

# Agreement of three or more trees (1995; Farach, Przytycka, Thorup)

Teresa M. Przytycka\*NCBI/NLM/NIH, [www.ncbi.nih.gov/~przytyck](http://www.ncbi.nih.gov/~przytyck)

entry editor: Ming-Yang Kao

**INDEX TERMS:** evolutionary trees, maximum agreement subtree, comparing evolutionary trees.

**SYNONYMS:** tree alignment.

## 1 PROBLEM DEFINITION

The Maximum Agreement Subtree problem for  $k$  trees (k-MAST) is a generalization of a similar problem for two trees (MAST). Consider a tuple of  $k$  rooted leaf-labeled trees  $(T_1, T_2 \dots T_k)$ . Let  $A = \{a_1, a_2, \dots a_n\}$  be the set of leaf labels. Any subset  $B \subseteq A$  uniquely determines the so called *topological restriction*  $T|B$  of the tree  $T$  to  $B$ . Namely,  $T|B$  is the topological subtree of  $T$  spanned by all leaves labeled with elements from  $B$  and lowest common ancestors of all pairs of these leaves. In particular, the ancestor relation in  $T|B$  is defined so that it agrees with the ancestor relation in  $T$ . A subset  $B$  of  $A$  such  $T^1|B, \dots, T^k|B$  are isomorphic is called an *agreement set*.

### Problem 1 (k-MAST).

INPUT: A tuple  $\vec{T} = (T^1, \dots, T^k)$  of leaf-labeled trees, with a common set of labels  $A = \{a_1, \dots, a_n\}$ , such that for each tree  $T^i$  there exists one-to-one mapping between the set of leaves of that tree and the set of labels  $A$ .

OUTPUT:  $k$ -MAST( $\vec{T}$ ) equal to the maximum cardinality agreement set of  $\vec{T}$ .

## 2 KEY RESULTS

In the general setting, k-MAST problem is NP-complete for  $k \geq 3$  [1]. Under the assumption that the degree of at least one of the trees is bounded, Farach *et al.* proposed an algorithm leading to the following theorem:

**Theorem 1.** *If the degree of the trees in the tuple  $\vec{T} = (T^1, \dots, T^k)$  is bounded by  $d$  then the  $k$ -MAST( $\vec{T}$ ) can be computed in  $O(kn^3 + n^d)$  time.*

In what follows, the problem is restricted to finding the cardinality of the maximum agreement set rather than the set itself. The extension this algorithm to an algorithm that finds the agreement set (and subsequently the agreement subtree) within the same time bounds is relatively straightforward.

Recall that the classical  $O(n^2)$  dynamic programming algorithm for MAST of two binary trees [11] processes all pairs of internal nodes of the two trees in a bottom up fashion. For each pair of such nodes it computes the MAST value for the subtrees rooted at this pair. There are  $O(n^2)$  pairs of nodes and the MAST value for the subtrees rooted at a given pair of nodes can be computed in constant time from MAST values of previously processed pairs of nodes.

---

\*Supported by the intramural research program of the National Library of Medicine National Institutes of Health.

To set the stage for the more general case, let  $\text{k-MAST}(\vec{v})$  be the solution to the k-MAST problem for the subtrees of  $T^1(v_1), \dots, T^k(v_k)$  where  $T^i(v_i)$  is the subtree if  $T^i$  rooted at  $v_i$ . We say that  $\vec{v}$  is *dominated* by  $\vec{u}$  (denoted  $\vec{v} \prec \vec{u}$ ) if, for all  $i$ ,  $u_i$  is a strict ancestor of  $v_i$  in  $T^i$ .

A naive extension of the algorithm for two trees to an algorithm for  $k$  trees would require computing  $\text{k-MAST}(\vec{v})$  for all possible tuples  $\vec{v}$  by processing these tuples in the order consistent with the domination relation. The basic idea that allows to avoid  $\Omega(n^k)$  complexity is to replace the computation of  $\text{k-MAST}(\vec{v})$  with the computation of a related value,  $\text{mast}(\vec{v})$ , defined to be the size of the maximum agreement set for the subtrees of  $(T^1, \dots, T^k)$  rooted at  $(v_1, \dots, v_k)$  subject to the additional restriction that the agreement subtrees themselves are rooted at  $v_1, \dots, v_k$  respectively. Clearly

$$\text{k-MAST}(T^1, \dots, T^k) = \max_{\vec{v}} \text{mast}(\vec{v}).$$

The benefit of computing  $\text{mast}$  rather than  $\text{k-MAST}$  follows from the fact that most of  $\text{mast}$  values are zero and it is possible to identify (very efficiently)  $\vec{v}$  with non-zero  $\text{mast}$  values.

**Remark 1.** If  $\text{mast}(\vec{v}) > 0$  then  $\vec{v} = (\text{lca}^{T^1}(a, b), \dots, \text{lca}^{T^k}(a, b))$  for some leaf labels  $a, b$  where  $\text{lca}^{T^i}(a, b)$  is the lowest common ancestor of leaves labeled by  $a$  and  $b$  in the tree  $T^i$ .

A tuple  $\vec{v}$  such that  $\vec{v} = (\text{lca}^{T^1}(a, b), \dots, \text{lca}^{T^k}(a, b))$  for some  $a, b \in A$  is called an *lca-tuple*. By Remark 1 it suffices to compute  $\text{mast}$  values for the lca-tuples only. Just like in the naive approach,  $\text{mast}(\vec{v})$  is computed from  $\text{mast}$  values of other lca-tuples dominated by  $\vec{v}$ . Another important observation is that only some lca-tuples dominated by  $\vec{v}$  are needed to compute  $\text{mast}(\vec{v})$ . To capture this, Farach *et al.* define the so called *proper domination* relation (introduced formally below) and show that the  $\text{mast}$  value for any lca-tuple  $\vec{v}$  can be computed from  $\text{mast}$  values of lca-tuples properly dominated by  $\vec{v}$  and that the proper domination relation has size  $O(n^3)$ .

**Proper Domination Relation.** Index the children of each internal node of any tree in an arbitrary way. With each pair  $\vec{w}, \vec{v}$  of lca-tuples such that  $\vec{w} \prec \vec{v}$  We associate a domination direction  $\vec{\delta}_{\vec{w} \prec \vec{v}} = (\delta_1, \dots, \delta_k)$  where  $w_i$  descends from the child of  $v_i$  indexed with  $\delta_i$ . Let  $v_i(j)$  be the child of  $v_i$  with index  $j$ . The direction domination is termed *active* is if the subtrees rooted at the  $v_1(\delta_1), \dots, v_k(\delta_k)$  have at least one leaf label in common. Note that each leaf label can witness only one active direction, and consequently each lca-tuple can have at most  $n$  active domination directions. Two directions  $\vec{\delta}_{\vec{w} \prec \vec{v}}$  and  $\vec{\delta}_{\vec{u} \prec \vec{v}}$  are called *compatible* if and only if the direction vectors differ in all coordinates.

**Definition 1.**  $\vec{v}$  properly denominates  $\vec{u}$  (denoted  $\vec{u} \prec \vec{v}$ ) iff  $\vec{v}$  dominates  $\vec{u}$  along an active direction  $\vec{\delta}$  and there exists another tuple  $\vec{w}$  which is also dominated by  $\vec{v}$  along an active direction  $\vec{\delta}_\perp$  compatible with  $\delta$ .

From the definition of proper domination and from the fact that each leaf label can witness only one active we make the following observations:

**Remark 2.** The strong domination relation  $\prec$  on lca-tuples has size  $O(n^3)$ . Furthermore, the relation can be computed in  $O(kn^3)$  time.

**Remark 3.** For any lca-tuple  $\vec{v}$ , if  $\text{mast}(\vec{v}) > 0$  then either  $\vec{v}$  is an lca-tuple composed of leaves with the same label or  $\vec{v}$  properly dominates some lca-tuple.

It remains to show how the values  $\text{mast}(\vec{v})$  are computed. For each lca-tuple  $\vec{v}$ , the so called compatibility graph  $G(\vec{v})$  is constructed. The nodes of  $G(\vec{v})$  are active directions from  $\vec{v}$  and there is an edge between two such nodes if and only if corresponding directions are compatible.

The vertices of  $G(\vec{v})$  are weighted and the weight of a vertex corresponding to an active direction  $\vec{v}$  equals the maximum mast value of a lca-tuple dominated by  $\vec{v}$  along this direction. Let  $MWC(G(\vec{v}))$  be the maximum weight clique in  $G(\vec{v})$ .

The bottom-up algorithm for computing non-zero mast values is based on the following recursive dependency whose correctness follows immediately from the corresponding definitions and Remark 3:

**Lemma 2.** *For any lca-tuple  $\vec{v}$*

$$\text{mast}(\vec{v}) = \max \begin{cases} 1 & \text{if all elements of } \vec{v} \text{ are leaves.} \\ MWC(G(\vec{v})) & \text{otherwise} \end{cases} \quad (1)$$

The final step is to demonstrate that once the lca-tuples and the strong domination relation is pre-computed, the computation of all non-zero mast values can be performed in  $O(n^d)$  time. This is done by generating all possible cliques for all  $G(\vec{v})$ . Using the fact that the degree of at least one tree is bounded by  $d$  one can show that all the cliques can be generated in  $O(\sum_{l \leq d} \binom{n}{l}) = O(d^3 (ne/d)^d)$  time and that there is  $O(d(ne/d)^d)$  of them [6].

### 3 APPLICATIONS

The k-MAST problem is motivated by the need to compare evolutionary trees. Recent advances in experimental techniques in molecular biology provide diverse data that can be used to construct evolutionary trees. This diversity of data combined with the diversity of methods used to construct evolutionary trees often leads to the situation when the evolution of the same set of species is explained by different evolutionary trees. Maximum Agreement Subtree problem has emerged as a measure of the agreement between such trees and as a method to identify subtree which is common for these trees. The problem was first defined by Finden and Gordon in the context of two binary trees [7]. These authors also gave a heuristic algorithm to solve the problem. The  $O(n^2)$  dynamic programming algorithm for computing *MAST* values for two binary trees has been given in [11]. Subsequently, a number of improvements leading to fast algorithms for computing *MAST* value of two trees under various assumptions about rooting and tree degrees [5, 8, 10] and references therein.

The *MAST* problem on 3 or more unbounded degree trees is NP-complete [1]. Amir and Keselman report an  $O(kn^{d+1} + n^{2d})$  time algorithm for the agreement of  $k$  bounded degree trees. The work described here provides in  $O(kn^3 + n^d)$  for the case where the number of trees is  $k$  and the degree of at least one tree is bounded by  $d$ . For  $d = 2$  the complexity of the algorithm is dominated by the first term. An  $O(kn^3)$  algorithm for this case was also given by Bryant [4] and  $O(n^2 \log^{k-1} n)$  implementation of this algorithm was proposed in [9].

k-MAST problem is fixed parameter tractable in  $p$ , the smallest number of leaf labels such that removal of the corresponding leaves produces agreement (see [3] and references therein). Approximability of the *MAST* and related problem has been studied in [2] and references therein.

### 4 OPEN PROBLEMS

None is reported.

### 5 EXPERIMENTAL RESULTS

None is reported.

## 6 DATA SETS

None is reported.

## 7 URL to CODE

None is reported.

## 8 CROSS REFERENCES

00178, 00186, 00190

## 9 RECOMMENDED READING

- [1] Amihoud Amir and Dmitry Keselman. Maximum agreement subtree in a set of evolutionary trees: Metrics and efficient algorithms. *SIAM J. Comput.*, 26(6):1656–1669, 1997.
- [2] Vincent Berry, Sylvain Guillemot, François Nicolas, and Christophe Paul. On the approximation of computing evolutionary trees. In *COCOON*, pages 115–125, 2005.
- [3] Vincent Berry and François Nicolas. Improved parameterized complexity of the maximum agreement subtree and maximum compatible tree problems. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 3(3):289–302, 2006.
- [4] David Bryant. Building trees, hunting for trees, and comparing trees: theory and methods in phylogenetic analysis. In *Ph.D. thesis, Dept. Math., University of Canterbury*, 1997.
- [5] Richard Cole, Martin Farach-Colton, Ramesh Hariharan, Teresa M. Przytycka, and Mikkel Thorup. An  $O(n \log n)$  algorithm for the maximum agreement subtree problem for binary trees. *SIAM Journal on Computing*, pages 1385 – 1404, 2001.
- [6] Martin Farach, Teresa M. Przytycka, and Mikkel Thorup. On the agreement of many trees. *Information Processing Letters*, 55(6):297–301, 1995.
- [7] C. R. Finden and A. D. Gordon. Obtaining common pruned trees. *Journal of Classification*, 2:255–276, 1985.
- [8] Ming-Yang Kao, Tak-Wah Lam, Wing-Kin Sung, and Hing-Fung Ting. An even faster and more unifying algorithm for comparing trees via unbalanced bipartite matchings. *J. Algorithms*, 40(2):212–233, 2001.
- [9] Chuan-Min Lee, Ling-Ju Hung, Maw-Shang Chang, and Chuan-Yi Tang. An improved algorithm for the maximum agreement subtree problem. *BIBE*, 533, 2004.
- [10] Teresa M. Przytycka. Transforming rooted agreement into unrooted agreement. *J. Comput. Biol.*, 5 (2):335–349, 1998.
- [11] Mike A. Steel and Tandy Warnow. Kaikoura tree theorems: Computing the maximum agreement subtree. *Inf. Process. Lett.*, 48(2):77–82, 1993.