Abstract—Genomic rearrangement operations can be very useful in inferring the phylogenetic relationship of gene orders representing species. We study the problem of finding potential ancestral gene orders for the gene orders of given taxa, such that the corresponding rearrangement scenario has a minimal number of reversals, and where each of the reversals has to preserve the common intervals of the given input gene orders. Common intervals identify sets of genes that occur consecutively in all input gene orders. The problem of finding such an ancestral gene order is called the preserving reversal median problem (pRMP). A tree-based data structure for the representation of the common intervals of all input gene orders is used in our exact algorithm TCIP for solving the pRMP. It is known that the minimum number of reversals to transform one gene order into another can be computed in polynomial time, whereas the corresponding problem with the restriction that common intervals should not be destroyed is already NP-hard. It is shown theoretically that TCIP can solve a large class of pRMP instances in polynomial time. Empirically we show the good performance of TCIP on biological and artificial data.

Index Terms—gene orders, reversals, common intervals, reversal median problem, strong interval trees

I. INTRODUCTION

The phylogenetic relationship between species is often analyzed by means of rearrangement scenarios for the gene orders from the species. A rearrangement scenario between two gene orders describes how one of the gene orders can be transformed into the other by a set of rearrangement operations. A very commonly used rearrangement operation is the reversal of a part of the genome. A median for a set of given gene orders is a gene order such that the sum of the reversal distances — i.e., the minimal number of reversals needed to transform one gene order into another — to the input gene orders is minimal. For inferring phylogenetic trees based on gene orders, solving this reversal median problem (RMP) is a basic operation used in several algorithms [1], [2], [3], [4].

It is known that certain gene groups are preserved during evolution. Since it is difficult to determine functionally what a gene group is, it has been proposed to consider common combinatorial structures between gene orders as gene groups. Conserved intervals [5], or — as used in this paper — common intervals [6], [7], [8], [9] are considered in evolutionary scenarios [10], [11], [12], [13], [14]. Unfortunately, even computing the preserving reversal distance, i.e., the minimum number of reversals that preserve all common intervals between two given gene orders, is an NP-complete problem [15]. Without the restriction that common intervals should be preserved, the reversal distance problem, as well as computing a corresponding minimum length sequence of reversals, is polynomial time solvable [16], [17], [18]. But finding a median is NP-hard already for three given gene orders [19]. Exact [19], [20] and heuristic [1], [4], [21] algorithms have been proposed to compute a median.

In the inspiring paper of Bérard et al. [22], it was shown that parsimonious preserving scenarios, that transform one given gene order into another given gene order, can be computed in polynomial time for certain interesting classes of problem instances. These results are based on the strong interval tree, which is a tree data structure for representing the set of common intervals of a set of gene orders. The strong interval tree can be computed in linear time for a constant number of input gene orders.

In this paper we use strong interval trees for the computation of preserving reversal median scenarios, i.e., for solving the preserving reversal median problem (pRMP). We develop an exact algorithm called TCIP (Tree Common Interval Preserving) for solving the pRMP for three given gene orders. It is shown experimentally on biological and artificial test data that TCIP is faster than the fastest existing exact algorithm for the reversal median problem (RMP), i.e., the corresponding problem in which common intervals are not considered.

Basic definitions are given in Section II. How the pRMP is solved with algorithm TCIP is described in Section III and Section IV. Empirical results on biological mtDNA data and on random data are presented in Section V. Conclusions are given in Section VI.

II. BASIC DEFINITIONS

A permutation \( \pi = (\pi(1) \pi(2) \ldots \pi(n)) \) of size \( n \) is a permutation of the elements in \([1,2,\ldots,n]\). A signed permutation of size \( n \) is a permutation of size \( n \) where every element has an additional “+” or “−” sign that defines its orientation. Sign “+” is usually omitted. In the following we call a signed permutation just permutation. A reversal \( \rho(i,j) \), \( 1 \leq i \leq j \leq n \), applied to a signed permutation \( \pi \) of size \( n \), transforms it into \( \pi \circ \rho(i,j) = (\pi(1) \ldots \pi(i-1)-\pi(j) \ldots -\pi(i) \pi(j+1) \ldots \pi(n)) \). When the context is clear, we identify a reversal with the set of elements that it reverses. A sorting scenario for two signed permutations \( \pi \) and \( \sigma \) is a sequence of reversals \( \rho_1,\ldots,\rho_d \) that transforms \( \pi \) into \( \sigma \). A shortest such sequence is called parsimonious and its length is the reversal distance \( d(\pi,\sigma) \) between \( \pi \) and \( \sigma \).

An interval of a permutation is a set of consecutive elements of this permutation. Let \( \Pi \) be a set of signed permutations of size \( n \). A common interval [6], [7] of \( \Pi \) is a subset of \([1,2,\ldots,n]\) that is an interval in each \( \pi \in \Pi \). The singletons \([i]\), \( i \in [1,2,\ldots,n] \), and the set \([1,2,\ldots,n]\) of all elements are called trivial common intervals. Observe that the signs of the elements are ignored in the definition of common intervals. Let

Manuscript received ; revised
Matthias Bernt and Martin Middendorf are with the Parallel Computing and Complex Systems Group, Department of Computer Science, University of Leipzig, Germany. E-mail: (bernt, middendorf)@informatik.uni-leipzig.de.
Daniel Merkle is with the Department of Mathematics and Computer Science, University of Southern Denmark, Denmark. E-mail: daniel@imada.sdu.dk.
C(Π) be the set of all common intervals of Π. Two intervals c and c' overlap if c ∩ c' ≠ ∅, c ≠ c', and c' ≠ c. If two intervals do not overlap they are said to commute. A common interval is called a strong common interval, if it does not overlap with any other common interval. The strong common interval tree [23], [22] T(Π) of Π is a tree where the nodes are exactly the strong common intervals of Π, such that the root node is the interval containing all elements, the leaves are the singletons, and the edges are defined by the minimal inclusion relation of the intervals, i.e., there is an edge between node c and c' iff c' ⊂ c and there is no node c'' with c' < c'' < c. As the common intervals of a set of permutations Π are independent of the signs of the elements, and therefore also the strong common intervals, the topology of T(Π) does not depend on the signs of the elements as well. The strong interval tree can be computed in time O(kn) for k signed permutations of size n [24], [25].

A reversal ρ applied to one of the permutations π ∈ Π preserves the common intervals of Π if it does not destroy any common interval c ∈ C(Π), i.e., C(Π) = C(Π ∪ (π ◦ ρ)). If there exists a common interval c ∈ C(Π), that does not exist after applying the reversal, i.e., c ∉ C(Π ∪ (π ◦ ρ)), then the reversal does not preserve C(Π). A reversal which preserves the common intervals of Π when applied to π ∈ Π is said to be preserving with respect to Π and π. The reversal is simply called preserving with respect to Π if π is obvious and preserving if Π and π are obvious. The preserving reversal distance dρ(π, σ) between two signed permutations π and σ is the minimum number of preserving with respect to (π, σ)) reversals necessary to transform π into σ (in [22] this distance was denoted as perfect reversal distance).

The reversal median problem (RMP) is to find for a set of signed permutations Π = {π1, π2, ..., πk}, a signed permutation μ which has a minimal total reversal distance to the permutations in Π, i.e., the so-called score ∑k i=1 d(πi, μ) has to be minimal. In this case permutation μ is called median of Π. For a signed permutation σ a scenario for Π to σ defines for each permutation in Π a sequence of reversals to σ. The sequence of permutations that is traversed from permutation πi, i ∈ {1, ..., k}, towards σ is called a trace of the scenario, or more specific the πi-trace of the scenario. If σ is a median, i.e., the total number of reversals in the scenario is minimal, the scenario is called median scenario. The preserving reversal median problem (pRMP) is as defined as the RMP with the additional restriction that only reversals which are preserving with respect to the k given permutations are allowed. The corresponding permutation μ is called preserving median or median of the pRMP. Observe that C(Π) = C(Π ∪ μ) holds for the median μ of the pRMP. Note that a slightly different definition of the pRMP was used in [13], where for the computation of the preserving reversal distance only pairwise common intervals of an input permutation and the median were taken into account. A scenario for a set of permutations Π to a permutation σ is preserving if each of its reversals is preserving with respect to Π. If σ is a preserving median this scenario is also called preserving median scenario.

In the following a slightly modified version of the RMP is introduced for which it is shown in this paper that it occurs as a subproblem when solving the pRMP. Solving this version of the RMP is a key element in our algorithm TCIP. It is shown that the number and size of the instances of this RMP variant which occur, have a large influence on hardness of the corresponding pRMP. The oriented reversal median problem (oRMP) is to find a median, such that for a given tuple of signs (s1, s2, ..., sk) the score ∑k i=1 d(πi, σ ◦ ρ) is minimal, where si ◦ ρ := σ if si = + and si ◦ ρ := −σ if si = − (i.e., the whole signed permutation is reversed). The oRMP is NP-hard, as it contains the RMP for minimizing ∑k i=1 d(πi, μ) as a special case when the tuple of signs is (+, +, +). Note that the oRMP for a given sign vector (s1, ..., sk) and signed permutations π1, ..., πk, i.e., minimizing ∑k i=1 d(πi, σ ◦ ρ), can be solved as a standard median problem where ∑k i=1 d(σ ◦ ρ, πi) is minimized. Hence, an oRMP instance can be solved with a standard RMP solver, e.g., Caprara’s exact median solver [19].

III. Solving the Preserving Reversal Median Problem

In [22] strong interval trees for pairs of permutations and algorithms computing the preserving reversal distance were introduced. In this Section we present a generalization of strong interval trees for more than two permutations. We present methods for solving the pRMP based on properties of this tree. The methods are used in the proposed algorithm (TCIP) for the pRMP.

A. k-signed Strong Interval Trees

Let π be a permutation and S = (I1, ..., Im) be an (ordered) partition of the elements of π into intervals. Similarly as in [22] we define the (unsigned) quotient permutation of a signed permutation π associated with S — denoted PπS — as an (unsigned) permutation of (1, 2, ..., m), such that x precedes y in PπS iff the elements of Ix precede the elements of Iy in π. The latter precedence relation is well defined in this definition since each set in S is an interval of π. Note that the quotient permutation depends on the order of the sets in S but not on the signs of the elements of π. A quotient permutation is called increasing linear if PπS is the identity permutation (1, ..., m), decreasing linear if PπS is the inverse of the identity permutation (m, ..., 1), and prime otherwise. A quotient permutation is called linear if it is increasing linear or decreasing linear.

Example 1: Let π = (3 7 1 4 5 6 2) be a permutation, and let S = (I1, I2, I3) with I1 = (6, 2), I2 = (3, 7, 1), and I3 = (4, 5) be a partition of π. The quotient permutation PπS of π is (2 3 1). Since it is not linear, it is prime.

From the definition of a strong interval tree T(Π) it follows easily that the children of an inner node define a partition of this node into strong common intervals. Using the definition of quotient permutations, the property linear or prime is assigned to the inner nodes of a strong interval tree as follows. Consider an inner node I of the strong interval tree. Node I corresponds to an interval in each permutation πi ∈ Π and the children of node I define a partition of this interval into intervals. Without loss of generality we assume that the sets in this partition (I1, ..., Im) are ordered, such that Ix precedes Iy in the first permutation πi for x < y. Then, consider the corresponding k quotient permutations that are induced for interval I in the permutations π1, ..., πk. Each of these quotient permutations is either prime or linear. Observe, that the quotient permutation for the interval I in π1 is increasing linear. If all k quotient permutations are linear, the inner node I is called linear. Otherwise, it is called prime.
Example 2: Let $\pi_1 = (1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \ 10)$, $\pi_2 = (1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \ 10)$, and $\pi_3 = (1 \ 2 \ 3 \ 5 \ 7 \ 5 \ 4 \ 3 \ 10)$ be the three input permutations (\(\tilde{I}\) is an abbreviation for \(-i\)). Their non-trivial strong common intervals are \((3, 5, 9)\) and \((5, 9, 10)\). Consider the node \(I\) in the strong interval tree (see Figure 1(b)), which corresponds to the strong common interval \((3, 5, 9)\). The children of \(I\) are \(I_1 = (3), I_2 = (4),\) and \(I_3 = (5, 9, 10)\). They are ordered relative to \(\pi_1\). Set \(I = (I_1, I_2, I_3)\) is a partition of \(I\). The quotient permutations that are induced for \(I\) are $P_{\tilde{I}} = (1 \ 2 \ 3)$, $P_{\tilde{I}} = (1 \ 2 \ 3)$, and $P_{\tilde{I}} = (3 \ 2 \ 1)$. All three quotient permutations are linear, hence the node \(I\) is linear.

In the following we define a variation of the strong interval tree which requires an order on a given set of permutations. Therefore, we assume that a sequence $Π = (π_1, π_2, ..., π_k)$ of permutation is given. To make our notation not unnecessarily complicated we use $Π$ to denote the sequence of permutations and also the corresponding set of permutations. It will be clear from the context whether $Π$ is a sequence or a set, if this is not explicitly stated. Some definitions that we have given for a set of permutations will also be used for a sequence of permutations when the meaning is clear.

A tuple $s = (s_1, ..., s_k)$, where $s_j \in \{+, -, \}$, is called a $s$-sign. The inverted tuple of $s$, i.e., the tuple for which each sign is inverted, is denoted as $\tilde{s}$. The $k$-signed strong interval tree $T^k(Π)$ is the strong interval tree of the sequence $Π = (π_1, π_2, ..., π_k)$ of $k$ permutations that has assigned a $s$-sign to some of its nodes, such that

i. each leaf has a $s$-sign $(s_1, ..., s_k)$ where $s_j$ is the sign of the corresponding element of the input permutation $π_j$,
ii. the $h$-th entry in the $s$-sign of a node $n$ is in the $s$-sign of the parent $p$, if and only if the parent node is linear.

Note that no $s$-sign is assigned to prime nodes with prime node parents and no $s$-sign is assigned to prime root nodes. For a node $I$ its $s$-sign is denoted $s(I)$. A k-signed strong interval tree is called unambiguous, if every prime node has a linear parent and ambiguous otherwise. Examples of 3-signed strong interval trees are given in the following sections.

Let a strong interval tree $T(Π)$ be given. A set of permutations $Π'$ is called consistent with $T(Π)$ if each node in $T(Π)$ is also a common interval of $Π'$. A set of permutations $Π'$ with $C(Π') = C(Π)$ is consistent. If a sequence of permutations $Π'$ is consistent with $T(Π)$ and $|Π'| = k$ then the quotient permutations and the $s$-signs for $Π'$ can also be computed with respect to the permutations in $Π'$. (Clearly, the quotient permutations and the $s$-signs can be different depending on whether they are computed with respect to $Π$ or $Π'$). In this case we need to order the sets in the partition $\{I_1, ..., I_m\}$, such that $I_s$ precedes $I_p$ in permutation $π_s$ if and only if $s(I_s) < s(I_p)$.

Let $Π = (π_1, ..., π_k)$ be a sequence of permutations and $I$ a node in $T(Π)$ with child nodes $I_1, ..., I_m$. Then $(I_1, ..., I_m)$ is a partition $\mathcal{F}$ of $I$ into intervals. Let $s_1, ..., s_m$ be the $s$-signs of the child nodes $(I_1, ..., I_m)$. We define the signed quotient permutation of the $i$-th permutation $π_i$ associated with $\mathcal{F}, s_1, ..., s_m$ — denoted $P_{\mathcal{F}, s_1, ..., s_m}(i)$ as a signed permutation of \((1, 2, ..., m)\), such that $x$ precedes $y$ in $P_{\mathcal{F}, s_1, ..., s_m}(i)$ iff the elements of $I_x$ precede the elements of $I_y$ in $π_i$ and the sign of an element $e$ in the quotient permutation equals the $i$-th sign of $s_e$.

Example 3: Consider the same input permutations and quotient permutations as in Example 2. Let furthermore be given 3-signs $s_1 = (+, +, -)$, $s_2 = (-, +, +)$, and $s_3 = (−, −, −)$. The signed quotient permutations are $P_{\mathcal{F}, s_1, s_2, s_3}(1) = (1 \ 2 \ 3)$ and $P_{\mathcal{F}, s_1, s_2, s_3}(3) = (3 \ 2 \ 1)$.

The $k$-signed strong interval tree data structure can be built in $O(nk)$ using the algorithms presented in [24], [25]. Tree $T^k(Π)$ has $O(nk)$ nodes [22] which makes it a nice data structure for designing efficient algorithms.

B. Some Theoretical Results

In this section we show some theoretical results that are the basis for the design of our algorithm TCIP. Let $Π = (π_1, ..., π_k)$ be a sequence of permutations. Let $σ$ be a preserving median of $Π$. Let $Π' = (π'_1, ..., π'_k)$ be a sequence of permutations consistent with $Π$ where $π'_i$ is on the $i$-th trace of the median scenario of $Π$ to $σ$ where the $i$-th trace is defined to be the $π'_i$-trace. Let $Π'' = (π''_1, ..., π''_t)$ be the sequence of permutations after a reversal $ρ$ of the preserving scenario has been applied to $π'_i$. Due to the application of $ρ$, the $k$-signed strong interval tree is changed from $T(Π, Π')$ and $T(Π, Π'')$. By definition of strong interval trees the topology of $T(Π, Π')$ and $T(Π, Π'')$ is the same, only the quotient permutations and $s$-signs of nodes may be different. The following proposition shows how preserving reversals can be identified in the strong interval tree. It is a modified version of a proposition of [22].

The proof of Proposition 1 is given in the Appendix.

Proposition 1: For a set $Π$ of permutations and a permutation $σ$ a scenario for $Π$ to $σ$ is preserving with respect to $Π$ if and only if each of the reversals in the scenario is either a node of the strong interval tree $T(Π)$ or the union of children of a prime node of $T(Π)$.

The next proposition describes the impact of a preserving reversal on the $s$-signs of the nodes of a $k$-signed strong interval tree.

Proposition 2: Let $Π, Π' = (π'_1, ..., π'_k), Π'' = (π''_1, ..., π''_k)$ be a sequence of permutations such that $Π'$ and $Π''$ are consistent with $T(Π)$. Let $σ$ be a preserving reversal for $Π$ with $π'_j σ π''_j$ and $π'_j σ π'_j$ for all $j \neq i$. Let $I$ be a node in the strong interval tree $T^k(Π, Π')$ with a $s$-sign. If $ρ \cap I = \emptyset$ or $ρ \cap I \neq \emptyset$ holds, then $I$ has the same $s$-sign in $T^k(Π, Π')$ and $T^k(Π, Π'')$. If $I \subset ρ$, then the $i$-th sign of $s(I)$ in $T^k(Π, Π')$ is inverse to the $i$-th sign of $s(I)$ in $T^k(Π, Π'')$ and the other signs of $s(I)$ are the same in both trees.

Proof: Let $Π' = (π'_1, ..., π'_k)$ be a sequence of permutations consistent with $Π$. Let $Π'' = (π''_1, ..., π''_t)$ be the sequence of permutations after a preserving reversal $ρ$ has been applied to the $i$-th permutation. Note that if a reversal is applied to the $i$-th permutation, only the $i$-th element of the $k$-sign of a node can be changed. The comparison of $k$-signs in a node $I$ of $T^k(Π, Π')$ and $T^k(Π, Π'')$ is as follows.

1) Let $ρ \cap I = \emptyset$. In the case that $I$ is a leaf node, the corresponding node is not influenced and the $k$-sign is not changed. In the case that $I$ is not a leaf node, no
which leads to either

\[ \Pi \]

or

\[ \Pi' \]

together with the trace \((\pi, \pi', \pi_2, \pi_3)\). If \(\Pi\) is a linear node, then the quotient permutations of \(\Pi\) are not changed as \(\rho\) does not change the precedence relation of the children of \(\Pi\). If \(\Pi\) is a prime node, then the \(k\)-sign of \(\Pi\) is not influenced by the change of the quotient permutation.

3. Let \(I \subseteq \rho\). If \(I\) is a leaf node, then the \(i\)-th entry of the \(k\)-sign is inverted, as in \(\pi_i'\) and \(\pi_i''\) the corresponding element is inverted. If \(I\) is not a leaf node, then the corresponding quotient permutation of node \(I\) is inverted and therefore the \(i\)-th entry of the \(k\)-sign is inverted.

The following theorem specifies which preserving reversals have to be applied in any preserving median scenario for \(\Pi\).

**Theorem 1:** (The Median Parity Theorem) Let \(I\) be a node of the strong interval tree \(T^3(\Pi, \Pi)\) of signed permutations \(\Pi = (\pi_1, \pi_2, \pi_3)\). If \(I\) has a linear parent \(J\), with \(s(I) \neq s(J)\) and \(s(I) \neq \overline{s}(J)\), then there exists a reversal of \(I\) in one of the traces, which leads to either \(s(I) = s(J)\) or \(s(I) = \overline{s}(J)\). This reversal belongs to any preserving median scenario for \(\Pi\).

**Proof:** Let \(S\) be a preserving median scenario for \(\Pi\) and let \(I\) be a node in the strong interval tree \(T^3(\Pi)\) for which \(s(I) \neq s(J)\) and \(s(I) \neq \overline{s}(J)\) holds. Assume that no reversal of \(I\) occurs in \(S\). By Proposition 1 and 2, \(s(I) \neq s(J)\) and \(s(I) \neq \overline{s}(J)\) still holds after applying all reversals from \(S\), as a reversal that inverts a sign in \(s(J)\) also inverts the corresponding sign in \(s(I)\). Therefore, \(s(I) = s(J)\) or \(s(I) = \overline{s}(J)\) is not achievable by \(S\). But then \(S\) cannot be a preserving median scenario because it is clear that after applying all reversals of a preserving median scenario, the \(3\)-sign of each node must be equal to \((+, +, +)\) or equal to \((-,-,-)\). From this contradiction follows immediately that a reversal of \(I\) must occur in any preserving median scenario. Because \(s(I) = s(J)\) or \(s(I) = \overline{s}(J)\) can always be achieved by applying the reversal of \(I\) on exactly one of the three traces to the median, this reversal has to occur in any preserving median scenario. Note that the possibility of applying the reversal of \(I\) on two traces would not lead to a median scenario.

A generalization of the Median Parity Theorem and its proof are given in the Appendix.

The following two propositions deal with the score differences between instances of median problems that differ only by signs of elements in the permutations or by reversals of whole permutations.

**Proposition 3:** Let \(\Pi\) be a set of three signed permutations with median \(\mu\) that has score \(s\). For \(\pi \in \Pi\) let \(\beta\) be a reversal of one element in \(\pi\) or a reversal of all elements in \(\pi\). Let \(\Pi'\) be the set of permutations that is obtained from \(\Pi\) by exchanging \(\pi\) by \(\pi \circ \beta\). For the score \(s'\) of the median problem defined by \(\Pi'\) holds \(|s - s'| \leq 1\).

**Proof:** Assume \(s - s' > 1\). Let \(\mu'\) be a median of \(\Pi'\) and \(\pi' = \pi \circ \beta\). The score of \(\Pi'\) to \(\mu'\) includes the trace \((\pi', ..., \mu')\). The two traces from the other two permutations in \(\Pi\) to \(\mu'\) together with the trace \((\pi, \pi' = \pi \circ \beta, ..., \mu')\) is a scenario for \(\Pi\) to \(\mu'\) that has score \(s' + 1\). As \(s' + 1 < s\) holds, \(\mu\) was not a median for \(\Pi\), which is a contradiction. The case \(s' - s > 1\) can be shown analogously.

Note that Proposition 3 does not imply that for a median \(\mu\) of \(\Pi\) and a median \(\mu'\) of \(\Pi'\) the reversal distance between \(\mu\) and \(\mu'\) is at most one. The next proposition is a generalization of Proposition 3.

**Proposition 4:** Let \(\Pi\) be a set of three signed permutations of \([1, ..., n]\) and \(\Pi'\) be the set of permutations that is obtained from \(\Pi\) by changing the sign of at most one element in each permutation or by inverting some of the permutations. Then for the score \(s\) of a median scenario for \(\Pi\) and the score \(s'\) for a median scenario of \(\Pi'\) holds \(|s - s'| \leq 1\). If \(\Pi'\) was obtained by changing a sign of an element in all three permutations of \(\Pi\) or by inverting all permutations in \(\Pi\), then \(s = s'\) holds.

**Proof:** First the inversion of the signs of single elements is considered. There are the following three cases. i) If only one input permutation has been changed, we can directly apply Proposition 3. ii) If the sign of an element in all three input permutations is changed, then any median \(\mu\) for \(\Pi\) can be converted to a median \(\mu'\) for \(\Pi'\) by inverting the sign of the corresponding element in \(\mu\). Obviously, then \(s = s'\) holds. iii) For the last case, assume that the sign of an element is changed in two permutations. Let \(\Pi''\) be the set of permutations that is obtained by changing the sign of the corresponding element in all input permutations of \(\Pi'\). Let \(\mu''\) be a median for \(\Pi''\) with score \(s''\). Obviously, \(s'' = s''\) holds. As in \(\Pi''\) the sign of only one element is changed relatively to \(\Pi\), we can apply Proposition 3. Therefore, it holds \(|s - s'| \leq 1\) which directly implies \(|s - s'| \leq 1\).

The case of inverting one, two, or three of the input permutations completely, can be shown analogously.

IV. Algorithm TCIP

Based on the theoretical results from the last section our algorithm TCIP (Tree Common Interval Preserving) is designed as described in the following. Algorithm TCIP is completely written in C++ and is freely available from the authors.

In the first step the 3-signed strong interval tree is computed. In the second step all reversals are applied on the traces starting from the three input permutations to change all 3-signs of linear nodes with linear parents according to Theorem 1. The third step in algorithm TCIP is — from a runtime point of view — the most complex one. After identifying all prime node components (i.e., all connected subtrees of prime nodes) in the strong interval tree, the quotient permutations of the prime nodes define oRMPs that have to be solved. For prime nodes with a linear parent the sign triple for the oRMP is known, but not for prime nodes with prime node parents. Recall, that the sign triple of an oRMP defines the signs of an individual element in the quotient permutations of the parent prime node. The sign triple assignment in the prime nodes that leads to a minimal number of reversals in a prime node component is determined with a dynamic programming approach.

In the following three subsections we explain how the pRMP is solved in the case of input permutations leading to strong interval trees that are A.) unambiguous and have no prime node, B.) unambiguous and have prime nodes, and C.) ambiguous.
A. Unambiguous Trees without Prime Nodes

By definition all initial 3-signs of a 3-signed strong interval tree \( T^3(\Pi, \Pi') \) with \( \Pi = \Pi' \) have a “+” as first entry. Suppose the 3-signs of a node \( i \) and its parent are \( s = (s_1, s_2, s_3) \), respectively \( s' = (s'_1, s'_2, s'_3) \). The tree is traversed in a bottom-up manner and 3-signs and \( s' \) are compared. If \( s \neq s' \) and \( \bar{s} \neq s' \), a trace is extended and the 3-sign of the child node is changed. (As only the 3-sign in the child node is changed and each child node can only be involved in one comparison, we can suppose \( s_1 = + \) and \( s'_1 = + \) for each comparison.) More precisely, as the signs of each node have to become equal, Theorem 1 implies that:

1. if the number of sign differences is 1, i.e., either \( s_2 \neq s'_2 \) or \( s_3 \neq s'_3 \), then the corresponding (second or third) trace is extended by inverting \( i \). This leads to \( s = s' \).
2. if the number of sign differences is 2, i.e., \( s_2 \neq s'_2 \) and \( s_3 \neq s'_3 \), then the first trace is extended by inverting \( i \).

This leads to \( s = \bar{s} \).

Note that three sign differences can not occur as \( s_1 = s'_1 = + \) for all comparisons. We illustrate the case of unambiguous trees without prime nodes with a small example.

Example 4: Let \( \Pi = (\pi_1, \pi_2, \pi_3) \) with \( \pi_1 = (1\ 2\ 3) \), \( \pi_2 = (1\ \bar{2}\ \bar{3}) \), and \( \pi_3 = (3\ 1\ \bar{2}) \) be the three input permutations. The only non-trivial strong common interval for these permutations is \( C = ([1\ 2]) \). The 3-signed strong interval tree \( T^3(\Pi, \Pi) \) is depicted in Figure 1(a). There are two differences between the signs of node \( 2 \) and its parent \( s(2) = (+, -, -) \) and \( s(1\ 2\ 1) = (+, +, +) \). This induces the reversal of \( 2 \) in \( \Pi \). The nodes \( 1 \) and \( 2 \) have a different sign compared to their parent leading to the reversals of \( 1\ 2 \) in \( \pi_3 \) and the reversal of \( 3 \) in \( \pi_2 \). The root node has signs \( (+, +, -) \), so \( \pi_3 \) has to be inverted completely. The traces of the parsimonious median scenario with 4 reversals and the median \( (1\ 2\ 3) \) are:

1. \( \pi_1 = (1\ 2\ 3) \Rightarrow (1\ 2\ 3) = \mu \),
2. \( \pi_2 = (1\ 2\ 3) \Rightarrow (1\ 2\ 3) = \mu \), and
3. \( \pi_3 = (3\ 1\ \bar{2}) \Rightarrow (1\ 2\ 3) \Rightarrow (1\ 2\ 3) = \mu \)

where \( \Rightarrow \) indicates a reversal operation.

If no prime node occurs in the strong interval tree \( T(\Pi) \), a preserving median scenario and the only existing median is directly defined by the tree. The unique set of reversals to be applied on a trace is inferred by 3-sign differences between nodes and their parents. As reversals implied by linear nodes commute by definition all orders of these reversals lead to the same permutation. Therefore, the resulting median is unique.

B. Unambiguous Trees with Prime Nodes

In this subsection we assume that each prime node of the strong interval tree \( T(\Pi) \) has a linear parent. The basic idea is to compute for each prime node a median scenario for the median problem that is induced by the three quotient permutations of the prime node. This can be done by solving an ORM, because no union of the children of prime nodes is a common interval, and therefore any reversal can be applied to the quotient permutations.

In the definitions of \( k \)-signed strong interval trees the root prime node of a prime node component inherits the \( 3 \)-sign of its linear parent. The following proposition shows the relation of the median sets solved for ORM with different 3-signs.

This proposition is necessary to explain why an inherited 3-sign assignment leads to a preserving median scenario in unambiguous trees with prime nodes, which will be shown in Corollary 1.

Proposition 5: Let \( T^3(\Pi) \) be a 3-signed strong interval tree with a prime node \( i \) that has only linear child nodes. Then the sequence of signed quotient permutations \( (\pi_1, \pi_2, \pi_3) \) of \( i \) together with the 3-sign \( s = (s_1, s_2, s_3) \) define an ORMP with score \( \zeta \) and set of medians \( M \), and \( (\pi_1, \pi_2, \pi_3) \) together with 3-sign \( s' = (s'_1, s'_2, s'_3) \neq s \) define another ORMP with score \( \zeta' \) and set of medians \( M' \). If \( \zeta > \zeta' \), then \( M \supset M' \).

Proof: First consider the case that one sign is different in \( s \) and \( s' \). Without loss of generality let \( s_1 \neq s'_1 \). Consider a median \( \mu \in M' \). With \( \sum_{i=1}^{3} d(\pi_i, s_1 \circ \mu) = \sum_{i=1}^{3} d(\pi_i, s'_1 \circ \mu) \) it holds by Proposition 4 that \( \zeta - \zeta' = 1 \). The traces \( (\pi_1, \pi_2, \pi_3) \circ \mu \) give permutation \( s_1 \circ \mu = s'_1 \circ \mu \), it holds \( s_2 = s_2, \) and \( s_3 = s_3 \). Thus \( \mu \in M \) and \( M \subseteq M' \). Now consider the case \( s \neq s' \). Let \( s_1 \circ \mu \in M \) be a median minimizing \( \sum_{i=1}^{3} d(\pi_i, s_1 \circ \mu) \). Then there is a corresponding median \( s'_1 \circ \mu \in M' \). The reverse \( \sum_{i=1}^{3} d(\pi_i, s'_1 \circ \mu) \), i.e., \( M = M' \). Furthermore \( \zeta = \zeta' \) by Proposition 4. The two cases of signs are different in \( s \) and \( s' \) follows immediately because there is only one sign difference between \( s \) and \( s' \).

Corollary 1: Let \( T^3(\Pi) \) be a 3-signed strong interval tree, and let \( i \) be a prime node of \( T^3(\Pi) \) with a linear parent and linear child nodes only. Prime node \( i \) can inherit the \( k \)-sign of its linear parent to find a preserving median scenario for \( T^3(\Pi) \).

Proof: Let \( s \) be the \( 3 \)-sign of the linear parent of prime node \( i \). Assume that \( i \) has a \( 3 \)-sign \( s' \) with \( \zeta < \xi \), where \( \zeta \) and \( \zeta' \) are defined as in Proposition 5. Then, by Proposition 3 \( \zeta = \xi - 1 \). Assigning the \( 3 \)-sign \( s' \) to node \( i \) instead of \( 3 \)-sign \( s \) would induce an additional reversal due to the sign difference of \( i \) and its linear parent, which would equalize this score reduction. Furthermore, due to Proposition 5 no additional median can be found by assigning \( s' \) to \( i \). Therefore a prime node can always inherit the sign of its linear parent, if existent, to find a preserving median scenario.

Example 5: Let \( \pi_1 = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10) \), \( \pi_2 = (1\ 2\ 3\ 4\ 6\ 7\ 9\ 8\ 10) \), and \( \pi_3 = (1\ 2\ 9\ 8\ 7\ 6\ 5\ 4\ 3\ 10) \) be the three input permutations. The non-trivial strong common intervals for \( (\pi_1, \pi_2, \pi_3) \) are \( C = (\{3, 4, 5, 6, 7, 8, 9\}, \{5, 6, 7, 8, 9\}) \). The 3-signed strong interval tree is depicted in Figure 1(b). The only prime node inherits the 3-sign \( (+, +, -) \) from the linear parent, and the three signed quotient permutations are \( \Gamma_1 = (1\ 2\ 3\ 4\ 5) \), \( \Gamma_2 = (2\ 5\ 3\ 1\ 4) \), and \( \Gamma_3 = (5\ 3\ 1\ 2\ 4) \). Solving the ORMP for 3-sign \( (+, +, -) \) — by solving the RMP for \( \Gamma_1, \Gamma_2, \) and \( -\Gamma_3 \) — leads to the only median \( \mu_p = (1\ 2\ 3\ 4\ 5) \) with a median score of 4. Note that in general there may be more than one median, but all of them have the same score. One parsimonious scenario for this ORMP is defined by the following traces:

1. \( \Gamma_1 \sim \mu_p \)
2. \( \Gamma_2 \sim \mu_p \)
3. \( \Gamma_3 \sim \mu_p \)

The reversals in the ORMP scenario correspond to reversals in the RMP scenario. The overall preserving reversal
known and each possible 3-sign, i.e., each 3-sign in \( \{ \pm \} \) for such prime nodes the 3-sign is unknown. In prime nodes. Each of these subtrees can be handled separately. the following we first explain an approach for this problem the size. Section V that the size of the quotient permutations in prime

The preserving median scenario has a score of 6.

Note that solving the pRMP with separate prime nodes can be accomplished by solving for each prime node one oRMP. The size of the quotient permutations in a prime node is always smaller than or equal to the original permutation size. This may lead to a computation time reduction compared to solving the original RMP. Empirically it will be shown in Section V that the size of the quotient permutations in prime nodes is usually much smaller than the original permutation size.

C. Ambiguous Trees

Similar to preserving sorting scenarios (compare [22]), the most difficult case for the pRMP occurs when there exist connected prime nodes in the strong interval tree \( T(\Pi) \). In the following we first explain an approach for this problem and suggest some improvements for this approach afterwards. Ambiguous trees posses subtrees that consist of connected prime nodes. Each of these subtrees can be handled separately. In order to compute a scenario with a minimum score for such a subtree the induced median problems within the subtree are solved bottom up.

Ambiguous trees have prime nodes with prime nodes as parents. For such prime nodes the 3-sign is unknown. In this case the sign vector of the corresponding oRMP is not known and each possible 3-sign, i.e., each 3-sign in \( \{ \pm \} \), is considered. Thus, for a prime node \( I \) that has a prime node parent and that has prime child nodes \( I_1, \ldots, I_p \), all combinations of 3-signs for nodes \( I, I_1, \ldots, I_p \) are considered. Recall, that the 3-sign of a prime node child defines the signs of the corresponding elements of the quotient permutations in the parent node. If prime node \( I \) is the root node, then only one 3-sign has to be considered, as the root may only have the 3-sign \( (+,+,+) \). If prime node \( I \) has a linear parent its 3-sign is known and it is enough to consider all combinations of the 3-signs of the child nodes. In the following we describe our approach more formally.

Let \( I \) be a prime node of the strong interval tree \( T^3(\Pi) \), \( I_1, \ldots, I_p \) be the \( p \) prime child nodes of \( I \), and \( I_{p+1}, \ldots, I_{p+m} \) be the \( m \) linear child nodes. Note that \( p = 0 \) or \( m = 0 \) is possible. Each linear child node \( I_{p+j} \) has a 3-sign \( s(I_{p+j}) \). Let \( \mathcal{F} = (I_1, \ldots, I_{p+m}) \) and let \( s, s_1, \ldots, s_p \) be 3-signs in \( \Sigma := \{ \pm \} \). Let \( \xi(I, s, s_1, \ldots, s_{p+m}) \) be the score of the solution of the oRMP that is defined by the three quotient permutations \( p_i^\mathcal{F}_{s_1, \ldots, s_{p+m}}(\pi_i), i \in \{1,2,3\}, \) that are associated with \( \mathcal{F}, s_1, \ldots, s_{p+m} \) and the input 3-sign is \( s \). Recall, that the elements of the input permutations for the oRMP have signs according to \( s_1, \ldots, s_{p+m} \). Let

\[
\xi_{\min}(I, s) = \min_{(s_1, \ldots, s_p) \in \Sigma^p} \xi(I, s, s_1, \ldots, s_{p+m}) + \sum_{i=1}^p \xi_{\min}(I, s_i)
\]

be the recursive definition of the score of prime node \( I \) with sign \( s \), i.e., the minimal number of preserving reversals needed to solve all oRMPs induced by the prime node component with root node \( I \) having sign \( s \). Note that if a prime node \( I \) has no prime child node, then \( \xi_{\min}(I, s) \) is just the score of the induced oRMP where the signs of all elements in the input permutations are all determined by the linear child nodes.

In order to show that the 3-sign of a linear parent node may be inherited by the root node of a prime node component, a slight modification of Proposition 5 and Corollary 1 is needed for the case of ambiguous strong interval trees.

**Proposition 6:** Let \( T^3(\Pi) \) be a 3-signed strong interval tree and let \( I \) be a root node of a prime node component of \( T^3(\Pi) \). Then the sequence of signed quotient permutations \( (\pi_1, \pi_2, \pi_3) \) of \( I \) together with 3-sign \( s = (s_1, s_2, s_3) \) define an oRMP with score \( \xi_{\min}(I, s) \) and set of medians \( M \) and \( (\pi_1, \pi_2, \pi_3) \) together with 3-sign \( s = (s_1, s_2, s_3) \) define another oRMP with score \( \xi_{\min}(I, s') \) and set of medians \( M' \). If \( \xi_{\min}(I, s) > \xi_{\min}(I, s') \), then \( M \subseteq M' \).

**Proof:** The proof is identical to the proof of Proposition 5 (\( \xi \) has to be replaced by \( \xi_{\min}(I, s) \) and \( \xi' \) has to be replaced by \( \xi_{\min}(I, s') \)).

**Corollary 2:** Let \( T^3(\Pi) \) be a 3-signed strong interval tree, and let \( I \) be a root node of a prime node component of \( T^3(\Pi) \) with a linear parent. Prime node \( I \) can inherit the \( k \)-sign of its linear parent to find a preserving median scenario for \( T^3(\Pi) \).

**Proof:** The proof is identical to the proof of Corollary 1 (\( \xi \) has to be replaced by \( \xi_{\min}(I, s) \) and \( \xi' \) has to be replaced by \( \xi_{\min}(I, s') \)).

In the following we show two propositions that are useful for improving the runtime of the first version of algorithm TCIP.

**Proposition 7:** It is sufficient to use the set \( \Sigma := \{ (+,+,+), (+,-,+), (+,-,-), (+,+,+) \} \) in (1) in order to compute \( \xi_{\min}(I, s) \).

**Proof:** Let \( (s_1, \ldots, s_p) \) be the tuple of signs as given in (1). Let \( s_i \in \{ s_1, \ldots, s_p \} \) be one of these 3-signs and let \( \gamma_i \) be the
inverted 3-sign of \( s_i \). The proof has three parts. In the first part it will be shown that for any node \( I \) of the strong interval tree \( \xi(I, s, s_1, \ldots, s_{p+1}, s_{p+2}) = (I, s, s_1, \ldots, s_{p+1}, s_{p+2}) \) holds. In the second part it will be shown that for any node of the strong interval tree \( \xi_{\min}(I, s) = \xi_{\min}(I, s_3) \) holds. In the third part this will be used to show Proposition 7.

1. The computation of \( \xi(I, s, s_1, \ldots, s_{p+1}, s_{p+2}) \) induces an oRMP to be solved that is defined by the three quotient permutations of \( I \) and the input 3-sign \( s \). The sign of the elements of the three quotient permutations are determined by the 3-signs \( s_i \). Replacing \( s_i \) by \( \bar{s} \) corresponds to inverting the sign of the one element in all three input permutations. With Proposition 4 the score of the median scenario does not change. Therefore, \( \xi(I, s, s_1, \ldots, s_{p+1}, s_{p+2}) = \xi(I, s, s_1, \ldots, s_{p+1}, \bar{s}_{p+2}) \) holds for any node \( I \).

2. Comparing \( \xi_{\min}(I, s) \) and \( \xi_{\min}(I, s_3) \) reduces to computing \( \xi(I, s, s_1, \ldots, s_{p+1}, s_{p+2}) \) and \( \xi(I, \bar{s}, s_1, \ldots, s_{p+1}, s_{p+2}) \). In the oRMP to be solved, replacing \( s \) with \( \bar{s} \) corresponds to inverting all three input permutations. With Proposition 4 the score of the median scenario does not change. Therefore, \( \xi_{\min}(I, s) = \xi_{\min}(I, s_3) \) holds for any node \( I \).

3. Let \( \{s_1, \ldots, s_p\} \) be the tuple of signs in (1) for which \( \xi_{\min}(I, s) \) is minimal. As any sign \( s_i \) can be replaced by sign \( \bar{s} \) in (1) we can replace any 3-sign \( s_i \) that has a “-” as first entry by \( \bar{s} \). Hence, there is a tuple of signs \( \{s'_1, \ldots, s'_p\} \) which also leads to a minimal value for \( \xi_{\min}(I, s) \) and the first entry of every \( s'_i \) is “+”. Therefore, we can restrict the set of 3-signs to be considered to \( \Sigma := \{(+, +, +), (+, +, -), (+, -, +), (+, -, -)\} \).

**Proposition 8:** For determining \( \xi_{\min}(I, s) \) in (1) it is sufficient to consider those tuples \( (s_1, \ldots, s_p) \in \Sigma^p \) with \( s_i \) in \( \{1, \ldots, p\} \) there exists no other 3-sign \( s'_i \) with \( \xi_{\min}(I, s'_i) < \xi_{\min}(I, s_i) \) for which the 3-sign is unknown since its parent is also a prime node. Hence, the sign of corresponding element 2 in the quotient permutation of I. The solutions of the oRMP of node I when element 2 is signed according to the 3-sign \( s_i \) of I are also given in Table I. In this example the 3-signs \((+, +, +)\) and \((+, -, -)\) lead to a minimal overall score \( \xi_{\min}(I, s) = 9 \) for the subtree induced by the prime nodes. The computation of one median by applying the permutations implied by oRMP solutions and reversals resulting from sign differences between linear nodes starting from \( \pi_1 \) is shown in Figure 2(b). Note that the traces connecting the quotient permutations with their medians can be easily computed with known algorithms, e.g. [18].

When the minimal score of a pRMP has to be computed, Proposition 8 can reduce the number of 3-sign vectors to be tested in a prime node significantly, especially if a prime node has many prime child nodes. Nevertheless, if all medians have to be computed, then all 3-sign vectors have to be evaluated in (1), as also non-optimal decisions in the prime child nodes may lead to a preserving median scenario.

The case of solving the pRMP for ambiguous trees is illustrated in the following example:

**Example 6:** Let \( \Pi = (\pi_1, \pi_2, \pi_3) \) be a set of three input permutations with \( \pi_1 = (1 2 3 4 5 6 7 8 9 10) \), \( \pi_2 = (5 4 3 2 1 6 9 10 7 8) \), and \( \pi_3 = (10 9 8 7 4 2 5 3 1 6) \). The non-trivial common intervals for these gene orders are \( C = \{[1, 2, 3, 4, 5, 6], [2, 3, 4, 5, 7, 8, 9, 10], [9, 10]\} \). The 3-signed strong interval tree \( T^3(\Pi, II) \) is depicted in Figure 2(a). There are two prime nodes in this tree, namely \( I_1 = \{2, 3, 4, 5\} \) and its parent node \( I = \{1, 2, 3, 4, 5, 6\} \). One child node of I is the prime node \( I_1 \) for which the 3-sign is unknown since its parent is also a prime node. Hence, the sign of corresponding element 2 in the quotient permutations of I is unknown and is marked by ". The signed quotient permutations (corresponding to \( \pi_1, \pi_2, \) and \( \pi_3 \)) of prime node I are \((1 2 3 4), (\{7\} \{3 \} \{2\}), \) and \((3 1 4 5)\) and for prime node \( I_1 \) they are \((1 \{+2\} 3), \{\pm2\} \{3\} \) and \((\pm2)\) respectively. For the four 3-sign assignments \( s_1 \) for I the oRMP is solved, the score \( \xi_{\min}(I, s_1) \) and the corresponding medians are given in Table I. Furthermore, Table I shows also the medians and scores of the four 3-sign assignments, which do not have to be computed according to Proposition 7. The assumed 3-sign for I defines the sign of element 2 in the quotient permutation of I. The solutions of the oRMP of node I when element 2 is signed according to the 3-sign \( s_i \) of I are also given in Table I. In this example the 3-signs \((+, +, +)\) and \((+, -, -)\) lead to a minimal overall score \( \xi_{\min}(I, s) = 9 \) for the subtree induced by the prime nodes. The computation of one median by applying the permutations implied by oRMP solutions and reversals resulting from sign differences between linear nodes starting from \( \pi_1 \) is shown in Figure 2(b). Note that the traces connecting the quotient permutations with their medians can be easily computed with known algorithms, e.g. [18].

The pRMP was determined by solving 4 oRMPs for \( I_1 \) and 4 oRMPs for I. The strong interval tree has 4 linear nodes with parents having a different 3-sign \((7, 7, 8, 9, 10), (9, 10), \) and \((10)\) and the 3-sign of the linear root node is not equal to \((+, +, +)\). Together this implies 5 necessary reversals for the linear nodes. Therefore the overall score is 14. There are two solutions of this pRMP \((1 6 2 3 4 5 7 8 9 10), (5 4 3 2 1 6 7 8 9 10)\), which can be found when the oRMP medians of I and I are combined for those k-signs where \( \xi_{\min}(I, s) \) is minimal.

If one wants to compute only one median, three of the oRMPs for I do not have to be computed. The oRMP for prime node \( I_1 \) has a unique minimum score of 4 for 3-sign \((+, +, -)\), the other 3-signs yield a score of 5. According to Proposition 8 it is sufficient to consider only this 3-sign for the computation of the oRMP for I. Hence for prime node I only one oRMP has to be solved. Altogether only 5 of the 8 oRMPs have to be solved.

**D. Pseudo Code of Algorithm TCIP: Runtime Analysis, and Extensions**

The pseudo code for TCIP is given in Algorithm 1 for the case that the output is a single preserving median of the given input permutations. For computing all preserving medians the pseudo code has to be modified, such that it iterates over all 3-sign vectors \( (s_1, \ldots, s_p) \) for which the property as given in line 38 of the pseudo code holds. Note that in the case, that only one preserving median has to be computed, it is sufficient to compute only one median in the induced ORMPs. This may also reduce the runtime significantly.

In the following we analyze the runtime of TCIP for three input permutations of size \( n \). The preserving median scenario as well as the median can be computed in linear time for strong interval trees that are unambiguous and have no prime node. More technically this can be done as follows (not shown...
in the pseudo code). Each of the permutations π₁, π₂, and π₃ is stored as a double-linked list with the signed elements as node entries. In each node of the strong interval tree the pointers to the first and last element of the corresponding 3-sign common interval in permutation πₖ, k ∈ [1,3], are stored. These pointers are determined when the k-signed strong interval tree is computed in linear time. Applying a reversal to a permutation is done by relinking the corresponding pointers in the permutations and an easy handling for the signs of the elements. Hence, a reversal corresponding to a linear node can be applied in constant time. In the case of unambiguous trees with no prime nodes the algorithm has to iterate over all nodes of the strong interval tree. For each node three sign comparisons, at most one sign update, and at most one reversal has to be done. Because the strong interval tree has O(n) nodes the overall runtime is in O(n) time in the case of unambiguous trees with no prime nodes.

If there exist prime nodes in the strong interval tree the induced oRMPs have to be solved. Recall, that solving the oRMP is NP-hard; nevertheless with Caprara’s median solver [19] oRMP instances with input permutations of size n ≤ 50 can often be solved within seconds or minutes.

1) Number of oRMPs to be solved: In the following we analyze the number of oRMPs that have to be solved for a given strong interval tree.

If a prime node I has p prime child nodes I₁, ..., Iₚ, then 4ᵖ different 3-sign vectors (s₁, ..., sₚ) have to be considered for these child nodes. For each combination of a 3-sign s ∈ Σ for I and a 3-sign vector for the prime child nodes, one oRMP has to be solved (compare lines 18-21 of Algorithm 1). An exception is the case that I is the root or I has a linear parent where the 3-sign is already known (not shown in Algorithm 1). It should be mentioned that the number of oRMPs to be solved per prime node can potentially be reduced according to Proposition 8, if only one preserving median has to be computed (not shown in Algorithm 1). If, for example, each child node Iₖ has a unique minimum ξₖ(Iₖ, sₖ) for different sₖ then only 4 oRMPs have to be solved for I. The total number of oRMPs that have to be solved in the worst case is \( \sum_{I \in \mathcal{P}} 4^{n(I)+1} \), where \( \mathcal{P} \) is the set of all prime nodes in the strong interval tree, n(I) is the number of prime child nodes of node I, and r(I) = 1 if I has a prime node parent and otherwise r(I) = 0. In the best case only \( \sum_{I \in \mathcal{P}} 4^{n(I)} \) oRMPs have to be solved.

Recall, that the size of the oRMPs induced by the prime nodes is given by the size of its quotient permutations. The number of oRMPs to be solved in the worst case can be calculated in linear time by the formula given above. Thus, the size of the quotient permutations and the number of oRMPs to be solved can be used for an estimation of the run runtime of TCIP.
2) Enumeration of all Preserving Medians: To compute the preserving medians after solving the orRMPs can lead to a large runtime simply because there can exist many medians. Theoretically, in each prime node \( i \) it may hold that \( \xi_{\min}(l, s) \) has the same values for all 3-signs \( s \in \Sigma \). In this case each combination of 3-sign assignments to prime nodes has to be analyzed, as each different 3-sign in each prime node may lead to a different preserving median. Hence, the number of 3-sign combinations to be analyzed is \( \Pi_{I \in \mathcal{P}} 4^{n(l)} \) in the worst case, where from each combination a set of preserving medians can be easily inferred with the solutions of the corresponding orRMPs in the prime nodes. If only one preserving median has to be computed, it is enough to analyze only one 3-sign combination per prime node component. In Algorithm 1 (lines 38-42) only one 3-sign vector \((s_1, \ldots, s_p)\) is assigned to the prime node children.

Since the number and size of prime nodes strongly influence the runtime of algorithm TCIP, the occurrence of prime nodes in the strong interval tree is analyzed in detail in Section V.

3) Extensions: So far we have considered only the case of non circular (i.e. linear) signed gene orders. But for evolutionary scenarios the input permutations might be circular (i.e., all circular shifts of a gene order are assumed to represent the same genome) or linear (not circular). Furthermore the permutations can be directed, (i.e., \( \pi \neq -\pi \)) or undirected (i.e., \( \pi = -\pi \)). The linear directed case has already been discussed.

A solution for the circular undirected case can be derived easily from the linear directed case as follows: Let \( \pi_1, \pi_2, \) and \( \pi_3 \) be three circular undirected gene orders with a preserving median scenario \( \mathcal{S} \). Then, by applying inversion and circular shift, there exist three gene orders \( \omega_1 = \pi_i, 1 \leq i \leq 3 \), such that the first element of \( \omega_i, 1 \leq i \leq 3, \) is 1. Note that in a circular scenario we can replace each reversal by its circular complement without changing the score or the resulting median. Hence, for \( \omega_i = \pi_i, 1 \leq i \leq 3, \) exists a (circular undirected) preserving scenario \( \mathcal{S}' \) with the same score as \( \mathcal{S} \) and such that no reversal \( \rho(i, j) \) has a start index \( i \) that is larger than its end index \( j \). \( \mathcal{S}' \) is also a preserving median scenario under the assumption that \( \omega_i, 1 \leq i \leq 3, \) are linear and directed. Therefore, we can solve the circular undirected case by shifting and inverting the input permutations as explained.

For linear undirected input permutations the handling of the root node has to be changed: i) linear root nodes can be left unchanged, since the relative orientation of the three permutations does not matter, ii) for prime root nodes all possible 3-sign assignments have to be tested (instead of only \((+, +, +)\)) and the assignments which lead to a minimal score have to be chosen.

V. Results

In this section properties of the strong interval trees for artificial and biological data will be investigated. Furthermore, an empirical runtime analysis of algorithm TCIP is performed and the improvements of TCIP that have been introduced on basis of the theoretical results are evaluated.

A. Data Sets

As test instances for the pRMP we have used randomly generated test instances and test instances that come from biological data sets.

1) Construction of Random Data Sets: The random test instances were generated as follows. Starting with the identity permutation, \( k \) random reversals were applied to generate a random permutation. Three permutations generate an instance of pRMP. More precisely, let \( C_1 := \{(n, k) | n \in \{20, 40, \ldots, 100\}, k \in \{1, 2, \ldots, 10\} \cup \{n/20, 2n/20, \ldots, 15n/20\}\} \) be the set of all considered combinations of \( n \) and \( k \) (this results in 103 different combinations). For each \((n, k) \in C_1\), 10000 pRMP instances have been generated. A subset of parameter value combinations, namely \( C_2 := \{(n, k) | (n, k) \in C_1 \text{ and } k < n/3 \lor k \leq 10\} \), was used for analyzing computation times and properties of the score distribution in the prime nodes. Set \( C_2 \) contains 62 different parameter values. For each of these values 1000 pRMP instances have been created.

2) Biological Data Sets: As biological data we use the mitochondrial genome orders from [26] that are marked in [26] as complete and have the standard set of 37 mitochondrial genes (13 protein coding-, 2 rRNA-, and 22 tRNA- genes). With the help of taxonomical information in the mitochondrial database and by manual inspection of the phylogenetic tree taken from the NCBI database the corresponding species were grouped into different taxa. The Deuterostomia were partitioned into the Chordata (Cho) and the remaining Deuterostomia — Hyperotreti, Cephalochordata, Echinodermata, and Hemichordata (HCEH). We also consider two subgroups of the Chordata the Actinopterygii (Act), and Sarcopterygii (Sar). The Protostomia were partitioned into the Arthropoda (Art) and the remaining Protostomia — Annelida, Pogonophora, Brachiopoda, and Mollusca (APBM). We also consider two subgroups of the Arthropoda: the Crustacea (Cru) and the Hexapoda (Hex). Also the species of the Nematoda and the Platyhelminthes (NP) form a group. Note that nearly all mitochondrial gene orders of the species in the last group have no gene atp8, hence here we have used a reduced set of 36 genes. Additionally the chloroplast gene orders (CP) from [27] are used (the dataset is included in the GRAPPA distribution [28]). This set consists of 13 distinct gene orders each having 105 genes. The number of different gene orders within each group can be found in Table II (#). For each taxonomic group all possible triples of different gene orders were used as test instances. Altogether, there are 36262 test instances.

B. Properties of Strong Interval Trees

In this subsection we analyze some properties of strong interval trees, that are crucial for the runtime of the pRMP solver TCIP. In Table II (respectively Table III) results are presented for the biological (respectively random) data sets. The results for the random data sets are only given for instances with permutation size \( n = 100 \). Results for different permutation sizes are very similar\(^1\). For random as well as biological data sets there are many instances which have only linear nodes. For each of the three Chordate groups (Act, Sar, Cho) the majority of the strong interval trees has no prime node (59-74%). Also for the Hexapoda there many of strong interval trees have only linear nodes (29%). For random data sets only triples generated with few reversals have no prime nodes. The strong interval trees that have prime nodes, mostly only have a very

\(^1\)Additional results are available at http://pacosy.informatik.uni-leipzig.de/pv/Forschung/IEEE-2008-pRMP-supp.pdf
the average over all instances and the second value (in parentheses) is the maximal value.

<table>
<thead>
<tr>
<th>p</th>
<th>q</th>
<th>r</th>
<th>l</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>Act</td>
<td>27</td>
<td>26.5</td>
<td>1.00 (1)</td>
<td>1.00 (1)</td>
</tr>
<tr>
<td>Sar</td>
<td>28</td>
<td>41.3</td>
<td>1.06 (2)</td>
<td>1.06 (8)</td>
</tr>
<tr>
<td>Cho</td>
<td>47</td>
<td>39.9</td>
<td>1.03 (2)</td>
<td>1.03 (8)</td>
</tr>
<tr>
<td>HCEH</td>
<td>11</td>
<td>98.9</td>
<td>1.10 (2)</td>
<td>1.33 (8)</td>
</tr>
<tr>
<td>Cru</td>
<td>18</td>
<td>90.6</td>
<td>1.09 (2)</td>
<td>1.51 (8)</td>
</tr>
<tr>
<td>Hex</td>
<td>15</td>
<td>70.7</td>
<td>1.08 (3)</td>
<td>1.10 (8)</td>
</tr>
<tr>
<td>Art</td>
<td>42</td>
<td>91.6</td>
<td>1.12 (4)</td>
<td>1.28 (24)</td>
</tr>
<tr>
<td>APBM</td>
<td>16</td>
<td>100.0</td>
<td>1.01 (2)</td>
<td>1.02 (8)</td>
</tr>
<tr>
<td>NP</td>
<td>9</td>
<td>98.8</td>
<td>1.05 (3)</td>
<td>1.05 (3)</td>
</tr>
<tr>
<td>CP</td>
<td>13</td>
<td>95.5</td>
<td>1.03 (2)</td>
<td>1.03 (2)</td>
</tr>
</tbody>
</table>

small number of prime nodes. The maximum number of prime nodes over all test instances is 4. The average number of prime nodes over all strong interval trees that have at least one prime node in each group of instances (biological taxonomic group, or random instances with the same value of $k$) is at most 1.12. When more than one prime node occurs in a strong interval tree, these prime nodes usually do not occur within one prime node component. Hence, the number of prime nodes with prime node children is small. Therefore the number of oRMPs to be solved per instance is also small.

The worst case that occurred in terms of the number of oRMPs that have to be solved, is a strong interval tree with 3 prime nodes for one prime node component. This case occurred only once in the Art data set and once in each of the artificial data sets with $k = 3$ and $k = 6$. In each of these cases 24 oRMPs had to be solved. For all test instances where the strong interval tree has 4 prime nodes the number of oRMPs to be solved was smaller, as the prime nodes did not occur within one prime node component. For all the test instances that have the largest number of prime nodes that has been found (i.e., 4 prime nodes) in the strong interval tree, the number of oRMPs that had to be solved was at most 10. The reason is that at most 2 of these prime nodes were connected. For sets of gene orders that have relatively small evolutionary distances (e.g., the Chordates or the Arthropods) the size of the oRMPs that had to be solved is small compared to the size of the original RMP (i.e., the size of the signed quotient permutations). The same holds for the sets of randomly generated gene orders with small value of $k$. For the biological test instances the maximum size of a quotient permutation of a prime node that occurred is 36. For the random instances with $k \leq 10$, the maximal size of a quotient permutation in a prime node is 53. Only for random instances with large values, $k \geq 25$, the largest quotient permutation of prime nodes that occurred is not much smaller than the input permutation. For example for $k = 75$ the average size of quotient permutations that occurred (for instances that have at least one prime node) is 98.7. Altogether, the results show that many pRMP instances are very suitable for TCIP; i.e., TCIP has to solve only few and small oRMP instances.

C. Computing RMP Medians

In this subsection we present computation times for computing pRMP medians with TCIP. As a comparison we also compute RMP medians with Caprara’s RMP solver [19] which is known to be very fast compared to other approaches for the RMP. Since Caprara’s RMP solver can compute only one median of the RMP we also used a slightly modified version that can compute all medians (this version is described in [4]). We empirically analyze the size of preserving median sets and present a comparison of pRMP and RMP median sets. Furthermore, an empirical analysis of the reduction of oRMP instances to be solved with TCIP, due to the theoretical results in this paper, is given.

1) Computation times: In Figure 3 boxplots are given for the run times of TCIP and Caprara’s RMP solver when, i) only one (preserving) median and, ii) all (preserving) medians are computed. All test runs were done on a PC with an AMD Opteron 2.0 GHz processor. It can be seen clearly, that in most cases it is faster to compute all pRMP medians with TCIP than to compute all RMP medians with Caprara’s RMP solver. For example TCIP was on average 768 times faster than Caprara’s median solver in the Sarcopterygii data set when computing all medians. The only exception is the NP data set where Caprara’s median solver is about 1.25 times faster.

10.8% of all instances of the biological data set TCIP was more than 1000 times faster when computing all medians. The maximal improvement was achieved in the Arthropoda data set, where TCIP was 7.28 million times faster on one instance. Note again, that Caprara’s median solver does not solve the pRMP but the RMP. Nevertheless, we compared the computation times to show the good performance of TCIP. In [13] an algorithm called ECIP was presented, which also solves the pRMP. ECIP was compared to a slightly modified version of Caprara’s median solver (for being able to solve the pRMP) and ECIP was shown to be faster than the latter. As ECIP and Caprara’s modified median solver were not able to solve all problem instances that are presented here in reasonable time, we have not included their computation times. Algorithm TCIP is much faster than ECIP, often achieving speedups of $10^2$ – $10^3$ for the instances presented in this paper. We empirically verified the correctness also by comparing the results of TCIP, ECIP, and Caprara’s modified median solver.
2) **Number of preserving medians:** The number of preserving medians in the different groups of test instances of the biological data is shown in Table IV. It can be seen that the number of preserving medians can be very large (more than 1 million in some instances of the NP group). Nevertheless, the fraction of the overall computation time that is needed for the enumeration of all medians (after all oRMPs were solved) is very small in the artificial as well as in the biological data set, even when the number of preserving medians is very large: it is always either < 0.1sec or < 2% of the overall computation time. This is due to the fact that the prime node components for the test instances are very small.

3) **Comparison of pRMP and RMP medians:** In Figure 4 the relation of solutions for the RMP and the corresponding pRMP is given. For each of the artificial and biological test instances we computed the score difference $\Delta s$ and the difference of the size of the median sets $\Delta \# \text{medians}$ relatively to the RMP solution, i.e. a positive value for $\Delta s$ (respectively $\Delta \# \text{medians}$) indicates that the pRMP solution has a larger score (respectively the median set of the pRMP solution is larger). For each combination of $\Delta s$ and $\Delta \# \text{medians}$ we computed the number of instances that occurred in the corresponding class. Based on 62000 artificial test instances with $(n, k) \in C_2$ and 35976 biological test instances, the number of instances that fall into a certain class are depicted in the histograms in Figure 4. For $\Delta s = 0$ the size of median sets can only decrease, as in this case each median of a pRMP instance is a median for the corresponding RMP instance, too. For larger values of $\Delta s$ the median set of a pRMP instance can have more or less elements than the median set of the corresponding RMP instance. Figure 4 clearly shows, that preserving medians are good approximations for the corresponding RMP: 55680 (i.e. 89.9%) of all artificial test instances and 32572 (i.e. 90.5%) of all instances have a score of $\Delta s \leq 1$. Consequently, TCIP can also be used as a good heuristic algorithm for the RMP.

4) **Specific Aspects of TCIP:** In Figure 3 it can be seen, that — as expected — computing one median with TCIP is much faster than computing all medians. For example, for the 560 triple instances of the APBM group, the average speedup per instance is 15.3, for some instances of the Art group even speedups of > 200 were achieved. As shown in the theoretical part of this paper, the reasons are threefold i) in the induced oRMPs only one median has to be computed, ii) due to Proposition

---

**Table IV**

<table>
<thead>
<tr>
<th></th>
<th>avg (max)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Act</td>
<td>2.75 (366)</td>
</tr>
<tr>
<td>Sar</td>
<td>2.14 (83)</td>
</tr>
<tr>
<td>Cho</td>
<td>2.91 (366)</td>
</tr>
<tr>
<td>HCEcH</td>
<td>95.53 (3553)</td>
</tr>
<tr>
<td>Cru</td>
<td>49.84 (3428)</td>
</tr>
<tr>
<td>Hex</td>
<td>21.44 (1184)</td>
</tr>
<tr>
<td>Art</td>
<td>61.53 (18360)</td>
</tr>
<tr>
<td>APBM</td>
<td>263.82 (24995)</td>
</tr>
<tr>
<td>NP</td>
<td>26583.54 (1192157)</td>
</tr>
<tr>
<td>CP</td>
<td>46.87 (1841)</td>
</tr>
</tbody>
</table>
8 the number of oRMPs to be solved is reduced, and iii) the preserving medians do not have to be enumerated.

If only one instead of all preserving medians are computed, the number of oRMP instances is potentially reduced. This reduction strongly depends on the number of 3-signs in a prime node \( I \) with a prime node parent, for which a minimal \( \xi(I, s) \) is found (compare Proposition 8). Let \( \Xi \) be the number of 3-signs that lead to such a minimal \( \xi_{\min}(I, s) \) (i.e., \( \Xi \) is the multiplicity of minimal 3-signs \( s \) in the multiset \( \{\xi_{\min}(I, s) \mid s \in \Sigma\} \)). In Figure 5 the frequency of \( \Xi = 1, \ldots, \Xi = 4 \) is depicted for all prime nodes with a prime node parent i) in all 1.03 million random test instances and ii) in all 36262 biological test instances. It can be clearly seen, that \( \Xi = 1 \) occurs very often. In the random data set this is the case in 828 of the 953 prime nodes with prime node parents. Due to small values for \( \Xi \), in the random data set the number of oRMPs to be solved is reduced from 662515 to 643303, i.e., 19212 oRMPs do not have to be solved due to Proposition 8. In the biological data set the number of oRMPs to be solved is reduced from 47828 to 47014. The number of saved oRMPs seems relatively small. This is because prime node components are very small in all test instances. Note that theoretically for prime nodes with a lot of prime node children for which \( \Xi = 1 \) holds, the number of saved oRMPs is exponential in the number of such prime node children.

VI. Conclusions

Based on an extension of strong interval trees, namely \( k \)-signed strong interval trees, we introduced a new algorithm called TCIP for solving the preserving reversal median problem (pRMP) for three given gene orders. The pRMP is to find a median for given signed input permutations, such that no reversal in the median scenario is allowed to break one of the common intervals of the input permutations. It was analyzed theoretically how the difficulty of the pRMP depends on the structure of the corresponding \( k \)-signed strong interval tree. Based on these results, algorithm TCIP was designed such that it can solve an pRMP instance by solving several smaller instances of the RMP (RMP is the pRMP without the requirement that reversals are preserving). Empirically it was shown that often no RMP instances have to be solved at all, leading to a linear runtime behavior for TCIP. For computing one preserving median instead of all, several non-trivial techniques are applied within algorithm TCIP. For data sets of mitochondrial gene orders and for randomly generated data sets, several important properties of the corresponding strong interval trees have been analyzed. Moreover, it was shown empirically, that preserving median scenarios can be computed even faster with TCIP than standard (i.e., not necessarily preserving) median scenarios can be computed with the state-of-the-art RMP solver by Caprara (and a variant thereof). This is interesting because even computing the preserving minimum reversal distance between two gene orders is an NP-hard problem (whereas the same problem with reversals that are not necessarily preserving can be solved in polynomial time). Furthermore it was shown that algorithm TCIP can be used as a good heuristic for the RMP. In our future work we will include TCIP in the algorithm amGRP [4] which is a state-of-the-art algorithm for computing phylogenetic trees that is based on frequently solving the RMP.

Acknowledgment

We thank Kai Ramsch for his helpful comments and Monika Popp for proofreading the article. We also thank the anonymous reviewers for their very helpful remarks. This work was...
For proving Proposition 1 we need the following definition and a theorem from [22].

Definition 1: (Reformulation of Definition 9 in [22]) Let $F$ be a set of common intervals of a set of permutations $\Pi$. The closure $F^*$ of $F$ is the smallest set of common intervals of $\Pi$ that contains $F$, all trivial common intervals of $\Pi$, and for any $I_1 \in F^*$ and $I_2 \in F^*$, with $I_1$ and $I_2$ overlapping, $I_1 \cup I_2$, $I_1 \cap I_2$, $I_1 \setminus I_2$, and $I_2 \setminus I_1$ belong to $F^*$.

Theorem 2: (Reformulation of Theorem 5 in [22]) Let $F$ be a set of common intervals of a set of permutations $\Pi$. Let $\pi$ be a permutation and $J = (I_1, \ldots, I_k)$ be the partition of $\pi$ into maximal strong intervals of $F^*$ other than $\pi$ itself. Then, exactly one of the following is true:

1. either every union of consecutive elements $I = \{i, \ldots, j\}$ of $P_J$ is a common interval of $P_J$ and $K = \bigcup_{i \leq h < j} I_h$ belongs to $F^*$, or
2. no union of intervals of $I$ belongs to $F^*$.

As the set of all common intervals $C(\Pi)$ for a set of permutations $\Pi$ is closed by definition of common intervals, Theorem 2 is in particular true for $F = F^* = C(\Pi)$.

Note that the following proof of Proposition 1 is basically the same as the proof of Proposition 2 in [22], where only pairwise preserving scenarios were considered.

Proof: [Proof of Proposition 1] Let $S$ be a scenario of $\Pi$ to $\sigma$. Suppose that every reversal of $S$ is either a node of $T(\Pi)$ or the union of children of a prime node of $T(\Pi)$. Let $I$ be a reversal of $S$. If $I$ is a node of $T(\Pi)$, $I$ is a strong common interval and then commutes with every common interval of $\Pi$. Now, assume that $I$ is the union of children of a prime node $J$. $I$ obviously commutes with any common interval not contained in $J$ and with any common interval contained in a child of $J$. Hence, it remains to show that $I$ commutes with any common interval that is a union of children of $J$, but there are none by definition of a prime node. It follows that $I$ commutes with every common interval of $\Pi$ and, then, $S$ is a preserving scenario (with respect to $\Pi$).

Conversely, suppose that $S$ is a scenario of $\pi \in \Pi$ and $\sigma$, that is preserving with respect to $\Pi$. Let $I$ be a reversal of $S$, and consider the partition $I_1, \ldots, I_k$ of $I$ in which the part containing an element $x$ of $I$ is the largest strong common interval included in $I$ and that contains $x$. If $k \geq 2$, then $I_1, \ldots, I_k$ must all be children of the same parent $J$ in $T(\Pi)$; otherwise, $I$ would not commute with the nodes of $T(\Pi)$ that are parents of $I_j$. If $J$ is a linear node, then $I$ must be equal to $J$; otherwise, $I$ would overlap an interval formed by a leftover child of $J$ and one of the intervals of $I$ and such an interval is a common interval of $\Pi$ by points 1 and 2 of Theorem 2 using $F = C(\Pi)$. Therefore, either $k = 1$ and $I$ is a node of $T(\Pi)$, or $k > 1$ and the node $I$ must be prime.

APPENDIX II

THE GENERALIZED MEDIAN PARITY THEOREM

Theorem 3: (The Generalized Median Parity Theorem) Let $I$ be a node of the strong interval tree $T^k(\Pi, \Pi)$ of a set of signed permutations $\Pi = \{\pi_1, \ldots, \pi_k\}$. Let $I$ be a node of $T^k(\Pi, \Pi)$ with linear parent $J$, where $s(I) \neq s(J)$ and $s(I) \neq \overline{s}(J)$ hold. Let $E$ denote the set of indices $i$ for which $s_i(I) = s_i(J)$ holds, where $s_i$ denotes the $i$-th entry of $s$. The reversal of $I$ applied to all $J$-th traces with $j \in E$, leads to $s(I) = \overline{s}(J)$. Let $\overline{E} = \{1, \ldots, n\} \setminus E$. If $|\overline{E}| < |E|$, then the reversals of $I$ in all $J$-th traces, $j \in \overline{E}$, are parents of $I$ and with any common interval contained in a child of $I$, that is preserving with respect to $I$. If $|\overline{E}| = |E|$ then the reversals of $I$ in all $J$-th traces with either all $j \in E$ or all $j \not\in E$ belong to any preserving median scenario for $\Pi$.

Proof: Let $S$ be a preserving median scenario for $\Pi$ and let $I$ be a node in the strong interval tree $T^k(\Pi, \Pi)$ for which $s(I) \neq s(J)$ and $s(I) \neq \overline{s}(J)$ holds. Let $R$ be the set of indices, for which the reversal of node $I$ is applied to the $j$-th trace, $j \in R$, in the scenario $S$.

Assume that $R \not\in E$ and $R \not\in \overline{E}$. Then $s(I) \neq s(J)$ and $s(I) \neq \overline{s}(J)$ holds after applying all reversals in all $J$-th traces with $j \in R$. By Propositions 1 and 2, $s(I) \neq s(J)$ and $s(I) \neq \overline{s}(J)$ still hold after applying all remaining reversals from $S$, as a reversal that inverts a sign in $s(J)$ also inverts the corresponding sign in $s(I)$ and all other reversals do not change $s(I)$ or $s(J)$. Therefore, $s(I) = s(J)$ or $s(I) = \overline{s}(J)$ is not achievable by $S$. But then $S$ can not be a preserving median scenario because it is clear that after applying all reversals of a preserving median scenario the $k$-sign of each node must be equal to $(+, \ldots, +)$ or equal to $(-, \ldots, -)$. From this contradiction follows immediately that all reversals on all $J$-traces must occur in any preserving median scenario, where indices $j$ are all from the smaller of the two sets $E$ and $\overline{E}$. If the sets $E$ and $\overline{E}$ have equal size then both choices — applying reversals in all $J$-th traces for $j \in E$ and $j \not\in \overline{E}$ respectively — are parsimonious.

REFERENCES


Daniel Merkle Daniel Merkle received the Diploma degree in Computer Science at the University of Karlsruhe, Germany, in 1997. He finished his Ph.D. in applied computer science at the Institute of Applied Informatics and Formal Description Methods, University of Karlsruhe, Germany in 2002. He worked as an assistant professor with the Parallel Computing and Complex Systems Group of the Department of Computer Science in Leipzig, Germany, till April 2008. Since then he is an Associate Professor at the Department of Mathematics and Computer Science, University of Southern Denmark. His research interests include phylogenetics, algorithms from nature, parallel algorithms, and social insect behavior.
Algorithm 1 \textsc{TCIP}(\pi_1,\pi_2,\pi_3)

1. \textbf{INPUT}: signed permutations \(\Pi := (\pi_1,\pi_2,\pi_3)\)
2. \textbf{OUTPUT}: a preserving median \(\mu\)
3. Initialize the traces \(\pi'_k := \pi_k, k \in [1:3]\), let \(\Pi' := (\pi'_1,\pi'_2,\pi'_3)\)
4. Compute the 3-signed strong interval tree \(
T^3(\Pi,\Pi')\)
5. // Linear node handling
6. \textbf{while} \(\exists\) linear node \(I\) with linear parent \(J\) with \((\pi(I) \neq \pi(J)\) and \(\pi(I) \neq \pi(J))\) (traversal bottom-up) \textbf{do}
7. \textbf{for} all \(I\) \textbf{do}
8. \textbf{end for}
9. \textbf{end while}

10. // Prime node handling
11. \textbf{for all} prime node components \(T_p\) in \(
T^3(\Pi,\Pi')\) \textbf{do}
12. \textbf{for all} prime nodes \(I\) in \(T_p\) (traversal bottom-up) \textbf{do}
13. \textbf{end for}
14. \textbf{end for}
15. \textbf{end for}

16. \textbf{for all} prime nodes \(I\) in \(T_p\) (traversal top-down) \textbf{do}
17. \textbf{if} the root node \(I\) of \(T_p\) has a linear parent \textbf{then}
18. \textbf{else}
19. \textbf{end if}
20. \textbf{end for}

21. \textbf{end for}

22. apply all reversal to \(\pi'_k, k \in [1:3]\) that are induced by the oRMP solutions, update \(\Pi'\)
23. \textbf{end for}

24. if root node \(I\) has not the 3-sign \((+,+,+)\) or \((-,-,-)\) \textbf{then}
25. apply a reversal to one trace, such that the signs of the 3-sign of \(I\) become identical
26. \textbf{end if}

27. \textbf{end for}

28. \textbf{end for}

29. \textbf{end for}

30. \textbf{end for}