MADALGO
CENTER FOR MASSIVE DATA ALGORITHMICS


Konstantinos Mampentzidis
Aarhus University


Gerth Stølting Brodal
Aarhus University

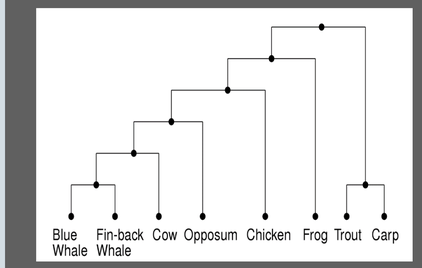
Danmarks Grundforskningsfond
Danish National Research Foundation

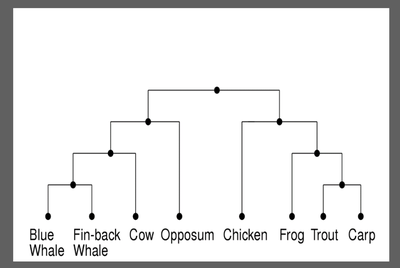
A Practical O($n \cdot \log n$) Algorithm for Computing the Triplet Distance on Binary Trees

The Problem

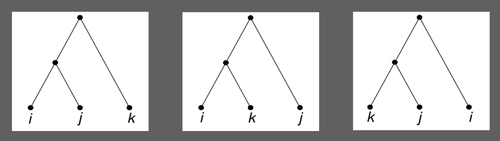
Trees are used in many scientific fields to represent relationships. For example in Biology a tree can represent evolutionary relationships, with the leafs corresponding to species that exist in the present and internal nodes to ancestor species that existed in the past.

For the same set of leafs, different data or construction methods can lead to different trees.



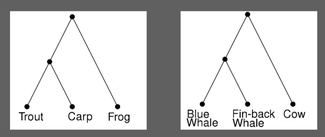


To quantify the difference, various distance measures between trees are used. When the trees are rooted, among the most popular ones is the **triplet distance**. A triplet is a set of three leaf labels { *i*, *j*, *k* } and it is the smallest possible leaf set that can induce different topologies in two rooted trees.

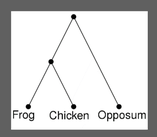


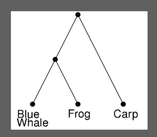
If T_1 and T_2 are two rooted trees on the same leaf set of size n, then the triplet distance $D(T_1,T_2)$ is defined as the total number of triplets that induce different topologies in the two trees. For the two example trees given above, we have $D(T_1,T_2) = 22$.

Common triplets



Different triplets





All algorithms that have been developed for this problem find $S(T_1,T_2)$, which is defined as the number of triplets that induce the same topology in T_1 and T_2 (commonly referred to as *shared triplets*).
Having S available, then $D(T_1,T_2) = \binom{n}{3} - S(T_1,T_2)$.

Results

Authors	Running time	Technique
	$O(n^3)$	naive
Critchlow *et al.* [1]	$O(n^2)$	Generational matrices
Brodal *et al.* [2]	$O(n \cdot \log^2 n)$	Tree coloring + hierarchical decomposition
Brodal *et al.* [3]	$O(n \cdot \log n)$	Tree coloring + hierarchical decomposition + contraction
Jansson *et al.* [4]	$O(n \cdot \log^3 n)$	Tree coloring + heavy light decomposition
new	$O(n \cdot \log n)$	**Tree coloring + centroid decomposition + contraction**

Motivation

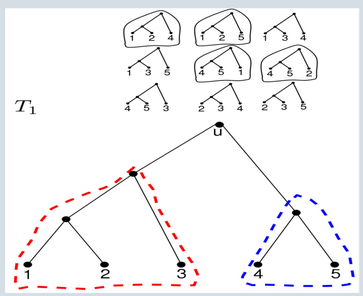
All previous solutions rely heavily on random access.

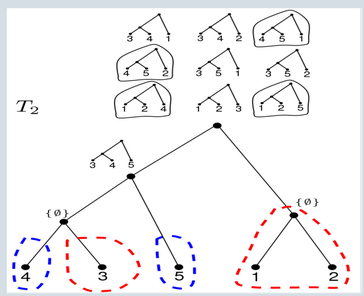
- The algorithms are being penalized by cache performance
- No chance that they can scale to external memory

The goal of this work is to take an approach that is more friendly towards the memory hierarchy, where the primary primitive will be scanning.

Basic Idea

Every triplet { *i*, *j*, *k* } is implicitly assigned to the lowest common ancestor in both T_1 and T_2. For an internal node $u \in T_1$ we find the number of shared triplets that are rooted in u by coloring the leafs red and blue in the two subtrees below u.





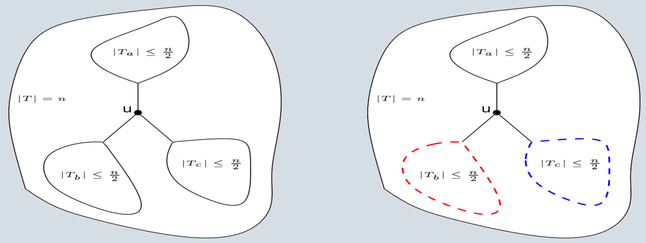
Visiting each internal node $u \in T_1$ in any order, coloring both trees according to u and applying a bottom up traversal on T_2 will give us an $O(n^2)$ time algorithm (*tree coloring* technique).

New Approach

Main ideas

- **Change the order in which we visit the nodes in T_1.**
- **Contract the trees to remove leafs with no color while maintaining the topologies induced by the leafs with color.**

1. By applying the centroid decomposition on T_1, we can find internal nodes that split T_1 into subtrees/components of small size.



2. After finding all the shared triplets that are rooted in an internal node u, we recurse to the three components that are generated. While doing so, we contract T_2 so that its size is always proportional to the size of the corresponding component in T_1.

Open Problems

- Compare the practical performance against the state of the art algorithms.
- Possible to extend to higher degree trees without increasing the theoretical running time?
- Possible to tweak the algorithm to make it O(n)? (**big open problem**)

References

[1] D.E. Critchlow, D.K. Pearl and C. Qian. *The triples distance for rooted bifurcating phylogenetic trees*. Syst. Biol. 1996.

[2] A. Sand, G.S. Brodal, R. Fagerberg, C.N.S. Pedersen and T. Mailund. *A practical O($n \cdot \log^2 n$) algorithm for computing the triplet distance on binary trees*. BMC Bioinformatics 2013.

[3] G.S. Brodal, R. Fagerberg, T. Mailund, C.N.S. Pedersen and A. Sand. *Efficient algorithms for computing the triplet and quartet distance between trees of arbitrary degree*. SODA 2013.

[4] J. Jansson and R. Rajaby. *A more practical algorithm for the rooted triplet distance*. AlCoB 2015.