Improving Tree Search in Phylogenetic Reconstruction from Genome Rearrangement Data

Fei Ye[1], Yan Guo[2], Andrew Lawson[1], and Jijun Tang[2,⋆]

[1] Department of Epidemiology and Biostatistics
University of South Carolina
Columbia, SC 29208, USA
[2] Department of Computer Science & Engineering
University of South Carolina
Columbia, SC 29208, USA
jtang@cse.sc.edu

Abstract. A major task in evolutionary biology is to determine the ancestral relationships among the known species, a process generally referred as phylogenetic reconstruction. In the past decade, a new type of data based on genome rearrangements has attracted increasing attention from both biologists and computer scientists. Methods for reconstructing phylogeny based on genome rearrangement data include distance-based methods, direct optimization methods (GRAPPA and MGR), and Markov Chain Monte Carlo (MCMC) methods (Badger). Extensive testing on simulated and biological datasets showed that the latter three methods are currently the best methods for genome rearrangement phylogeny. However, all these tools are dealing with extremely large searching spaces; the total number of possible trees grows exponentially when the number of genomes increases and makes it computationally very expensive. Various heuristics are used to explore the tree space but with no guarantee of optimum being found. In this paper, we present a new method to efficiently search the large tree space. This method is motivated by the concept of particle filtration (also known as Sequential Monte Carlo), which was originally proposed to boost the efficiency of MCMC methods on massive data. We tested and compared this new method on simulated datasets in different scenarios. The results show that the new method achieves a significant improvement in efficiency, while still retains very high topological accuracy.

1 Introduction

The goal of phylogenetic analysis is to determine the evolutionary relationships among organisms and their genomes. A *phylogeny* for a set of N genomes (species) is a preferably N leaves binary tree, with each leaf labeled by a distinct element of the input set. Fig 1 shows two proposed phylogenies [18], the left one is for 12 species of the Campanulaceae (bluebell flowers) family (with Tobacco as an outgroup), represented in the form of *cladogram*; and the right one is for herpesviruses that are known to affect humans, represented in the form of an *unrooted* tree.

⋆ Corresponding author.

C. Demetrescu (Ed.): WEA 2007, LNCS 4525, pp. 352–364, 2007.

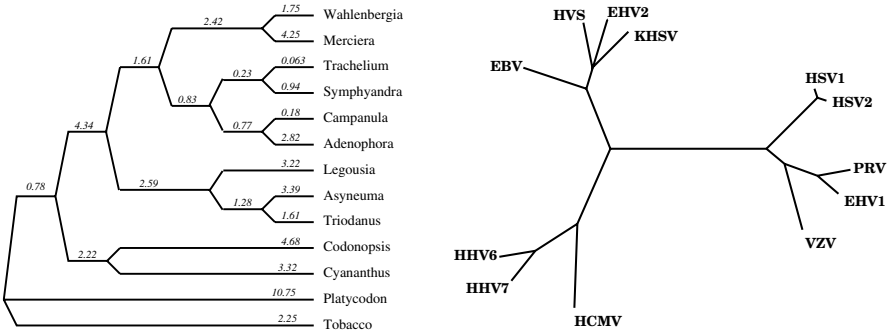


Fig. 1. Phylogenies for Campanulaceae (left) and herpesviruses (right)

To date, DNA (or protein) sequence data is the primary source of information for phylogenetic analysis. In the last decade, genome rearrangement (also known as gene-order) data are emerging into the field and many researchers showed great interests about it [8,20,21,24].

Biologists can infer the ordering and strandedness of genes on a chromosome and thus represent each chromosome by an ordering of signed genes (where the sign indicates the strand). These gene orders can be rearranged by evolutionary events such as inversions and transpositions. The relative rarity of genomic rearrangements, together with the increasing availability of complete genome sequences, make them very attractive as new sources for genome comparison. Developing appropriate tools for analyzing such data is therefore an important area of research. During the past several years, computer scientists have been able to make substantial progress in genome rearrangement research. With the solution for inversion distance [12] and inversion median [4], we were able to estimate phylogenies and ancestral genomes based on inversions (the dominant events in organellar genomes).

There are several widely used methods for genome rearrangement analysis, including neighbor-joining [25], `GRAPPA` [16], `MGR` [2] and `Badger` [14]. Using the later three generally will achieve better accuracy than using distanced based methods such as neighbor-joining. However, since all these three methods need to find the best tree from a large number of possible trees (tree space), they all face the similar problem of scalability: none of these three methods can be used for more than 15 genomes (species) because the total number of possible trees increases exponentially with the number of genomes [1]: there are 13 billion trees for 10 genomes, more than $7,000$ billion trees for 15 genomes, and more than 2^{67} trees for merely 20 genomes. Using today's most powerful machines, `GRAPPA` (the fastest among the three) can finish searching the tree space for 15 genomes within several minutes, but such search will take more than $6,000$ centuries for 20 genomes. Even if the CPU power continues to increase under Moore's law (doubled for every 18 months), such computation will still take at least 20 years.

[1] The number of all possible binary trees is $(2N-5)!! = (2N-5) \times (2N-7) \cdots \times 3$ for N genomes [10].

One way to remedy this problem is to use disk-covering methods (DCM), introduced by Warnow and her group. Tang et al. combined the DCM1 [13] approach with GRAPPA, and produced DCM-GRAPPA [28], which can analyze datasets of up to 1,000 taxa with high accuracy. DCM-GRAPPA works in three steps: it first decomposes the dataset into smaller overlapping subproblems, then runs GRAPPA as the base method on these subproblems to obtain subtrees, and finally combines the subtrees to build a tree for the original dataset. Since GRAPPA has a limit on the number of genomes it can handle, DCM-GRAPPA has to recursively call DCM until each subproblem size falls below that limit (currently set to 13 genomes). Because the threshold is small, large problems require many levels of recursive decomposition, which is not only time-consuming but also risks propagating and amplifying error in the assembly of the subtrees. For $1,000$ genomes, it generally requires $6-7$ level of recursively calls if the maximum subproblem size is limited to 13, but only $2-3$ levels when the limit is raised to 20. As a result, improving the efficiency of GRAPPA is still desirable.

Another standard approach to scaling is to compute the smallest possible nontrivial trees: quartet trees, defined on just four taxa. Quartet methods rely on finding the optimal 4-leaf tree for each quartet and using this information to build the overall tree. Liu et al. developed an optimization algorithm [15] for the NP-hard problem of computing optimal trees for each quartet, as well as a variation of the dyadic method [9] to choose suitable short quartets. This method is able to handle $20-30$ genomes. However, it begins to loose accuracy when the input dataset has more than 18 genomes, hence is not suitable to be used as a base method for DCM-GRAPPA.

In the past five years, facing similar problem in Bayesian analysis of massive datasets, statisticians developed particle filtration techniques (also known as sequential Monte Carlo methods) [7,23] to improve the efficiency in MCMC samplings. In this paper, we present a new tree search method motivated by particle filtrations. After some background review and definitions, we describe in Section 5 our new tree search method in detail; in Section 6, we evaluate the new method on simulated datasets. The results suggest that our method is much faster than GRAPPA, with very limited loss of accuracy. This new method can be integrated with DCM methods to further improve the analysis of large scale datasets.

2 Backgrounds

2.1 Genome Rearrangements

We assume a reference set of n genes $\{g_1, g_2, \cdots, g_n\}$, thus a genome can be represented as a signed ordering of these genes, and each gene is given an orientation that is either positive, written g_i, or negative, written $-g_i$. Genomes can evolve through events such as inversions, transpositions and transversion, as well as many other events.

Let G be the genome with signed ordering of $g_1, g_2, \cdots, g_n$. An *inversion* between indices i and j $(i \leq j)$, transforms G to a new genome with linear ordering

$$g_1, g_2, \cdots, g_{i-1}, -g_j, -g_{j-1}, \cdots, -g_i, g_{j+1}, \cdots, g_n$$

A *transposition* on genome G acts on three indices i, j, k, with $i \leq j$ and $k \notin [i, j]$, picking up the interval $g_i, g_{i+1}, \cdots, g_j$ and inserting it immediately after g_k. Thus genome G is replaced by (assume $k > j$):

$$g_1, \cdots, g_{i-1}, g_{j+1}, \cdots, g_k, g_i, g_{i+1}, \cdots, g_j, g_{k+1}, \cdots, g_n$$

An *transversion* is a transposition followed by an inversion of the transposed subsequence; it is also called an *inverted transposition*.

The *edit distance* between two genomes is the minimum number of evolutionary events required to transform one genome into the other. When only inversions are allowed, the edit distance is the *inversion distance*. The *score* of a tree is the sum of the costs of its edges, where the cost of an edge can be defined as the distance between the two genomes that label the endpoints of the edge. Finding the (minimum) score of a tree generally requires the determination of gene orders on each internal node, which is itself very difficult. For a three-leave tree, the tree score can be obtained by finding a genome that minimizes the sum of pairwise distances between itself and each of the three leaf genomes. Such procedure is generally referred as *median problem of three*, or *median problem* for short, which is NP-hard [3,22] when inversion distances are used.

2.2 Phylogenetic Reconstruction from Genome Rearrangements

Methods for phylogeny analysis based on genome rearrangement data include distance-based methods (for example, neighbor-joining [25]), maximum parsimony methods based on encodings [31], and direct optimization methods. The latter, pioneered by Sankoff et al. [26] in their package `BPAnalysis` and improved by `GRAPPA` [16] and `MGR` [2], are the most accurate. A Markov Chain Monte Carlo method (`Badger`) [14] developed by Larget et al. is also widely used with comparable accuracy. The reconstruction goal of both `GRAPPA` and `MGR` is to find the tree(s) with the lowest score. One should note that such minimum score tree may not be the true phylogeny–it is just an estimation of the history. In deed, all phylogenetic methods so far cannot guarantee that the true phylogeny be found, thus the accuracy of a method should be carefully assessed using both simulated and biological data. Besides returning a phylogeny, all these methods can also give an estimate of ancestral genomes, which will have great utility for biologists interested in the process of genome rearrangement.

2.3 `GRAPPA`

`GRAPPA` is an exhaustive search method, moving systematically through the space of all $(2N-5)(2N-7)\cdots 3$ possible trees on N genomes. For each tree, the program tests a lower bound to determine whether the tree is worth scoring; if so, then the program will determine the tree score by iteratively solving the median problems at internal nodes until convergence, as outlined in Fig. 2.

The speed of `GRAPPA` is regulated by two factors: the efficiency of median computation and the pruning rates, i.e. how many trees can be discarded before being scored. Moret et al. developed several lower bounds based on triangular inequalities [17]. All these bounds are very tight and easy to compute (much easier than the scoring procedure). As a result, more than 99.99% trees can be discarded without being scored. Further speed-up is achieved by using a branch-and-bound approach that can discard most trees without even generating them. Overall, `GRAPPA` achieves a billion-fold speed up over its predecessor of `BPAnalysis` and can finish the original 13-genome *Campanulaceae* dataset [6] within 20 minutes on a single workstation. Even with such speedup,

```
Initially label all internal nodes with gene orders
Repeat
    For each internal node v, with neighbors A, B and C, do
    Solve median problem on A, B, C to yield m
    If relabeling v with m improves the tree score, then do it
Until no change occurs
```

Fig. 2. The GRAPPA scoring procedure

GRAPPA is not suitable for datasets with more than 15 genomes. Both MGR and Badger face similar limitations [2,14].

3 Searching the Large Tree Space

GRAPPA is not the only package that faces the dilemma of dealing with the fast growing tree space. Indeed, except for distance based methods (such as neighbor-joining [25]), all popular phylogeny packages–including those developed for DNA sequence data–have to search this space for phylogenies. For dataset with many species, people have to rely on various heuristics to explore the tree space with no guarantee that the optimal trees will be found.

Three natural approaches for searching the tree space are nearest-neighbor interchanges (NNI), subtree pruning and regrafting (SPR), and Tree-Bisection- Reconnection (TBR) [27]. In NNI, one of the internal edges is chosen at random and the four subtrees (by removing the edge and its two nodes) are reconnected randomly. In SPR, a random edge is selected and two subtrees are created, then one of the two subtrees is removed at random and reinserted along a random edge in the other subtree. In TBR, similar to SPR, one edge is removed and the tree is divided into two subtrees, then they are joined by an edge connecting two midpoints of edges of the two subtrees. Figure 3 shows two examples of these heuristics.

These methods form the core of many phylogeny analysis packages, including Bayesian methods such as Badger. Since there is no guarantee of convergence by using these methods, various schemes (such as choosing multiple starting points) are developed with the hope that the search will converge. However, Mossel et al. [19] recently proved that many of the popular Markov Chain Monte Carlo methods using the above tree searching techniques take exponentially long time to converge.

On the other hand, GRAPPA uses an exhaustive approach to examine all possible trees. In order to perform this exhaustive search, a fast tree generating procedure using depth-first approach is implemented, which in turn provides a new way of defining the tree space.

Let's examine the depth-first procedure in detail. Assuming the tree generating procedure works on levels, and each level k has all the partial trees containing the first k genomes. We pick the first three genomes (genome 1, 2 and 3) and create the (only) binary tree in level 3. named $T_{3\cdot1}$ (tree number one in level 3). Then, we will attach the next genome (genome 4) to the first edge of this tree, and generate tree $T_{4\cdot1}$. We repeatedly add the next genomes until it reaches the last level (genome N is included), resulted in the first complete N-genome tree ($T_{N\cdot1}$) being generated. The next tree ($T_{N\cdot2}$)

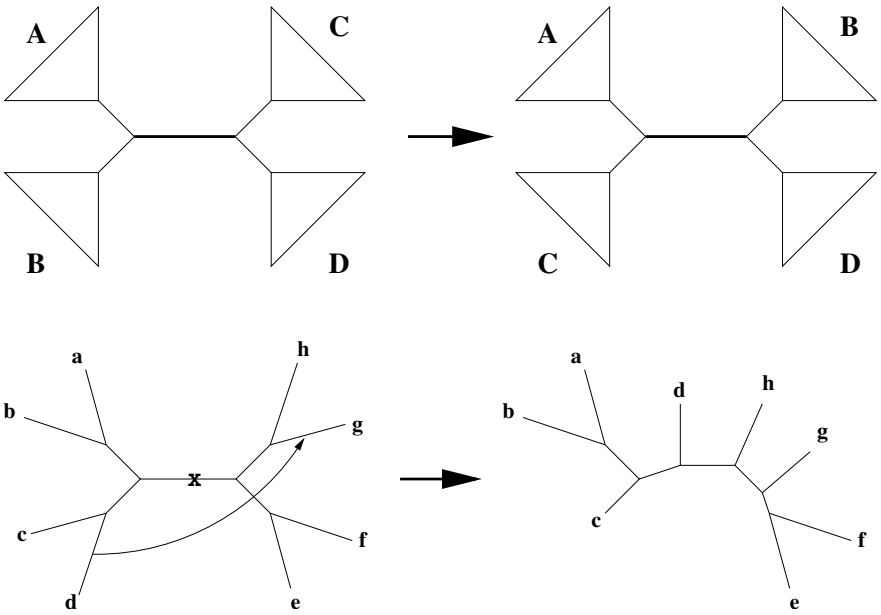


Fig. 3. Examples of NNI (top) and TBR (bottom)

is generated by attaching genome N to the second edge of $T_{N-1\cdot1}$, and tree ($T_{N\cdot2N-5}$) is generated by attaching genome N to the last edge of $T_{N-1\cdot1}$. The depth-first search has reached the depth, so it will move a level up and attach genome $N-1$ to the second edge of tree $T_{N-2\cdot2}$, and generates the next block of $2N-5$ trees for N genomes. We then move up and down the levels as in the standard depth-first search, until all trees are generated.

We can define a tree space by assigning every tree a unique tree number with respect to the ordering of its appearance in the depth-first generating procedure. This space has a unique property that trees with close tree numbers are *generally* similar in topology. For example, the first $2N-5$ trees has difference of only one edge. The branch-and-bound method developed in GRAPPA [29] was based on this tree generating procedure and is the fastest options for using GRAPPA.

GRAPPA also provides a stepping function to quickly explore the tree space: if user sets the stepping interval of S, then it will only generate trees with number $1+S, 1+2S, \ldots$ as defined in the above space. Of course, the problem is obvious: it has a high probability of missing the best tree if the stepping is set too high, and is too slow if such value is small. The exhaustive search approach can be viewed as a special case when the stepping interval is set to one.

4 Particle Filtration Technique

In the context of sequential estimation, full MCMC algorithms must re-sample when new data arrives, hence the parameter and data spaces are enlarged to fit the new

situation, which is computationally very expensive when the data space is large. Sequential Monte Carlo methods (or particle filtration) techniques were developed to deal with such data-parameter enlargement problems.

Particle filtration techniques can be applied to the situation where subsets of the target data space are progressively included in the analysis [5,23]. They are also used to deal with massive non-sequential datasets where the data space can be divided into subsets and sequentially included into the analysis. The method starts by initializing a set of parameter estimates and uses importance sampling to filter out the particles, i.e., the parameters that have the least posterior probability after incorporating the additional data, by reweighing the parameter estimates derived from the current posterior distribution, using importance sampling. With this approach, the time saving can be dramatic. In addition, particle filtration only requires the adoption of a single step Metropolis update within re-sampling steps that are guaranteed to come from the posterior distribution and hence the usual convergence requirement does not apply to it.

Although particle filtration is developed for MCMC sampling on massive data and therefore requires a definition of posterior distribution, the idea of particle filtration, especially the concept of importance sampling and re-sampling scheme, can be adopted by any phylogeny packages to explore the large tree space.

5 The New Algorithm

5.1 Overview

Our new algorithm integrates particle filtration with the various tree search techniques discussed in Section 3. Let D denote the whole tree topology space for N genomes, and let D_1 denote a subset of trees from D, which is served as the initial trees for updating.

The algorithm of our method can be described with the following steps:

1. (Loading) Load a subset of D_1 trees into the memory. These trees can be randomly picked from the whole phylogenetic tree space.
2. (Resampling) For every tree D_{1i} in D_1, search its neighborhood in the tree space. If a tree with higher weight exists, replace the current D_{1i} tree with this one.
3. (Updating) After getting a new set of D_1, we then call the MCMC updating methods (such as NNI, TBR and SPR) to update the D_1 trees. The update will stop after certain number of updates are performed.
4. (Reporting) Choose the best tree found in the previous steps, and return the result.

5.2 Implementation Details

The above procedures are standard in particle filtration methods. However, there are a few modifications in each step since the new algorithm does not handle posterior distributions.

In step one, all particle filtration methods require loading as much as possible data into the initial subset (D_1). Since the tree space is too large, the portion of trees can

be loaded into memory is always a small percentage. In our experiments, we found that the final results are not very sensitive to the size of D_1 (denote $|D_1|$), and only a few hundred trees are needed in forming the initial D_1. In fact, selecting which trees to form set D_1 is more important. Our strategy is to divide the whole tree space into $|D_1|$ subspaces (using tree numbers) and therefore each subspace contains $\frac{(2N-5)!!}{|D_1|}$ trees. Within each subspace, we then randomly select one tree into D_1. The purpose of doing so is to go deep enough into the tree space so that the trees are not chosen only from a small portion of the whole dataset. Since the tree number can be very large, we use the GNU Multiple Precision Arithmetic Library (GMP) to handle large integer numbers required in this step.

In the re-sampling procedure (step 2), trees in D_1 are updated by comparing the importance weight of each tree, i.e., a tree with higher weight will substitute the old tree in D_1. Such weight is originally defined to be proportional to the posterior density. We define a weight that is proportional to it inversion tree score, and a tree with lower tree score is given higher weight. Such score can be computed by the `GRAPPA` scoring procedure, or it can be estimated using the linear programming method [30] if the median is too difficult to compute. The searching for new trees should be done locally, which is utilized by using the stepping function in `GRAPPA`. Specifically, the local search is controlled by stepping interval S and number of steps U_1, all can be kept in the range of several hundred.

In standard particle filtration methods, the updating procedure (step 3) is performed according to the importance weight of each tree in D_1, i.e., trees with higher weight will have higher chances to be updated. In our current implementation, we choose to keep it simple and all trees in D_1 are updated exactly the same number of times, denoted as C (number of *cycles*). As we can see in the experimental results, we only need a small number of cycles to generate very satisfying results. MCMC methods are requested for this update, thus all the standard MCMC update methods like NNI, TBR and SPR can be used. However, since SPR and NNI are localized updates compared to TBR, and a local search has already been done in step 2, we only use TBR in the new method.

6 Experimental Results

6.1 Setup of Simulations

We set out to examine the performance (in terms of speed and accuracy) of the new method. We concentrated our experiments on simulated datasets because topological accuracy can be easily assessed when the true trees are known, using measurements such as *false negative* and *false positive*. Let T be a tree leaf-labelled by a set of genomes, deleting some edge e from T produces a bipartition which splits the genomes into two sets. Let T be the true tree and let T' be the inferred tree; then the false negatives (FN) are those bipartitions that appear in the true tree T but do not appear in the inferred tree T'. Similarly, false positives (FP) are those bipartitions that appear in T but not in T' [17]. The goal of all phylogeny methods is to obtain both lower false

negative and false positive. In our simulations, since all true trees and reconstructed trees are binary trees, thus $FP = FN$, and we only report FN here. The rate of FN is defined as the number of false edges divided by the number of internal edges of the true tree ($N - 2$ for N genomes). A rate of smaller than 5% is generally considered acceptable [27].

We generated datasets of 14, 16, 18 and 20 genomes with 100 genes for each genome (approximately the size of small organelle genomes). We used various number of evolutionary rates: letting r denote the expected number of evolutionary events along an edge of the true tree, we used values of r in the range of 2 to 12. The actual number of events along each edge is sampled from a uniform distribution on the set $\{\frac{r}{2}, \ldots, \frac{3r}{2}\}$. While all computations were based on inversion distances and inversion medians, we generated the data with a deliberate model mismatch to test the robustness of the methods, using a mix of 80% inversions and 20% transpositions. For each combination of parameter settings, we ran 10 datasets and averaged the results. All the experiments are conducted on a Linux cluster with 152 Intel Xeon CPUs, but each CPU works independently on a test task.

Since GRAPPA can not handle more than 15 genomes, we only tested GRAPPA on datasets with 14 genomes and compared its accuracy with the new method. For GRAPPA, each dataset was tested with the branch-and-bound method, which is the fastest version available. For the new method, there are several parameters that have impact on the results: $|D_1|$ (number of trees in $D1$), U_1 (number of steps in re-sampling), S (stepping intervals) and C (number of TBR updates). In our experiments, we found that C has the biggest impact on both speed and accuracy, thus we only tested different C values (100, 300 and 500) for each dataset while the other variables were fixed. Since the tree space of 20 genomes is very large, we also tested those datasets with $C = 1000$. The choice of other parameters is listed in table 1:

Table 1. Parameters in the experiments

number of genomes	$\|D_1\|$	U_1	S
14	200	200	400
16	400	200	4000
18	600	400	40000
20	1000	1000	400000

6.2 Topological Accuracy

Fig 4 shows the accuracy in term of average FN rate, each graph contains results from different number of TBR updates C. Not surprisingly, the accuracy increases with the number of TBR updates, and using $500 - 1000$ cycles of such updates produced results with very high accuracy (with less than 5% errors). Fig 4 (a) also shows the result obtained from exhaustive search using the original GRAPPA, which is almost identical to those obtained from the new method.

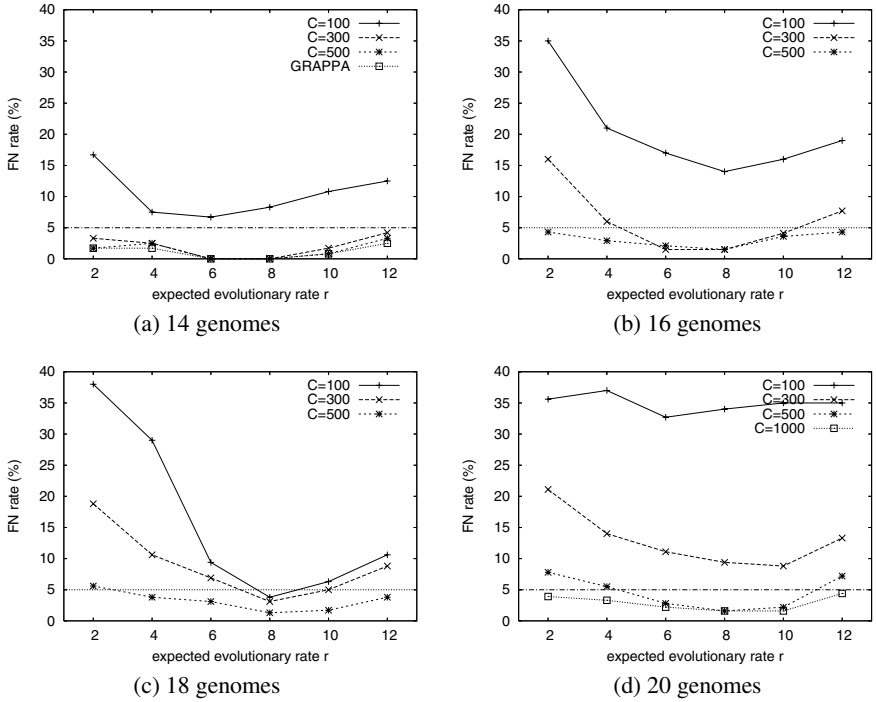


Fig. 4. Average FN rates as a function of r and number of cycles C

Fig 5 shows the accuracy results of using $500 - 1000$ TBR updates. Compared to the results showed in [15], this figure suggests that this new method is more accurate than the quartet methods, thus can be used as a better base method for disk-covering methods.

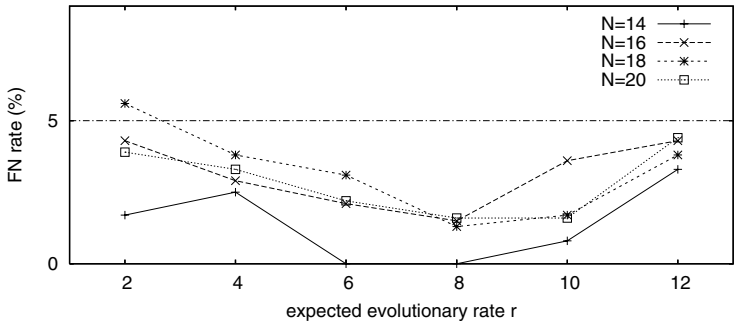
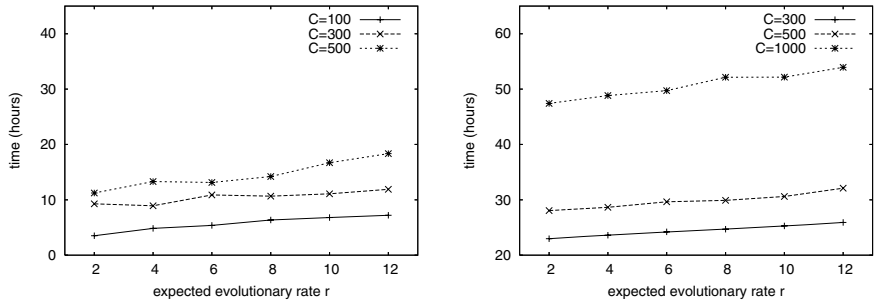


Fig. 5. Best average FN rates as a function of r and N

6.3 Speed

Our method is also very fast–it took only $1 \sim 2$ days to compute 20 genomes, instead of $6,000$ centuries as we projected by using GRAPPA. In other words, we achieved ten billion-fold speed-up while still retain very high accuracy. Fig 6 shows the average time spent for 18 and 20 genomes. Roughly speaking, the number of trees examined needs to be doubled when number of genomes N increases by one, thus the new method scales better than using exhaustive approach and has the potential to handle several more genomes.



(c) 18 genomes (d) 20 genomes

Fig. 6. Average running time as a function of r and number of cycles C

7 Conclusions

This paper presents a new method to search the large phylogeny space based on the concept of particle filtration. Although this method is now presented along with GRAPPA, such technique can be easily applied to other phylogeny packages. Since this method is originally designed to extend the range of base methods for DCMs, we will further assess the impact of using it over the direct use of GRAPPA as a base method.

This paper verifies the particle filtration to be a powerful tool in searching large space. There are many research subjects in Bioinformatics that face similar problem. For example, the median computation in genome rearrangement analysis needs to explore the space consists of all possible permutations–for signed genomes with n genes, there are $2^n n!$ possible permutations. The fact that inversion median problem is in APX reflects the difficulty of searching such large space. Applying particle filtration may provide a more efficient method for the median problems.

Acknowledgments

The authors were supported by US National Institutes of Health (NIH grant number R01 GM078991-01) and by the University of South Carolina.

References

1. Blanchette, M., Sankoff, D.: The median problem for breakpoints in comparative genomics. In: Jiang, T., Lee, D.T. (eds.) COCOON 1997. LNCS, vol. 1276, pp. 251–263. Springer, Heidelberg (1997)
2. Bourque, G., Pevzner, P.: Genome-scale evolution: Reconstructing gene orders in the ancestral species. Genome Research 12, 26–36 (2002)
3. Caprara, A.: Formulations and hardness of multiple sorting by reversals. In: RECOMB'99. Proc. 3rd Int'l Conf. on Comput. Mol. Biol, pp. 84–93. ACM Press, New York, NY, USA (1999)
4. Caprara, A.: On the practical solution of the reversal median problem. In: Gascuel, O., Moret, B.M.E. (eds.) WABI 2001. LNCS, vol. 2149, pp. 238–251. Springer, Heidelberg (2001)
5. Chopin, N.: A sequential particle filter method for static models. Biometrika 89, 539–552 (2002)
6. Cosner, M.E., Raubeson, L.A., Jansen, R.K.: Chloroplast DNA rearrangements in Campanulaceae: Phylogenetic utility of highly rearranged genomes. BMC Evol. Biol. vol. 4(27) (2004)
7. Doucet, A., de Freitas, N., Gordon, N.: Sequential Monte Carlo Methods in Practice. Springer, Heidelberg (2001)
8. Downie, S., Palmer, J.: Use of chloroplast DNA rearrangements in reconstructing plant phylogeny. In: Soltis, P., Soltis, D., Doyle, J. (eds.) Plant Molecular Systematics, pp. 14–35 (1992)
9. Erdős, P.L., Steel, M.A., Székely, L.A., Warnow, T.: Local quartet splits of a binary tree infer all quartet splits via one dyadic inference rule. Computers and Artif. Intell. 16(2), 217–227 (1997)
10. Felsenstein, J.: The number of evolutionary trees. Systematic Zoology 27, 27–33 (1978)
11. Gilks, W., Berzuini, C.: Following a moving target-Monte Carlo inference for dynamic Bayesian models. J. of the Royal Statistical Society (2001)
12. Hannenhalli, S., Pevzner, P.A.: Transforming cabbage into turnip polynomial algorithm for sorting signed permutations by reversals. In: Proc. 27th Ann. Symp. Theory of Computing, pp. 178–189. ACM Press, New York, NY, USA (1995)
13. Huson, D., Nettles, S., Warnow, T.: Disk-covering, a fast converging method for phylogenetic tree reconstruction. J. Comput. Biol. 6(3), 369–386 (1999)
14. Larget, B., Simon, D.L., Kadane, J.B., Sweet, D.: A Bayesian analysis of metazoan mitochondrial genome arrangements. Mol. Biol. and Evol. 22(3), 486–495 (2005)
15. Liu, T., Tang, J., Moret, B.M.E.: Quartet methods for phylogeny reconstruction from gene orders. In: Gorodetsky, V., Liu, J., Skormin, V.A. (eds.) AIS-ADM 2005. LNCS (LNAI), vol. 3505, pp. 63–73. Springer, Heidelberg (2005)
16. Moret, B.M.E., Wyman, S., Bader, D.A., Warnow, T., Yan, M.: A new implementation and detailed study of breakpoint analysis. In: Proc. 6th Pacific Symp. on Biocomputing (PSB 01), pp. 583–594. World Scientific Pub, Singapore (2001)
17. Moret, B.M.E., Tang, J., Wang, L.-S., Warnow, T.: Steps toward accurate reconstructions of phylogenies from gene-order data. J. Comput. Syst. Sci. 65(3), 508–525 (2002)
18. Moret, B.M.E., Tang, J., Warnow, T.: Reconstructing phylogenies from gene-content and gene-order data. In: Gascuel, O. (ed.) Mathematics of Evolution and Phylogeny, pp. 321–352. Oxford Univ. Press, Oxford (2005)
19. Mossel, E., Vigoda, E.: Limitations of Markov Chain Monte Carlo Algorithms for Bayesian Inference of Phylogeny. Quantitative Biology, vol. 4(12) (2006)
20. Olmstead, R., Palmer, J.: Chloroplast DNA systematics: a review of methods and data analysis. Amer. J. Bot. 81, 1205–1224 (1994)

21. Palmer, J.: Chloroplast and mitochondria genome evolution in land plants. In: Herrmann, R. (ed.), Cell Organelles, pp. 99–133 (1992)
22. Pe'er, I., Shamir, R.: The median problems for breakpoints are NP-complete. Elec. Colloq. on Comput. Complexity, vol. 71 (1998)
23. Ridgeway, G., Madigan, D.: A sequential Monte Carlo method for Bayesian analysis of massive datasets. Data Mining and Knowledge Discovery 7, 301–319 (2002)
24. Raubeson, L., Jansen, R.: Chloroplast DNA evidence on the ancient evolutionary split in vascular land plants. Science 255, 1697–1699 (1992)
25. Saitou, N., Nei, M.: The neighbor-joining method: A new method for reconstructing phylogenetic trees. Mol. Biol. Evol. 4, 406–425 (1987)
26. Sankoff, D., Blanchette, M.: Multiple genome rearrangement and breakpoint phylogeny. J. Comput. Biol. 5, 555–570 (1998)
27. Swofford, D.L., Olson, G., Waddell, P., Hillis, D.M.: Phylogenetic inference. In: Hillis, D.M., Moritz, C., Mable, B. (eds.) Molecular Systematics, 2nd edn. chapter 11 (1996)
28. Tang, J., Moret, B.M.E.: Scaling up accurate phylogenetic reconstruction from gene-order data. In: Proc. 11th Conf. on Intelligent Systems for Mol. Biol. ISMB'03, in Bioinformatics, vol. 19, pp. i305–i312 (2003)
29. Tang, J.: Large-scale Phylogenetic Reconstruction from Arbitrary Gene-order Data. Ph.D. Dissertation, available online at `http://www.cse.sc.edu/~jtang/dissertation.ps` (2004)
30. Tang, J., Moret, B.M.E.: Linear programming for phylogenetic reconstruction based on gene rearrangements. In: Apostolico, A., Crochemore, M., Park, K. (eds.) CPM 2005. LNCS, vol. 3537, pp. 406–416. Springer, Heidelberg (2005)
31. Wang, L.-S., Jansen, R., Moret, B.M.E., Raubeson, L., Warnow, T.: Fast phylogenetic methods for genome rearrangement evolution: An empirical study. In: Proc. 7th Pacific Symp. on Biocomputing (PSB 02), pp. 524–535. World Scientific Pub, Singapore (2002)