem *exactly*. The search space for the ILP formulation is encoded using the matrix-based tree representation introduced in [23]. Building on top of that we introduced variables and constraints allowing to infer path-lengths between all pairs of leaves. The main challenge in this formulation was to account for the minus method [24] that is used in this work for the path-difference distance calculation. In our final experimental study we further demonstrate the accuracy of our local search heuristic by comparing its results with the exact solutions provided by our ILP formulation on small simulated data sets.

2 Basics and Preliminaries

2.1 Basic definitions

Throughout this paper we adhere to the definitions and notation introduced in [10]. A *(phylogenetic) tree* T is a rooted tree, where each leaf is uniquely labeled with a taxon, each internal node v has exactly two children nodes, denoted by $\mathsf{Ch}_T(v)$, and each node u except for the root has a single parent node denoted by $\mathsf{Pa}_T(v)$. In addition, we denote the node set, edge set and leaf set of T by $V(T)$, $E(T)$ and $\mathsf{L}(T)$ respectively. We denote the root by $\mathsf{Rt}(T)$ and a sibling of each non-root node u by $\mathsf{Sb}(u)$. For convenience we say the root node has an additional (auxiliary) edge coming into it, denoted by $\{\infty, \mathsf{Rt}(T)\}$. We also set $T(v)$ to be a subtree of T rooted at some $v \in V(T)$, and $\overline{T(v)}$ to be a phylogenetic tree obtained by pruning $T(v)$ from T (in a sense, the tree complement of $T(v)$).

We define a partial order $\preceq_T$ on the vertex set $V(T)$, such that $u \preceq v$, if v is a node on the path from u to $\mathsf{Rt}(T)$. Additionally, we say $u \prec v$, if $u \preceq v$ and $u \neq v$. The *least common ancestor (LCA)* of two nodes $u, v \in V(G)$, $\mathsf{LCA}_T(u,v)$, is the furthest from the root node, w, such that $v \prec w$ and $u \prec w$.

A set of leaves $\mathsf{L}(T(v))$ is called a *(lower) cluster* of the node v, and is denoted by C_v. Additionally, we define $\overline{C_v} := \mathsf{L}(\overline{T(v)}) = \mathsf{L}(T) \backslash C_v$ to be the *upper cluster* of the node v. Note that for convenience we identify the leaves in a phylogenetic tree with the respective labels (taxa).

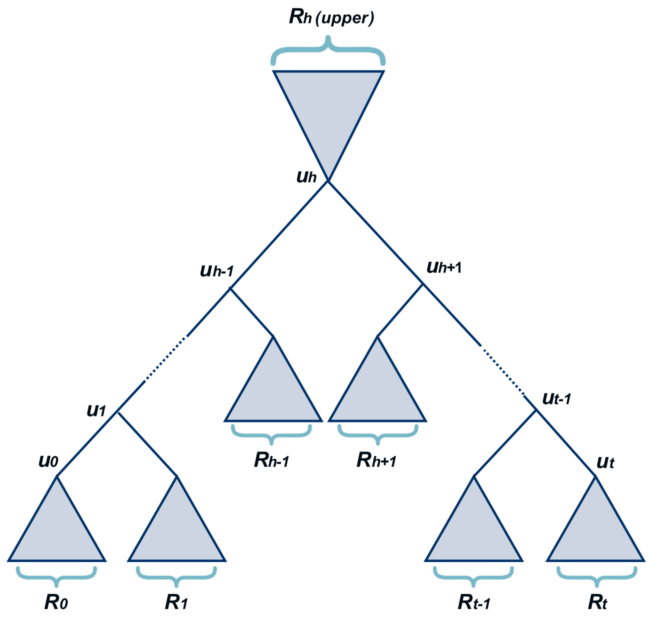


Fig. 1: Path-separation example in regards to a path $(u_0, \ldots, u_t)$; the clusters of the illustrated subtrees are labeled R_0, R_1 through R_t, where $t > 0$. Note that the subtree with the cluster $R_h = \overline{C_{u_h}}$ contains the root of the tree.

Let $L \subseteq \mathsf{L}(T)$ and T' be the minimal subtree of T with the leaf set L. We define the *leaf-induced subtree* $T[L]$ of T to be the tree obtained from T' by successively removing each node of degree two (except for the root) and adjoining its two neighbors (a parent and a child).

Let $\mathcal{P}$ be a set of phylogenetic trees $\{G_1, \ldots, G_k\}$. We extend the definition of a leaf set to a set of trees as follows: $\mathsf{L}(\mathcal{P}) := \cup_{i=1}^{k} \mathsf{L}(G_i)$. A tree S is called a *supertree* of $\mathcal{P}$, if $\mathsf{L}(S) = \mathsf{L}(\mathcal{P})$. A set of trees $\mathcal{P}$ is called *compatible* if there exist a supertree T consistent with every tree in $\mathcal{P}$, and a tree T is *consistent* with a tree G if $T[\mathsf{L}(G)] \equiv G$.

2.2 Path-difference distance

Given a tree T and two leaves $u, v \in \mathsf{L}(T)$, let $\mathsf{d}_{u,v}(T)$ denote the length in edges of the unique path between u and v in T. Let $\mathsf{d}(T)$ be an associated *path-length vector* obtained by a fixed ordering of pairs i, j [9], e.g., $\mathsf{d}(T) = (\mathsf{d}_{1,2}(T), \mathsf{d}_{1,3}(T), \ldots, \mathsf{d}_{n-1,n}(T))$, where n is the number of leaves. Then the *path-difference distance (PDD)* between two trees G and S over the same leaf set is defined (for a fixed $p \in [1, \infty)$) as

$$\mathsf{d}_p(G, S) := ||\,\mathsf{d}(G) - \mathsf{d}(S)||_p,$$

where $||\cdot||_p$ denotes an L_p norm of a vector.

Further, we define $\Delta(G, S)$ to be a *path-difference matrix* of size $|\,\mathsf{L}(T)| \times |\,\mathsf{L}(T)|$. Thats is, $\Delta_{i,j}(G,S) = \mathsf{d}_{i,j}(G) - \mathsf{d}_{i,j}(S)$.

2.3 Path-separation of a phylogenetic tree

Let $P = (u_0, u_1, \ldots, u_t)$ be a simple path in a phylogenetic tree T, then we say that P induces a set of *disjoint* clusters $C_P := \{R_0, R_1, \ldots, R_t\}$, such that

- $R_0 := C_{u_0}$;
- For $0 < i < t$, we have
$$R_i := \begin{cases} \overline{C_{u_i}}, & \text{if } u_{i-1}, u_{i+1} \in \mathsf{Ch}(u_i); \\ C_w, & \text{for } w \in \mathsf{Ch}(u_i) \backslash \{u_{i-1}, u_{i+1}\}. \end{cases}$$
Note that in the second case w is uniquely defined, since either u_{i-1} or u_{i+1} is the parent of u_i and $|\mathsf{Ch}(u_i) \backslash \{u_{i-1}, u_{i+1}\}| = 1$;
- $$R_t := \begin{cases} \overline{C_{u_{t-1}}}, & \text{if } u_t = \mathsf{Pa}(u_{t-1}); \\ C_{u_t}, & \text{otherwise.} \end{cases}$$
- $\bigcup_{i=0}^{t} R_i = \mathsf{L}(T)$.

Note that one of the clusters R_i for $0 < i \leq t$ can be an empty set. A schematic example of the defined here path-separation is shown in Figure 1.

3 Path-difference median trees

Here we introduce a class of path-difference based median tree problems defined under different vector norms.

We extend the definition of the path-difference distance to a set of trees. Note that so far PDD is defined only for two trees over the same leaf set. However, we do not want to enforce such a restriction on the set of input trees, because tree-size variations are typically present in real world data. Therefore, in order to calculate a path-difference distance under an L_p norm between two trees S and G, where $\mathsf{L}(G) \subseteq \mathsf{L}(S)$ we use the *minus method* [24]. That is, we calculate a distance between G and the subtree of S induced by $\mathsf{L}(G)$: $\mathsf{d}_p(S, G) = \mathsf{d}_p(S[\mathsf{L}(G)], G)$. We now define PDD for an input set $\mathcal{P}$ and a supertree S as a sum

$$\mathsf{d}_p(\mathcal{P}, S) := \sum_{i=1}^{|\mathcal{P}|} \mathsf{d}_p(G_i, S[\mathsf{L}(G_i)]).$$

Given that we establish the following general problem:

Problem 1 (PD median tree (for an L_p norm) – decision version).
Instance: A set of input trees $\mathcal{P}$ and a real number q;
Question: Determine whether there exists a supertree S, such that $\mathsf{d}_p(\mathcal{P}, S) \leq q$.

The PD median tree problem is NP-complete.
We show NP-hardness by describing a polynomial time reduction from the MaxRTC problem, which is known to be NP-complete (see [25]).

Problem 2 (Maximum Compatible Subset of Rooted Triplets – MaxRTC).
Instance: A set of rooted triplets R and an integer $0 \leq c \leq |R|$;
Question: Is there a subset $R' \subseteq R$, such that R' is compatible and $|R'| \geq c$.

Where a *rooted triplet* is a phylogenetic tree with exactly three leaves.

Theorem 3.1. *The PD median tree problem under an L_p norm is NP-complete for any $1 \leq p < \infty$.*

Proof. This problem belongs to NP. That is, given a supertree S we can in polynomial time determine whether $\mathsf{d}_p(R, S) \leq q$ by directly calculating the distance $\mathsf{d}_p(R, S)$. Next, we generalize the NP-hardness proof given in [10] for a special case of $p = 2$ to any L_p-norm for $1 \leq p < \infty$

We map an instance $\langle R, c \rangle$ of the MaxRTC problem to an instance $\langle R, 2^{\frac{1}{p}}(|R| - c) \rangle$ of the PD median tree problem. To explain why this transformation works we observe the following. Assume that S is a supertree of a set of rooted triplets $R = \{T_1, \ldots, T_k\}$. Then

$$\mathsf{d}_p(S[\mathsf{L}(T_i)], T_i) = \begin{cases} 0, & \text{if } S \text{ is } \textit{consistent} \text{ with } T_i; \\ 2^{\frac{1}{p}}, & \text{otherwise.} \end{cases}$$

Therefore, $\mathsf{d}_p(R, S) = 2^{\frac{1}{p}}(|R| - c')$, where c' is the number of triplets in R, which are consistent with S. That is, there are at least c' compatible triplets in R, which implies that $\langle R, 2^{\frac{1}{p}}(|R| - c) \rangle$ is a yes-instance for the PD median tree problem if and only if $\langle R, c \rangle$ is a yes-instance of the MaxRTC problem. □

3.1 Unrooted median trees

While in this work we focus on rooted phylogenetic trees, it is not difficult to see that the unrooted version of the PD median tree problem is NP-hard as well. Note that in the unrooted case we seek an *unrooted* phylogenetic supertree S that minimizes the $d_p(\mathcal{P}, S)$ distance. To show NP-hardness we use the classic NP-complete quartet compatibility problem [26].

Problem 3 (Quartet compatibility).
Instance: A set of quartets Q;
Question: Is there a supertree S that is consistent with every quartet in Q.

Where a *quartet* is an unrooted phylogenetic tree with four leaves. We observe that $\mathsf{d}_p(S, Q) = 0$ for a supertree S of Q if and only if S is consistent with *every* quartet in Q. Therefore, the reduction is given by the mapping of an instance $\langle Q \rangle$ of the quartet compatibility problem to the instance $\langle Q, 0 \rangle$ of the unrooted PD median tree problem.

Although results in this section are shown for *sets* of input trees, they easily extend to *multi-sets*. Note that further in this work we focus on an L_1 norm PDD, or as we additionally refer to it, *Manhattan distance*. Additionally, we refer to the PD median tree problem under the L_1 norm as simply *Manhattan median tree problem*

4 Local search for Manhattan median tree problem

NP-hard supertree problems are traditionally approached by standard local search heuristics defined over a tree search space. Tree search space is commonly defined in terms of a tree edit operation. In this paper we focus on *SPR* – one of the most applied tree edit operations (see [27], [28], [29]). Below we introduce the needed definitions and formally state the local neighborhood search problem as applied to the Manhattan median tree problem.

4.1 Local neighborhood search problem

Definition 4.1. *Given a node $v \in V(S)\backslash\{\mathsf{Rt}(S)\}$, and a node $u \in V(S)\backslash(V(S(v))\cup\{\mathsf{Pa}(v)\})$, $SPR_S(v,u)$ is a tree obtained by the following modifications of the tree $S' = S|v$:*

1) *If u is a root of S', then a new root w' is introduced, so that u is a child of w'. Otherwise, an edge $(\mathsf{Pa}(u), u)$ is subdivided by a new node w'.*
2) *Connect the pruned subtree $S(v)$ to the node w'.*

Further, we define the following sets of trees that can be obtained from S by performing SPR:

$$SPR_S(v) := \bigcup_{u} SPR_S(v,u);$$
$$SPR_S := \bigcup_{v,u} SPR_S(v,u).$$

SPR_S is called an *SPR-neighborhood* of a tree S. It is easy to see from the definition that $|SPR_S| = O(n^2)$, where $n = |\mathsf{L}(S)|$.

Given a set of input trees $\mathcal{P} = \{G_1,...,G_k\}$, the search space in an SPR neighborhood search problem could be viewed as a graph $\mathcal{T}$, where nodes represent all existing supertrees of $\mathcal{P}$. $\{S_1,S_2\}$ is an edge in $\mathcal{T}$, if S_1 could be transformed to S_2 with a single SPR operation.

At each iteration local search heuristic finds a supertree S' in the neighborhood of a current tree S, such that S' minimizes the cost function that we are interested in. In case $S \equiv S'$, the local search stops (reaches a local minimum). Otherwise, it proceeds to the next iteration with a tree S'. An instance (single iteration) of the SPR-based local search algorithm could be formalized as the following problem:

Problem 4 (PD (local) neighborhood search).
Instance: An input set $\mathcal{P}$ and a supertree S;
Question: find a tree $S' = \operatorname*{arg\,min}_{S' \in SPR_S} \mathsf{d}_p(\mathcal{P}, S')$.

Naïve algorithm for the neighborhood search problems. Given two trees S and G, one can compute $\mathsf{d}_p(S,G)$ in $O(n^2)$ time for any p. Therefore, direct computation of the $\mathsf{d}_p(\mathcal{P},S')$ score for each $S' \in SPR_S$ would take $O(n^4k)$ time, where $n = |\mathsf{L}(\mathcal{P})|$ and $k = |\mathcal{P}|$. Next, we show how to improve on this complexity under $p = 1$ (*Manhattan* distance).

Let $\mathbf{G} \in \mathcal{P}$ be a fixed input tree, and let S_i be a supertree in the i-th iteration of the local search. Throughout this section we refer to the restricted tree $S_i[\mathsf{L}(G)]$ as simply $\mathbf{S}$.

4.2 Local search environment for PD median trees

The SPR neighborhood of a candidate median tree under the path-difference metric was already well studied for both Manhattan and Euclidean norms [10]. Here we present the structure of the SPR neighborhood under the Manhattan norm in a compact and consistent form.

Let T be a tree in SPR_S obtained by pruning some node $v \in V(S)\backslash\{\mathsf{Rt}(S)\}$ and regrafting it above another node w. It follows that $e_w = \{w, \mathsf{Pa}_S(w)\}$ is the edge, where the new parent of v was placed as a result of the regrafting operation (note that e_w can be the auxiliary edge $\{\infty, \mathsf{Rt}(S)\}$). Then let $U_T = \{v = u_0, u_1, \ldots, u_t\}$ denote the path in S that starts with the node v and ends with e_w (i.e., $e_w = \{u_{t-1}, u_t\}$. This path is illustrated in Figure 2a.

Let $C_{U_T} = \{R_0, \ldots, R_t\}$ be the set of clusters induced by the path U_T (see section 2.3 for more details).

The key observation in our analysis is that the distance between two leaves from a same cluster R_i is not affected by the regrafting operation. Further, the inter-cluster difference is shown in Table 1 (i.e., it depicts the path-difference matrix $\Delta(T,S)$). Having this table, we can find the Manhattan PD distance $\mathsf{d}_1(T,G)$ given the distance $\mathsf{d}_1(S,G)$ for any $T \in SPR_S$. That is, let A, B be two elements (clusters) from the cluster set $\{C_v, R_1, \ldots, R_t\}$ and let Δ denote $\Delta(S,G)$ for brevity, then

$$\mathsf{d}_1(T,G) - \mathsf{d}_1(N,G) = \sum_{\forall\{A,B\}} \sum_{\substack{i \in A \\ j \in B}} |\Delta_{i,j} + d_{A,B}| - |\Delta_{i,j}|$$

$$= \sum_{\forall\{A,B\}} \begin{pmatrix} d_{A,B} \cdot \#\{(i \in A, j \in B) | \Delta_{i,j} \geq -d_{A,B})\} \\ -d_{A,B} \cdot \#\{(i \in A, j \in B) | \Delta_{i,j} < -d_{A,B})\} \\ +2 \displaystyle\sum_{i \in A, j \in B: -d_{A,B} \leq \Delta_{i,j} < 0} \Delta_{i,j} \\ -2 \displaystyle\sum_{i \in A, j \in B: 0 \leq \Delta_{i,j} < -d_{A,B}} \Delta_{i,j} \end{pmatrix}, \tag{1}$$

where $d_{A,B}$ denotes the value in the Table 1 corresponding to the clusters A and B.

4.3 Preprocessing

As the Equation 1 implies, we need to be able to efficiently answer queries of kind: find a sum/count of elements of a certain submatrix of Δ, such that those elements are between specified bounds. In order to do that, we preprocess the tree S and the matrix Δ by precomputing certain sums and counts.

First, we introduce new notation. Let $\mathsf{L}_1, \mathsf{L}_2 \subseteq \mathsf{L}(N)$, then $\Sigma_{\geq}(\mathsf{L}_1, \mathsf{L}_2)$ is a vector indexed from $-(n-2)$ to $n-2$, such that

$$\Sigma_{\geq}(\mathsf{L}_1, \mathsf{L}_2)[x] = \sum_{\substack{i \in \mathsf{L}_1 - \mathsf{L}_2;\ \Delta_{i,j} \geq x \\ j \in \mathsf{L}_2 - \mathsf{L}_1}} \Delta_{i,j}$$

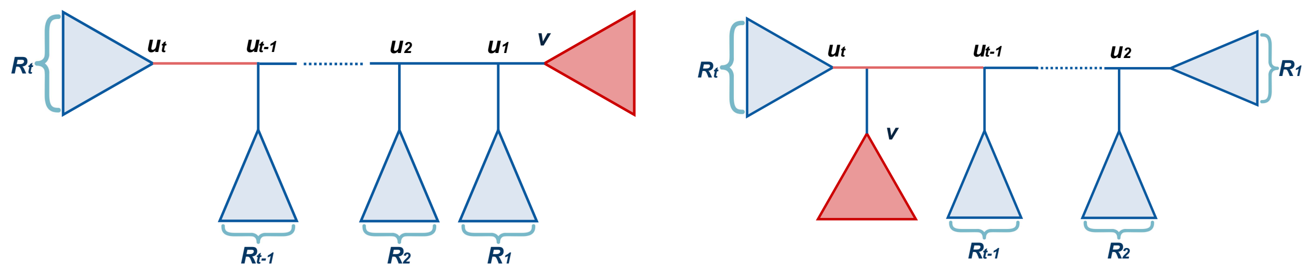
That is, a sum of path-differences (or cophenetic differences) across two subsets of leafs, such that those differences are greater than or equal to a parameter x. Similarly for "counts", we have $\#_{\geq}(\mathsf{L}_1, \mathsf{L}_2)$ is a vector indexed from $-(n-2)$ to $n-2$, such that

$$\#_{\geq}(\mathsf{L}_1, \mathsf{L}_2)[x] = \#\{(i \in \mathsf{L}_1 - \mathsf{L}_2, j \in \mathsf{L}_2 - \mathsf{L}_1) | \Delta_{i,j} \geq x\}$$

Assume now that we have computed the $\Sigma_{\geq}$ and $\#_{\geq}$ vectors for all pairs of *clusters* C_1 and C_2 of S (here we consider both *upper* and *lower* clusters). Then, using Equation 1, one can compute $\mathsf{d}_1(T,G) - \mathsf{d}_1(S,G)$ or $\mathsf{d}_{\phi,1}(T,G) - \mathsf{d}_{\phi,1}(S,G)$ using the following formula:

$$\sum_{\forall\{A,B\}} \begin{pmatrix} d_{A,B} \cdot \#_{\geq}(A,B)[-d_{A,B}] \\ -d_{A,B} \cdot (|A||B| - \#_{\geq}(A,B)[-d_{A,B}]) \\ +2\,\mathsf{sign}(d_{A,B}) \begin{pmatrix} \Sigma_{\geq}(A,B)[\max(-d_{A,B},0)] \\ -\Sigma_{\geq}(A,B)[\min(-d_{A,B},0)] \end{pmatrix} \end{pmatrix}$$

Next, we present an efficient algorithm for computing those vectors for all pairs of clusters in S.



(a) The original tree S. The edge shown in red is e_w.

(b) The tree $T = SPR_S(v, w)$.

Fig. 2: The two figures schematically show the SPR operation and the way it affects the SPR-path U_T. Note that one of the subtrees, except for $S(v)$ that is depicted in red, could contain a root. Additionally, one of these subtrees could be empty.

TABLE 1: Here $1 < p < t$. Values inside the table indicate the difference in path lengths between leaves from different subsets, i.e., for $i \in C_v$ and $j \in R_1$: $\mathsf{d}_{i,j}(T) = \mathsf{d}_{i,j}(S) + t - 2$.

	$\mathbf{C_v}$	$\mathbf{R_1}$	$\dots$ $\mathbf{R_p}$ $\dots$	$\mathbf{R_t}$
$\mathbf{C_v}$	0	$t-2$	$-2p+1$ $+t$	$2-t$
$\mathbf{R_1}$	$t-2$	0	-1	0
$\vdots$				
$\mathbf{R_p}$	$-2p+1$ $+t$	-1	0	1
$\vdots$				
$\mathbf{R_t}$	$2-t$	0	1	0

4.3.1 Pairs of lower clusters

We are going to compute the vectors $\Sigma_{\geq}(C_x, C_y), \#_{\geq}(C_x, C_y)$ for some $x, y \in V(S)$ using dynamic bottom-up approach.

Base case. Assume that x and y are both leaves, then $\Sigma_{\geq}(C_v, C_y)[p] = \Delta_{x,y}$ if $p \leq \Delta_{x,y}$, and 0 otherwise. Similarly, $\#_{\geq}(C_v, C_y)[p] = 1$ if $p \leq \Delta_{x,y}$, and 0 otherwise.

Step up. Without loss of generality assume that x is an internal node (not a leaf), and it has two children c_1, c_2. Then we observe that

$$\Sigma_{\geq}(C_x, C_y) = \Sigma_{\geq}(C_{c_1}, C_y) + \Sigma_{\geq}(C_{c_2}, C_y)$$
$$\#_{\geq}(C_x, C_y) = \#_{\geq}(C_{c_1}, C_y) + \#_{\geq}(C_{c_2}, C_y)$$

4.3.2 Pairs including upper clusters

Note that we are not interested in vectors for a pair of upper clusters, since there is exactly *one* upper cluster among $\{R_1, \dots, R_t\}$. Next, we show that we can compute vectors for pairs, where one of the clusters is upper, using the *top-down strategy*.

Base case. Note that $\Sigma_{\geq}(\overline{C_{\mathsf{Rt}(S)}}, C_x) = \vec{0}$ and $\#_{\geq}(\overline{C_{\mathsf{Rt}(S)}}, C_y) = \vec{0}$ for any $y \in V(S)$, since $\overline{C_{\mathsf{Rt}(S)}} = \emptyset$.

Step down. We observe the following relationship for some node $x \in V(S) \backslash \{\mathsf{Rt}(S)\}$:

$$\Sigma_{\geq}(\overline{C_x}, C_y) = \Sigma_{\geq}(\overline{C_{\mathsf{Pa}_S(x)}}, C_y) + \Sigma_{\geq}(C_{\mathsf{Sb}_S(x)}, C_y)$$
$$\#_{\geq}(\overline{C_x}, C_y) = \#_{\geq}(\overline{C_{\mathsf{Pa}_S(x)}}, C_y) + \#_{\geq}(C_{\mathsf{Sb}_S(x)}, C_y)$$

That is, to compute vectors for some $\overline{C_x}$, we need to know the vectors for the upper cluster of the parent of x and for the lower cluster of the sibling of x.

4.3.3 Preprocessing time complexity

The goal of our preprocessing algorithm is to compute $\Sigma_{\geq}$ and $\#_{\geq}$ vectors for all pairs of clusters in S. Note that there are $\Theta(n)$ clusters in S, where n is the number of taxa. Therefore, we need to compute $\Theta(n^2)$ vectors, and each vector is of length $2(n-2)+1 = \Theta(n)$. Note that we can calculate an entry of each vector in $O(1)$ (constant) time using the dynamic programming approach outlined above. Therefore, the whole preprocessing step takes an order of $\Theta(n^3)$ time.

4.4 Overall neighborhood search time complexity

The preprocessing algorithm outlined in Section 4.3 can be executed in $\Theta(n^3)$ time. Note that this algorithm allows us to compute $\mathsf{d}_1(T, G)$ in $O(n)$ time for *every* $T \in SPR_S$. Since $|SPR_S| = O(n^2)$, we are now enabled to solve problem 4 for the Manhattan norm in $\boldsymbol{\Theta(n^3 \cdot |\mathcal{P}|)}$ time.

5 Integer Linear Programming Solution

In this section we present an integer linear programming (ILP) formulation for the Manhattan median tree problem, which allows us to solve small instances of the problem *exactly*.

5.1 ILP formulation

In order to encode a space of phylogenetic trees we utilize the structure presented in [23]. A phylogenetic tree is encoded as a *binary hierarchy* – a binary matrix of size $n \times n-2$, where rows represent taxa (one row for each taxon), and columns represent non-trivial clusters of the tree. A binary variable $M_{i,c} = 1$ if and only if a taxon i is contained in the cluster c.

To ensure that such a matrix contains only non-trivial clusters, we add the following constraint:

$$2 \leq \sum_{i=1}^{n} M_{i,c} \leq n-2, \quad \forall 1 \leq c \leq n-2.$$

Compatibility. Further, the *three-gamete* conditions [30] are used in order to make sure that a matrix M represents a valid phylogenetic tree, i.e., clusters are compatible. The $(0,1)$, $(1,0)$ and $(1,1)$ *gametes* are inferred using the following variables for all $1 \leq i \leq n$, $1 \leq c_1, c_2 \leq n-2$ (see [23] for more details):

$$\begin{aligned} C_{01}(c_1,c_2) &\geq -M_{i,c_1} + M_{i,c_2} \\ C_{10}(c_1,c_2) &\geq M_{i,c_1} - M_{i,c_2} \\ C_{11}(c_1,c_2) &\geq M_{i,c_1} + M_{i,c_2} - 1 \end{aligned}$$

Now, to enforce compatibility for any two clusters c_1 and c_2 we add the constraint $C_{01}(c_1,c_2) + C_{10}(c_1,c_2) + C_{11}(c_1,c_2) = 2$.

Uniqueness. If we rearrange the columns of a matrix M, the corresponding tree would not change. Therefore, to enforce unique matrix representation of a tree we supply a linear order on the columns (here we treat columns as binary integers), and thereby prohibiting column rearrangements:

$$\sum_{i=1}^{n} 2^{i-1} M_{i,c} \geq \sum_{i=1}^{n} 2^{i-1} M_{i,c+1}, \quad \forall 1 \leq c \leq n-3$$

Path-Difference variables. Note that all the variables that we are going to introduce are integers. Let $PL(i,j)$ be a variable that equals to a path-length between two taxa i and j in a tree S represented by the matrix M. We define it as follows:

$$PL(i,j) = D(i) + D(j) - 2D(i,j), \quad \forall 1 \leq i,j \leq n$$

Here $D(i)$ is a depth of a taxon i in S (length of the path between $\mathsf{Rt}(S)$ and i). While $D(i,j)$ is a depth of the least common ancestor of i and j, $\mathsf{LCA}(i,j)$, in S. $D(i)$ could be calculated by counting the number of clusters, where i appears, i.e.

$$D(i) = \sum_{c=1}^{n-2} M_{i,c} + 1, \quad \forall 1 \leq i \leq n.$$

In order to calculate $D(i,j)$ we need to introduce a few more variables:

$$\begin{aligned} In(c,i,j) &\geq M_{i,c} + M_{j,c} - 1, && \forall 1 \leq i,j \leq n,\ 1 \leq c \leq n-2 \\ In(c,i,j) &\leq M_{i,c}, && \forall 1 \leq i,j \leq n,\ 1 \leq c \leq n-2 \\ In(c,i,j) &\leq M_{j,c}, && \forall 1 \leq i,j \leq n,\ 1 \leq c \leq n-2 \end{aligned}$$

$In(c,i,j)$ is a binary variable, which is 1 if and only if a cluster c contains both taxa i and j. Note that the first inequality assures that $In(c,i,j)$ equals one, when c contains i and j, while the second and third constraints assure that $In(c,i,j)$ is 0 otherwise. Now, we can calculate $D(i,j)$ as follows:

$$D(i,j) = \sum_{c=1}^{n-2} In(c,i,j), \quad \forall 1 \leq i,j \leq n$$

Note that if all the trees in the input to our median tree problem were over the same taxa set, the ILP formulation would be almost complete. However, when it is not the case, we need to accommodate for the minus method used for calculating path-difference distance. Therefore, we introduce the following binary variables:

$$\begin{aligned} &\mathit{Above}(k,i) \geq D(k,i) - D(k) + 1 && \forall 1 \leq i,k \leq n \\ &\overline{\mathit{Above}}(k,i) \geq \frac{(D(k) - D(k,i))}{n-2} && \forall 1 \leq i,k \leq n \\ &\mathit{Above}(k,i) + \overline{\mathit{Above}}(k,i) = 1 && \forall 1 \leq i,k \leq n \end{aligned}$$

Here $\mathit{Above}(k,i)$ and $\overline{\mathit{Above}}(k,i)$ are two complementary variables. The constraints assert that $\mathit{Above}(k,i) = 1$ if and only if the parent of a node with a taxon k lies on the path from $\mathsf{Rt}(S)$ to i.

Let now k,i,j be three taxa in the tree S represented by a matrix M. Then a binary variable $Affects(k,i,j)$ equals 1 if and only if $\mathsf{Pa}(k)$ is on the path from i to j in S. The following constraints enforce this relationship:

$$\begin{array}{lcl} Max\_above(k,i,j) & \geq & \mathit{Above}(k,i) \\ Max\_above(k,i,j) & \geq & \mathit{Above}(k,j) \\ Min\_above(k,i,j) & \leq & \mathit{Above}(k,i) \\ Min\_above(k,i,j) & \leq & \mathit{Above}(k,j) \\ Max\_above(k,i,j) + Min\_above(k,i,j) & = & \mathit{Above}(k,i) \\ & & +\mathit{Above}(k,j) \\ Max\_above(k,i,j), Min\_above(k,i,j) & \in & \{0,1\} \end{array}$$

$$Affects(k,i,j) = Max\_above(k,i,j) - Min\_above(k,i,j)$$

Now we are ready to introduce path-difference variables that would be used in the objective function. Let G be some tree from the input $\mathcal{P}$ and let M_G be a set of "missing" taxa in G. That is, $M_G = \mathsf{L}(\mathcal{P}) - \mathsf{L}(G)$. We introduce a variable $PD(G,i,j)$ for two taxa i and j which equals $d_{i,j}(G,S)$ if $i,j \in \mathsf{L}(G)$ and 0 otherwise. That is

$$PD(G,i,j) = \begin{cases} \mathsf{d}_{ij}(G) - PL(i,j) & \\ \quad + \sum_{k \in M_G} Affects(k,i,j), & i,j \in \mathsf{L}(G) \\ 0, & \text{otherwise.} \end{cases}$$

This leads us to the following objective function:

$$\min \sum_{G \in \mathcal{P}} \sum_{i<j} |PD(G,i,j)|.$$

Finally, we need to eliminate absolute values from the objective function. In order to do it we introduce a set of auxiliary variables:

$$\begin{aligned} APD(G,i,j) &\geq PD(G,i,j) \\ APD(G,i,j) &\geq -PD(G,i,j). \end{aligned}$$

It is easy to verify that the following objective function is equivalent to the original one:

$$\min \sum_{G\in\mathcal{P}} \sum_{i<j} APD(G,i,j).$$

5.2 Complexity analysis

The $Affects$ and PD, APD variable groups contribute the largest number of variables as well as constraints. There are $O(n^3)$ $Affects$ variables and $O(n^3)$ constraints are needed to guarantee its correctness. Further, there are $O(kn^2)$ of PD and APD variables and the same order of constraints involving them. In total, we have $O(n^3+kn^2)$ variables and constraints in our ILP formulation.

6 Experimental Evaluation

In this section we evaluate the efficiency and effectiveness of local search heuristics implemented on the base of the novel algorithm introduced in Section 4. We conducted four studies designed to demonstrate the applicability of the Manhattan median tree (MMT) heuristics. In the first study we present a scalability analysis for the simplest *random-restart* heuristic type on artificially generated tree datasets. Next, we present an extensive applicability study of the MMT framework by comparing two powerful heuristic paradigms using different parameters. This study is designed to help practitioners with a choice of the heuristic that provides the most credible results. In our third study we validate our method against other popular supertree software on empirical phylogenetic datasets, and compare resulting trees under multiple relevant objectives. At last, we demonstrate the performance of our ILP formulation and compare the heuristic results with the corresponding exact ones on simulated data.

6.1 Scalability analysis

Here we demonstrate the difference in runtime between the heuristic that uses the naïve local search algorithm and the heuristic that uses our improved algorithm that employs the dynamic programming approach. The runtime curve of the improved heuristic depicts how an expected local search convergence (to the local minimum) time grows with the increase of the total number of taxa.

When we presented the local search heuristic in Section 4 we did not describe how to determine the starting tree that is used to initiate searching the tree-space. A simple approach is to select this tree at random, and the corresponding local search heuristic is referred to as *random restart heuristic*. This heuristic is most appropriate for our scalability analysis, since other more involved methods, which will be discussed in the following study, utilize time-expensive tree building heuristics to determine the starting tree, and thus introduce bias to the pure local-search runtime.

6.1.1 Experimental setting

The two heuristics were implemented in Java 1.8 and tested on a standard workstation with an Intel Core i7 2.5 GHz CPU under the Windows 7 operating system. For the runtime evaluation of the methods we generated multiple datasets with random input trees. All datasets consist of 10 trees where the number of taxa is varying from 10 to 120 with a step of 10. Both the naïve and the improved heuristics were executed 5 times on each dataset.

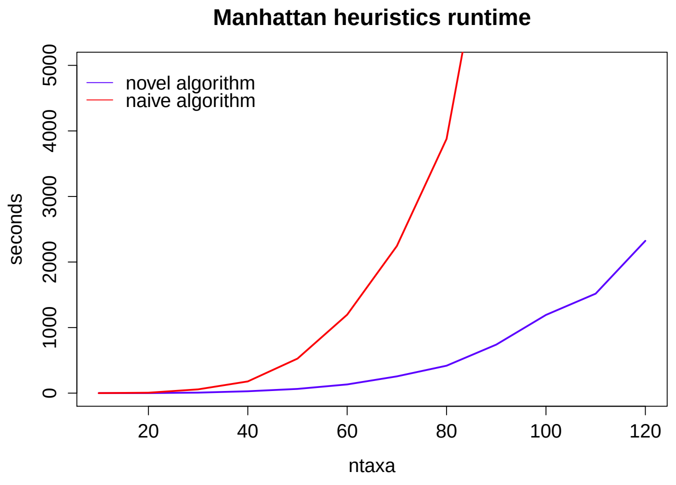


Fig. 4: The graph shows the growth of Manhattan median tree heuristics' runtime with the increase of the number of taxa

6.1.2 Results

Figure 4 depicts the average run-times over the 5 trials. We observe that the naïve heuristic's runtime grows much faster than our improved heuristic's runtime. Note that the naïve heuristic's curve "cuts out" at around 80 taxa, while the other curve carries on much further. The figure exhibits the significance of the here introduced fast algorithm, since it allows us to apply the MMT heuristics to datasets of much larger scale than it was possible previously.

6.2 Heuristic performance in practice

While above we discuss the theoretical algorithm for speeding up the local search approach, in practice it is very important to use the right local search paradigm in order to obtain the most credible median tree estimates in reasonable time. In this section we describe two state-of-the-art heuristic approaches that were efficiently implemented in Java, and evaluate them under different parameter settings.

When applying the local search idea, it is very important to carefully select a starting tree (also referred to as a *seed*) for the final local search phase. That is, instead of taking a random tree as a seed, as we did in the previous section, we would like to *construct* a starting tree that by itself is already a good estimate for a median tree. This approach is expected to result in much better estimates of the median trees as opposed to the random restart method. Therefore, traditionally, computation of supertrees using a local search approach (in a standard tree-search space) is divided into two phases. In Phase I the seed is built incrementally by greedily adding taxa one-by-one minimizing the objective distance-function. In Phase II the local search heuristic is launched with the seed tree as a starting tree. We refer to this approach as a *two-phase heuristic*.

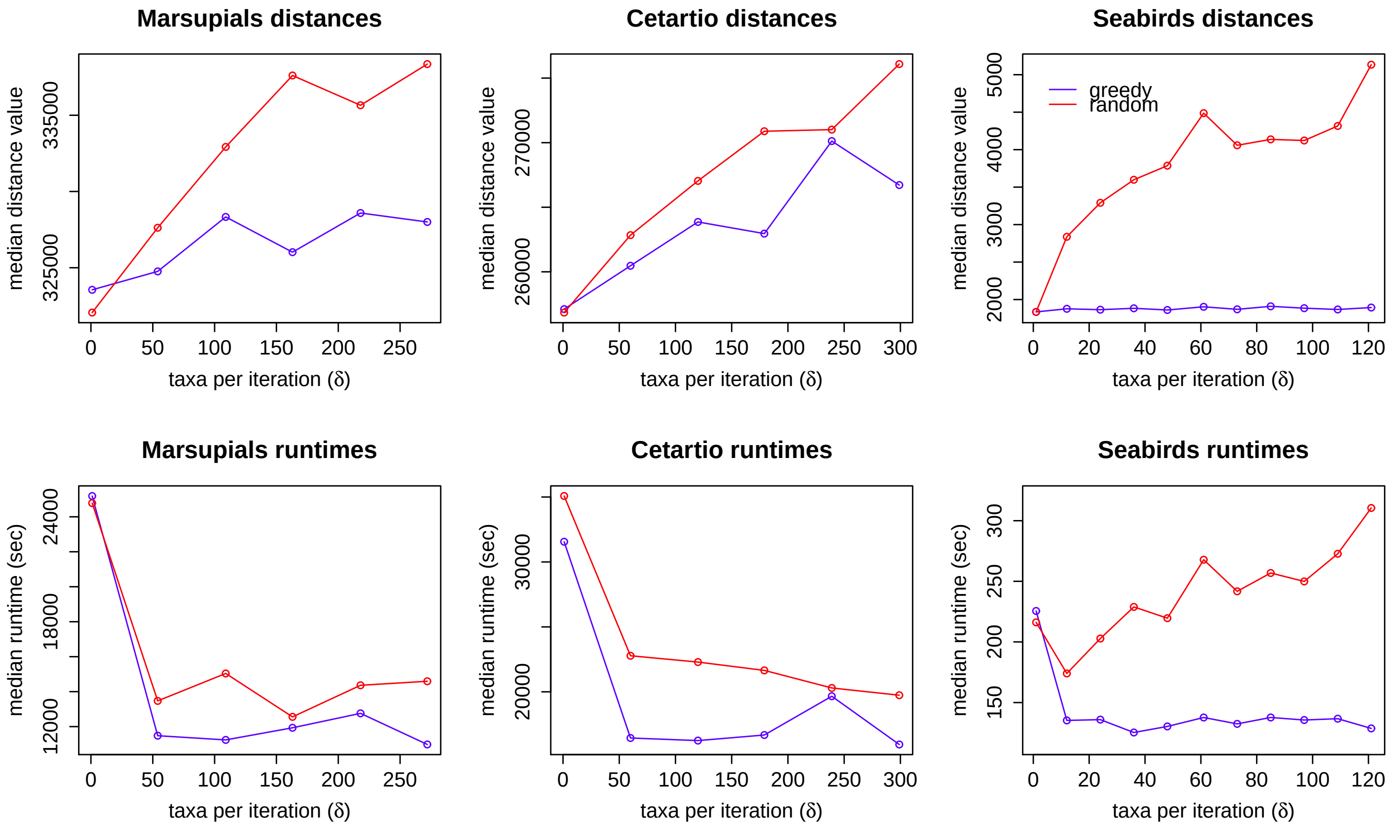


Fig. 3: Comparison of the random and greedy hybrid heuristics for MMT under different parameters on three baseline phylogenetic datasets.

An extension of this idea, called a *hybrid heuristic*, was first introduced in [10]. The hybrid heuristic, in a sense, intervenes the two phases of the traditional approach by making use of the local search method while constructing the seed. In this scenario local search is launched multiple times and not only on the final phase. The hybrid approach requires a user-specified parameter, $1 \leq \delta \leq |\mathsf{L}(\mathcal{P})|$ (lies between 1 and the number of taxa in the input), that determines how often the local search is attempted. Algorithmically this idea can be formalized as the following function:

```
1: function HybridHeuristic(Input tree collection $\mathcal{P}$, parameter $\delta$)
2:     Initialize an empty tree $S$
3:     while $|\mathsf{L}(S)| < |\mathsf{L}(\mathcal{P})|$ do
4:         Add $\delta$ new randomly chosen taxa from $\mathsf{L}(\mathcal{P})$ to the tree $S$;
5:         Run local search starting with the augmented $S$;
6:         Set $S$ to be the tree on the last iteration of the local search run;
7:     end while
8:     return $S$
9: end function
```

We deliberately do not formalize the step 4 in the above algorithm outline, the addition of δ new taxa to a partially constructed supertree S, because there are multiple ways to achieve that. Ideally, we would want to solve the following computational problem:

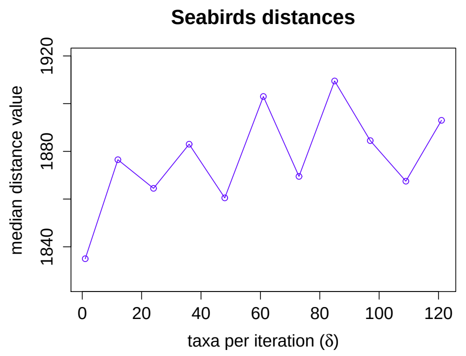


Fig. 5: The Manhattan distance statistics for the greedy hybrid heuristic on the Seabirds dataset. The graph shows only the greedy hybrid approach to exhibit the points clearer then it is done in Figure 3

Instance:	A set of input trees $\mathcal{P}$, partially constructed supertree S, and a set of taxa X, such that $X \cap \mathsf{L}(S) = \emptyset$ and $|X| = \delta$;
Goal:	Find a tree S', such that $\mathsf{L}(S') = \mathsf{L}(S) \cup X$, S is compatible with S' (i.e., S' contains S), and $\mathsf{d}_1(S', \mathcal{P})$ is minimized.

However, this problem is computationally hard, as follows from the NP-hardness of the Manhattan median tree problem. Instead, to make the idea applicable in pactice, we use one of the strategies explained below to add δ new taxa, and then run local search to approximate the desired tree S'. There are two computationally feasible strategies for adding new taxa

1) **Greedy hybrid heuristic**. We choose a random ordering of the δ new taxa, and add them greedily one-by-one on the *best* positions to the tree; that is, we consider all positions for a new taxon in the tree and choose the one that minimizes the distance to the input trees.
2) **Random hybrid heuristic**. We again choose a random ordering of the δ new taxa and add them on-by-one, but at the *random* positions in the tree.

Note that if we set $\delta = |\mathsf{L}(\mathcal{P})|$, then the random hybrid heuristic simply becomes the random-restart method, while the greedy hybrid heuristic turns into the traditional two-phase heuristic. Therefore, the presented hybrid framework is a generalization of the two. In this section we compare the performance of these hybrid approaches given different parameters δ. This comparison will give practitioners the insight on how to choose the most suitable parameter δ.

6.2.1 Data Sets

Following the pioneer work on the path-difference median trees [10] we evaluate the heuristics on the following baseline phylogenetic datasets, the Marsupials dataset [31] (contains 272 taxa and 158 trees) and the Cetartiodactyla dataset [32] (contains 299 taxa and 201 trees). These datasets are considerably large in size and appear frequently in phylogenetic studies (see, for example, [28], [33], [34], [35]). Additionally, we analyze the Seabirds dataset [36] (contains 121 taxa and 7 trees), which is significantly smaller in size than the other two, and, therefore, allows for a more extensive investigation statistically.

6.2.2 Experimental setting

We designed a study to obtain multiple median tree estimates using different values for the parameter δ. For the larger datasets, Marupials and Cetartiodactyla, we estimated median trees under the values of $\delta = 1$ and $\delta = 20\%, 40\%, ..., 100\%$ of the respective dataset's taxa number. For each of the listed values of δ, the heuristics (random and greedy hybrid heuristics) were set to run for 24 hours (under each setting). During this period of time both heuristics were able to compute from 5 to 10 different trees for each listed value of δ.

As for the Seabirds dataset, due to its relatively small size, we were able to analyze a larger number of different values for the parameter δ by computing median trees for each of them. That is, we considered a range of $\delta = 1$ and $\delta = 10\%, 20\%, 30\%, ..., 100\%$. For each of those δ settings we estimated 50 median trees using each of the two heuristics. For example, we constructed 50 trees using the random hybrid heuristic with the setting $\delta = 1$, and another 50 trees were constructed using the greedy hybrid heuristic with the same setting.

From every run we gathered statistics about the runtime of computing a single median tree estimate and the resulting Manhattan distance of the estimated tree to the corresponding input trees.

6.2.3 Results

For each dataset and each parameter δ used, we report the *median runtime* among all runs performed by each heuristic and the *median distance* values (note that we use the median, since we want the statistic to be less affected by the possible outliers). Figures 3 and 5 depict the obtained results. First of all, we observe that for $\delta > 1$ the greedy heuristic outperforms the random heuristic on average both in terms of the runtime and distance. As for $\delta = 1$ we see that the results do not differ very much; this can be explained by the fact that the strategy for adding a single taxon does not affect the result too much. Note that wherever we position the new taxon, the SPR neighborhood will contain all other possibilities.

Focusing on the **greedy hybrid heuristic**, we observe that for $\delta = 1$ the mean runtime is significantly worse than the mean runtime for other applied $\delta > 1$. We observe that for all three datasets – for all of them the $\delta = 1$ runtime is approximately two times the observed runtimes for other values of δ. On the other hand, it appears that the mean runtime for the values of $\delta \geq 10\%$ is rather stable, although fluctuates noticeably. Note that the runtimes, for instance on the settings $\delta = 20\%$ and $\delta = 60\%$ are very close for all three datasets, despite the fact that for $\delta = 20\%$ we launch the local search heuristic much more frequently. This suggests that when we run the local search more frequently, we can expect that for every run it takes fewer iterations to converge to a local minimum, which results in a compensation in the runtime. Next, we observe that the median Manhattan distance *tends to grow* with the increase of δ – this, we again observe for all three datasets. While we present the median values, we exclude the analysis of the sample variances of the runtimes and distances from this study. In fact, we were not able to observe any significant patterns in the variances, which might be explained by the insufficient number of samples for variance estimation (maximum 10 runs due to the vast complexity of the study on large datasets).

To sum up, the performed analysis suggests that setting smaller values for δ (e.g., $10\% - 20\%$ of the overall taxa number) is beneficial in terms of the expected distance (i.e., this would result in better median tree estimates), and does not give any observable difference in runtime as compared to the traditional two-phase method (i.e., $\delta = 100\%$). For smaller datasets it appears more reasonable to set $\delta = 1$, since the runtime is not a critical component in this case.

6.3 Evaluation against other supertree methods

The MMT heuristic comparison presented in the previous study does not provide sufficient insight into how well those heuristics achieve their objective. Given the scale of the data sets used previously, it is not possible to compute the Manhattan median tree exactly. Therefore, we deducted a comparative study evaluating how heuristics with other objectives perform under the L_1 PD distance. Indeed, we expect that our Manhattan median tree heuristics perform best in this study. However, a negative outcome would question the ability of our method.

6.3.1 Experimental setting

In order to evaluate the performance of the MMT heuristic we compare it against other phylogenetic methods includ-

TABLE 2: Empricial evaluation of supertree methods over two published phylogenetic datasets. The best scores under each objective function are shown in bold.

Data set	Method	L_1 PD score	L_2 PD score	Triplet-sim	MAST-sim	Pars. score
Marsuplial 158 input trees 272 taxa	MMC	1,681,015	16,670.45	51.73 %	53.4 %	3901
	MRP	515,257	5,694.59	98.29 %	**71.6** %	**2274**
	TH(SPR)	515,906	5,866.27	**98.99** %	70.3 %	2312
	TH(TBR)	517,274	5,888.22	**98.99** %	70.4 %	2317
	EMT	327,379	**4,380.77**	85.24 %	67.0 %	2869
	MMT	**323,909**	5,063.34	54.68 %	57.6 %	3817
Cetartiodactyla 201 input trees 299 taxa	MMC	918,639	16,206.17	70.03 %	51.5 %	4929
	MRP	365,870	6,991.36	96.49 %	**65.2** %	**2603**
	TH(SPR)	403,233	7,630.03	**97.28** %	63.1 %	2754
	TH(TBR)	401,327	7,591.13	**97.28** %	63.0 %	2754
	EMT	258,836	**5,639.24**	85.98 %	61.0 %	3394
	MMT	**258,424**	6,142.98	66.28 %	54.2 %	4218

ing the Euclidean (L_2 norm) median tree (EMT) heuristic [10]. In addition, we included the traditional maximum representation with parsimony (MRP) supertree heuristic [29], the exact modified min-cut (MMC) algorithm [37], and the triplet heuristic (TH) for the triplet median tree problem [33].

While the MRP supertree problem is NP-hard [38], existing MRP heuristics are among the most applied supertree methods in evolutionary biology [5]. In this study we use supertrees obtained by the MRP local search algorithm implemented in PAUP* [29] under the tree bisection and reconnection (TBR) edit operation [33]. Note that the *TBR edit operation* is an extension of the SPR operation, where the pruned subtree is allowed to be re-rooted before regrafting it. The MMC algorithm is the only polynomial-time supertree algorithm in this study. In addition, this algorithm was shown to satisfy certain “desirable” properties formulated by Steel et. al. [39], and therefore was suggested for its use on large-scale phylogenetic inference problems [37]. Finally, the triplet heuristic is a local search algorithm that addresses the well-studied NP-hard triplet median tree problem [33]. We use supertrees constructed using the triplet heuristic based on both SPR and TBR local searches, abbreviated by TH(SPR) and TH(TBR) respectively.

For estimating Manhattan median trees we use the greedy hybrid heuristic approach with the parameter setting $\delta = 30$. Note that $\delta = 30$ is slightly higher than 10% of the overall taxa for both datasets, following the recommendation of the previous study. That way we present a “fair” comparison, while we demonstrated that we obtain more credible median tree estimates using the setting $\delta = 1$.

6.3.2 Results

The evaluation results are summarized in Table 2. The phylogenetic supertrees obtained by different methods were assessed using relevant objectives: Manhattan (L_1) and Euclidean (L_2) PD distances, triplet similarity (which is used as a maximization criterion in the triplet heuristic), parsimony score (minimization criterion for the MRP heuristic). In addition, we present the maximum agreement subtree (MAST) scores (i.e., the average percentage over the largest agreement subtrees between a supertree and each input tree in proportion to the total tree size) for the computed supertrees. As expected, we observe that our MMT algorithm produced the best supertrees in regards to the Manhattan PD distance. We also see that EMT heuristic produced trees that perform comparably well in terms of Manhattan distance to the trees produced by MMT. Further, we observe a similar relationship under the Euclidean distance. This suggests that Manhattan and Euclidean distance search spaces for local search are highly correlated.

On the other hand, the table reveals that the MMT median trees perform worse than traditional MRP and TH supertrees on such objectives as triplet similarity, MAST similarity, and parsimony score. It is worth noting that all three objective functions depend heavily on the rootings of the trees under comparison, while this is not quite the case with the path-difference distances. That is, the root affects the path-difference score, but not as significantly as with other objectives.

Distributions. In general, having exact distance distributions over supertree spaces is highly beneficial while evaluating experimental results. However, currently there are no known efficient algorithms that would allow us to obtain PDD distributions even for a single input tree under any vector-norm [14]. Therefore, to further analyze the results we estimated the distribution of Manhattan distances on random sample data for both datasets as it was performed previously for the Euclidean distance [10]. That is, we generated two supertree collections with 5000 trees for each phylogenetic dataset discussed above. One collection was generated under the uniform binary tree distribution, and another collection using the Markovian branching process [40]. Next, we processed each supertree collection and obtained corresponding sample datasets with raw Manhattan distances. These samples allowed us to estimate the distance distributions and map the results from Table 2 on them (see Figure 6).

Once again, we observe how close the results for Manhattan and Euclidean heuristics are. In addition, both triplet heuristic and MRP produced trees that are better in distance terms than any of the random trees drawn from the uniform distribution. On the other hand, Markovian trees show a quite significant bias in regards to our distance measure (i.e., the sample mean value is almost twice as small as the

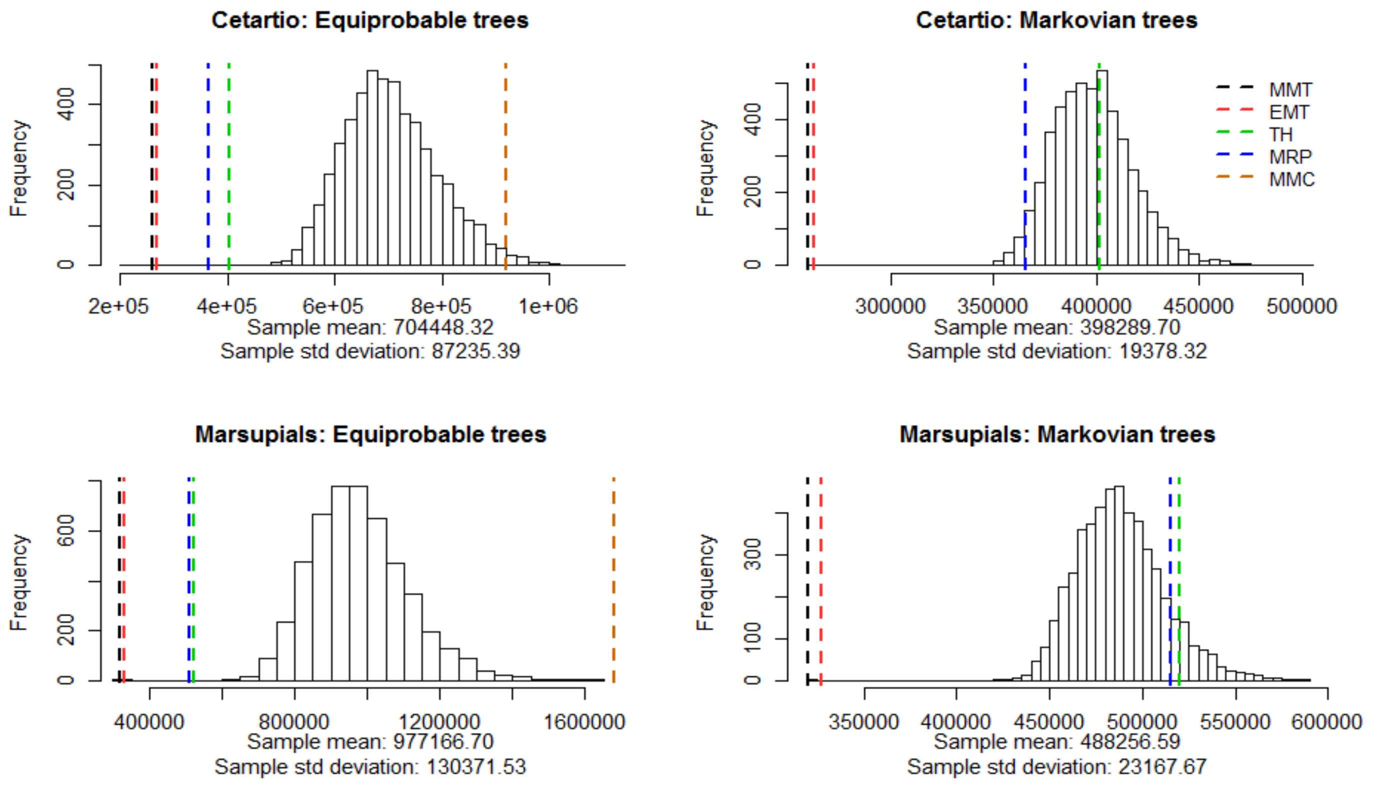


Fig. 6: Histograms of the Manhattan PD distances based on the generated tree samples. The dotted lines mark the distances for the supertrees assembled by different methods.

TABLE 3: MAST similarity scores between supertree estimates constructed via different phylogenetic methods.

		MMT	EMT	MRP	TH(SPR)	TH(TBR)
Marsupials	EMT	0.2390	-	-	-	-
	MRP	0.2059	0.4154	-	-	-
	TH(SPR)	0.2169	0.4081	0.6029	-	-
	TH(TBR)	0.2169	0.4044	0.6103	0.8272	-
	MMC	0.1434	0.1949	0.2243	0.2279	0.2279
Cetartiodactyla	EMT	0.2742	-	-	-	-
	MRP	0.2742	0.4181	-	-	-
	TH(SPR)	0.2542	0.3679	0.5385	-	-
	TH(TBR)	0.2575	0.3712	0.5518	0.9097	-
	MMC	0.1472	0.1605	0.2843	0.2408	0.2375

sample mean value for the equiprobable trees). Surprisingly, as a result, many Markovian random trees are better as Manhattan median trees than MRP and TH supertrees (this is observed for both empirical datasets). Note that this was not the case for the estimated distributions of Euclidean distances [10], where MRP and TH trees were better than any of the generated Markovian trees. On a separate note, MMC supertrees show a negative bias in regards to the Manhattan distance, which is also true for the Euclidean distance.

Comparing supertrees. Table 2 exhibits how well supertrees generated by different methods fit the corresponding input trees under multiple objectives. While it is difficult to argue, which objective is the most accurate and important one (in terms of evolutionary history), we can analyze how the computed supetrees compare among themselves.

To do that we choose a single best-fitting tree for each supertree method. For example, among 20 trees computed by the MRP heuristic, we choose the one with the smallest parsimony score over the input dataset. Note that we do so for both Marsupial and Cetartiodactyla datasets. Next, we compare those best-fitting trees using the maximum agreement subtree similarity function. Note that we choose MAST for comparison, since it is an “independent” similarity-function not used as an objective by any of the supertree methods under consideration. Table 3 depicts the resulting similarity matrix.

From this matrix we observe that MMT supertrees are most closely related to EMT supetrees. However, this similarity is still very subtle when compared to the similarities

TABLE 4: Evaluation of ILP performance against the local search heuristic on small instances of the median tree problem. "LS runs" row shows how many local search runs it took to find an optimal solution by the heuristic, and "LS runtime" records a cumulative runtime of the first LS runs.

ntaxa	n=4				n=6				n=8				n=10
ntrees	k=5	k=10	k=20	k=100	k=5	k=10	k=20	k=100	k=5	k=10	k=20	k=100	k=5
Optimal MPDD	17	34	87	431	66	173	322	1686	181	375	799	4154	350
ILP runtime (s)	0.07	0.16	0.20	0.53	1.21	3.03	9.47	7.10	704	481	402	653	10531
LS runs	1	1	1	2	1	5	1	3	1	2	10	3	8
LS runtime (s)	1.75	1.06	1.13	3.99	2.32	9.25	3.11	17.39	4.67	8.38	65.46	39.23	51.39

between other supertrees (e.g., EMT and MRP, or MRP and TH). It suggests that although the Euclidean and Manhattan median tree objectives are correlated, the corresponding heuristics produce structurally different trees.

6.4 ILP evaluation

The ILP approach enables us to compute Manhattan median tree exactly for small instances of phylogenetic datasets (to a scale that is prohibitive for exact brute-force approaches). By comparing the local search algorithm results to the exact solutions, we estimate the effectiveness of the SPR-driven heuristic.

6.4.1 Experimental setting

For evaluation of our ILP formulation we artificially generated several instances of the Manhattan median tree problem with random input trees. The size of the generated datasets varied both in terms of the number of taxa and trees. To carry out our experiments we used a Gurobi MILP optimizer [41]. The largest instance of the Manhattan median tree problem that we obtained exact results for consisted of 5 phylogenetic trees over 10 taxa. In addition, we ran the presented local search heuristic on the generated problem instances (20 times each).

6.4.2 Results

Table 4 depcits the results of the performed experiments. It can be seen that the runtime of the ILP solver grows very fast with increase of the number of taxa. We also observe that the local search algorithm was able to reach a global minimum within the first ten runs on each instance, while being significantly faster than the ILP solver on larger inputs (i.e., $n = 8$ and 10).

7 Discussion and conclusion

Large-scale phylogenetic inference problems are among the central computational problems in evolutionary biology. Computationally efficient median tree and supertree approaches bring the scale of phylogenetic analysis to unprecedented levels.

While MRP is one of the predominant supertree method in this area [5], different tree assembly methods could be applied to provide alternative viewpoints on evolutionary events. As the presented experimental studies suggest, phylogenetic trees constructed using two different path-difference distance heuristics (for Manhattan and Euclidean median tree problems) do not strongly correlate with the supertrees constructed using other popular methods. Thus, we can pose these methods as a valuable alternative to MRP. At the same time, Table 3 showed that Manhattan and Euclidean median trees do not bear much similarity according to the maximum agreement subtree criterion. Therefore, the two cases are worth being examined independently. Further, the distribution analysis performed for Manhattan and Euclidean distances suggests the distribution space of the two metrics is rather different.

In this work we generalized the path-difference median tree problem for any L_p vector norm and showed that the corresponding problems are NP-hard. While there already exists an efficient local search heuristic for the Euclidean median tree problem [10], here we introduce a related algorithm for the Manhattan median tree problem with a more involved preprocessing step that accommodates the complexity of dealing with absolute values. Interestingly, the described preprocessing schema can be directly applied to the Euclidean median tree algorithm, but not vice versa. The developed algorithm solves an instance of the local neighborhood search problem in $O(kn^3)$. The experimental results suggest that our heuristic can handle input datasets with several hundreds of taxa in reasonable time and is scalable in regards to the number of trees in the input (as k is a linear factor in the runtime).

While an efficient algorithm enables us to estimate large-scale Manhattan median trees, in our hybrid heuristic study we show that the local search approach used makes an enormous difference in the quality of estimates. The greedy hybrid approach turned out to be able to provide a significant boost in accuracy of the heuristic without negatively affecting the speed on average.

Further, we developed an ILP formulation that can be applied to find exact solutions for considerably small Manhattan median tree problem instances (up to 10 taxa as experiments show). The comparison of the ILP performance to the local search performance on the simulated data enabled us to additionally justify the merit of the developed heuristic, since it was able to find *global* minima quite fast.

Encouraged by the promising results of our method, future research will investigate further into the ability and applicability of our method in practice in large-scale comparative studies, as well as studying its theoretical properties.

While there is a great interest in weighted phylogenetic trees (see TimeTree database [42] for example), none of the widely applied supertree methods is capable of constructing credible weighted trees. The path-difference median tree approache, however, has a clear potential of changing that and boosting the development of large time-annotated trees of life.

Acknowledgments

The authors would like to thank the three anonymous reviewers for their constructive and valuable comments that helped to improve the quality of this work. This material is based upon work supported by the National Science Foundation under Grant No. 1617626.

References

[1] S. Nik-Zainal and et al., "The life history of 21 breast cancers," *Cell*, vol. 149, no. 5, pp. 994–1007, 2012.

[2] R. A. Hufbauer, R. A. Marrs, A. K. Jackson, R. Sforza, H. P. Bais, J. M. Vivanco, and S. E. Carney, "Population structure, ploidy levels and allelopathy of Centaurea maculosa (spotted knapweed) and C. diffusa (diffuse knapweed) in North America and Eurasia," in *Proceedings of the XI International Symposium on Biological Control of Weeds, Canberra Australia*. Morgantown, WV.: USDA Forest Service. Forest Health Technology Enterprise Team, 2003, pp. 121–126.

[3] J. J. L. Roux, A. M. Wieczorek, M. M. Ramadan, and C. T. Tran, "Resolving the native provenance of invasive fireweed (Senecio madagascariensis Poir.) in the Hawaiian Islands as inferred Poir.) in the Hawaiian Islands as inferred from phylogenetic analysis," *Diversity and Distributions*, vol. 12, pp. 694–702, 2006.

[4] S. R. Harris, E. J. Cartwright, M. E. Török, M. T. Holden, N. M. Brown, A. L. Ogilvy-Stuart, M. J. Ellington, M. A. Quail, S. D. Bentley, J. Parkhill, and S. J. Peacock, "Whole-genome sequencing for analysis of an outbreak of meticillin-resistant staphylococcus aureus: a descriptive study," *Lancet Infect Dis*, vol. 13, no. 2, pp. 130–6, 2013.

[5] O. R. Bininda-Emonds, Ed., *Phylogenetic Supertrees: Combining Information to Reveal the Tree of Life*, ser. Computational Biology. Springer Verlag, 2004, vol. 4.

[6] H. Philipe and M. J. Telford, "Large-scale sequencing and the new animal phylogeny," *TRENDS in Ecology and Evolution*, vol. 21, no. 11, pp. 614–620, 2006.

[7] H. Gee, "Evolution: ending incongruence," *Nature*, vol. 425, no. 6960, p. 782, Oct 2003.

[8] J. O. McInerney, J. A. Cotton, and D. Pisani, "The prokaryotic tree of life: past, present... and future?" *Trends Ecol Evol*, vol. 23, no. 5, pp. 276–81, May 2008.

[9] M. A. Steel and D. Penny, "Distributions of tree comparison metrics - some new results," *Systematic Biology*, vol. 42, no. 2, pp. 126–141, 1993.

[10] A. Markin and O. Eulenstein, "Path-difference median trees," in *Bioinformatics Research and Applications: 12th International Symposium, ISBRA 2016, Minsk, Belarus, June 5-8, 2016, Proceedings*, A. Bourgeois, P. Skums, X. Wan, and A. Zelikovsky, Eds. Cham: Springer International Publishing, 2016, pp. 211–223.

[11] J. Farris, "A successive approximations approach to character weighting." *Systematic Zoology*, vol. 18, pp. 374–385, 1969.

[12] J. Bluis and D. Shin, "Nodal distance algorithm: Calculating a phylogenetic tree comparison metric," in *3rd IEEE International Symposium on BioInformatics and BioEngineering (BIBE 2003), 10-12 March 2003, Bethesda, MD, USA*. IEEE Computer Society, 2003, pp. 87–94.

[13] P. Puigbò, S. Garcia-Vallvé, and J. O. McInerney, "TOPD/FMTS: a new software to compare phylogenetic trees," *Bioinformatics*, vol. 23, no. 12, pp. 1556–1558, 2007.

[14] A. Mir and F. Rosselló, "The mean value of the squared path-difference distance for rooted phylogenetic trees," *CoRR*, vol. abs/0906.2470, 2009.

[15] W. Williams and H. Clifford, "On the Comparison of Two Classifications of the Same Set of Elements," *Taxon*, vol. 20, no. 4, pp. 519–522, 1971.

[16] J. B. Phipps, "Dendogram topology," *Systematic Zoology*, vol. 20, pp. 306–308, 1971.

[17] W. P. Maddison and L. L. Knowles, "Inferring phylogeny despite incomplete lineage sorting," *Syst Biol*, vol. 55, no. 1, pp. 21–30, 2006.

[18] C. Than and L. Nakhleh, "Species tree inference by minimizing deep coalescences," *PLoS Comput Biol*, vol. 5, no. 9, p. e1000501, 2009.

[19] M. S. Bansal, J. G. Burleigh, and O. Eulenstein, "Efficient genome-scale phylogenetic analysis under the duplication-loss and deep coalescence cost models," *BMC Bioinformatics*, vol. 11 Suppl 1, p. S42, 2010.

[20] R. Chaudhary, M. S. Bansal, A. Wehe, D. Fernández-Baca, and O. Eulenstein, "iGTP: a software package for large-scale gene tree parsimony analysis," *BMC Bioinformatics*, vol. 11, p. 574, 2010.

[21] H. T. Lin, J. G. Burleigh, and O. Eulenstein, "Consensus properties for the deep coalescence problem and their application for scalable tree search," *BMC Bioinformatics*, vol. 13 Suppl 10, p. S12, 2012.

[22] C. Semple and M. A. Steel, *Phylogenetics*. Oxford: University Press, 2003.

[23] W.-C. Chang, J. G. Burleigh, D. F. Fernández-Baca, and O. Eulenstein, "An ILP solution for the gene duplication problem," *BMC Bioinformatics*, vol. 12 Suppl 1, p. S14, 2011.

[24] J. A. Cotton and M. Wilkinson, "Majority-rule supertrees," *Syst Biol*, vol. 56, no. 3, pp. 445–452, 2007.

[25] D. Bryant, "Hunting for trees in binary character sets: efficient algorithms for extraction, enumeration, and optimization." *J Comput Biol*, vol. 3, no. 2, pp. 275–288, 1996.

[26] M. A. Steel, "The complexity of reconstructing trees from qualitative characters and subtrees," *Journal of Classification*, vol. 9, pp. 91–116, 1992.

[27] K. Takahashi and M. Nei, "Efficiencies of fast algorithms of phylogenetic inference under the criteria of maximum parsimony, minimum evolution, and maximum likelihood when a large number of sequences are used," *Molecular Biology and Evolution*, vol. 17, no. 8, pp. 1251–1258, 2000.

[28] M. S. Bansal, J. G. Burleigh, O. Eulenstein, and D. Fernández-Baca, "Robinson-foulds supertrees," *Algorithms for Molecular Biology*, vol. 5, no. 1, pp. 1–12, 2010.

[29] D. L. Swofford, "PAUP*. Phylogenetic analysis using parsimony (*and other methods). Version 4. Sinauer Associates, Sunderland, Massachusetts." 2002.

[30] D. Gusfield, *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. New York, NY, USA: Cambridge University Press, 1997.

[31] M. Cardillo, O. R. P. Bininda-Emonds, E. Boakes, and A. Purvis, "A species-level phylogenetic supertree of marsupials," *Journal of Zoology*, vol. 264, pp. 11–31, 2004.

[32] S. A. Price, O. R. P. Bininda-Emonds, and J. L. Gittleman, "A complete phylogeny of the whales, dolphins and even-toed hoofed mammals (cetartiodactyla)," *Biological Reviews*, vol. 80, no. 3, pp. 445–473, 2005.

[33] H. T. Lin, J. G. Burleigh, and O. Eulenstein, "Triplet supertree heuristics for the tree of life," *BMC Bioinformatics*, vol. 10, no. Suppl 1, 2009.

[34] S. Snir and S. Rao, "Quartets maxcut: A divide and conquer quartets algorithm," *IEEE/ACM TCBB*, vol. 7, no. 4, pp. 704–718, 2010.

[35] D. Chen, O. Eulenstein, D. Fernández-Baca, and J. Burleigh, "Improved heuristics for minimum-flip supertree construction," *Evolutionary Bioinformatics*, vol. 2, 2006.

[36] M. Kennedy and R. D. M. Page, "Seabird supertrees: Combining partial estimates of procellariiform phylogeny." *The Auk*, vol. 119, no. 1, pp. 88–108, 2002.

[37] R. D. M. Page, "Modified mincut supertrees," in *Proceedings of the Second International Workshop on Algorithms in Bioinformatics*, ser. WABI '02. London, UK: Springer-Verlag, 2002, pp. 537–552.

[38] S. Moran, S. Rao, and S. Snir, "Using semi-definite programming to enhance supertree resolvability," in *Proceedings of the 5th International Conference on Algorithms in Bioinformatics*, ser. WABI'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 89–103.

[39] M. Steel, A. W. M. Dress, and S. Bocker, "Simple but fundamental limitations on supertree and consensus tree methods," *Systematic Biology*, vol. 49, no. 2, pp. 363–368, 2000.

[40] N. G. Bean, N. Kontoleon, and P. G. Taylor, "Markovian trees: properties and algorithms," *Annals of Operations Research*, vol. 160, no. 1, pp. 31–50, 2007.

[41] Gurobi Optimization, Inc., "Gurobi optimizer reference manual," 2015.

[42] A. D. Leaché, "The timetree of life. S. Blair Hedges and Sudhir Kumar, editors." *Integrative and Comparative Biology*, vol. 50, no. 1, pp. 141–142, 2010.

Alexey Markin received a B.S. degree in Computer Science from Higher School of Economics (Russia) in 2015. Since then he is a Ph.D. student of Computer Science at Iowa State University. His research interests include graph theory and computational biology.

Oliver Eulenstein is a professor of computer science at Iowa State University. He earned his doctoral degree at the University of Bonn (Germany) in 1998, and held a postdoctoral position at the University of California Davis before joining the department of Computer Science at Iowa State University in 2000. His research interest is in Combinatorial Optimization, with special emphasis on Computational Biology and Bioinformatics.