Reversing Gene Erosion—Reconstructing Ancestral Bacterial Genomes from Gene-Content and Order Data

Joel V. Earnest-DeYoung[1], Emmanuelle Lerat[2], and Bernard M.E. Moret[1]

[1] Dept. of Computer Science, Univ. of New Mexico, Albuquerque, NM 87131, USA,
`joeled,moret@cs.unm.edu`
[2] Dept. of Ecology and Evolutionary Biology, Univ. of Arizona, Tucson, AZ 85721, USA,
`lerat@email.arizona.edu`

Abstract. In the last few years, it has become routine to use gene-order data to reconstruct phylogenies, both in terms of edge distances (parsimonious sequences of operations that transform one end point of the edge into the other) and in terms of genomes at internal nodes, on small, duplication-free genomes. Current gene-order methods break down, though, when the genomes contain more than a few hundred genes, possess high copy numbers of duplicated genes, or create edge lengths in the tree of over one hundred operations. We have constructed a series of heuristics that allow us to overcome these obstacles and reconstruct edges distances and genomes at internal nodes for groups of larger, more complex genomes. We present results from the analysis of a group of thirteen modern γ-proteobacteria, as well as from simulated datasets.

1 Introduction

Although phylogeny, the evolutionary relationships between related species or taxa, is a fundamental building block in much of biology, it has been surprisingly difficult to automate the process of inferring these evolutionary relationships from modern data (usually molecular sequence data). These relationships include both the evolutionary distances within a group of species and the genetic form of their common ancestors.

In the last decade, a new form of molecular data has become available: gene-content and gene-order data; these new data have proved useful in shedding light on these relationships [1–4]. The order and the orientation in which genes lie on a chromosome changes very slowly, in evolutionary terms, and thus together provide a rich source of information for reconstructing phylogenies. Until recently, however, algorithms using such data required that all genomes have identical gene content with no duplications, restricting applications to very simple genomes (such as organelles) or forcing researchers to reduce their data by equalizing the gene content (deleting all genes not present in every genome and all "copies" of each gene, e.g., using the *exemplar* strategy [5]). The former was frustrating to biologists wanting to study more complex organisms, while the latter resulted in data loss and consequent loss of accuracy in reconstruction [6].

Our group recently developed a method to compute the distance between two nearly arbitrary genomes [7, **?**] and another to reconstruct phylogenies based on gene-content and gene-order in the presence of mildly unequal gene content [6]. In this paper, we bring together these methods in a framework that enables us to reconstruct the genomes

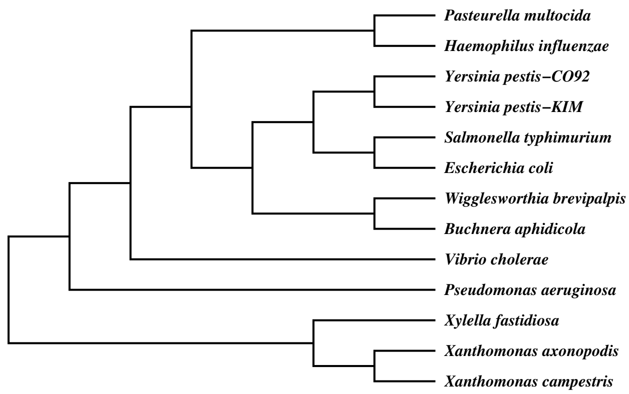

Fig. 1. The 13 gamma-proteobacteria and their reference phylogeny [8].

of the common ancestors of the 13 modern bacteria shown in Fig. 1 (from [8]). Gamma-proteobacteria are an ancient group of bacteria, at least 500 million years old [9]; the group includes endosymbiotic, commensal, and pathogenic species, with many species playing an important medical or economic role. The evolutionary history of the group is quite complex, including high levels of horizontal gene transfer [10–12] and, in the case of *B. aphidicola* and *W. brevipalpis*, massive levels of gene loss. These factors make a phylogenetic analysis of this group both interesting and challenging.

The rest of this paper is organized as follows. Section 2 presents the problem. Section 3 summarizes prior work on phylogenetic reconstruction from gene-content and gene-order data. Section 4 presents our framework for tackling the problem of ancestral genome reconstruction given a reference phylogeny; it is itself divided into three subsections, one on each of our three main tools: median-finding, content determination, and gene clustering. Section 5 discusses our approach to the testing of our framework: given that we have only one dataset and that ancestral genomes for that dataset are entirely unknown, our testing was of necessity based on simulations. Section 6 presents the results of this testing.

2 The Problem

We phrase the reconstruction problem in terms of a parsimony criterion:

> Given the gene orders of a group of genomes and given a rooted tree with these genomes at the leaves, find gene orders for the internal nodes of the tree that minimize the sum of all edge lengths in the tree.

The length of an edge is defined in terms of the number of evolutionary events (permissible operations) needed to transform the genome at one end of the edge into the genome at the other end. The permissible operations in our case are inversions, insertions (and duplications), and deletions; all operations are given the same cost in computing edge lengths. Restricting rearrangements to inversions follows from findings that the inversion phylogeny is robust even when other rearrangements, such as transpositions, are used in creating the data [13]. Our assignment of unit costs to all operations simply reflects insufficient biological knowledge about the relative frequency of these operations.

In our setting, one insertion may add an arbitrary number of genes to a single location and one deletion may remove a contiguous run of genes from a single location, a convention consistent with biological reality. Gene duplications are treated as specialized insertions that only insert repeats. Finally, on each edge a gene can either be inserted or deleted, but not both; the same holds for multiple copies of the same gene. Allowing deletion and insertion of the same genes on the same edge would lead to biologically ridiculous results such as deleting the entire source genome and then inserting the entire target genome in just two operations.

Finding internal labels that minimize edge distances over the tree has been addressed by our group—this is the main optimization performed by our software suite GRAPPA [14]. However, even the most recent version of GRAPPA [6] is limited to relatively small genomes (typically of organellar size, with fewer than 200 genes), with modestly unequal content and just a few duplications. In stark contrast, the bacterial genomes in our dataset contain 3,430 different genes and range in size from 540 to 2,987 genes, with seven containing over 2,300 genes; moreover, these genomes contain a large number of duplications, ranging from 3% to 30% of the genome. Thus, in our model, most pairwise genomic distances are very large: a simple pairwise comparison along the tree of Fig. 1 indicates that some edges of the tree must represent at least 300 events. Such lengths are at least an order of magnitude larger than GRAPPA can handle. The large genome size, vastly unequal gene content, large number of duplications, and large edge lengths all combine to make this dataset orders of magnitude more difficult to analyze than previously analyzed genome sets.

3 Prior Work

A thorough recent review of the current work in phylogenetic reconstruction based on gene content and gene order appears in [15]; we review only the relevant points here.

The GRAPPA software package [16] computes internal labels in two phases. First, it initializes internal labels of the tree by some method. Then it iteratively refines labels until convergence: each newly labeled (or relabeled) node is pushed on a queue and, while the queue is not empty, the node at the head of the queue is removed, a new label computed for it (by computing the median of its three neighbors), and, if the new label reduces the total distance to the three neighbors, the existing label is replaced with the improved label and the three neighbors are placed on the queue. Thus GRAPPA relies on the computation of the *median* of three genomes, that is, a fourth genome which minimizes the sum of the number of operations needed to convert it into each of the three given genomes. GRAPPA finds optimal inversion medians with an algorithm that runs in worst-case exponential time, but finishes quickly when the edge lengths are small (10 to 40 operations per edge) [6, 17]. GRAPPA treats groups of genes that occur in the same order and orientation in all genomes as a single genetic unit; this *condensation* step reduces computational costs and does not affect the final result [18].

Our group developed a method to find the distance between two genomes with arbitrary gene content [7, **?**]; this method relies on a *duplication-renaming* heuristic that matches multiple copies of genes between genomes and renames each pair and each unmatched copy to a new, unused gene number. Thus arbitrary genomes are converted into duplication-free genomes. We proved that, given two genomes with unequal gene content and no duplications, any optimal sorting sequence can be rearranged to contain first all insertions, then all inversions, and finally all deletions—a type of *normal form* for edit sequences [7]. (Deletions here are genes unique to the source genome, while insertions are genes found only in the target genome.) Using the genomes produced by the duplication-renaming method, an optimal inversion sequence can be calculated in time quadratic in the size of the consensus genomes [19, 20]. The number of deletions is calculated by counting the number of Hannenhalli-Pevzner cycles that contain deletions, as described in [21]. Finally, the number of insertions is estimated by calculating all possible positions in the source genome to which the inversion sequence could move insertions, then choosing the final position for each insertion that minimizes the number of groups of inserted genes.

In some genomes, especially bacterial ones, genes with similar function are often located together on one strand of a chromosome; these functional units are called *operons*. In bacteria, at least, while the order of genes in an operon may change, the gene content of the operon is much less likely to do so [22]. In gene-order data, an operon appears as a cluster of gene numbers with the same sign, with content, but not order, preserved across genomes. Heber and Stoye developed a linear-time *cluster-finding* algorithm to identify these operon-like clusters within equal-content genomes [23].

McLysaght *et al.* [4] reconstructed ancestral genomes for a group of poxviruses; she determined gene content by assuming that the phylogenetic tree contained a single point of origin for each gene family in the modern genomes. Each point of origin was assigned to that internal node which minimized the number of loss events necessary to achieve the gene content of the leaf genomes.

4 Designing an Algorithmic Framework

To address the problem of reconstructing ancestral genomes at the level of complexity of gamma-proteobacteria, we use condensation of gene clusters in order to reduce the size of the genomes, describe a procedure similar to that of McLysaght *et al.* to determine the gene content of every internal node, and present a new heuristic to compute the median of three very different genomes.

4.1 Medians

We use the queue-based tree-labeling heuristic described in Section 3. Since leaves contain the only labels known to be correct, we update the nodes in order of their distance from the leaves, as shown in Fig. 2. The heart of the top-level heuristic is the median computation. Exact median-finding algorithms are limited to small genomes, small edge lengths in the tree, and few changes in content—and none of these properties holds in our problem. We therefore pursue a simple heuristic inspired by geometry. The median of a triangle in the plane can be found by drawing a line from one vertex to the middle of the opposite segment, then moving two thirds of the way along this line. By analogy,

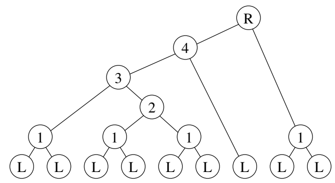

Fig. 2. Internal nodes ordered by their distance from the leaves. Nodes with lower indices will be labeled first; no label is generated for the root.

we generate a sorting sequence from one genome to another (an edge of the triangle), then choose a genome halfway along this sorting sequence and generate a new sorting sequence from it to the third genome, stopping one-third along the way.

We extend the method of Marron *et al*. [7] to enumerate all possible positions, orientations, and orderings of genes after each operation. Deleted genes at the endpoint of an inversion are moved to the other endpoint if doing so avoids "trapping" the deleted genes between two consensus genes that are adjacent in the target genome. Inserted genes are moved so as to remain adjacent to one of the two consensus genes between which they lie in the target genome. We can thus generate the genomes produced by "running" a portion of the sorting sequence, then use these intermediate genomes for the median heuristic just described, all in polynomial time.

This handling of inserted genes leads to an *overestimate* of the edit distance, which Marron *et al*. showed at most doubles the number of operations [7]. Their original method calculates all possible positions in the source genome to which the inversion sequence could move insertions and chooses the final position (for each insertion) to minimize the number of groups of inserted genes; it may *underestimate* the edit distance because the grouping of inserted genes may require an inversion to join inserted genes and simultaneously split deleted genes, which is not possible. We compared pairwise distances produced by our method and by theirs to get an upper bound on the overestimation: average and maximum differences between the overestimate and underestimate were 11.3% and 24.1%, respectively.

4.2 Gene Content

We predetermine the gene content of every internal node before computing any median: once the gene content of an internal node is assigned, it remains unchanged. Since the tree is rooted, we know the direction of time flow on each tree edge; we also assume that deletions are far more likely than insertions, The number of copies of each gene g is decided independently of all other genes; at internal node i, it is set to the maximum number of copies of g found in any of the leaves in i's subtree if: (i) there are leaves both inside and outside i's subtree that contain g; or (ii) there are leaves containing g in each half of i's subtree. Otherwise the number of copies of gene g in node i is set to zero.

This value can be calculated in $O(NG)$ time, where N is the number of nodes in the tree and G is the number of distinct genes in all the leaves, as follows. For each node in the tree, we determine the maximum number of copies of each gene from among the

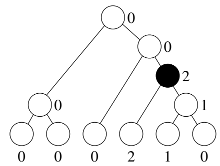

Fig. 3. The number of copies of a gene in internal nodes.

leaves of that node's subtree, using a single depth-first traversal. We use a second depth-first traversal to set the actual number of copies of each gene at each internal node. If either of the root's children has a subtree maximum of zero, then we set the root's actual number to zero as well. For each internal node other than the root, if its parent's actual number of copies is zero and at least one of its two children's subtree maximum is zero, then we set the number of copies for the gene to zero; otherwise we set the number of copies to the node's subtree maximum for the gene.

Internal nodes will thus possess at least as many copies of a gene as the majority consensus of their neighbors' gene contents. An internal node will always possess a copy of a gene if two or more of its neighbors do. (We consider the two children of the root to be neighbors.) Moreover, if a node is the nearest common ancestor of all genomes possessing the gene, it may have more copies of the gene than its parent and one of its children, as in the case of the black node in Fig. 3. The gene content of intermediate genomes along sorting sequences is a union of the gene contents of the starting genomes, because the sorting sequence of operations that we use always involves first insertions, then inversions, and finally deletions. Therefore, when calculating medians from sorting sequences, we face three cases in which the number of copies of a gene differ between the intermediate genome, the median genome, and the median's parent—see Fig. 4. In Fig. 4a, the intermediate genome has the same number of copies as the median, but fewer than the parent, as with the black node's right child in Fig. 3. Each copy in the parent that is not matched by the duplication-renaming algorithm will be excluded from the median genome. The case of Fig. 4b only arises when the median genome is the nearest common ancestor of all genomes containing the gene in question, as with the black node in Fig. 3. Genomes along the intermediate sequence have the same number of copies as the median, while the parent of the median contains no

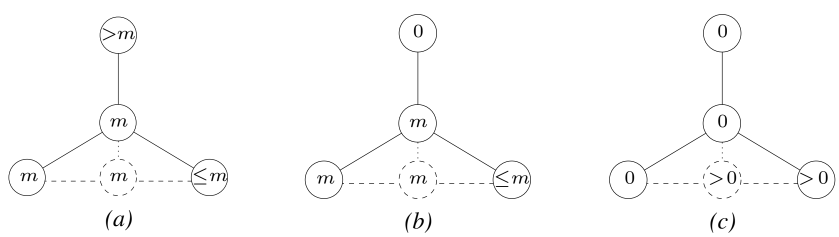

Fig. 4. Cases where the median and its neighbors have different numbers of copies of a gene. Solid lines are tree edges; dashed and dotted lines are fractions ($\frac{1}{2}$ and $\frac{1}{3}$, resp.) of sorting sequences.

copy at all. Finally, the case of Fig. 4c can only arise when the right child of the median is the nearest common ancestor of all genomes containing the gene, as with the parent of the black node in Fig. 3.

Biologically, this process of finding which duplicates to include in the median corresponds to matching orthologous duplicates of each gene between genomes and to discarding unmatched paralogous duplicates. Since the original nucleotide sequences are abstracted away before the analysis begins, this ortholog matching is decided entirely on the basis of which other genes are located next to the different homologs. Fortunately, orthologs and paralogs that can be distinguished by a nucleotide-based analysis are assigned different gene numbers before our analysis begins. Therefore, our method represents a reasonable way to integrate both nucleotide and gene-order data in differentiating orthologous and paralogous homologs of genes.

4.3 Cluster Condensation

To extract information from larger and more complex biological datasets, we need fast algorithms with fast implementations; faster processing also enables a more thorough analysis and thus produces results of higher quality. The key factor here is the size of the genomes—their number is a much smaller issue. We thus developed a technique to identify and condense gene clusters in order to reduce the size of the genomes. Our approach generalizes that used in genomes with equal content [23]; in contrast, GRAPPA only condenses identical subsequences of genes, because it aims to preserve the identity of edit sequences. Our method allows the condensation of clusters based only on content (not order, at least as long as genes stay on the same strand) and also handles the difficult cases that arise out of unequal gene content (such as an insertion within a cluster).

To identify clusters, we first use the duplication-renaming technique of Marron *et al.* to create duplication-free genomes. After renaming, we remove any genes not present in all of the genomes under examination. This step creates a group of genomes with equal gene content. We then use the cluster-finding algorithm of Heber and Stoye [23] to find equivalent clusters of genes within the equal-content genomes. Once clusters are identified, each one is condensed out of the original genomes and replaced with a single marker (as if it were a single gene). In a set of genomes with unequal gene content, there can be genes inside a cluster that are not present in the corresponding equal-content genomes. We deal with these genes in one of two ways. If every occurrence of that gene is located inside the cluster in each of the genomes that possesses the gene, then the gene is condensed along with the rest of the cluster. Otherwise, the extra gene is moved to one side of the cluster and the cluster condensed. When a median genome is computed, a median for each cluster is also computed and each cluster's marker in the median genome is replaced with the cluster's median. At this point, if any extra genes moved to the side of the cluster are still beside it, they are moved back inside the cluster to a position similar to their original one.

4.4 Putting It All Together

Ancestral genome reconstructions are performed using these three main components. Initialization of the internal nodes of the tree is done from the leaves up by taking either the midpoint or one of the two endpoints (along the inversion portion of an edit sequence) of an internal node's two children and discarding any genes not allowed by the median gene content. This method accounts for all three of the cases in Fig. 4 and produces labels with the desired gene content. New medians are computed locally node by node in a postorder traversal of the tree, so as to propagate information from the leaves towards the root. Whenever a median is found that reduces the local score at a node, it immediately replaces the previous label at that node; that node and all its neighbors are then marked for further update.

5 Testing

We used our label reconstruction method on the bacterial dataset as well as on simulated datasets. With simulated datasets, we know the true labels for the internal nodes as well as the exact evolutionary events along each edge, so that we can test the accuracy of the reconstruction. The goal of our experiments was to generate datasets roughly comparable to our biological dataset so that our experimental results would enable us to predict a range of accuracy for the results on the biological dataset.

The simulated data were created using the tree of Fig. 1; edge lengths were assigned to the tree based on our best estimate of the edge lengths for the bacterial genomes. To keep the data consistent, edge lengths were interpreted as the number of operations per gene rather than as an absolute number, allowing us to use the same relative value for genomes of different sizes. The tree was labeled by first constructing a root genome. The number of genes g and the total size n of the root genome were set as variable user parameters. One of each gene from 1 to g was added to the root genome, after which $n - g$ additional genes were chosen uniformly at random in the range $[1, g]$ and added to the root genome. The root genome was then randomly permuted and each gene assigned a random sign. The other nodes were then labeled from the root by evolving the genomes down the tree according to the prescribed number of operations. The allowed operations were insertions, deletions, and inversions. Although the total number of operations was fixed, the proportion of each of the three types of operations was left as a variable parameter by setting the ratio of inversions to insertions to deletions. This mix of operations was used over all edges of the tree.

The characteristics of each type of operation were determined separately. The length of each inversion was chosen uniformly at random between 1 and half the size of the genome, with a start point chosen uniformly at random from the beginning of the genome to the size of the genome minus the length of the inversion. The average insertion length was set via a user parameter as a portion of the size of the root genome and was used unchanged over the entire tree, while the actual length of each insertion was drawn from a Poisson distribution with this expectation and its location was chosen uniformly at random from the beginning to the end of the genome. In moving from the root to the leaves, it was assumed that a particular gene could only be inserted along one edge of the tree—multiple insertions of the same gene, even along separate paths, were not allowed. The average deletion length was chosen as a user-specified portion of the genome from which genes were being deleted, thus varying from edge to edge as well as along each edge with each successive deletion, while the actual size of each deletion was drawn from a Poisson distribution with this expectation and with a start location chosen uniformly at random from the beginning of the genome to the size of the genome minus the length of the deletion. With the constant expected insertion length, genomes grow linearly in the absence of deletions, while, with a proportional expected deletion length, genomes shrink exponentially in the absence of insertions. When both insertions and deletions are used, genomes farther from the root tend towards a stable size. Along each edge, the prescribed number of insertions are performed first, then inversions, and finally deletions. Once all nodes have been been assigned genomes, the resulting leaf genomes are fed into our reconstruction procedure. The results of the reconstruction, in terms of gene content and gene order at each internal node, are compared with the "true" tree, i.e., that generated in the simulation.

We constructed trees using five different models: an "inversion-only" model, a "no-deletions" model with a 6:1 inversion-to-insertion ratio, a "no-insertions" model with a 6:1 inversion-to-deletion ratio, a "low-insertion/deletion" model with a 40:4:1 ratio of inversions to deletions to insertions, and a "high-insertion/deletion" model with a 30:10:3 ratio of inversions to deletions to insertions. The average insertion length was set to 2% of the root genome and the average deletion length to 3% of the local genome.

In order to test the efficacy of cluster condensation, we tested the technique on triples among the bacterial genomes that lie close to each other on the tree in Fig. 1. Triples were chosen by selecting internal nodes, then, for each of the three edges leading out from the internal node, by choosing a nearby leaf reachable by following the edge. For each set of three genomes, we measured the sum of the lengths of all clusters that were found.

6 Results

Our discussion and summaries of results refer to Fig. 5. Reconstruction of ancestral genomes for the bacterial genomes takes around 24 hours on a typical desktop computer. The midpoint-initialization proved quite strong: the only genomes to be updated in the

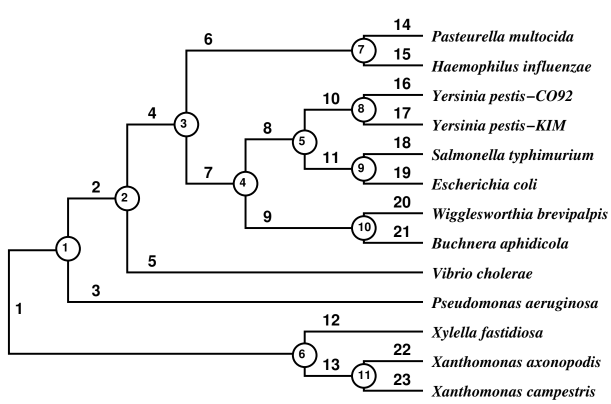

Fig. 5. The bacterial tree with numbered edges and internal nodes.

Table 1. Error Percentage in Tree Scores

	Avg error	Min error	Max error
Inversion only	63.2%	57.3%	67.4%
No deletions	62.6%	54.8%	70.7%
No insertions	45.2%	37.6%	54.3%
Low insertion/deletion	56.4%	46.7%	64.8%
High insertion/deletion	34.9%	25.1%	46.4%

subsequent local improvement procedure were the two children of the root (nodes 1 and 6 in Fig. 5), the only neighboring genomes in which one neighbor was not used to create the other. When we used endpoint-initialization, three internal nodes were updated (nodes 3, 4, and 6 in Fig. 5) and the score of the entire tree was lowered by 2.8%. This finding may indicate that the initialization is very good, but it may also reflect the large numbers of local optima in the search space—a similar finding was reported for the simpler GRAPPA [18]. It should be noted that, when calculating medians, only four different midpoints in the child-to-child sorting sequence are used; from each of these midpoints, only three midpoints in the sorting sequence from the intermediate genome to the parent are tested. Thus we only perform a very shallow search and could easily miss a better solution. Interestingly, though, when we did a slightly more thorough search with ten midpoints from child to child and four midpoints from intermediate to parent, using endpoint-initialization, the tree score was slightly worse than in the shallower analysis, although the search, which took about 3.5 times longer, updated the same three internal nodes. Of course, this larger search remains very shallow; going beyond it will require a much more efficient implementation of the duplication-renaming heuristic of Marron *et al*. [7]—in our current version, it uses up over 90% of the computing time.

We simulated 100 labelings of the tree with a root genome size of 200 genes for each of the five previously described scenarios. Endpoint-initialization was used in all scenarios. The leaf genomes produced in our simulations ranged in size from 70 genes to 400 genes. We compared the predicted gene content of the internal nodes with the actual gene content. As expected (due to our restriction on generation), the predicted gene content always matched, except when a gene copy that was present at an internal node was lost in all leaves. Failure to detect this kind of missing gene is unavoidable in an analysis since the deletion from all leaves means that no historical record is left to attest the presence of that gene in ancestral genomes. When we compared the number of operations over all edges in reconstructed trees versus the original simulated tree, the score for the tree was fairly inaccurate, consistently overestimating the true score, as illustrated in Table 1. The rather tight distribution for the tree indicates that the error is not a random process, but a result of some aspect of our reconstruction method, one that may lend itself to reverse mapping.

We compared edge lengths in the reconstructed trees with those in the true trees by calculating the ratio of the lengths for each edge (Fig. 6). A perfect reconstruction would give a ratio of 1.0; as the figure shows, most ratios are higher, with edges further from leaves having larger ratios (and also larger variances). About half of the 23 edges are within a factor of two and another quarter are within a factor of four.

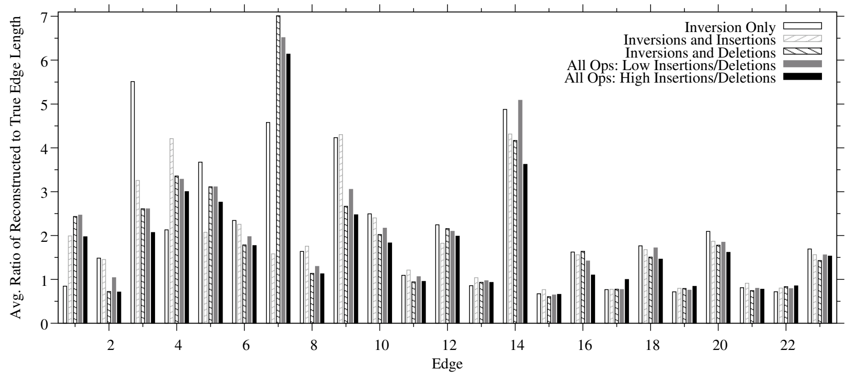

Fig. 6. Average ratio of reconstructed to true edge lengths for each edge, numbered as in Fig. 5.

We calculated the number of operations needed to convert the reconstructed genome labels at internal nodes into the corresponding labels from the true tree. We normalized these values by dividing them by the size of the true tree genome, thus a perfect reconstruction would give edit distances of zero. Here again, internal nodes closer to leaves are closer to the true ancestral gene orders.

Finally, we tested cluster condensation on triples of closely-related bacterial genomes. The number of genes that fell into clusters, and thus the number of genes that could be condensed away, is a lower bound on the clustering potential in the actual tree. Condensation would remove the same number of genes from each genome, so the maximum possible condensation is determined by the smallest of the three genomes considered. In the cases we examined, it was possible to condense away on average 21% (ranging from 13% to 31%) of the size of the smallest genome. Computationally, however, the cluster condensation is heavily dependent on the duplication-renaming heuristic, the slowest of the various algorithmic components; thus, the benefits of working with

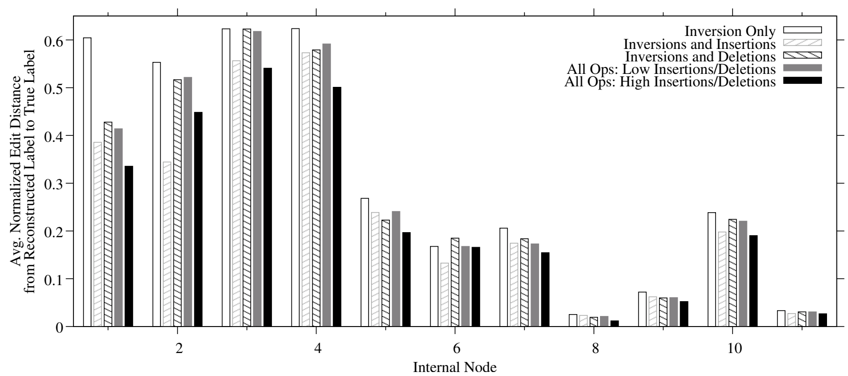

Fig. 7. Average normalized edit distance from reconstructed to true labels for each internal node of the tree, numbered as in Fig. 5.

smaller genomes will not be apparent until the time required to condense the genomes can be substantially reduced.

7 Conclusions

We have successfully produced a framework under which we are able to compute ancestral gene orders for modern bacteria. The number of operations over the tree is somewhat inaccurate in absolute terms, but rather accurate in relative terms—the error is a systematic bias towards overestimation. Accuracy is, unsurprisingly, greater for internal nodes and edges closer to the leaves (the modern data). We also have shown that, under certain simplifying assumptions, we are able to recover consistently the gene content of the ancestral genomes of simulated genomes. The size and complexity of the genomes mean that only a very shallow search of the space of possible ancestral genomes is possible: our results are undoubtedly heavily impacted by that problem, but we have pushed the size boundary for phylogenetic analysis with gene orders by an order of magnitude.

8 Acknowledgments

We thank our colleagues at the U. of Arizona for data, analysis, and advice: Nancy Moran (E. Lerat's postdoctoral advisor) and Howard Ochman and his postdoctoral student Vincent Daubin. We also thank Jens Stoye (U. Tübingen) for providing the source code to the cluster-finding program and Mark Marron and Krister Swenson (U. of New Mexico) for many useful discussions.

Research on this topic at the University of New Mexico is supported by the National Science Foundation under grants EF 03-31654, IIS 01-13095, IIS 01-21377, and DEB 01-20709 and by the NIH under grant 2R01GM056120-05A1 (through a subcontract to the U. of Arizona); research at the University of Arizona on this topic is supported by the NIH under grant 2R01GM056120-05A1.

References

1. Cosner, M., et al.: An empirical comparison of phylogenetic methods on chloroplast gene order data in Campanulaceae. In Sankoff, D., Nadeau, J., eds.: Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment, and the Evolution of Gene Families. Kluwer Academic Pubs., Dordrecht, Netherlands (2000) 99–121
2. Waterston, R., et al.: Initial sequencing and comparative analysis of the mouse genome. Nature **420** (2002) 520–562
3. Hannenhalli, S., Chappey, C., Koonin, E., Pevzner, P.: Genome sequence comparison and scenarios for gene rearrangements: A test case. Genomics **30** (1995) 299–311
4. McLysaght, A., Baldi, P., Gaut, B.: Extensive gene gain associated with adaptive evolution of poxviruses. Proc. Nat'l Acad. Sci. USA **100** (2003) 15655–15660
5. Sankoff, D.: Genome rearrangement with gene families. Bioinformatics **15** (1999) 990–917
6. Tang, J., Moret, B., Cui, L., dePamphilis, C.: Phylogenetic reconstruction from arbitrary gene-order data. In: Proc. 4th Int'l IEEE Conf. on Bioengineering and Bioinformatics BIBE'04, IEEE Press (2004)

7. Marron, M., Swenson, K., Moret, B.: Genomic distances under deletions and insertions. In: Proc. 9th Int'l Conf. Computing and Combinatorics (COCOON'03). Volume 2697 of Lecture Notes in Computer Science., Springer Verlag (2003) 537–547
8. Lerat, E., Daubin, V., Moran, N.: From gene trees to organismal phylogeny in prokaryotes: The case of the γ-proteobacteria. PLoS Biology **1** (2003) 101–109
9. Clark, M., Moran, N., Baumann, P.: Sequence evolution in bacterial endosymbionts having extreme base composition. Mol. Biol. Evol. **16** (1999) 1586–1598
10. Lawrence, J., Ochman, H.: Amelioration of bacterial genomes: Rates of change and exchange. J. Mol. Evol. **44** (1997) 383–397
11. Parkhill, J., et al.: Complete genome sequence of a multiple drug resistant *Salmonella enterica* serovar Typhi CT18. Nature **413** (2001) 848–852
12. Stover, C., et al.: Complete genome sequence of *Pseudomonas aeruginosa* PAO1, an opportunistic pathogen. Nature **406** (2000) 959–964
13. Moret, B., Tang, J., Wang, L.S., Warnow, T.: Steps toward accurate reconstructions of phylogenies from gene-order data. J. Comput. Syst. Sci. **65** (2002) 508–525
14. Bader, D., Moret, B., Warnow, T., Wyman, S., Yan, M.: (GRAPPA (Genome Rearrangements Analysis under Parsimony and other Phylogenetic Algorithms)) `www.cs.unm.edu/~moret/GRAPPA/`.
15. Moret, B., Tang, J., Warnow, T.: Reconstructing phylogenies from gene-content and gene-order data. In Gascuel, O., ed.: Mathematics of Evolution and Phylogeny. Kluwer Acad. Publ. (2004)
16. Tang, J., Moret, B.: Scaling up accurate phylogenetic reconstruction from gene-order data. In: Proc. 11th Int'l Conf. on Intelligent Systems for Molecular Biology ISMB'03. Volume 19 (Suppl. 1) of Bioinformatics. (2003) i305–i312
17. Moret, B., Siepel, A., Tang, J., Liu, T.: Inversion medians outperform breakpoint medians in phylogeny reconstruction from gene-order data. In Guigó, R., Gusfield, D., eds.: Proc. 2nd Int'l Workshop on Algorithms in Bioinformatics WABI'02. Volume 2452 of Lecture Notes in Computer Science., Springer Verlag (2002) 521–536
18. Moret, B., et al.: A new implementation and detailed study of breakpoint analysis. In: Proc. 6th Pacific Symp. on Biocomputing (PSB'01), World Scientific Pub. (2001) 583–594
19. Bergeron, A.: A very elementary presentation of the Hannenhalli-Pevzner theory. In: Proc. 12th Ann. Symp. Combin. Pattern Matching (CPM'01). Volume 2089 of Lecture Notes in Computer Science., Springer Verlag (2001) 106–117
20. Bergeron, A., Stoye, J.: On the similarity of sets of permutations and its applications to genome comparison. In: Proc. 9th Int'l Conf. Computing and Combinatorics (COCOON'03). Volume 2697 of Lecture Notes in Computer Science., Springer Verlag (2003) 68–79
21. El-Mabrouk, N.: Genome rearrangement by reversals and insertions/deletions of contiguous segments. In: Proc. 11th Ann. Symp. Combin. Pattern Matching (CPM'00). Volume 1848 of Lecture Notes in Computer Science., Springer Verlag (2000) 222–234
22. Overbeek, R., et al.: The use of gene clusters to infer functional coupling. Proc. Nat'l Acad. Sci. USA **96** (1999) 2896–2901
23. Heber, S., Stoye, J.: Algorithms for finding gene clusters. In: Proc. 1st Int'l Workshop on Algorithms in Bioinformatics WABI'01. Volume 2149 of Lecture Notes in Computer Science., Springer Verlag (2001)