# Early Eukaryote Evolution Based on Mitochondrial Gene Order Breakpoints

David Sankoff [*] David Bryant [*] Mélanie Deneault [*]

B. Franz Lang [†] Gertraud Burger [†]

June 17, 2004

[*]Centre de recherches mathématiques, Université de Montréal, CP 6128 succursale Centre-ville, Montréal, Québec H3C 3J7. E-mail: {sankoff,bryant,deneault}@crm.umontreal.ca.

[†]Département de biochimie, Université de Montréal, CP 6128 succursale Centre-ville, Montréal, Québec H3C 3J7. E-mail: {langf,burgerg}@bch.umontreal.ca.

**Abstract**

The comparison of the gene orders in a set of genomes can be used to infer their phylogenetic relationships and to reconstruct ancestral gene orders. For three genomes this is done by solving the "median problem for breakpoints"; this solution can then be incorporated into a routine for estimating optimal gene orders for all the ancestral genomes in a fixed phylogeny. For the difficult (and most prevalent) case where the genomes contain partially different sets of genes, we present a general heuristic for the median problem for induced breakpoints. A fixed-phylogeny optimization based on this is applied in a phylogenetic study of a set of completely sequenced protist mitochondrial genomes, confirming some of the recent sequence-based groupings which have been proposed and, conversely, confirming the usefulness of the breakpoint method as a phylogenetic tool even for small genomes.

# 1 Introduction.

The origin and early diversification of the eukaryotes is one of the fundamental problems of evolutionary theory. The widely accepted endosymbiotic origin of the mitochondrion and its consequent evolution, in key respects independent of the evolution of the nuclear genome, make it a natural focus of phylogenetic studies. Indeed phylogenies based on a number of mitochondrial genes have led to a far clearer understanding of the phylogeny of unicellular eukaryotes—the protists and the fungi—than was ever possible based on morphological classifications alone (Burger et al. 1999, Gray et al. 1998, 1999, Lang et al. 1997,1998a,b, 1999, Paquin et al. 1997, Turmel et al. 1999). Nevertheless, this approach is limited by the relatively small number of genes present in all or most mitochondria, and the finite amount of phylogenetic information that can be extracted from the sequence comparison of any of these genes. For some time we have advocated the quantification of gene order changes within the mitochondrion as an independent measure of genomic divergence that can be used to supplement sequence comparison data (Sankoff et al. 1992).

Early work in the construction of phylogenies from gene order data used a distance based approach. Sankoff (1992) estimated the distance between two gene orders using a heuristic algorithm for minimizing a weighted measure of the number of reversals and transpositions of chromosome fragments, as well as the insertion and deletion of individual genes, necessary to transform one order into the other. Exact polynomial time algorithms for calculating the minimum number of reversals, of translocations, or of reversals and translocations, needed

to transform one gene order into another were developed by Hannenhalli and Pevzner (1995a), Hannenhalli (1995), and Hannenhalli and Pevzner (1995b), respectively. Distance matrix methods could then be used to reconstruct a phylogeny.

As attention focused on exact and efficient algorithms for rearrangement distances, phylogenetic questions were somewhat neglected. This was partly due to the lack of realism in measuring genomic divergence in terms of reversals only, with all reversals of equal weight. It was also due to the limited aim of distance matrix methods in estimating only the phylogenetic tree, and not the characters of the ancestors associated with the internal nodes of the tree. This contrasts with methods such as parsimony or likelihood which avoid reducing the data to pairwise distances prior to phylogenetic reconstruction, and thus conserve information necessary for inferring ancestral structures. The latter methods would require us to compare more than two genomes at a time, whereas two only suffice with distance-based methods. Comparison of more than two genomes at a time, however, has been shown to be NP-hard (Caprara 1999). Moreover, even heuristic approaches to such comparisons work well only for very small problems (cf Hannenhalli et al. 1995, Sankoff et al. 1996).

To circumvent these difficulties, Sankoff and Blanchette (1998) introduced the notion of breakpoint phylogeny. For two genomes containing the same genes, breakpoints are simply pairs of consecutive genes $g_1$ and $g_2$ which occur in order $g_1 g_2$ in one genome but not in the other. Adjacency is also considered disrupted if the two genes have different orientation to each other in the two

genomes; e.g. $g_2$ succeeds $g_1$ but with opposite reading direction in one genome while they are adjacent on the same DNA strand in the other genome. The number of breakpoints correlates with the evolutionary divergence of the two genomes. In contrast to rearrangement distances, this is easily extended to three or more gene orders: the *median problem*, though it is computationally costly for large genomes (Pe'er and Shamir 1998, Bryant 1998). Solutions to the median problem can be combined and iterated to optimize the ancestral genomes in a given tree topology. For moderate numbers of genomes, all possible topologies can be evaluated to solve the phylogenetic problem. Blanchette et al. (1999) have demonstrated the applicability of the method by showing the plausibility of breakpoint phylogenies constructed on the basis of the relatively small (37 genes) mitochondria of metazoans, from humans to nematodes.

Unfortunately the notion of breakpoint does not carry over in a straightforward way when the genomes being compared do not have the same set of genes. The shared genes in two genomes may be ordered in exactly the same way but because of intervening genes that belong to only one or the other, the number of breakpoints may be large. It is more appropriate in this context to consider *induced breakpoints*, the breakpoints remaining when the genes belonging to only one or the other genome are discarded. When comparing a set of three or more genomes which vary greatly in the number of genes they contain, it also becomes necessary to normalise the number of induced breakpoints between two genomes by the number of genes they share.

In the present paper we propose and test heuristics for the median problem

for (normalised) induced breakpoints on unequal genomes, including a particularly rapid version – $O(n \log n)$, where $n$ is the length of the longest genome. We then incorporate this routine into a procedure for optimizing the gene order inferred at each ancestral node of a fixed phylogenetic tree. This methodology is applied to a set of completely sequenced protist mitochondrial genomes, confirming some of the recent sequence-based groupings which have been proposed and, conversely, confirming the usefulness of the breakpoint method as a phylogenetic tool even for small genomes.

# 2 Definitions and problems

## 2.1 Signed genomes and the breakpoint distance

A genome $A$ is represented by a (circular) ordering $A = \langle a_1, a_2, \ldots, a_n, a_1 \rangle$. To indicate which genes lie on the same DNA strand, each gene is signed either positive ($a_i$) or negative ($-a_i$), depending on whether it is on the strand read (transcribed) in the clockwise or the counterclockwise direction, respectively. Reversing both the gene order and the signs of all the genes of a genome therefore gives an alternative representation of the same genome. We let $\mathcal{G}(A)$ denote the set of genes in $A$, including both positive and negative copies of each gene. The set of genes in a genome is referred to as its **gene content**. For each $a_i \in \mathcal{G}(A)$ we let $succ(a_i, A)$ denote the **successor** of $g$ in $A$. In this example, $succ(a_i, A) = a_{i+1}$ and $succ(a_n, A) = a_1$. Note that $succ(g, A) = h$ if and only if $succ(-h, A) = -g$.

A **breakpoint** in a genome $A$ with respect to another genome $B$ on the same gene set is an ordered pair of genes $(g, h)$ such that $succ(g, A) = h$ and $succ(g, B) \neq h$. An **adjacency** in a genome $A$ with respect to another genome $B$ is an ordered pair of genes $(g, h)$ such that $succ(g, A) = h = succ(g, B)$.

Let $A$ be a genome and let $W$ be a set of genes. We let $A|_W$ denote the genome $A$ with all genes *not* in $W$ removed, and all remaining genes left in the same order. This is called an **induced genome** of $A$. The **(normalised) breakpoint distance** between two genomes $A$ and $B$ is then defined

$$d(A, B) = \frac{1}{2|\mathcal{G}(A) \cap \mathcal{G}(B)|} |\{g \in \mathcal{G}(A) \cap \mathcal{G}(B) : succ(g, A|_{\mathcal{G}(B)}) \neq succ(g, B|_{\mathcal{G}(A)})\}| \tag{2.1}$$

The scaling factor $\frac{1}{2|G(A) \cap G(B)|}$ is pertinent only when there is variation in gene content between genomes, as in the mitochondrial genomes we study here.

Because it focuses only on *induced* breakpoints, and is normalised, our breakpoint distance is relatively robust against missing data, such as genes absent in some organisms or excluded for the methodological reasons invoked in Section 3.2.

## 2.2 The breakpoint median problem

Let $\mathcal{A} = A_1, \ldots, A_N$ be a collection of genomes and define

$$\mathcal{G}(\mathcal{A}) = \mathcal{G}(A_1) \cup \mathcal{G}(A_2) \cdots \cup \mathcal{G}(A_N) \ . \tag{2.2}$$

We will assume that each gene $g \in \mathcal{G}(\mathcal{A})$ appears in at least two genomes, as genes appearing in only one genome do not contribute to gene order information.

The median score $\Psi(X, \mathcal{A})$ of a genome $X$ with gene set $\mathcal{G}(X) = \mathcal{G}(\mathcal{A})$ is defined to be

$$\Psi(X, \mathcal{A}) = \sum_{i=1}^{N} d(X, A_i) \tag{2.3}$$

and the breakpoint median problem is to find $X$ with $\mathcal{G}(X) = \mathcal{G}(\mathcal{A})$ that minimizes $\Psi(X, \mathcal{A})$. The breakpoint median problem is NP-hard, even for three genomes with equal gene content (Pe'er and Shamir 1998). In Section 5.1 we present a fast heuristic for the breakpoint median problem for genomes with unequal gene sets.

## 2.3 The breakpoint phylogeny problem

Let $T = (V, E(T))$ be a binary rooted tree with $N$ leaves. We direct all edges of $T$ towards the root. To each leaf we assign a different genome $A_i$ from the input collection $\mathcal{A} = A_1, A_2, \ldots, A_N$. A **valid assignment** of genomes to *internal* nodes of $T$ is a function $\mathcal{X}$ from $V$ to the set of genomes satisfying:

1. For each leaf $v \in V$, $\mathcal{X}(v)$ is the genome already assigned to $v$.

2. For each internal node $v \in V$ with children $u_1, u_2$ we have $\mathcal{G}(\mathcal{X}(v)) = \mathcal{G}(\mathcal{X}(u_1)) \cup \mathcal{G}(\mathcal{X}(u_2))$.

The second condition models the situation in eukaryotes where, it is believed, all known mitochondrial genes were present in the ancestral mitochondrial genome and gene content has evolved only through deletion. For ease of presentation we let $\mathcal{G}(v)$ denote $\mathcal{G}(\mathcal{X}(v))$, which is constant over all valid assignments $\mathcal{X}$.

The **length** of a tree $T$ with respect to the input genomes $\mathcal{A}$ and a valid assignment $\mathcal{X}$ is defined

$$l(\mathcal{X}, \mathcal{A}, T) = \sum_{(u,v)\in E(T)} d(\mathcal{X}(u), \mathcal{X}(v)) \tag{2.4}$$

and the **breakpoint phylogeny problem** is to find a valid assignment $\mathcal{X}$ minimizing $l(\mathcal{X}, \mathcal{A}, T)$.

The median breakpoint problem is equivalent to the breakpoint phylogeny problem applied to a rooted tree with one internal node that has $N$ children labelled by the input genomes to the median problem. Hence, the breakpoint phylogeny problem is also NP-hard. In Section 5.1 we present an iterative heuristic for the breakpoint phylogeny problem for genomes with unequal gene sets.

# 3 Gene order data for early eukaryotes

## 3.1 The evolution of the eukaryotes

Prior to plants, animals and fungi, a large number of mostly unicellular eukaryotes (protists) diverged from the common eukaryotic ancestor. Their classification has traditionally been difficult since they lack the differentiated tissues organized into organs that help categorize plants and animals. However, contrary to prokaryotes that virtually lack morphological character, protists can be classified based on ultrastructural features, such as features of the flagellar apparatus or the shape of mitochondrial cristae. Although not without exception, animals, fungi, plants, green and red algae all manifest flattened cristae, while

*Euglena*, the trypanosomatids (like *Leishmania*) and heteroloboseans possess discoidal cristae. Still another large grouping, including the ciliates (such as *Parmecium*), the slime molds and the stramenopiles, have tubular cristae (Gray et al. 1998).

Among the organisms characterized by flattened cristae, sequence analysis of several mitochondrial genes has indicated a common ancestry for animals and fungi (Paquin et al. 1997), a close relationship between red and green algae (Burger et al. 1999), and the origin of the land plants within the latter. Several of the subgroups within the discoidal cristae grouping can also be linked through gene sequence comparison. The same can be said within the tubular cristae group, particularly those within the stramenopiles, where links are evident between the chrysophytes, the synurophytes, the oomycetes and the bicosoecids. Within each of these large groupings, however, the earliest relationships remain unclear.

## 3.2 The data

GOBASE (http://megasun.bch.umontreal.ca/gobase/ gobase.html) is a relational organelle genome database, which integrates sequence data, information on evolution, taxonomy, biochemistry, RNA secondary structure, physical maps and more (Korab-Laskowska et al. 1998).

The major body of data contained in GOBASE consists of mitochondrial sequence data drawn from the Entrez database system and taxonomy data extracted from the NCBI Taxon database. These data are obtained on a regular

basis by a custom-made tool, POP2, which reads the Entrez data in ASN.1 format, extracts the data relevant to the molecular features defined in GOBASE, and stores this information in GOBASE tables.

The production of gene orders from sequence data cannot be totally automated. Genes may overlap, may be fragmented and scattered across the genome (especially the rRNA genes), may be unrecognized or unannotated in the Entrez file, or annotated in an idiosyncratic way. As a first step, maps are produced from an entry, and a gene order with signed genes is derived from that. Maps are posted on the GOBASE website. Many of the mitochondrial sequences are produced in the laboratories affiliated with the Organelle Genome Megasequencing Program and the Fungal Mitochondrial Genome Project (e.g. Burger et al. 1999, Lang et al. 1997, Paquin et al. 1996, Paquin et al. 1997, Turmel et al. 1999), and prepublication maps of these are also posted.

For the purposes of the analysis in the present paper, duplicate genes were excluded from some of the genomes because of the inability of our method to handle these duplicates. Most of these were tRNA genes. As far as possible, we tried to identify homologous sets of mitochondrial tRNA genes across as many protists as possible, taking into account the corresponding amino acid, the anticodon, the translation table appropriate to the organism and, in the few cases where it was possible, positional correspondences in closely related genomes. In the remaining instances where the duplicates remained indistinguishable, we deleted both from the gene order.

As explained at the end of Section **??**, this introduces little bias into the

comparison, though the loss of data does decrease the precision of the estimates. In other cases, where entire fragments of the genome were duplicated, we simply deleted the fragment which seemed the secondary one, based on comparisons with closely related genomes or by conforming to the strandedness of the majority of the genome.

For some genes in some genomes, part of the gene is located in one position and the rest elsewhere, in such a way that other genes intervene between two fragments. We retained only the position of the longer fragment. Where there was fragmentation of a gene into several pieces, the genome was excluded from the analysis.

Finally, we excluded all ORFs (hypothetical proteins) from the data unless there was good evidence that the same ORF appeared in two or more genomes.

Two other criteria served to exclude other genomes from the analysis; too few genes, such as in the trypanosomatids where, moreover, no tRNAs appear in the gene order, and an *a posterior* filter where a genome turned out to bear no more than random resemblance to any of the other genomes in the data set.

The data we analyzed are summarized in Table 1.

TABLE 1 GOES NEAR HERE

# 4 A practical heuristic for the breakpoint median problem

We now discuss algorithms for the breakpoint median problem. In Section 4.1 we discuss connections with the traveling salesman problem (TSP) that have been exploited in earlier work. The reduction to the TSP breaks down when the input genomes have unequal gene sets, but we can still use the literature on TSP as a source for algorithms for the general case. We found that a simple insertion algorithm performed best. A high level description of the algorithm is given in Section 4.2 and an $O(Nn \log n)$ time version is described in Section 4.3. Here, $N$ is the number of genomes in the input to the median problem, while $n$ is the number of genes in the union of their gene sets. The application and variability of the heuristic are explored in Section 4.4.

## 4.1 The breakpoint median problem and the TSP

The breakpoint median problem was first investigated by Sankoff and Blanchette (1997) who showed that when all genomes have the same gene set, the median problem can be converted into an instance of the traveling salesman problem (TSP). Though the TSP is itself NP-hard, there exist an impressive range of heuristics, lower bounds, branch and cut methods and iterative improvement techniques that can be used to solve the TSP for even moderately large problem instances. (Reinelt (1991) gives a comprehensive survey of TSP methods.)

The reduction to TSP breaks down when we consider the *induced* break-

point median problem for genomes with unequal gene sets. The equivalent TSP problem would be one with multiple distance matrices defined on overlapping sets of cities—the length of a tour being the summed length of the respective induced tours. Many of the standard lower bounds, heuristic and exact methods for TSP cannot be translated to this general framework (e.g. spanning tree and matching based methods, Lin-Kernighan local search, Held-Karp lower bound). While we were able to construct a linear programming formulation of the induced breakpoint problem, the large number of variables required prevented the practical application of branch and cut style methods.

We implemented divide and conquer methods, tour amalgamation methods, $k$-opt search methods and various insertion methods. These were applied to triples of genomes taken from the eukaryote mitochondrial data set. Though the comparison was relatively informal, the insertion-based heuristics were clearly superior, and were adopted as our heuristics of choice.

## 4.2 A suprisingly effective heuristic for gene insertion

In this section we give a high level description of the algorithm. Implementation details, and efficiency gains, will be discussed in Section 4.3.

The insertion algorithm works by inserting the genes one by one into a partially complete genome. The place of insertion is chosen to minimize the induced breakpoint median score $\Psi(X, \mathcal{A})$, where $X$ is the partially completed genome. The heuristic is therefore an analogue of insertion-based heuristics from the traveling salesman problem (Reinelt 1991). There is a subtle difference that makes

the breakpoint median heuristic perform well whereas the original TSP heuristic performs quite badly. At each iteration we solve an *induced* subproblem, since the definition of the breakpoint distance is only affected by the relative order of shared genes. The induced subproblem captures far more important structural information of the whole problem than the simple restriction of a distance matrix to a subset of cities.

We pay particular attention to the insertion order of the genes. Clearly we want to use as much information as we can as early as possible. If the first genes that we insert only appear in a few genomes then the induced input genomes, with gene sets restricted to those genes already inserted, will be relatively small and uninformative. We therefore select an ordering that inserts genes in such a way that the genes appearing in the largest number of genomes are inserted first.

To formalise this intuition, let $\mathcal{A} = A_1, A_2, \ldots, A_N$ be a collection of genomes and put $\mathcal{G} = \mathcal{G}(\mathcal{A})$. For each $g \in \mathcal{G}$ define the frequency of $g$ by

$$freq(g, \mathcal{A}) = |\{A_i \in \mathcal{A} : g \in \mathcal{G}(A_i)\}|.$$

A linear ordering $g_1 < g_2 < \cdots < g_n$ of genes in $\mathcal{G}$ is **monotonic decreasing** if $freq(g_i, \mathcal{A}) > freq(g_j, \mathcal{A})$ implies $i < j$. We can generate a random monotonic decreasing ordering of $\mathcal{G}$ by partitioning $\mathcal{G}$ according to frequency, randomly ordering the set of genes with a particular frequency, and concatenating these orderings.

So as not to disguise the basic structure of the algorithm we present high level pseudocode here and leave detailed implementation details to the next

section.

**Priority insertion heuristic**

1. Choose a random monotonic decreasing order $g_1, g_2, \ldots, g_n$ of $\mathcal{G}$.

2. $med \leftarrow \langle g_1 \rangle$

3. **for** $k$ equals 2 to $|\mathcal{G}|$ **do**

4. Insert $g_k$ into $med$ so as to minimize $\Psi(med, \mathcal{A})$.

5. **end(for)**

6. **return** $med$

**end.**

## 4.3 Implementing the heuristic

The bottleneck in the Priority insertion heuristic is line 4 where we determine where to insert the next gene. At iteration $k$ there are $2(k-1)$ places to insert the new gene $g_k$ since we could insert before $h$ or $-h$ for each of the $k-1$ genes $h$ already in $med$. For each insertion position we need to calculate the corresponding increase in $\Psi$.

Suppose that we are maintaining a table containing the successors of every gene in a genome $A$. If we insert a new gene $x$ into $A$ between genes $g$ and $h$ then we will change exactly four entries in the successor table: the entries for $x$, $-x$, $g$ and $-h$. Let $B$ be another genome on the same gene set, and suppose we have inserted $x$ into $B$ between genes $g'$ and $h'$. The change in $d(A, B)$ resulting from these insertions can be computed by examining the successor table entries for $x$, $-x$, $g$, $-h$, $g'$, and $-h'$, counting the number of new adjacencies and the

number of new breakpoints. Hence we can update $d(A, B)$ in constant time.

In order to exploit the constant time update of breakpoint distances when the genomes have unequal gene sets we need to have some way of updating the induced genomes. Put $\mathcal{G}_k = \{g_1, g_2, \ldots, g_k\}$. At iteration $k$ we use the following induced genomes:

1. $A_i|_{\mathcal{G}_k}$ for each $i = 1, \ldots, N$.

2. $med|_{\mathcal{G}(A_i)}$ for each $i = 1, \ldots, N$.

We use a data structure that enables us to store each of these $2N$ genomes and update them at each iteration in $O(N \log n)$ time.

Given a signed genome $A$ and a subset of genes $X$ we construct a balanced binary tree with leaves corresponding to genes in $A$ and subtrees ordered in such a way that a postorder traversal of the tree gives the genes in the correct order. We assign a flag to each node in the tree indicating whether or not a subtree below that node contains any elements of $X$ (Figure 1).

FIGURE 1 GOES NEAR HERE

We can insert new genomes into $A$ or $X$, rebalance the tree, and update flags, all in $O(\log n)$ time. The successor of a gene in the induced genome is found by continuing a postorder traversal, skipping over subtrees that do not contain elements of $X$. Hence we can update our table of genomes in $O(N \log n)$ time.

Running simulations we noticed that the algorithm tended to insert a gene $g_k$ so as to introduce an adjacency with one of the input genomes. Since the search through all insertion locations is a bottleneck in the priority insertion algorithm, we implemented a second version of the heuristic. In this version, which we call **fast priority insertion**, we restrict the insertion locations for $g_k$ to those positions immediately before or after genes that are a predecessor or successor of $g_k$ in one of the input genomes. This reduces the number of positions to $O(N)$ at each iteration, giving an algorithm taking $O(Nn \log n)$ time in total.

We note that it is possible to construct a pathological case where fast priority insertion and priority insertion will insert in different positions. However we found (see Section 4.4) that the advantages of an increase in speed far outweighed the possible loss of optimality.

## 4.4 Applying the heuristic

Both insertion heuristics have some degree of randomness in the way they break ties. Improved genomes can be obtained by repeating the heuristics many times. In order to choose an appropriate number of iterations we require an idea of the distribution of median scores returned by the heuristics.

While the median heuristic can be applied to an arbitrary number $N$ of genomes, the application to the breakpoint phylogeny problem below requires the use of the median heuristic for up to three genomes. Hence our experimental work focused on the case of three input genomes.

We choose ten arbitrary triples of genomes from the data set, the only criterion affecting our choice of triples being that some of the triples should contain closely related species while others should contain distantly related species. The composition of the triples is given in Table 2.

The priority insertion algorithm and the fast priority insertion algorithm were run 10000 times on each triple. Our aim was to estimate how many iterations are necessary to be adequately confident that the best score returned is equal to, or very close to, the best score that would be returned by 10000 iterations.

For each heuristic and each triple, we determined the best median score, the number $f(< \epsilon)$ of times the heuristic returned a score that is within $\epsilon = 0.05$ (normalised) breakpoints of this best score. This gives an estimate $p_\epsilon$ of the probability of getting to within $\epsilon$ of the best score in a particular iteration. The probability of getting within $\epsilon$ of the best in $k$ iterations is then $1 - (1 - p_\epsilon)^k$. Therefore, to be 99% confident of performing enough iterations we need to perform

$$k_\epsilon = \frac{\log(0.01)}{\log(1 - p_\epsilon)}$$

iterations.

In Table 2 we summarise our experimental results for the median heuristic algorithms.

TABLE 2 goes near here.

First note that there is little difference in the scores obtained by the two heuristics. In triple 5, the priority insertion heuristic returns a score slightly less than the fast priority insertion—but the situation is reversed in the next triple. The number of iterations required is close to the same between each heuristic. Since the fast priority takes a lot less time to compute, we adopt it as the median heuristic in later analyses.

In most of the triples, 50 iterations more than suffices to be 99% sure of obtaining a score close to the score that would be obtained after 10000 iterations. There are two exceptions: triple 2 and triple 6, where neither heuristic frequently locates a near-optimal median. We would hope that detailed examination of these two examples might lead to future improvements of the heuristics. In the meantime we need to increase the number of iterations performed.

# 5 A heuristic for the fixed topology breakpoint phylogeny problem

## 5.1 The median iterate heuristic for breakpoint phylogeny

The heuristic for the median problem developed in Section 4 is at the heart of our approach to breakpoint phylogeny.

The basic idea is to assign random genomes to the internal vertices of the tree, then iteratively improve the tree by changing one genome at a time. For a given vertex $v$, we apply the median algorithm to the three genomes assigned to neighbours of the vertex. If the median score of the median returned by

the algorithm improves on the sum of the existing edge lengths from $v$ to its neighbours, then we assign the new genome to $v$. By repeated passes through the tree we will eventually obtain a local optimum. We can then repeat the whole process many times, each time initializing the genomes randomly then iterating until reaching a local optimum.

We implemented two strategies for passing through the tree at each iteration. In the first, we traversed the tree in a postorder traversal, finishing the iteration with an assignment to the root vertex. In the second version we performed two passes at each iteration, a post-order traversal followed by a pre-order traversal. The double pass approach was motivated by the desire to communicate order information as quickly downward through the tree as upward through the tree. The experimental performances of both strategies are described in the following section.

The double pass iterative improvement heuristic is outlined in the following algorithm. Refer to Section 2.3 for notation and definitions.

**Median iterate heuristic for breakpoint phylogeny**

1. Let $\mathcal{X}$ be a randomly chosen valid assignment for $T$.
2. **repeat**
3. **for** each internal vertex $v$ in a postorder traversal of $T$ **do**
4. Let $u_1, u_2, u_3$ be the neighbours of $v$.
5. Let $M$ be the median genome returned by a breakpoint median heuristic applied to $\mathcal{X}(u_1)$, $\mathcal{X}(u_2)$, and $\mathcal{X}(u_3)$.
6. **If** $d(\mathcal{X}(u_1), M) + d(\mathcal{X}(u_2), M) + d(\mathcal{X}(u_3), M)$

$< d(\mathcal{X}(u_1), \mathcal{X}(v)) + d(\mathcal{X}(u_2), \mathcal{X}(v)) + d(\mathcal{X}(u_3), \mathcal{X}(v))$ **then**

7. $\mathcal{X}(v) \leftarrow M$

8. **end for**

9. **for** each internal vertex $v$ in a pre-order traversal of $T$ **do**

10. Let $u_1, u_2, u_3$ be the neighbours of $v$.

11. Let $M$ be the median genome returned by a breakpoint median heuristic applied to $\mathcal{X}(u_1)$, $\mathcal{X}(u_2)$, and $\mathcal{X}(u_3)$.

12. **If** $d(\mathcal{X}(u_1), M) + d(\mathcal{X}(u_2), M) + d(\mathcal{X}(u_3), M)$
$< d(\mathcal{X}(u_1), \mathcal{X}(v)) + d(\mathcal{X}(u_2), \mathcal{X}(v)) + d(\mathcal{X}(u_3), \mathcal{X}(v))$ **then**

13. $\mathcal{X}(v) \leftarrow M$

14. **end for**

15. **until** convergence conditions reached.

**end.**

There are a number of possibilities for convergence conditions. In our analysis of eukaryote mitochondrial data (Section 6.2) we simply repeated this loop 50 times, later observing that at this point there had been no, or very little, improvement in the tree length for some time.

Finally, the randomness introduced in the selection of initial genomes necessitates repeated calls to the median iterate heuristic. Each time, a new random, initial assignment is chosen. We made 120 calls to the heuristic when analysing the eukaryote mitochondrial data (see below, Section 6.2).

### 5.2 Selecting phylogenies

When the number of genomes is small, we can evaluate each possible tree individually, then select the tree with the minimum breakpoint phylogeny length. However when the number $N$ of genomes is large, as in the mitochondrial genome data set, an exhaustive tree search becomes infeasible. At present, we use the breakpoint phylogeny algorithms to discriminate between three competing phylogenetic hypotheses. There is clearly scope for future work in the simultaneous construction of phylogenies and ancestral gene orders.

## 6 Experimental results: estimating the phylogeny of early eukaryotes

### 6.1 Three evolutionary hypotheses

Applying the normalised induced breakpoint distance (eqn. **??**) to all pairs of genomes in Table 1 for which we could establish a usable gene order (indicated by an entry under "Genes"), resulted in a matrix which we used for the initialization of our phylogenetic analysis. Two genomes, *Malawimonas* and *Monosiga*, manifested uniformly high values indicating random genome order with respect to all the other genomes, and were thus dropped from the analysis. The matrix remaining was submitted to two distance matrix analyses, neighbor-joining and the Fitch-Margoliash procedure, which both produced the tree in Figure 2.

FIGURE 2 GOES NEAR HERE.

These initial results indicate that the mitochondrial gene orders, as compared by our normalised induced breakpoints measure, contain a clear phylogenetic signal. The red algae form a monophyletic group; so do the stramenopiles. The large jakobid mitochondrial genomes, thought to most closely represent the ancestral form (Lang et al. 1997), group with other early-branching lineages. In addition, the ciliates group with the stramenopiles, a configuration which is sometimes seen in phylogenies constructed with single gene sequences. Only the plants and green algae, which, according to a great diversity of scientific evidence, should also form a monophyletic group, do not seem to have conserved sufficient commonality in their mitochondrial gene orders for them to be grouped together. This is, however, consistent with the rapid evolution of these orders known to occur among other green algae, such as those listed in Table 1.

More detailed phylogenetic techniques, to be discussed in the following sections, do not do any better in reconstructing the plant-green algae group. Indeed, the noise caused by the inclusion of the green algal genomes has a distorting effect on other parts of the tree, particularly the ciliate branching and possibly the remoteness of *Marchantia* from the red algal group. (There is a consensus that the plant-green algae group shares some common ancestry with the red algae.) For further investigation of protist phylogeny, then, we reduced our data set through the elimination of the *Prototheca* gene order, which seems highly derived, as well as that of *Pedinomonas*, which has a very small number

of genes.

FIGURE 3 GOES NEAR HERE

The distance matrix methods applied to the 18 remaining genomes produced the phylogeny in Figure 3. This is almost identical to what is obtained from the tree in Figure 2 by simply deleting the *Prototheca* and *Pedinomonas* branches. In the ensuing sections, we will refer to the evolutionary history implied by this tree as Hypothesis $H_1$.

Note that the what remains of the plant-green algae group still does not group with the red algae. Thus for the more detailed analyses below, we postulate another hypothesis, $H_2$, as represented in Figure 4.

FIGURE 4 GOES NEAR HERE

Finally, we construct a more speculative hypothesis, $H_3$, which would place the cryptophyte *Rhodomonas salina* on the lineage leading to the red and green algae on plants, based on the possibility that flattened cristae may be monophyletic (Figure 5). This also has the effect that *Naegleria gruberi*, the sole representative of the organisms with discoidal cristae, now branches earlier, more in line with the ancient divergence thought to have occurred with this group.

FIGURE 5 GOES NEAR HERE

## 6.2 Practical tree length evaluation

We then applied the breakpoint phylogeny algorithms to compare the three competing phylogenetic hypotheses. The aim of our experiment was to examine the relative efficiencies of the single pass and double pass breakpoint phylogeny algorithms and to determine whether there was significantly more support for one of the tree hypothesis $H_1$, $H_2$, or $H_3$. The results of any heuristic algorithm must always be interpreted with a degree of caution, as must any exact algorithm for any simplified model.

We ran both the single pass and double pass breakpoint phylogeny algorithms on each of the three trees $H_1$, $H_2$ and $H_3$. We used the fast priority insertion algorithm to evaluate median genomes at the internal nodes, calling the algorithm 50 times for each median calculation. After each initial random assignment of genomes to internal nodes we made 50 passes through the tree. In the case of the double pass algorithm, this corresponds to 100 single passes through the tree. Finally, we restarted the single pass and double pass algorithms 120 times with different assignments of initial internal genomes.

The results of the experiment are summarised in Table 3.

TABLE 3 GOES NEAR HERE.

After 120 calls to the median iterate heuristics, both with single and double passes, the shortest tree length found was for tree $H_1$. In fact, every call to the median iterate heuristics for tree $H_1$ returned a length that was shorter than all calls to the median iterate heuristics for trees $H_2$ and $H_3$.

To estimate whether the result was caused by variability in the heuristic, we calculated the mean and standard deviations of the tree lengths returned by each of the 120 calls to the heuristic. While it is not clear which distribution is best suited for the calculation of confidence evidence, what is clear is that for any reasonable choice of distribution, the minimum tree length returned for $H_1$ is significantly shorter than those returned for $H_2$ or $H_3$.

That our analysis selects the same tree produced by distance matrix methods suggests that this tree indeed represents the phylogenetic signal contained in the induced breakpoint data and, we believe, the mitochondrial gene orders themselves. In the case of the green plants, for example, these orders do not seem sufficiently conserved to reflect their common ancestry with the red algae.

It is of interest that this same methodology – normalisation of the number of induced breakpoints as input to phylogenetic methods – does a much better job on the phylogeny of green plants and other plastid-containing protists when applied to the chloroplast genome instead of the mitochondrial genome (Sankoff et al. 2000b).

A final observation from our experimental study is that there is little difference in performance between the single and double pass heuristics—except that the double pass heuristic runs, of course, twice as slowly.

# 7 Conclusions

This work highlights the potential of the induced breakpoint method for genomes with differing gene sets. The fact that we were able to obtain phylogenies as clear as many that are derived from sequence comparison attests to the amount of phylogenetic information which resides in gene order.

The results of our provisional phylogenetic analysis may be summarized as follows:

- The stramenopiles cluster together, usually but not always monophyletically, and there is a tendency for the ciliates to be a sister group.
- The jakobids, and other early branching (as previously revealed by sequence analysis) protists group together.
- The red algae group together.
- The phylogenetic signal is too weak to group all plants and green algae together. And after discarding the most highly diverged green algae from the analysis, though the green plants are close to the other phyla with flattened cristae mitochondrial genomes in the phylogeny, they do not all form a monophyletic group.
- The phylogenetic relationships at the earliest level – among the stramenopiles, alveolates and other tubular cristae mitochondrial genomes, and among the flattened, discoidal and tubular groups, remain uncertain, awaiting further mitochondrial sequences which fit the criteria for inclu-

sion in our analyses.

Our conclusions regarding the breakpoint median and breakpoint phylogeny algorithms may be summarized:

- The fast priority insertion heuristic is an effective and rapid construction algorithm for breakpoint median heuristics. The loss of accuracy in comparison with the priority insertion algorithm is negligible, but the gains in speed are immense.

- Breakpoint parsimony tree lengths can be effectively estimated using a median iterate algorithm with a single pass through the tree. Uncertainty due to the variability of tree lengths returned can be addressed by making repeated calls and determining confidence statistics.

There are a number of directions for further work. The first problem is the development of a practical tree search method. There are a number of obstacles. Though we have dramatically improved the computation time required to evaluate a tree since the work in (Sankoff et al. 2000a), this factor still restricts the total number of tree evaluations that can be made. Further improvements in efficiency could be expected from an optimized implementation of the algorithm. Currently we are developing a method where, instead of focusing on one median at a time, genes are inserted into all internal genomes simultaneously. This will hopefully improve both speed and accuracy.

The variability in scores resulting from the use of a heuristic can still make it difficult to discriminate between neighbouring trees with close tree lengths.

This might possibly be resolved by application of the breakpoint phylogeny lower bound of (Bryant 2000) to determine whether a global optimum has been obtained.

Though it is a useful exercise to consider gene order only, a more accurate approach might take into account both gene order and gene complement (i.e. the genes present in each genome) in a single measure.

Finally, this work underscores the interest inherent in the evolution of the mitochondrial genomes, especially at the earliest times.

## Acknowledgments

Research supported by grants to the authors from the Natural Sciences and Engineering Research Council (NSERC) and the Medical Research Council of Canada. Thanks to C.J. O'Kelly for helpful comments. DS and BFL are Fellows, GB an Associate, and DB a postdoctoral fellow in the Evolutionary Biology Program of the Canadian Institute for Advanced Research. MD holds an NSERC summer studentship.

## References

[1] Blanchette, M., Kunisawa, T. and Sankoff, D. 1999. Gene order breakpoint evidence in animal mitochondrial phylogeny. *Journal of Molecular Evolution* 49, 193-203.

[2] Bryant, D. 2000. A lower bound for the breakpoint phylogeny problem. In: Giancarlo, R. and Sankoff, D. (eds.) *Combinatorial Pattern Matching. 11th Annual Symposium. Lecture Notes in Computer Science* 1848. Springer Verlag, New York, pp 241-254.

[3] Burger, G., Saint-Louis, D., Gray, M.W. and Lang, B.F. 1999. Complete sequence of the mitochondrial DNA of the red alga *Porphyra purpurea*: cyanobacterial introns, and shared ancestry of red and green algae. *Plant Cell* 11, 1675-1694.

[4] Caprara, A. 1999. Formulations and hardness of multiple sorting by reversals. In: Istrail, S., Pevzner, P.A. and Waterman, M. (eds) *Proceedings of the 3rd Annual International Conference on Computational Molecular Biology (RECOMB 99)*. ACM, New York, pp 84-93.

[5] Gray, M.W., Burger, G. and Lang, B.F. 1999. Mitochondrial evolution. *Science* 283, 1476-81.

[6] Gray, M.W., Lang, B.F., Cedergren, R.J., Golding, B., Lemieux, C., Sankoff, D., Turmel, M., Brossard, N., Delage, E., Littlejohn, T.G., Plante, I., Rioux, P., Saint-Louis, D., Zhu, Y. and Burger, G. 1998. Genome structure and gene content in protist mitochondrial DNAs. *Nucleic Acids Research* 26, 865-878.

[7] Hannenhalli, S. 1996. Polynomial-time algorithm for computing translocation distance between genomes. *Discrete Applied Mathematics* 71, 137-151.

[8] Hannenhalli, S., Chappey, C., Koonin, E.V. and Pevzner, P.A. 1995. Genome sequence comparison and scenarios for gene rearrangements: a test case. *Genomics* 30, 299-311.

[9] Hannenhalli, S. and Pevzner, P.A. 1995a. Transforming cabbage into turnip. (polynomial algorithm for sorting signed permutations by reversals). In: *Proceedings of the 27th Annual ACM-SIAM Symposium on the Theory of Computing*, pp 178-189.

[10] Hannenhalli, S. and Pevzner, P.A. 1995b. Transforming men into mice (polynomial algorithm for genomic distance problem). In: *Proceedings of the IEEE 36th Annual Symposium on Foundations of Computer Science*, pp 581-592.

[11] Korab-Laskowska, M., Rioux, P., Brossard, N., Littlejohn, T.G., Gray, M.W., Lang, B.F. and Burger, G. 1998. The Organelle Genome Database Project (GOBASE). *Nucleic Acids Research* 26, 139-146.

[12] Lang, B.F., O'Kelly, C.J. and Burger, G. 1998a. Mitochondrial genomics in protists, an approach to probing eukaryotic evolution. *Protist* 149, 313-322.

[13] Lang, B.F., Seif, E., Gray, M.W., O'Kelly, C.J. and Burger, G. 1998b. A comparative genomics approach to the evolution of eukaryotes and their mitochondria. *Journal of Eukaryote Microbiology* 46, 320-326.

[14] Lang, B.F., Gray, M.W. and Burger, G. 1999. Mitochondrial genome evolution and the origin of the eukaryotes. *Annual Review of Genetics* 33, 351-97.

[15] Lang, B.F., Burger, G., O'Kelly, C.J., Cedergren, R.J., Golding, B., Lemieux, C., Sankoff, D., Turmel, M. and Gray, M.W. 1997. An ancestral mitochondrial DNA resembling a eubacterial genome in miniature. *Nature* 387, 493-497.

[16] Paquin, B., Laforest, M.-J., Forget, L., Roewer, I., Zhang, W., Longcore, J. and Lang, B.F. 1997. The Fungal Mitochondrial Genome Project: evolution of fungal mitochondrial genomes and their gene expression. *Current Genetics* 31, 380-395.

[17] Paquin, B. and Lang, B.F. 1996. The mitochondrial DNA of *Allomyces macrogynus*: the complete sequence from an ancestral fungus. *Journal of Molecular Biology* 255, 688-701.

[18] Pe'er, I. and Shamir, R. 1998. The median problems for breakpoints are NP-complete. Electronic Colloquium on Computational Complexity Technical Report 98-071, http://www.eccc.uni-trier.de/eccc

[19] Reinelt, G. 1991. *The traveling salesman - computational solutions for TSP applications.* Springer Verlag.

[20] Sankoff, D. 1992. Edit distance for genome comparison based on non-local operations. In: Apostolico, A., Crochemore, M., Galil, Z. and Manber, U. (eds) *Combinatorial Pattern Matching. 3rd Annual Symposium. Lecture Notes in Computer Science* 644. Springer Verlag, New York, pp 121-135.

[21] Sankoff, D. and Blanchette, M. 1997. The median problem for breakpoints in comparative genomics. In: Jiang, T. and Lee, D.T. (eds) *Computing and*

*Combinatorics, Proceeedings of COCOON '97. Lecture Notes in Computer Science* 1276. Springer Verlag, New York, pp 251-263.

[22] Sankoff, D., Blanchette, M. 1998. Multiple genome rearrangement and breakpoint phylogeny. *Journal of Computational Biology* 5, 555-570.

[23] Sankoff, D., Bryant. D., Deneault, M., Lang, B.F., Burger, G. 2000a. Early eukaryote evolution based on mitochondrial gene order breakpoints. *Proceedings of the 4th Annual International Conference on Computational Molecular Biology (RECOMB 2000)* (R.Shamir, S.Miyano, S.Istrail, P. Pevzner, M. Waterman eds.) ACM, New York, pp 254–262.

[24] Sankoff, D., Leduc, G., Antoine, N., Paquin, B., Lang, B.F. and Cedergren, R.J. 1992. Gene order comparisons for phylogenetic inference: Evolution of the mitochondrial genome. *Proceedings of the National Academy of Sciences USA* 89, 6575-6579.

[25] Sankoff, D., Sundaram, G. and Kececioglu, J. 1996. Steiner points in the space of genome rearrangements. *International Journal of the Foundations of Computer Science* 7, 1-9.

[26] Sankoff, D., Deneault, M., Bryant, D., Lemieux, C. and Turmel, M. 2000b. Chloroplast gene order and the divergence of plants and algae, from the normalized number of induced breakpoints. In: Sankoff, D. and Nadeau, J.H. (eds.) *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment and the Evolution of Gene Families* Kluwer, Amsterdam, to appear.

[27] Turmel, M., Lemieux, C., Burger, G., Lang, B.F., Otis, C., Plante, I. and Gray, M.W. 1999. The complete mitochondrial sequences of *Nephroselmis olivacea* and *Pedinomonas minor*: two radically different evolutionary patterns within green algae. *Plant Cell* 11, 1717–1729.

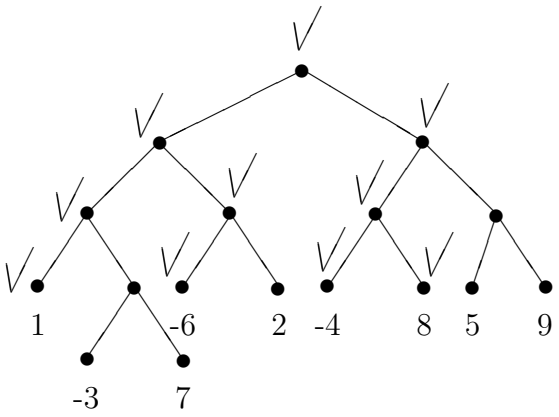


Figure 1: A data struture for quickly calculating sucessors in an induced genome. The genome is $\langle 1, -3, 7, -6, 2, -4, 8, 5, 9, 1\rangle$ and $X = \{1, 6, 4, 8\}$. We place a tick on nodes that have descendents in $X$ (positive or negative).

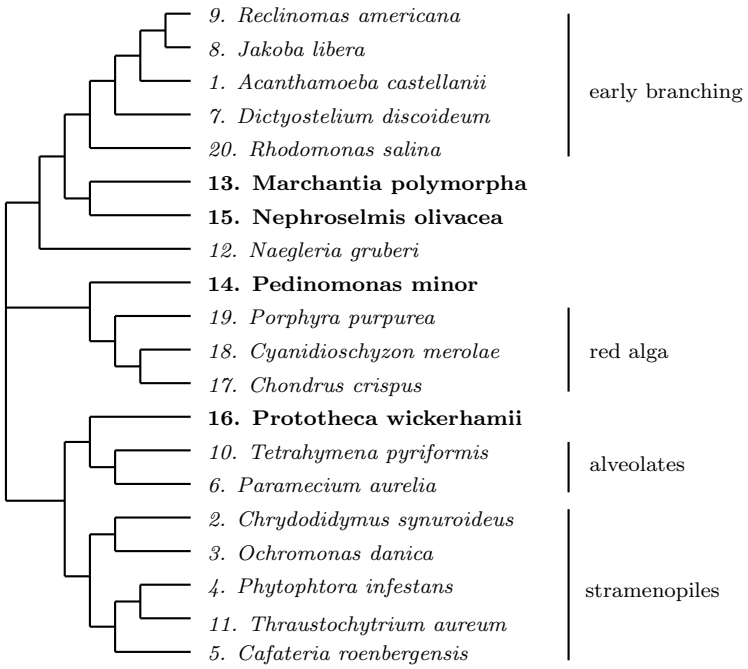


Figure 2: Distance-matrix analysis of protist evolution. Lengths of branches not to scale. Root is near *Reclinomonas* (9). Stramenopiles and alveolates cluster together. Jakobids and other early branching protists group together. Green algae and plants (bold) scattered throughout the phylogeny.

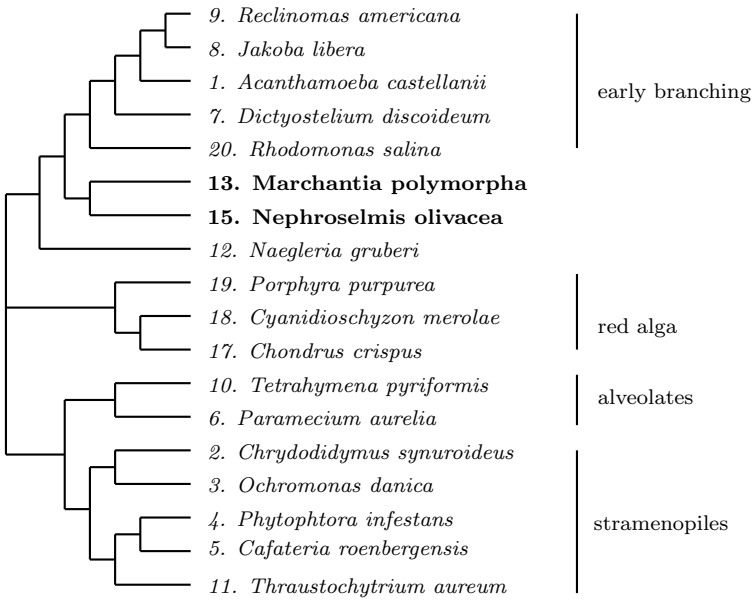


Figure 3: Distance tree without *Prototheca* and *Pedinomonas*. Hypothesis $H_1$.

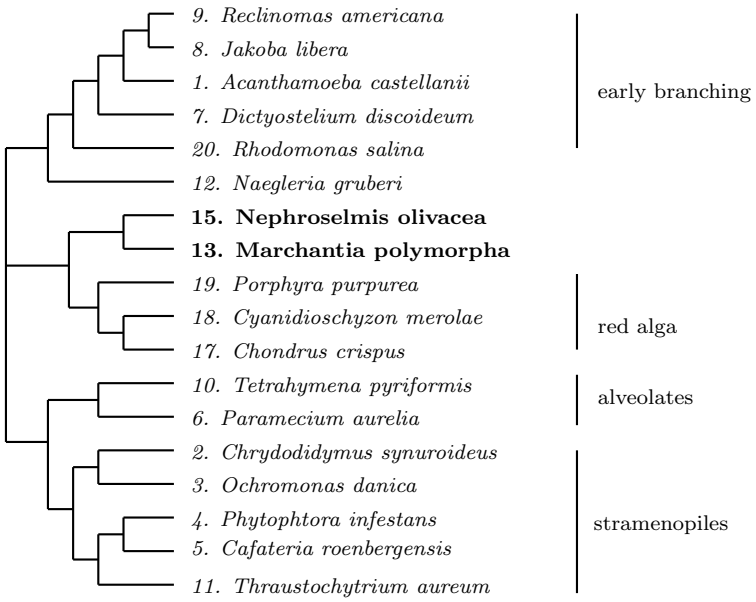


Figure 4: Hypothesis $H_2$ grouping plants and green algae with red algae

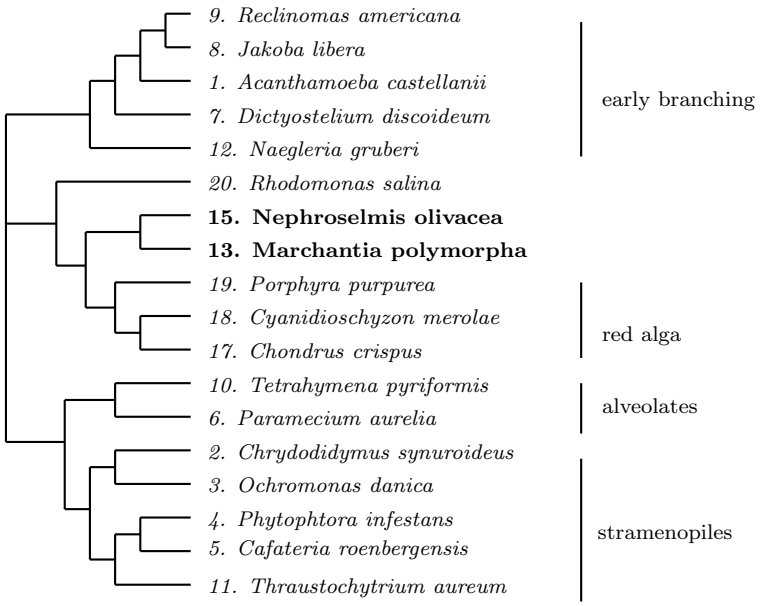


Figure 5: Hypothesis $H_3$ grouping all organisms with flattened cristae

| Organism (Accession number) | Classification | | Genes | tRNAs |
|---|---|---|---|---|
| TUBULAR CRISTAE | | | | |
| 1. *Acanthamoeba castellanii* (U12386) | lobose amoeba | | 56 | 16 |
| 2. *Chrysodidymus synuroideus* | stramenopile | (synurophyte) | 53 | 19 |
| 3. *Ochromonas danica* | stramenopile | (chrysophyte) | 57 | 22 |
| 4. *Phytophthora infestans* | stramenopile | (oomycete) | 60 | 23 |
| 5. *Cafeteria roenbergensis* | stramenopile | (bicosoecid) | 54 | 22 |
| 6. *Paramecium aurelia* (X15917) | alveolate | (ciliate) | 39 | 3 |
| 7. *Dictyostelium discoideum* (D16466) | slime mold | | 48 | 17 |
| 8. *Jakoba libera* | jakobid | | 88 | 24 |
| ∗ *Plasmodium falciparum* (M76611) | alveolate | (apicomplexan) | | |
| ∗ *Plasmodium yoelii* (M29000) | alveolate | (apicomplexan) | | |
| 9. *Reclinomonas americana* (AF007261) | jakobid | | 97 | 26 |
| 10. *Tetrahymena pyriformis* (AF160864) | alveolate | (ciliate) | 43 | 7 |
| † *Theileria parva* (Z23263) | alveolate | (apicomplexan) | | |
| 11. *Thraustochytrium aureum* | stramenopile | (labyrinthulid) | 53 | 19 |
| DISCOIDAL CRISTAE | | | | |
| ‡ *Malawimonas jakobiformis* | malawimonad | | 68 | 25 |
| 12. *Naegleria gruberi* | heterolobosean | | 61 | 17 |
| † *Leishmania tarentolae* (M10126 ) | trypanosomatid | | | |
| † *Trypanosoma brucei* | trypanosomatid | | | |
| FLATTENED CRISTAE | | | | |
| 13. *Marchantia polymorpha* (M68929) | land plant | | 69 | 24 |
| § *Arabidopsis thaliana* (Y08502) | land plant | | | |
| 14. *Pedinomonas minor* ( AF116775) | green alga | | 21 | 8 |
| 15. *Nephroselmis olivacea* | green alga | | 65 | 26 |
| 16. *Prototheca wickerhamii* (U02970) | green alga | | 63 | 26 |
| ∗ *Chlamydomonas eugametos* (AF008237) | green alga | | | |
| ∗ *Chlamydomonas reinhardtii* (U03843) | green alga | | | |
| ∗ *Chlorogonium elongatum* (Y13644) | green alga | | | |
| 17. *Chondrus crispus* (Z47547) | red alga | | 50 | 23 |
| 18. *Cyanidioschyzon merolae* (D89861) | red alga | | 59 | 22 |
| 19. *Porphyra purpurea* (AF114794) | red alga | | 55 | 24 |
| 20. *Rhodomonas salina* | cryptophyte | | 67 | 27 |
| ‡ *Monosiga brevicollis* | choanoflagellate | | 50 | 22 |
| ‡ fungal,animal | fungal,animal | | | |

Table 1: Sequenced mitochondrial genomes. Data from gene maps in GOBASE (Korab-Laskowska et al. 1998). Gene numbers affected by the exclusion of some duplicate genes (see text). Organisms numbered 1-20 used in analysis. Other organisms excluded for the following reasons: ∗ fragmented rRNA genes, † too few genes, ‡ no gene order resemblances with (other) protist mitochondrial genomes, § trans-spliced genes.

| | Genomes | Min. score | $f(< 0.05)$ | $k_{0.05}$ | actual iter. |
|---|---|---|---|---|---|
| PRIORITY INSERTION | | | | | |
| 1 | 4,5,11 | 1.387 | 1532 | 28 | 7 |
| 2 | 1,5,10 | 1.087 | 530 | 85 | 39 |
| 3 | 2,9,16 | 1.339 | 1062 | 42 | 1 |
| 4 | 17,18,19 | 1.127 | 1780 | 24 | 6 |
| 5 | 4,12,20 | 1.471 | 2302 | 18 | 3 |
| 6 | 8,9,19 | 1.015 | 177 | 258 | 76 |
| 7 | 12,15,20 | 1.089 | 1862 | 23 | 2 |
| 8 | 3,11,19 | 1.373 | 1973 | 21 | 1 |
| 9 | 3,7,8 | 1.115 | 1637 | 26 | 9 |
| 10 | 7,12,17 | 1.252 | 1790 | 24 | 6 |
| FAST PRIORITY INSERTION | | | | | |
| 1 | 4,5,11 | 1.387 | 1498 | 29 | 4 |
| 2 | 1,5,10 | 1.087 | 515 | 88 | 3 |
| 3 | 2,9,16 | 1.339 | 1037 | 43 | 22 |
| 4 | 17,18,19 | 1.127 | 1776 | 24 | 2 |
| 5 | 4,12,20 | 1.473 | 2234 | 19 | 2 |
| 6 | 8,9,19 | 1.014 | 133 | 344 | 26 |
| 7 | 12,15,20 | 1.089 | 1811 | 24 | 2 |
| 8 | 3,11,19 | 1.373 | 1935 | 22 | 1 |
| 9 | 3,7,8 | 1.115 | 1622 | 27 | 5 |
| 10 | 7,12,17 | 1.252 | 1623 | 27 | 19 |

Table 2: Performance of the priority insertion and fast priority insertion heuristics. Triples taken from the mitochondrial genome data set. We performed 10000 iterations of each heuristic on each triple. $f(< 0.05)$ is the number of iterations returning a score that is within 0.05 (normalised) breakpoints of the minimum. $k_{0.05}$ is the estimated number of iterations required to get within 0.05 of the minimum after 10000 iterations. actual iter. is the number of the first iteration within 0.05 of the minimum.

| Hypothesis: | $H_1$ | | $H_2$ | | $H_3$ | |
|---|---|---|---|---|---|---|
| Passes: | Single | Double | Single | Double | Single | Double |
| min. length | 8.14 | 8.125 | 8.517 | 8.517 | 8.547 | 8.563 |
| max. length | 8.436 | 8.396 | 8.853 | 8.867 | 8.928 | 8.894 |
| av. length | 8.265 | 8.253 | 8.651 | 8.643 | 8.691 | 8.697 |
| st. dev. | 0.063 | 0.056 | 0.059 | 0.056 | 0.068 | 0.065 |

Table 3: Application of the median iteration heuristics to the gene order data or sequenced mitochondrial genomes. Both single pass and double pass heuristics were called 120 times. The statistics presented are those for the tree lengths returned by each call to each heuristic.