Efficiently Calculating Evolutionary Tree Measures Using SAT

Maria Luisa Bonet[1] and Katherine St. John[2]

[1] Lenguajes y Sistemas Informáticos, Universidad Politécnica de Cataluña, Spain
[2] Math & Computer Science Dept., Lehman College, City U. New York, USA

Abstract. We develop techniques to calculate important measures in evolutionary biology by encoding to CNF formulas and using powerful SAT solvers. Comparing evolutionary trees is a necessary step in tree reconstruction algorithms, locating recombination and lateral gene transfer, and in analyzing and visualizing sets of trees. We focus on two popular comparison measures for trees: the hybridization number and the rooted subtree-prune-and-regraft (rSPR) distance. Both have recently been shown to be NP-hard, and efficient algorithms are needed to compute and approximate these measures. We encode these as a Boolean formula such that two trees have hybridization number k (or rSPR distance k) if and only if the corresponding formula is satisfiable. We use state-of-the-art SAT solvers to determine if the formula encoding the measure has a satisfying assignment. Our encoding also provides a rich source of real-world SAT instances, and we include a comparison of several recent solvers (minisat, adaptg2wsat, novelty+p, Walksat, March KS and SATzilla).

1 Introduction

Phylogenies, or evolutionary histories, play a central role in biology. While traditionally represented as trees, due to evolutionary processes such as hybridization, horizontal gene transfer and recombination [16], the relationship between many species is better represented by networks, or directed graphs. These nontree events connect nodes from different branches of a tree, and they are usually called *reticulations* (see Figure 1). Given two trees that represent the evolutionary history of different genes of a set of species, the *hybridization number* between the trees characterizes the number of reticulation events needed to explain the evolution of the set of species. With the recent explosion in biological data available, it is now possible to compute multiple phylogenetic trees for a set of taxa (species), based on many different gene sequences. Calculating the differences between species and gene trees very efficiently is essential to building evolutionary histories, and in turn to understanding the underlying properties of the species. Further, comparing phylogenies play important roles in locating recombination and lateral gene transfers, and analyzing searches in treespace.

Our primary focus is on calculating the hybridization number. The related rooted subtree-prune-and-reconnect (rSPR) distance is often used as a surrogate.

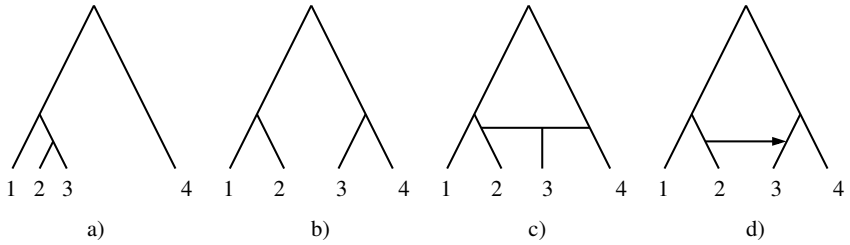

Fig. 1. Hybridization events: a) and b) represent two different gene trees on the same set of species, and c) and d) show two possible evolutionary scenarios. In c), species 2 and 4 hybridize (combine genetic information) to form a new species 3. In d), we show lateral gene transfer where some of the genetic information from species 3 is derived along one lineage as in tree in a), while other information is derived along the lineages shown in b).

rSPR captures individual hybridization events but misses an important acyclicity condition that taxa cannot have themselves as ancestors. Further, while often similar in size, there exist instances where the difference between the rSPR and hybridization number are arbitrarily large [5].

Calculcating tree measures is of great interest, and the focus of much recent work. Bordewich and Semple [6] showed that the hybridization number is NP-hard and fixed parameter tractable, by relating it with an appropriately defined agreement forest. Agreement forests were developed for evolutionary tree metrics in the pioneering work of Hein *et al.* [14] and Allen and Steel [1] that linked the tree distance to the size of the maximum agreement forest (MAF). With the development of a MAF for the rooted subtree-prune-and-reconnect (rSPR) distance [5] (see Figure 2), Bonet *et al.* [4] showed these algorithms are a 5-approximation for rSPR distance. Algorithms for biologically relevant restricted cases of rSPR were also developed by Hallett and Lagergren [13] and Beiko and Hamilton [3]. Nakhleh *et al.* [20] developed a very fast heuristic for rSPR distance, which due to its basis on maximum agreement subtrees, also yields bounds on the hybridization number. Wu [28] encodes the rSPR problem into an integer linear programming instance, achieving good results for the rSPR problem only. To find exact answers for hybridization numbers, Linz *et al.* [7] used clever combinatorial characterizations to yield an exhaustive search that does well for surprisingly large values.

We have developed new software tools to calculate hybridization number and rSPR distance, by transforming these into satisfiability (SAT) questions. Using combinatorial characterizations and insights of past work, we can often reduce the scope of the problem to several smaller subproblems for hybridization, or a single smaller problem for rSPR. We use two different approaches to calculating these measures: exact calculation and an upper bound heuristic. Our novel contribution is the use of powerful SAT solvers to finish this final part of the computation on the reduced trees. We do this by encoding the problem as a

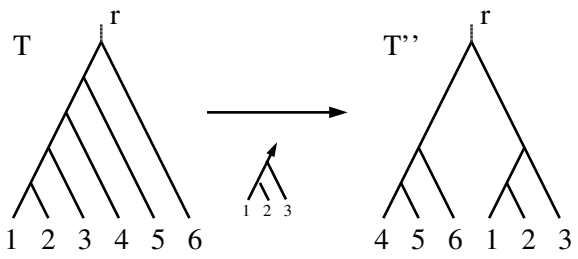

Fig. 2. rSPR Move: A rooted SPR move breaks off a subtree from the first tree and reattaches the subtree to another tree. For technical reasons, we represent our rooted trees as "planted trees" and allow rSPR moves to reattach subtree to the edge of the root, as done with the rSPR move above.

Boolean formula such that two trees have some particular or hybrid number (or rSPR distance) if and only if the corresponding formula is satisfiable. Then we give the formula as input to one of the best SAT solvers. Due to the large community focused on techniques to solve SAT more efficiently, there are many different choices of SAT solvers, optimized for differing criteria.

For our upper bound heuristic (SAT Descent), we work down from an upper bound (instead of eliminating possibilities counting up from zero). In this case we do a comparison among several solvers. They are walksat [24, 25], adaptg2wsat [8], novelty+p [8], minisat [10, 11], SATzilla [29] and March KS [15]. Notice that we compare all kinds of different solvers: local search algorithms (the first three), DPLL with learning (minisat), SAT solver portfolio (SATzilla) and solver specialized on random instances (March KS). The performance of minisat on our instances was worse in general than the performance of the local search solvers. Using local search algorithms yields excellent results in both accuracy and performance. For example, we find solutions for biological data sets in 48 seconds that take over 11 hours with the exact program, HybridNumber and do not finish after two days of compute time using the complete solver minisat.

This paper is organized as follows: we give background on tree measures and agreement forests in Section 2. Section 3 details our methods, with more information on the SAT encoding in Section 4. Section 5 describes the data analyzed. Results are in Section 6, followed by discussion and future work in Section 7.

2 Hybridization Networks and Agreement Forests

The recent theoretical results have linked tree measures to the size of maximum agreement forests [14]. This link has been used to show NP-hardness, fixed parameter tractability, and is the basis for approximation algorithms. Roughly, each measure corresponds to the size of the appropriately defined maximum agreement forest. For a more thorough treatment, see [5, 18, 26].

Subtree Prune and Regraft (SPR): A **subtree prune and regraft** (SPR) operation [1] on a binary tree T is defined as cutting any edge and thereby pruning a subtree t, then regrafting the subtree by the same cut edge to a new vertex obtained by subdividing a pre-existing edge in $T - t$. We apply a forced contraction to maintain the binary property of the resulting tree (see Figure 2). The **SPR distance** between two trees T_1 and T_2 is the minimal number of SPR moves needed to transform T_1 into T_2. When working with rooted trees, we refer to this distance as **rooted SPR** or **rSPR**. Bordewich and Semple [5] showed that the rSPR distance of two trees is the same as the size of an appropriately defined maximum agreement forest for rooted trees of the two trees. This number is related to another measure between trees that we next define.

Hybridization Number: A **hybridization network** on a leaf set X [5, 26] is a rooted acyclic directed graph with root ρ in which

- X is the set of leaves (vertices of outdegree zero);
- $d^+(\rho) \geq 2$;
- for all the vertices v with $d^+(v) = 1$, we have $d^-(v) \geq 2$.

Let $d^-(v)$ be the indegree of v and $d^+(v)$ be the outdegree of v. The vertices with indegree at least two represent the hybridization vertices. Now, we define the **hybridization number** of a hybridization network H with root ρ as

$$h(H) = \sum_{v \neq \rho} (d^-(v) - 1).$$

Let T be a rooted phylogenetic tree and H a hybridization network. We say H **displays** T [5, 26] if T can be obtained from H by first deleting a subset of edges of H and any resulting isolated vertices, and then contracting edges. Then given two trees T_1 and T_2,

$h(T_1, T_2) = min\{h(H) : H \text{ is a hybridization network that displays } T_1 \text{and } T_2\}$.

We define the **hybridization number** of two trees T_1 and T_2 as the minimal hybridization number of all hybridization network H that display T_1 and T_2.

Agreement Forest: Originally linked to tree measures [14], agreement forests are an essential tool for calculating and showing hardness for tree measures. Roughly, an **agreement forest** for T_1 and T_2 with identical leaf set X, is a set of subtrees that occur in both the initial trees T_1 and T_2, where:

1. The subtrees partition the leaf set X into $\{X_0, \ldots, X_k\}$.
2. The subtrees occur as induced subtrees of T_1 and T_2. i.e. for each $i, 0 \leq i \leq k$, T_1 restricted to the set of leaves X_i, and T_2 restricted to the set of leaves X_i are the ith subtree.
3. The subtrees are vertex disjoint in both T_1 and T_2.

For two trees, T_1 and T_2, with the same leaf set, a **maximum agreement forest (MAF)** is an agreement forest with the minimal number of subtrees. Allen and Steel [1] show the size of the MAF corresponds to another tree measure, the tree-branch-and-reconnect (TBR) distance. Augmenting this forest definition to handle rooted trees, Bordewich and Semple [5] link these new MAFs to rSPR distance. Figure 3 illustrates agreement forests for rSPR distance.

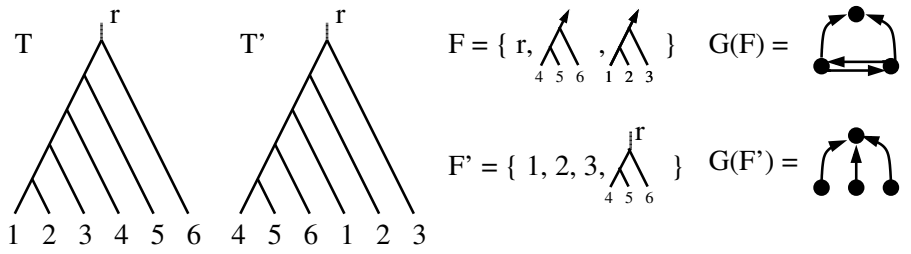

Fig. 3. Agreement Forests: F and F' are two possible forests for the trees T and T'. F is also maximal for rSPR, but its associated graph, $G(F)$ contains a cycle and is thus not a good agreement forest for hybridization. The second, larger forest, is acyclic, and is the maximum agreement forest for hybridization. The rSPR distance is 2, while the hybrid number is 3.

Hybrid Number and Acyclicity of the Forest: We define the graph, G_F of a MAF F of two trees T_1 and T_2 as follows: the nodes are the trees of F, and there is an edge from one node (F_1) to (F_2) corresponding to two trees of F if the root of (F_1) is a descendant of the root of (F_2) in either T_1 or T_2. Adding the simple condition that the graph of the forest is acyclic yields a MAF for hybridization number. That is, a forest that is maximal with respect to all agreement forests that have acyclic associated graphs has size equivalent to the hybridization number of the two trees [6]. See Figure 3.

Hardness Results: Both of these measures, hybridization number and rSPR distance have been shown to be NP-hard and fixed parameter tractable [5, 6]. The following operations help reduce the size of the trees and provide additional efficiency for our methods by "shrinking" the size of the problem encoded:

Subtree Reduction (Rule 1 of [5]): Replace any pendant subtree that occurs identically in both trees T_1 and T_2 by a single leaf with a new label.

Our second rule looks at clusters in trees. While not part of the fixed parameter tractability reduction for hybridization number, it gives important reductions on the sizes of the trees and improves the performance. A is a **cluster** for T_1 and T_2 if there is a node in each tree that has A as its set of descendants in X. We note that this reduction preserves hybridization number but does not preserve rSPR distance [2]:

Cluster Reduction (Rule 3 of [2]): Let T_1 and T_2 be two rooted binary X-trees, and $A \subset X$ a cluster of both T_1 and T_2. Then,

$$h(T_1, T_2) = h(T_1 \mid A, T_2 \mid A) + h(T_{1a}, T_{2a})$$

where T_{1a} (T_{2a}) is the result of substituting the subtree of T_1 (T_2) having leaf set A by the new leaf a and $T_1 \mid A$ ($T_2 \mid A$) is the restriction of T_1 (T_2) to A.

3 Methods

We develop four related algorithms for calculating the tree measures: exact solutions ('SAT Ascent') and upper bound heuristics ('SAT Descent') for both hybridization number and rSPR distance. Our input is two trees, T_1 and T_2, that represent the evolution of two different genes of a set of species. Our methods break into several parts:

1. Efficient preprocessing to reduce size, using known reductions (see §2),
2. Encoding the questions "hybridNumber$(T_1, T_2) = r$?" and "$d_{rSPR}(T_1, T_2) = r$?" as Boolean formulas,
3. Using fast heuristics [20] to give starting upper bounds, and
4. Using different search strategies and solvers to answer these questions.

Efficient Preprocessing: Each of the reduction rules can be performed in linear time, following a clever coding of trees by Day [9]. His coding stores sufficient information about each internal vertex to identify internal structure. This takes $O(1)$ space per internal vertex, allowing linear time algorithms for the reduction rules presented in the previous section (see [4] for more details).

Encoding: We describe the SAT encoding in more detail in the next section.

Efficient Heuristics: We use RIATA-HGT from the PhyloNet program suite [20] to give starting points for our upper bounds. While not an approximation algorithm (since families of trees can be constructed whose distance is fixed, but whose distance found by the algorithm is arbitrarily large), RIATA-HGT performs very well in practice (see Figures 4 and 5). It takes the input trees and calculates a maximum agreement subtree. The maximum agreement subtree is added to the forest and then used as a "backbone" and the algorithm is then repeated for each subtree hanging from the backbone. While not explicitly stated, the resulting forest is acyclic by construction and thus gives an upper bound for both rSPR distance and hybridization number.

Different Search Strategies and SAT Solvers: We use Minisat [10, 11] to find exact solutions for rSPR and hybrid number. On the other hand, we use Walksat [24, 25], adaptg2wsat [8], novelty+p [8] for the upper bounds of both measures. We use the UBCSAT implementation [27] for the latter two since it was significantly faster than the stand-alone versions. We compare the performance of these three local search solvers among themselves and also with the performance of the complete solvers minisat,March KS and SATzilla. As we will see in the experimentation, the local search algorithms work much faster in general.

Software: We built four different methods that calculate upper bounds for hybridization numbers, upper bounds for d_{rSPR}, exact solutions for hybridization number, and exact solutions for d_{rSPR}. The software is written in perl and java, using the TreeJuxtaposer [19] java code base. All four have similar format, so, we only describe the upper bound for hybridization numbers in detail:

1. Preprocess by the reduction rules to yield smaller pairs of trees.
2. Find a starting upper bound for each pair using RIATA-HGT [20].
3. Starting with the upper bound, r, encode the formula for hybridization is r and use a SAT solver to find a satisfiable assignment (i.e. a MAF).
4. Decrement r and loop to 3, until a satisfiable assignment is *not* found. Return $r + 1$.

We similarly define the algorithm for upper bounds for d_{rSPR}. For the SAT Ascent algorithm, we begin by looking for an agreement forest of size 1 and work upwards until a forest is found.

4 Encoding

Our program takes pairs of phylogenetic trees on the same leaf set and a proposed size for the MAF and produces SAT instances in DIMACS SAT format:

INPUT: Two trees, T_1 and T_2, and an integer $r > 0$.
OUTPUT: An encoding into a SAT instance, in the DIMACS SAT format.

The resulting formula will be satisfiable if the hybridization number (rSPR distance) between T_1 and T_2 is $\leq r$. We rely on the correspondence to agreement forests, described in Section 2. Namely, that $d_{rSPR}(T_1, T_2) = r$ iff there is a maximum agreement forest for T_1 and T_2 of size r. Similarly, the hybridization number of T_1 and T_2 is r iff there is a maximum *acyclic* agreement forest for T_1 and T_2 of size r. Thus, most of the encoding focuses on saying that a agreement forest exists:

Literals: For each subtree i in the forest and leaf j from the original leaf set, we have a literal l_{ij} which is true iff leaf j is part of subtree i in the agreement forest. We have similar sets of literals for internal vertices of T_1 and T_2. We also have literals to reduce the number of clauses needed (explained below) and to represent the acyclic conditions. The number of literals is $O(rn + r^2)$. Since $r < n$, this yields $O(nr)$.

Clauses for Subtrees Partition Leaf Sets: It is easy to say that every leaf is in at least one subtree, by having clauses for each leaf j, $l_{0j} \vee l_{1j} \vee \ldots \vee l_{rj}$, that literally say, "leaf j is in subtree 0 or leaf j is in subtree 1 or ... leaf j is in subtree r. This takes $O(rn)$ clauses.

To say that every leaf occurs in at most one subtree is more difficult. The obvious encoding takes $O(rn^2)$. Following [17], we introduce $O(rn)$ new literals, s_{ij} and use them to reduce the number of clauses needed to $O(rn)$. The intuition for these new literals and corresponding clauses is that they encode $\sum_i l_{ij} \leq 1$. The new variables signal when leaf j occurs in some tree i, and the clauses ensure that this happens for only one i.

Clauses for Subtrees Occurring as Induced Trees: The clauses below assert that the $r + 1$ subtrees occur in both T_1 and T_2. This is done in a similar manner as above: we show that every internal vertex is in at most one subtree. Note that we do not need to say that every internal node is in at least one

subtree. We need new variables to say to which subtrees of the agreement forest the internal vertices of T_1 and of T_2 belong to. If a rooted binary tree has n leaves, then it has $n-1$ internal vertices. For tree T_1, we have variables v_{ij}, for $0 \leq i \leq r$ and $1 \leq j \leq n-1$ such that v_{ij} is true iff the jth internal vertex is part of the ith subtree. Similarly, for tree T_2, we have variables v'_{ij}.

We will further have two sets of variables to reduce the number of clauses needed: $t_{i,j}$ and $t'_{i,j}$ for $i = 0, \ldots, r$ and $j = 1, \ldots, n-1$ (these are similar to the s variables used for the leaves of the trees). The clauses for the internal nodes of the trees state:

1. Every internal vertex of T_1 (and of T_2) is in at most one subtree.
   This follows the same idea as in the previous step with v and t for T_1 and with v' and t' for T_2. This is done twice to require that all the internal vertices of both the input trees occur at most once in the subtrees of the forest.
2. If two leaves occur in a subtree, then internal vertices on the path between them must also occur in the same subtree.
   First, look at tree T_1 (the clauses for T_2 will be almost identical). For every pair of leaves, j and k in T_1, there exists a unique path between them of internal vertices, $v_{p_1}, v_{p_2}, \ldots, v_{p_x}$ (x and the internal vertices on the path depend on the leaves chosen and could be 0, if $i = j$, or up to $n-1$). Our clauses state that if j and k occur in subtree i, then so do the nodes on the path between them: $v_{p_1}, v_{p_2}, \ldots, v_{p_x}$. So for $i = 0, \ldots, r$ and $j, k = 1, \ldots, n-1$ we need the clauses saying

   $$(l_{ij} \wedge l_{ik}) \rightarrow (v_{ip_1} \wedge v_{ip_2} \wedge \ldots \wedge v_{ip_x})$$

   Note that the internal vertices and the paths *depend* on the particular tree.

Clauses for Checking that Subtrees are Equal: Once we have that the leaves form subtrees, we add clauses to guarantee that the structure of the subtrees is the same in both T_1 and T_2. This is the last condition needed to have that the subtrees form an rSPR agreement forest for T_1 and T_2. To do this, we look at triples of all leaves and their structure in T_1 and T_2. If the structure differs, then we add clauses preventing that triple of leaves from occurring in the same tree. In the worst case, this takes $O(rn^3)$ clauses, but in practice it is significantly smaller.

Clauses for Acyclic Conditions: For hybridization, the agreement forest also needs to be acyclic. Adding variables to represent that there is a directed edge between subtrees is $O(r^2)$. The clauses needed to encode the initial edges, transitive closure of the edge relationship, and forbid cycles takes $O(r^3)$.

Expected Number of Clauses: The theoretical bound on the number of clauses in this encoding is quite high, $O(rn^3)$ where n is the number of taxa in the trees and r is the hybridization number (rSPR distance) that is encoded. However, in practice, we see significantly smaller number of clauses generated by the encoding. This large difference in sizes is due to the clauses needed to

check that the internal substructure of the subtrees are equal. It is possible that all the $O(n^3)$ triplets of taxa will differ in structure in T_1 and T_2, resulting in $O(rn^3)$ clauses. In practice, most trees compared have are similar and as such most of triplets agree, and few are needed. For example, the theoretical upper bound for unreduced trees with 50 taxa and with a starting upper bound of 13 is 1,625,000. For a pair chosen at random from our simulated dataset, the reduction rules shrunk the size of the trees to 39 taxa from the initial 50 taxa and the starting upper bound is 13. The number of literals and clauses depend on the size of the reduced tree pairs and the starting upper bound. They are 3,416 literals and 370,571 clauses, a huge reduction from the worst case bound for the full trees and half of the bound calculated for the reduced trees.

5 Data

We analyze both biological and simulated data. The biological data set, from the analysis of HybridNumber [7] and described more fully there, is from the Poaceae (Grass) family. Hybridization is a well-recognized occurrence in grasses [12], making this an excellent test data set. The data set consists of sequence data for six loci: internal transcribed spacer of ribosomal DNA (ITS); NADH dehydrogenase, subunit F (ndhF); phytochrome B (phyB); ribulose 1,5-biphosphate carboxylase/oxygenase, large subunit (rbcL); RNA polymerase II, subunit (rpoC2); and granule bound starch synthase I (waxy). For each loci, a tree was built using the fastDNAmL program [21] by Heiko Schmidt [23]. As in [7], we looked at pairs of trees, reduced to their common taxa. In all, we have 15 pairs of trees. The pairs and the number of overlapping taxa are listed in Figure 4.

The simulated datasets were generated to capture small and medium distances between reasonably sized trees. All trees have 50 taxa. For each run, we generated a "species" tree, and then 10 "gene" trees by making k rSPR-moves from the species tree for $k = 2, 4, 6, 8, 10, 12, 14$. These give tree pairs with rSPR distance at most k, since it is possible for some of the sequence of moves to "cancel" each other out. The hybridization number could be larger than k, since its corresponding maximum agreement forest is that for rSPR with additional acyclic conditions. Each of the species trees was generated with Sanderson's `r8s` program [22], using Yule-Harding distribution. The program that alters the species tree by k rSPR moves chooses a non-pendant edge uniformly and at random (software written by the authors in Java). For each k, 10 trials were generated, yielding 100 species-gene tree pairs, for a total of 700 pairs of trees.

6 Results

We show the results for the hybridization number algorithms. The rSPR distance results have similar, and often worst running times, since cluster reduction rule does not apply to rSPR distance. This rule often breaks the problem into reasonably sized subproblems, speeding computation.

Poaceae (Grass) Dataset: The results for this dataset are presented in Figure 4. Our exact solution algorithm does well at small cases, as HybridNumber does but slows down for larger instances sooner. On the other hand, our SAT Descent algorithm performs extremely well using the local search algorithm, Walksat, finding the true number in 11 out of 12 of the known cases and doing so in under five minutes time. Surprisingly, Walksat outperforms more recent local search algorithms including adaptg2wsat (which recently won a silver medal in SAT2007 competition in satisfiable random formula category). All the local search algorithm outperformed the complete solvers, which often ran out of time before completing the calculations. In Figure 4, we do not include the results for March KS, since this solver performed very poorly on almost all these instances. RIATA-HGT returns answers extremely quickly, all in less than 12 seconds, but overestimates by average of 9%.

Simulated 50 Taxa Dataset: Figure 5 contains the graphs for the simulated data for both accuracy and speed. Both HybridNumber and SAT Ascent solver could not calculate the solutions for $r \geq 6$ in the 24 hour time-limit used for these experiments. Since the SAT Ascent solver's results mirror HybridNumber, we report only the latter. Our upper bound software did extremely well in both accuracy and speed. By construction, SAT Descent with local search algorithms always gave answers that were closer to the true answer. RIATA-HGT finished in under 15 seconds for all runs. SAT Descent with local search algorithms completed all runs in less than 15 minutes. The standard deviations were omitted from Figure 5 but are worth noting. For small values of k, they are below 5% for the time and accuracy of both RIATA-HGT and SAT upper bound. The standard deviation for the time for RIATA-HGT remains below 2% for all values. For all other algorithms, the standard deviations rise for both time and accuracy to almost 20%, illustrating the variability of difficulty of problems even for small and medium values.

7 Discussion & Conclusion

Encoding problems as SAT instances has positive and negative points. On the negative side, we must build a SAT instance that may be even bigger than the original problem. On the positive side, once the hard work of encoding is done, we can use the variety of SAT tools to try many different search strategies to improve our algorithms in both efficiency and time. In a way, it is like having several solvers in one, since we can benefit from all the different tools that the SAT community has developed over the years and from future improvements of SAT solvers.

Our novel approach of encoding the NP-hard problems of calculating hybridization number and rSPR distance into SAT instances yields an elegant and efficient algorithm for estimating these measures. While not an exact answer, our algorithms often find the true answer in a fraction of the time needed to search for the exact solution. Given the ever-improving state of SAT-solvers, these results will only improve, allowing for better bounds. Future work includes

improving the encoding, finding tighter bounds via combinatorial analysis of the inputs, and uses for related tree problems such as TBR distance.

One final observation is that our grass instances are an unusual case of combinatorial real problems better solved by local search algorithms than by DPLL solvers. Even though the instances come from real data, we are encoding an NP-hard problem of complexity similar to random instances, and local search solvers win the Random Satisfiable category in competitions.

Acknowledgements

The first author was partially supported by the Spanish grant TIN2007-68005-C04-03. Also, this project was partially supported by USA NSF grants SEI 0513660, and by MRI 0215942 and the computational research center at the CUNY Graduate Center. The second author would like to thank the Centre de Recerca Matemàtica in Barcelona for hosting her visits in 2007. We also thank Charles Semple, Simone Linz, and Carlos Ansotegui for helpful conversations and the Munzner group (UBC) for the TreeJuxtaposer [19] code base.

References

1. B. Allen and M. Steel. Subtree transfer operations and their induced metrics on evolutionary trees. *Annals of Combinatorics*, 5:1–13, 2001.
2. Mihaela Baroni, Charles Semple, and Mike Steel. Hybrids in real time. *Systematic Biology*, 55:46–56, 2006.
3. RG Beiko and N Hamilton. Phylogenetic identification of lateral genetic transfer events. *BMC Evol Biol.*, 6:15, 2006.
4. Maria Luisa Bonet, Katherine St. John, Ruchi Mahindru, and Nina Amenta. Approximating subtree distances between phylogenies. *Journal of Computational Biology*, 13(8):1419–1434, 2006.
5. M. Bordewich and C. Semple. On the computational complexity of the rooted subtree prune and regraft distance. *Annals of Combinatorics*, 8:409–423, 2005.
6. M. Bordewich and C. Semple. Computing the minimum number of hybridization events for a consistent evolutionary history. Discrete Applied Mathematics, 2007.
7. Magnus Bordewich, Simone Linz, Katherine St. John, and Charles Semple. A reduction algorithm for computing the hybridization number of two trees. *Evolutionary Bioinformatics*, 3:86–98, 2007.
8. Harry Zhang Chu Min Li, Wanxia Wei. Combining adaptive noise and look-ahead in local search for SAT. In *In proceedings of 10th international conference on the Theory and Applications of Satisfiability Testing (SAT2007), Lisbon, Portugal*, 2007.
9. W. H. E. Day. Optimal algorithms for comparing trees with labeled leaves. *Journal of Classification*, 2:7–28, 1985.
10. Niklas Eén and Niklas Sörensson. Software available from `http://www.cs.chalmers.se/Cs/Research/FormalMethods/MiniSat/`.
11. Niklas Eén and Niklas Sörensson. An extensible SAT-solver. In *Proc. of the 6th Int. Conf. on Theory and Applications of Satisfiability Testing, SAT'03*, volume 2919 of *Lecture Notes in Computer Science*, pages 502–518, Santa Margherita Ligure, Italy, 2003. Springer.

12. Grass Phylogeny Working Group. Phylogeny and subfamilial classification of the grasses (poaceae). *Annals of the Missouri Botanical Garden*, 88(3):373?457, 2001.
13. M.T. Hallett and J. Lagergren. Efficient algorithms for lateral gene transfer problems. In ACM, editor, *Proceedings of the Fifth Annual International Conference on Computational Molecular Biology (RECOMB 2001)*, pages 149–156. ACM, 2001.
14. J. Hein, T. Jiang, L. Wang, and K. Zhang. On the complexity of comparing evolutionary trees. *Discrete Applied Mathematics*, 71:153–169, 1996.
15. Marijn J.H. Heule and Hans van Maaren. March dl: Adding adaptive heuristics and a new branching strategy. *Journal on Satisfiability, Boolean Modeling and Computation*, 2:47–59, mar 2006.
16. Daniel H. Huson and David Bryant. Application of phylogenetic networks in evolutionary studies. *Molecular Biology and Evolution*, 23(2):254–26, 2006.
17. I. Lynce and J.P. Marques Silva. Efficient haplotype inference with boolean satisfiability. In *Proceedings of oceedings of National Conference on Artificial Intelligence (AAAI)*, 2006.
18. B. Moret, L. Nakhleh, T. Warnow, C. R. Linder, A. Tholse, A. Padolina, J. Sun, and R. Timme. Phylogenetic networks: Modeling, reconstructibility and accuracy. *IEEE Transactions on Computational Biology and Bioinformatics*, 1(1):13–23, 2004.
19. Tamara Munzner, François Guimbrètiere, Serdar Tasiran, Li Zhang, and Yunhong Zhou. TreeJuxtaposer: Scalable tree comparison using Focus+Context with guaranteed visibility. *SIGGRAPH 2003 Proceedings, published as special issue of Transactions on Graphics*, pages 453–462, 2003.
20. Luay Nakhleh, Derek Ruths, and Li-San Wang. RIATA-HGT: A fast and accurate heuristic for reconstructing horizontal gene transfer. In L. Wang Editor, editor, *Proceedings of the Eleventh International Computing and Combinatorics Conference (COCOON 2005)*, pages 84–93. LNCS 3595, 2005.
21. G. J. Olsen, H. Matsuda, R. Hagstrom, and R. Overbeek. fastdnaml: A tool for construction of phylogenetic trees of dna sequences using maximum likelihood. *Comput. Appl. Biosci.*, 10:41–48, 1994.
22. MJ Sanderson. r8s; inferring absolute rates of evolution and divergence times in the absence of a molecular clock. *Bioinformatics*, 19:301–302, 2003.
23. HA Schmidt. *Phylogenetic trees from large datasets.* PhD thesis, Heinrich-Heine-Universitat, Dusseldorf, 2003.
24. Bart Selman, Henry A. Kautz, and Bram Cohen. Software available at `http://www.cs.rochester.edu/u/kautz/walksat/`.
25. Bart Selman, Henry A. Kautz, and Bram Cohen. Local search strategies for satisfiability testing. In M. Trick and D. S. Johnson, editors, *Proceedings of the Second DIMACS Challange on Cliques, Coloring, and Satisfiability*, Providence RI, 1993.
26. Charles Semple. Hybridization networks. New Mathematical Models for Evolution. Oxford University Press., 2007.
27. Dave A.D. Tompkins and Holger H. Hoos. UBCSAT: An implementation and experimentation environment for SLS algorithms for SAT and MAX-SAT. In *Proceedings of the 7th Int. Conf. on Theory and Applications of Satisfiability Testing, SAT'04*, volume 3542 of *Lecture Notes in Computer Science*, pages 306–320, Vancouver, Canada, 2005. Springer.
28. Yufeng Wu. A practical method for exact computation of subtree prune and regraft distance. *Bioinformatics*, 25(2):190–196, 2009.
29. Lin Xu, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. SATzilla:portfolio-based algorithm selection for SAT. *Journal of Artificial Intelligence Research 32 (2008) 565-606*, 32:565–606, 2008.

trees [23]	# of taxa	Hybrid Number[7]	SAT Exact	RIATA -HGT[20]	SAT Descent w [24]	a [8]	n[8]	m [11]	z [?]
ndhf	40	14	≥ 9	15	14	16	14	≤ 15	16
phyB		11h	2d	11s	4m	24s	48s	6h	44s
ndhf	36	13	≥ 9	16	13	17	14	≤ 14	18
rbcl		11.8h	2d	11s	4m	28s	51s	6h	48s
ndhf	34	12	≥ 9	15	12	15	12	≤ 12	15
rpoC2		26.3 h	2d	7s	3m	14s	35s	6h	34s
ndhf	19	9	9	9	9	10	9	≤ 9	10
waxy		5m	46h	3s	19s	4s	7s	6h	2m
ndhf	46	≥ 15	≥ 9	24	22	22	21	≤ 20	22
xits		2d	2d	12s	3m	50s	1.2m	6h	1m
phyB	21	4	4	4	4	4	5	4	4
rbcl		1s	6s	4s	7s	4s	4s	3s	5s
phyB	21	7	7	7	7	7	7	7	10
rpoC2		3m	1.5m	3s	33s	11s	13s	77s	11s
phyB	14	3	3	3	3	3	4	3	3
waxy		1s	3s	2s	5s	3s	2s	2s	2s
phyB	30	8	8	9	8	9	9	8	10
xits		19s	1.5h	6s	1m	10s	11s	1.7h	10s
rbcl	26	13	9	16	14	15	15	≤ 15	14
rpoC2		29.5h	2d	5s	1m	9s	10s	6h	36s
rbcl	12	7	7	7	7	7	7	7	8
waxy		4m	42s	1s	10s	3s	3s	40s	7s
rbcl	29	≥ 9	≥ 9	15	14	19	14	≤ 15	19
xits		2d	2d	6s	271s	20s	1m	6h	40s
rpoC2	10	1	1	1	1	1	1	1	1
waxy		1 s	1s	1s	3s	1s	1s	1s	1s
rpoC2 xits	31	≥ 10	≥ 9	17	15	18	15	≤ 15	18
		2d	2d	7s	4m	18s	50s	6h	1h
waxy	15	8	8	10	9	10	9	8	9
xits		10m	1s	2s	13s	6s	11s	1m	14s

Fig. 4. The Grass (Poaceae) Data Set: We compare the exact solver, HybridNumber [7], the fast heuristic, RIATA-HGT [20], and our program using the SAT encodings. The data for HybridNumber in the third column is from [7]. First: HybridNumber finds the exact solution, but due to the NP-hardness of the problem, often does not find a solution. Second: The performance of the SAT Ascent solver which works upward from the smallest distance until the true distance is found. Its performance echos HybridNumber. Third: RIATA-HGT gives very quickly a reasonable, but not tight, upper bound. Right: Our software gives excellent results in reasonable time. It employs five different solvers: the incomplete solvers: Walksat [24, 25] and two high scoring solvers from SAT 2007: adaptg2wsat and novelty+p [8] implemented in [27], as well as the complete solvers minisat [11] and SATzilla [29]. Solutions listed as upper or lower bounds did not halt before the time limit and estimates based on the log files are listed.

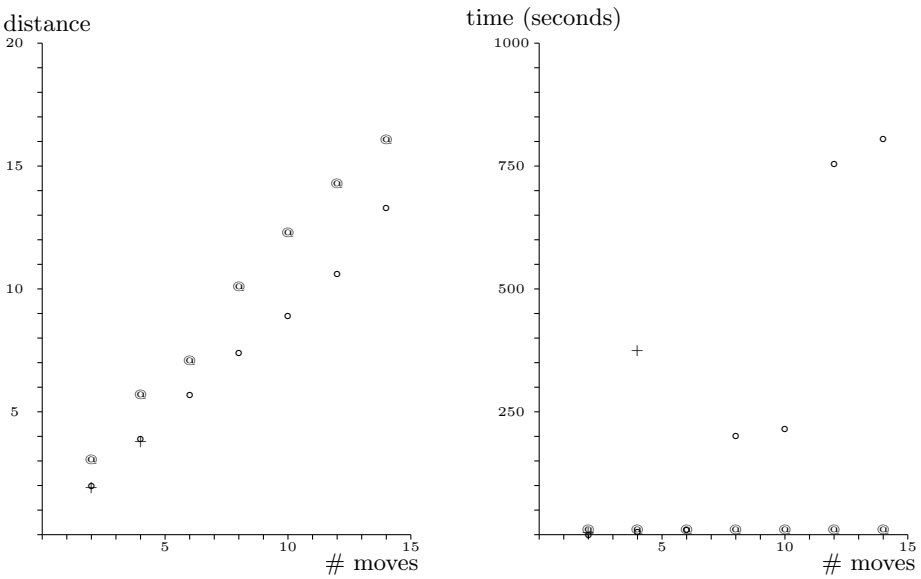

Fig. 5. Simulated Data Set: 50-taxa trees were generated under the Yule-Harding distribution to be the "species tree" and then for each distance and each species tree, 10 "gene trees" of that distance were generated. In both graphs, @ is RIATA-HGT [20], ∘ is the SAT Descent using Walksat [25], and + is the exact algorithm HybridNumber [7]. Due to the similarity in results to HybridNumber, the results for SAT Ascent solution are omitted. All runs had a 24 hour time limit. This did not affect RIATA-HGT and SAT Descent, but limited the runs that completed for HybridNumber to values 2 and 4. The left graph shows the hybridization number returned by the programs; the right graph shows the time, in seconds, to accomplish the task.