Inference of a Phylogenetic Tree: Hierarchical Clustering versus Genetic Algorithm

Glenn Blanchette, Richard O'Keefe, and Lubica Benuskova

Computer Science Department, Otago University, New Zealand
{gblanche,ok,lubica}@cs.otago.ac.nz

Abstract. This paper compares the implementations and performance of two computational methods, hierarchical clustering and a genetic algorithm, for inference of phylogenetic trees in the context of the artificial organism *Caminalcules*. Although these techniques have a superficial similarity, in that they both use agglomeration as their construction method, their origin and approaches are antithetical. For a small problem space of the original species proposed by Camin (1965) the genetic algorithm was able to produce a solution which had a lower Fitch cost and was closer to the theoretical evolution of *Caminalcules.* Unfortunately for larger problem sizes its time cost increased exponentially making the greedy directed search of the agglomerative clustering algorithm a more efficient approach.

Keywords: phylogenetics, genetic algorithm, agglomerative clustering, Caminalcules.

1 Introduction

Phylogenetic trees are a diagrammatic representation of the evolutionary relationships between taxonomic units (TU). They can be inferred using a variety of methods (Cotta 2002, Felsenstein 2004): supervised optimization methods based on cost (Guindon 2003), unsupervised distance methods (Russo 1996) and probability models of evolution.

Recent research has centered on probability models (Sullivan 2005) and clustering. Tamura (2004) has encouraged the use of clustering algorithms even for large taxonomies and these neighbour-joining methods form the basis of an integrated software package 'MEGA' (Tamura 1994). By contrast, genetic algorithms seem to have been neglected or only implemented for small problems (Lewis 1989). Moreover, Hudson and Bryant (2006) have suggested that tree structures do not have the flexibility required to represent complex phylogenies and that networks are better suited.

Joseph Camin invented the *Caminalcules* species in 1965 as a means of evaluating phylogenetic inference methods (Camin 1965). They are used in many universities for teaching phylogenetics (Gendron 2011, Ausich 2011). There are 29 'currently existing' species and 48 'fossil' species. Each individual has 85 morphological characteristics: a variety of Boolean, nominal and ordered numeric attributes. Sokal published an extensive four-part article on the *Caminalcules* in 1983 (Sokal 1983). A cladistic approach to phylogenetic analysis is usually based on evolutionary relationships, which include idiosyncratic heuristic information from the 'fossil' records. However, a

M. Thielscher and D. Zhang (Eds.): AI 2012, LNCS 7691, pp. 300–312, 2012.

strict computational approach to the phylogenetic analysis can be applied using just morphological differences (phenetics) between the 'existing' species.

This paper aims to directly compare two different computational methods of inference in the construction of a phylogenetic tree for the *Caminalcules* species. Although other authors have considered which varieties of genetic algorithm are best suited for phylogenetic inference (Cotta 2002), as far as we are aware this is the first direct comparison of a clustering and a genetic algorithm approach for inference of the same *Caminalcules* phylogeny.

2 Implementations

Agglomerative hierarchical clustering was chosen as a distance method and a genetic algorithm as an optimization method. Both applications were implemented using object-oriented design in C++ and share support classes representing a phylogenetic (binary) tree: an abstract parent *TU_Node* class that is extended in its children *OTU_Node* (observed taxonomic unit) and *HTU_Node* (hypothetical taxonomic unit). The data members for each of the *Node* classes include: an identifier and the three tree pointers (parent, left and right child).

The *Caminalcules* data vectors were initially read from a text file of character vectors into bit fields. Non-applicable characteristics 'x' were represented as binary 0001, '0' was represented as binary 0010, '1' was represented as binary 0100, '2' was represented as binary 1000, and so on.

Two methods were implemented for calculating the tree cost metric: the Fitch/Hamming distance and a combined Fitch/Manhattan distance to better adjust for continuous characteristics. The former method measures the minimum number of substitutions required to change one string into the other. The later method finds the difference between two bit fields using intersection and converts the difference into a scaled ordinal distance. The result is a semi-quantitative cost. The advantage of this metric is that continuous measurements such as 'flange-length' are treated as ordinals. The disadvantage is that nominal characteristics such as 'top-of-head' (depressed, flat or crested) may be treated incorrectly if there are more than two values. The cost metric was converted to a fitness metric: fitness = constant – cost. The range of values for the Fitch costing was experimentally determined and a constant value sufficient to maintain a positive value for the fitness was used.

The phylogenetic trees produced by the applications were output in two standard formats: Newick format (wikipedia.org 2012) as text and GraphViz Dot format (graphviz.org 2011) for a diagrammatic tree.

2.1 Agglomerative Clustering

The implementation of bottom-up agglomerative clustering is from O'Keefe [2006]. The algorithm is deceptively simple: starting with the matrix for distances between single OTU clusters it successively merges the closest clusters by forming a tree with an HTU node as the new root, it re-calculates the distances from the new cluster to all

the existing clusters and reduces the matrix by moving the last cluster (row/column) up until all the clusters are included in a single tree. The matrix is symmetric around the diagonal, so only one half of the matrix needs to be calculated.

O'Keefe recommends using an average linkage criterion, since this includes a contribution from each of the vectors within a cluster. The maximal and minimal linkage criteria use only one vector difference, which may not be representative of the overall inter-cluster distance. The basis of the clustering algorithm is reported in Sokal [1983] and Murtagh [1984]. A segment of pseudo-code for the clustering agglomeration is shown in Figure 1.

```
  Find the two closest existing clusters a & b
       size ab = size a + size b
  //Calculate all distances to the new cluster
  for every existing cluster x {
       calculate its distance to the new cluster
       based on average linkage: distance ab-x =
       (dist a-x * size a + dist b-x * size b) / size ab
  }
  //Make new tree from cluster - join a & b with an HTU
  tree a = new HTU_Node (HTU_node, tree a, tree b);
  //Move last cluster up the matrix, last row/col ->
row/col b

  for every column in the matrix {
       make the distance at [col][b] & [b][col] =
       the distance at [col][last]
       and then clear the distance at [col][last]
  }
  tree b = tree last;
       size b = size last;
```

Fig. 1. Pseudo-code for the Clustering Algorithm agglomeration

2.2 Genetic Algorithm

The base process of any genetic algorithm is identical, it repetitively selects and breeds individuals within the population. Only the individual chromosome representation, which underlies breeding and the fitness/selection metric, are specific to the particular problem.

In this case a chromosome representation outlined by O'Keefe [2009] was used. Here, the chromosome consists of pairs of sub-tree indices. The agglomerative process for building the new subtrees is very similar to the clustering algorithm. Initially Pairs of OTUs are successively merged with a new HTU at the root to form the new sub-trees, as the process continues the indices in the chromosome will apply to progressively larger trees until a single tree is formed. The difference between the algorithms is that the sub-tree indices for agglomeration by the genetic algorithm are

chosen randomly on creation of the chromosome, whereas the clustering algorithm chooses clusters dynamically based on shortest distance. A segment of pseudo-code for the tree building is shown in Figure 2. Both the clustering algorithm and the genetic algorithm use the same overloaded constructor in the support class.

```
tree_build() { //method in Phylogeny Class
     //using an array of sub-trees
     //start at the bottom, from the leaves
     bottom = number of OTUs - 1
     for each allele in the chromosome {
     //get the pair of node indices p & q
     p = chromosome->x
     q = chromosome->y
     //combine these as a new sub-tree
     tree p = new HTU_Node (bottom, tree p, tree q)
     //reduce the array
     bottom – 1
     }
     //make a new node at the root
     tree 0 = new HTU_Node (bottom, tree 0, tree 1);
}
```

Fig. 2. Pseudo-code for the Genetic Algorithm agglomeration

The chromosomal representation used is robust and has closure because:

- It is able to represent every tree in the problem space,
- It is well formed and does not require any correction to the chromosome after crossover or mutation
- And it is modular in that changes produced by crossover or mutation are locally limited.

The framework of the genetic algorithm itself is more complex than a single execution of clustering; it requires a separate support class to manage the individuals within the population. The *Phylogeny* class keeps track of the raw and scaled fitness of each individual and its chromosome representation.

Two different selection methods were tried. The more complex method, proportional selection based on scaled fitness over the whole population was not a very strong driver for selection. Most of the random trees created were in the same fitness range, 400 – 800. The simpler method of just sorting the population by fitness and transfering a proportion into the next generation (truncation selection) was more effective. This is a common lay-person's (or 'breeder') method but suffers from the impurity of the attribute selection (Voigt & Muhlenbein, 1996), ie. it may reduce the fitness for some attributes.

Most of the methods in the *Phylogeny* class are just wrappers for methods within the Node classes: Newick and GraphViz Dot print methods and Fitch costing methods are shared with the clustering algorithm. However, the crossover and mutation methods are only required for the genetic algorithm and use an iterative loop to permit

multiple crossover or mutation. Both of these methods operate directly on the chromosome, they do not involve any re-arrangement of the constructed trees. They conserve the chromosome structure because of the sound original choice of representation (O'Keefe 2009).

3 Results

3.1 Agglomerative Clustering

Agglomerative clustering using the Fitch metric and the average linkage criterion, produced a constant result, which appeared to have an optimal cost, 277 by Fitch, but 291 by the combined metric. The tree is given in Newick format and GraphViz Dot format, in Figure 3.

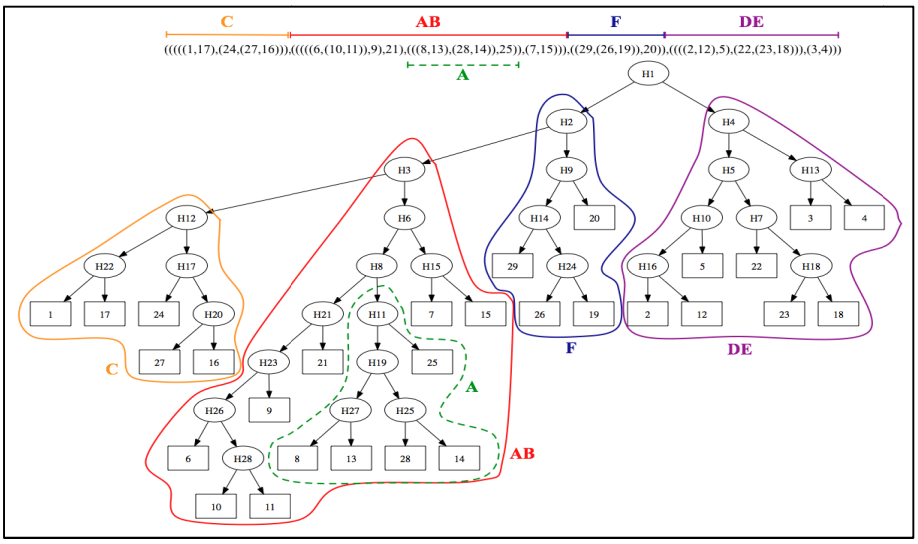

Fig. 3. Phylogenetic tree infered by clustering. Fitch Cost 277

Agglomerative clustering using the combined metric did not produce any marked improvement. There were four major branch groups (Figure 3) as illustrated by their OTUs, which were identical in both trees.

Group AB: ((((6, (10, 11)), 9), 21),
(((8, 13), (14, 28)), 25)), (7, 15))
Group C: ((1, 17), (24, (16, 27)))
Group DE: (((2, 12), 5), (22, (18, 23)), (3, 4))
Group F: ((29, (19, 26)), 20)

The groups are labeled in this way to match the classification published by Sokal (1983).

3.2 Genetic Algorithm

One of the time-consuming phases of using any genetic algorithm is tuning. The results of trials for tuning the parameters are not presented. However, in summary the optimal values (and range) were:

- Generations: 900 (100 – 18,000)
- Population size: 40 individual trees (20 – 100)
- Selection proportion: 50% (20 – 60)
- Crossover rate: > 0.25 (0.2 – 0.8)
- Mutation rate: >0.35 (0.1 – 0.5)

The low crossover and high mutation rates are unusual. However, there was good propagation of 'beneficial' chromosomes in early generations even at this low crossover rate. A high mutation rate was required to achieve solutions close to the theoretical trees of Sokal (1983) and of lowest Fitch cost. Epistasis may have influenced the high mutation rate. Our mutation algorithm assumed that each attribute was independent of the others; this was in fact not the case. Although Camin and Sokal (1965) have not explicitly defined dependencies within the attributes, implicit dependencies are present in the morphological drawings.

Experiments with two-point crossover and mutation were not beneficial, with a larger number of runs being required to reach a result close to that of agglomerative clustering. So most of the experimental testing was done with just single point mutations and single crossovers.

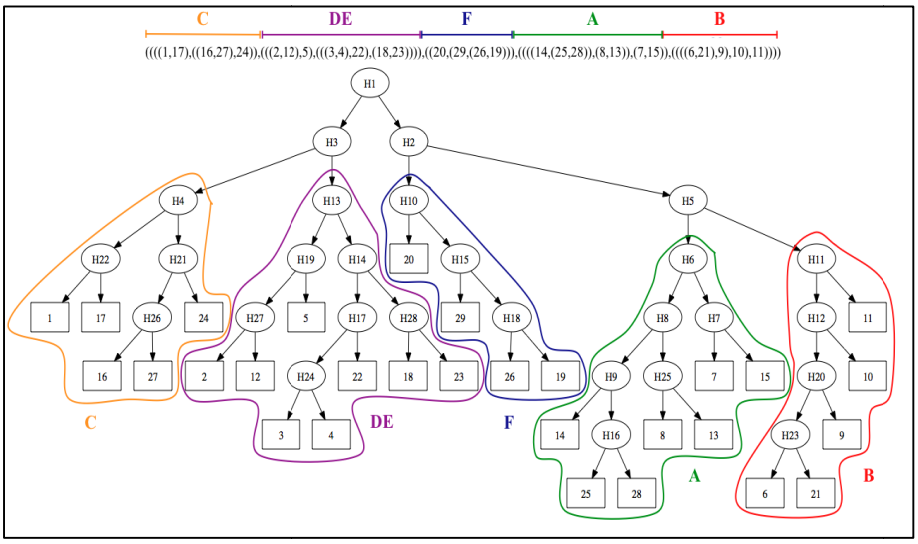

Fig. 4. Phylogenetic tree infered by genetic algorithm. Fitch Cost 266

After much trial and error, a phylogenetic tree with a Fitch cost of 266 was discovered; see Figure 4. This tree was superior to the tree produced by the clustering algorithm. In terms of the previously identified tertiary level branch groupings it is identical for groups C, DE and F but group BA is divided into two subtrees:

Group A: ((8, 13), (25, 28), 14)
Group B: ((((6, 9), 10), (11, 21)), (7, 15))
Group C: (((1, 17), (16, 24)), 27)
Group DE: (((2, 12), 5), ((3, 4), 22), (18, 23))
Group F: (29, (19, (20, 26)))

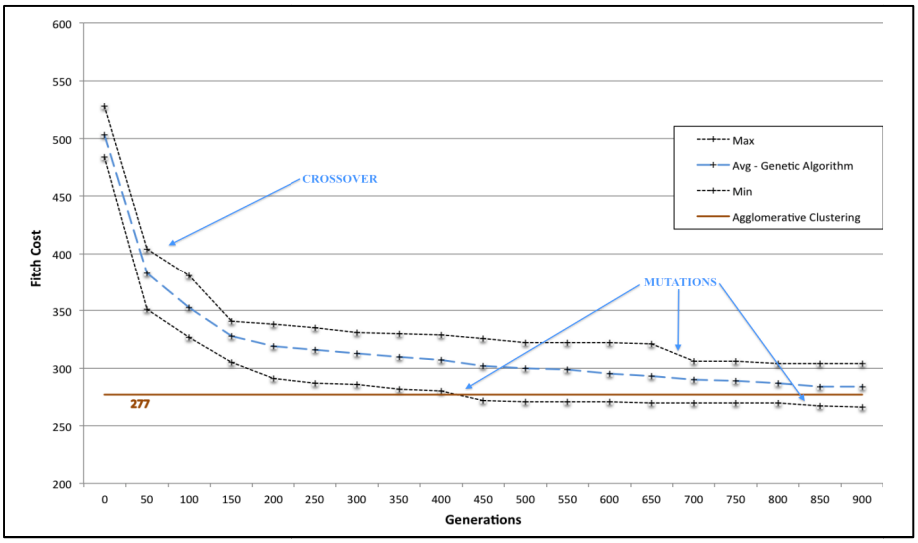

Fig. 5. 'Evolution' of a phylogeny using the genetic algorithm

It is interesting to look at the evolution of this solution over time. Figure 5 shows the data collected from the genetic algorithm compared to the static cost of the agglomerative clustering. Initially it can be seen that the fittest chromosomes are quickly spread through the whole population by crossover and then the process of improvement plateaus at some steady state. Over time 'beneficial' mutations occur and there is a further smaller change in the population. Average solutions for the genetic algorithm are worse than for clustering but the best genetic solution had a lower cost.

4 Discussion

Even for this artificial biological problem: that phylogenetic trees can be inferred by a number of different approaches, phenetic and cladistic, suggests that no single tree is likely to be perfect. A 'history' of the *Caminalcules* has been published by Sokal (1983), which includes the complete cladistic classification.

According to Sokal the *Caminalcules* evolved into five genera over 19 periods, four lines terminating in fossils. The current genera are:

Group A: (((14, (25, 28)), (18, 13)), (17, 15))
Group B: ((((6, 21), 9), 10), 11)
Group C: ((1, 17), ((16, 27), 24))
Group DE: (((2, 12), 5), (((3, 4), 22), (18, 23)))
Group F: (20, (29, (26, 19)))

A binary phylogenetic tree was adapted from Sokal's classification using hypothetical taxonomic units to represent the fossil species and internal node branch points, see Figure 6.

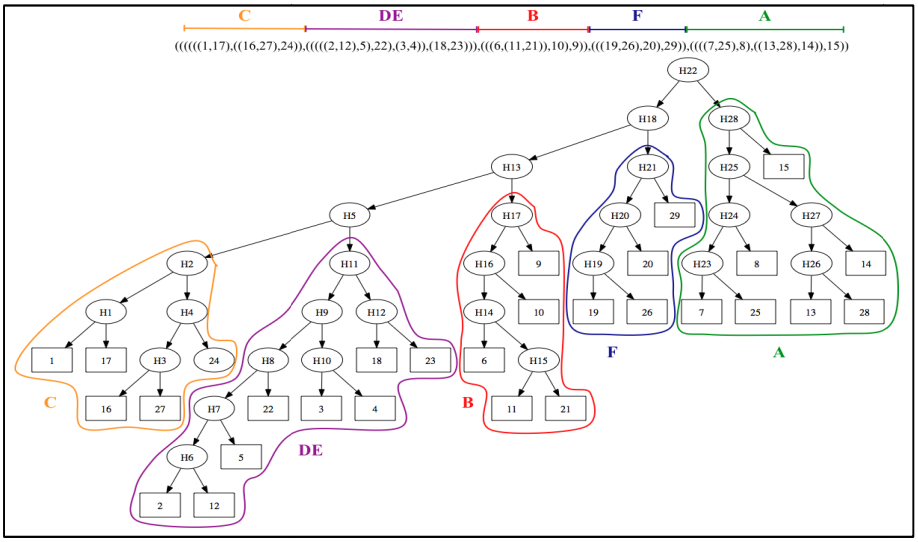

Fig. 6. Phylogenetic tree adapted from Sokal. Fitch Cost 270

It should be noted that the heuristics Sokal used for construction of his cladistic phylogeny are quite different to those used for the current implementations. Sokal's original phylogeny is not a binary tree; it does not require that each pair of OTUs have a common HTU ancestor. In addition Sokal's cladistic tree has current OTUs as branch points and their own leaves. Because the implementations presented are purely computational they have been strictly constructed; as binary trees, with OTUs as leaves and HTUs as internal nodes.

If Sokal's classification is compared to the phenetic tree produced by the genetic algorithm (Figure 4), the distribution of species within each genus is identical. Considering the very different construction of the trees and that the current implementation did not have the evidence of the 48 'fossils' to work with, the results from the genetic algorithm seem very respectable. Why does the agglomerative clustering algorithm fail to find the best solution? One answer could be that it is a

greedy algorithm; it chooses only the best current subtrees (at the lower levels). It misses the optimization that occurs at a higher level in the phylogeny.

These two implementations of a phenetic classification (clustering and genetic algorithm) may seem similar, they both construct trees by aggregation of subtrees, but the similarity is superficial because most trees are commonly constructed by aggregation. In fact, the approaches come from entirely different branches of science. Agglomerative clustering is an engineering approach. It aims to provide a single fast concise solution based on an exact as possible mathematical measurement of distance. This exact measurement intentionally drives the process of construction. The genetic algorithm comes from biology. It is inexact; most of its 'solutions' are poor. No plan is applied during construction of the trees. In fact, it is not aiming to find a solution, only explore the problem space. There is no driving force, except chance. It is only after construction that selection and time filter the results; by its nature it is a slow process.

After testing the algorithms on the base *Caminalcules* phylogeny, four ideas emerged for further investigation and comparison of these phylogenetic inference methods.

4.1 Including the Fossil Evidence

The first idea was: could the algorithms be extended to produce a cladistically based phylogenetic tree? Given all the available fossil evidence (48 additional taxons) could the tree published by Sokal (1983) be duplicated exactly? It is perhaps easier to see how the genetic algorithm could be adapted to use this additional information. A separate randomly shuffled array of fossil HTUs could be added to the implementation. As each chromosome pair is aggregated, the next of these random HTUs would be used as the parent node. This would leave 20 terminating fossil lines but still might achieve a result close to Sokal's. Crossover of the chromosome representation would require some structural fix. However, for the agglomerative clustering algorithm, adaption would be more difficult. The fossil HTUs could not just be added to the original distance matrix, as there has to be some distinction between them and the OTUs which cannot be parent nodes. A separate distance matrix for the HTUs as a mid-point between clusters might be needed, but this would greatly increase the complexity of the algorithm.

4.2 Extending the Time for Evolution

The second idea was hinted at previously, in Section 3 Results. If the genetic algorithm was given sufficient time would mutation produce further improvement in the phylogenetic tree? This idea was interesting enough to check immediately. A change was made in the statistical sampling, to look for a particularly 'lucky' run of the algorithm (Fitch cost <= 270) and to continue it for up to 18,000 generations. After hundreds of runs the tree in Figure 7 was discovered. It had a Fitch cost of only 262, thus confirming our hypothesis. The last reduction in the cost of this tree was produced by a mutation after 10,000 generations.

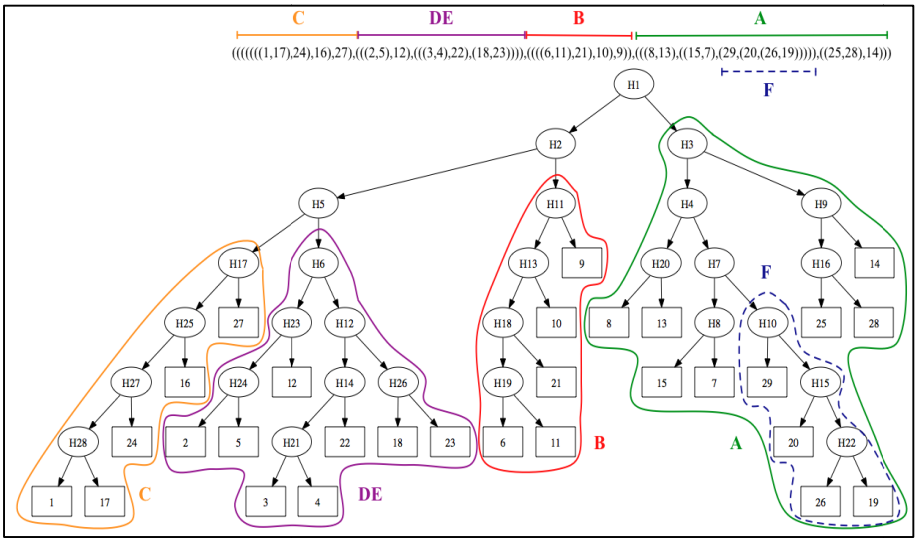

Fig. 7. Phylogenetic tree infered by genetic algorithm. Fitch Cost 262

Surprisingly, this tree had branch groupings, which were quite different from the earlier clustering result (Figure 5) and from the previous results for the genetic algorithm (Figure 6). Particularly genus F was included as a sub-tree of genus A:

Group AF: (((8, 13), ((15, 7),
(F): (29, (20, (26, 19))))),
((25, 28), 14)))
Group B: ((((6, 11), 21), 10), 9)
Group C: ((((1, 7), 24), 16), 27)
Group DE: (((2, 5), 12), (((3, 4), 22), (18, 23))))

So this tree was not as close to the theoretically correct tree produce by Sokal (1983). Unfortunately, while it is possible to find trees with lower objective morphological cost, they may not always be cladistically as acceptable. It would be difficult to build idiosyncratic cladistic parameters into an objective/fitness function.

4.3 Scaling the Problem Size

The third idea was to examine the complexity of the genetic algorithm, scaling the problem size by inventing further *Caminalcules* species. Considering there are 85 attributes with at least two values; billions of un-utilized combinations are available. A further 50,000 unique *Caminalcules* species were generated. The time cost and solution quality of the inferred phylogenetic trees for both the agglomerative clustering and genetic algorithm were tested against increasing numbers of OTUs, using the optimal tuning parameters for the original problem. Figure 8 shows that while the time complexity of the genetic algorithm is almost linear, it is inferior to the almost constant time cost of the clustering algorithm. Further, Figure 9 shows that the quality of the

genetic algorithm solutions were deteriorating in comparison to those of the clustering algorithm because the algorithm was not optimally tuned for these larger problems. Satisfactory tuning of the genetic algorithm for some of these larger problem sizes was attempted, by for example proportionately increasing the population size, in addition to experimenting with the other parameters. This experimentation was unable to improve the quality of the genetic algorithm solutions but did result in an exponential increase in the time cost.

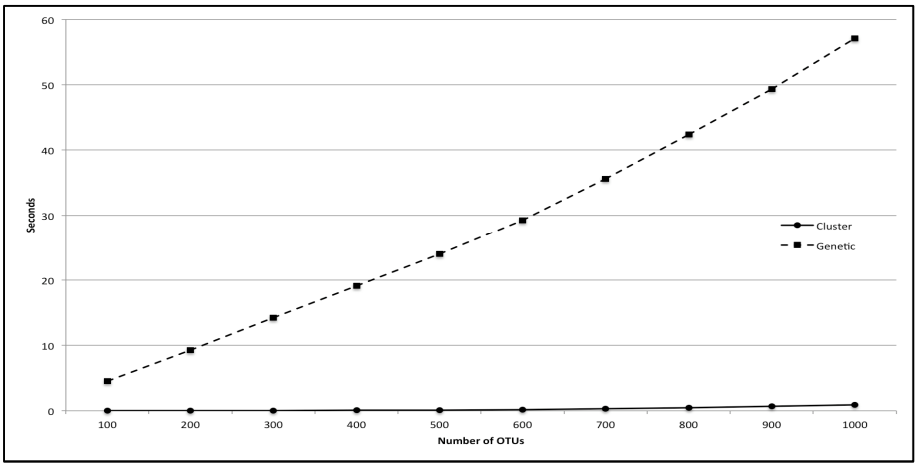

Fig. 8. Comparison of real time costs for the genetic and clustering algorithms

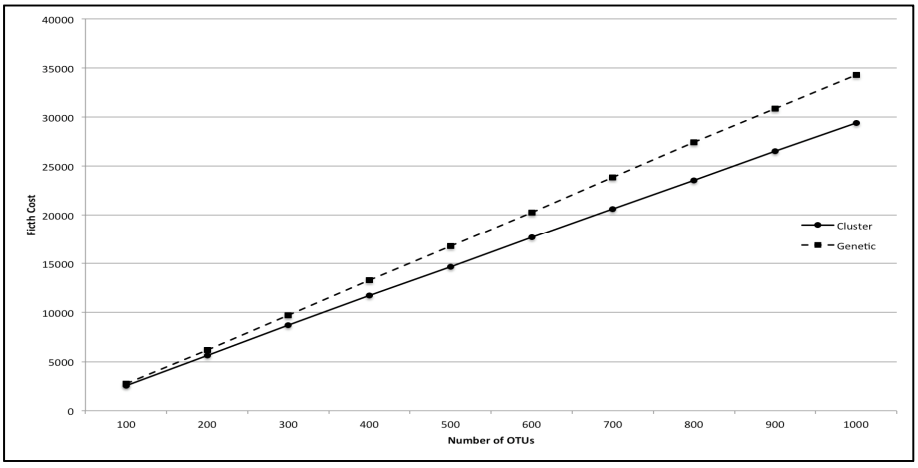

Fig. 9. Comparison of solution quality for the genetic and clustering algorithms

4.4 Extending the Comparisons

If space permitted, it would have been possible to include the results of comparisons for other data sets. However, based on our current experiments, we felt these were likely to show the same difficulties with scaling of the genetic algorithm and would not have altered the conclusion.

Currently, the more modern approaches to phylogenetic inference are based on probabilistic models of evolution; see Ronquist & Huelsenbeck (2003) and Drummond & Rambaut (2007) implemented in the software "MrBayes" and "Beast", respectively. It would be very interesting and informative to compare the phylogeny trees for *Caminalcules* inferred based on a probabilistic model of their evolution, but this work is beyond the scope of the present article.

5 Conclusion

Although these techniques have a superficial similarity, in that they both use agglomeration as their construction method, their origins and approaches are antithetical. The genetic algorithm permits more thorough exploration and for a small problem space, such as the 29 original *Caminalcules* species, it achieves a solution which has a lower Fitch cost and is closer to the theoretical evolution proposed by Sokal (1983). However there is a time cost for this exploration and tuning the algorithm for larger problems is exponentially less efficient than agglomerative clustering. A directed search even though greedy and unable to guarantee an optimal solution, may be advantageous.

References

1. Ausich, W.: Caminalcule Phylogenetic Exercise. Ohio State University (2011), http://www.serc.carleton.edu/NAGTWorkshops/paleo/activities
2. Drumond, A., Rambaut, A.: BEAST: Bayesian Evolutionary Analysis by Sampling Trees. BMC Evolutionary Biology 7, 214–221 (2007)
3. Camin, J., Sokal, R.: A Method for Deducing Branching Sequences in Phylogeny. Evolution 19, 311–326 (1965)
4. Cotta, C., Moscato, P.: Inferring Phylogenetic Trees Using Evolutionary Algorithms. In: Guervós, J.J.M., Adamidis, P.A., Beyer, H.-G., Fernández-Villacañas, J.-L., Schwefel, H.-P. (eds.) PPSN 2002. LNCS, vol. 2439, pp. 720–729. Springer, Heidelberg (2002)
5. Felsenstein, J.: Infering Phylogenies. Sinauer Associates, Sunderland (2004)
6. GraphViz Dot, http://www.graphviz.org/News.php
7. Gendron, R.: Caminalcule Evolution Lab. Indiana University Pennsylvania (2011), http://www.nsm1.nsm.iup.edu
8. Guindon, S., Gascuel, O.: A Simple, Fast, and Accurate Algorithm to Estimate Large Phylogenies by Maximum Likelihood. Systematic Biology 52(5), 696–704 (2003)
9. Hudson, D., Bryant, D.: Application of Phylogenetic Networks in Evolutionary Studies. Molecular Biology and Evolution 23(2), 254–267 (2006)
10. Lewis, P.: A genetic algorithm for maximum likely-hood phylogeny inference using nucleotide sequence data. Molecular Biology and Evolution 15(3), 277–283 (1989)

11. Murtagh, F.: Complexities of Hierarchical Clustering Algorithms: the State of the Art. Computational Statistics Quarterly 1, 101–113 (1984)
12. Newick Format, http://www.wikipedia.org/wiki/Newick_format
13. O'Keefe, R.: Cluster Analysis, notes for Cosc348. Otago University (2006), http://www.cs.otago.ac.nz/cosc348/phylo/phylo5.html
14. O'Keefe, R.: Laboratory 07, notes for Cosc348. Otago University (2009), http://www.cs.otago.ac.nz/cosc348/labs/lab07.html
15. Ronquist, F., Huslsenbeck, J.: MrBayes 3: Bayesian Phylogenetic Inference Under Mixed Models. Bioinformatics 19(12), 1572–1574 (2003)
16. Russo, C., Takezaki, N., Lei, M.: Efficiences of genes and different tree-building methods in recovering a known vertebrate phylogeny. Molecular Biology and Evolution 13(3), 525–536 (1996)
17. Sokal, R.: A Phylogenetic Analysis of the Caminalcules. I. The Data Base. Systematic Zoology 32(2), 159–184 (1983)
18. Sokal, R.: A Phylogenetic Analysis of the Caminalcules. II. Estimating the True Cladogram. Systematic Zoology 32(2), 185–201 (1983)
19. Sokal, R.: A Phylogenetic Analysis of the Caminalcules. III. Fossils and Classification. Systematic Zoology 32(3), 248–258 (1983)
20. Sokal, R.: A Phylogenetic Analysis of the Caminalcules. IV. Congruence and Character Stability. Systematic Zoology 32(3), 259–275 (1983)
21. Sokal, R., Michener, C.: A Statistical Method for Evaluating Systematic Relationships. University of Kanas Science Bulletin 38, 1409–1438 (1985)
22. Sullivan, J., Joyce, P.: Model Selection in Phylogenetics. Annual Review of Ecology, Evolution and Systematics 36, 445–466 (2005)
23. Tamura, K., Nei, M., Kumar, S.: Prospects for infering very large phylogenies by using neighbour-joining method. Procedings of the National Academy of Sciences 101(30), 11030–11035 (2004)
24. Tamura, K., Nei, M., Kumar, S.: MEGA: Molecular Evolutionary Genetics Analysis software for microcomputers. Bioinformatics 10(2), 189–191 (1994)
25. Voigt, H., Muhlenbein, H.: Erroneous Truncation Selection – A Breeder's Decision Making Perspective (1996), http://www.muehlenbein.org/errtrun96.pdf