Evolving Better Multiple Sequence Alignments

Luke Sheneman and James A. Foster

Initiative for Bioinformatics and Evolutionary Studies (IBEST)
Department of Computer Science
University of Idaho, Moscow, ID 83844-1010, USA
+1 208.885.7062
{sheneman,foster}@cs.uidaho.edu

Abstract. Aligning multiple DNA or protein sequences is a fundamental step in the analyses of phylogeny, homology and molecular structure. Heuristic algorithms are applied because optimal multiple sequence alignment is prohibitively expensive. Heuristic alignment algorithms represent a practical trade-off between speed and accuracy, but they can be improved. We present EVALYN (EVolved ALYNments), a novel approach to multiple sequence alignment in which sequences are progressively aligned based on a *guide tree* optimized by a genetic algorithm. We hypothesize that a genetic algorithm can find better guide trees than traditional, deterministic clustering algorithms. We compare our novel evolutionary approach to CLUSTAL W and find that EVALYN performs consistently and significantly better as measured by a common alignment scoring technique. Additionally, we hypothesize that evolutionary guide tree optimization is inherently efficient and has less time complexity than the commonly-used neighbor-joining algorithm. We present a compelling analysis in support of this scalability hypothesis.

1 Introduction

Aligning multiple DNA or amino acid sequences is an extremely important task in modern biology. Researchers apply multiple sequence alignment (MSA) to a diverse set of problems. MSA is used to find positional similarity across distinct biological sequences as a first step in inferring sequence homology and the evolutionary relationships between organisms. MSA is used in gene identification and discovery and in identifying similarity in molecular structure and function. Among other practical applications, MSA plays a critical role in the diagnoses of genetic disease and the development of modern pharmaceuticals.

A sequence alignment is composed of two or more biological sequences which are arranged such that similar positions within the sequences are grouped (aligned). Alignments are often represented as a two-dimensional matrix where rows are sequences and columns are sequence positions. A good alignment is one which maximizes the positional similarity across all columns and all sequences. Alignments are constructed by inserting or deleting sequence segments in order to group similar characters into columns. Since it is impossible to know whether positions have been in

K. Deb et al. (Eds.): GECCO 2004, LNCS 3102, pp. 449–460, 2004.

serted or deleted relative to one another, these insertions/deletions are simply called *indels*. Indels can be represented as *gaps* in a sequence. Placing a single gap in a sequence causes the remainder of the sequence to shift by one position. Gaps placed in the optimal positions will result in an alignment where positional similarity is maximized as shown in Fig. 1.

```
A-CTTCAACTAAGT-ATTG-AATAAA-CT-GCTTAGATATATCTCCAAATTATTAGCTATCGCTTAT-GGATTATATTAC
ACCTTTA--TAAGTCATTG-ACT-AAGCTCGCCTAGAT---------AATTACCCGCTATCG---ATATCC-CCTATTAC
-CC-TCAACTAAGT-ATTG-AATAAAG---GCTTAGATATATCTCCAAATTACTAGCTAT----TATATCCTCATAT---
```

Fig. 1. Here is an example of a multiple sequence alignment of three DNA sequences in which gaps are denoted by the dash (-) character

Before the advent of alignment algorithms, researchers laboriously aligned multiple sequences by hand. This task was both error-prone and time-consuming. In the 1970's, researchers developed simple pairwise alignment algorithms based on dynamic programming (DP) and proved that they produce optimal alignments with respect to any given scoring system. [1, 2]. Although these algorithms extend easily to the simultaneous and optimal alignment of multiple sequences, they are NP-Hard [3] and have a time-complexity of $O(L^N)$ where L is the average length of the sequences being aligned and N is the number of sequences being aligned. Using DP, the simultaneous optimal alignment of more than a handful of sequences is prohibitively expensive. As a result, heuristic approaches trade quality for speed.

Progressive multiple sequence alignment is the most common heuristic [4] and is depicted in Fig. 2. In traditional progressive MSA, a distance matrix is formed by using DP to compute the optimal edit distance between all possible combinations of sequence pairs. A clustering algorithm such as neighbor-joining [5] takes a distance matrix as input and deterministically constructs a *guide tree* based on these distances, grouping closely related sequences prior to more divergent sequences. Once a guide tree has been constructed, sequences are progressively pairwise aligned in the order dictated by the guide tree. Closely related sequences are aligned prior to more distant sequences. Progressive MSA avoids the computationally intractable problem of the simultaneous alignment of multiple sequences by instead performing incremental pairwise alignments.

However, traditional progressive MSA has fundamental problems. Most importantly, after sequences are pairwise aligned, any inserted gaps in that pairwise alignment become immutable, and subsequent alignments with other sequences cannot retroactively add additional information to improve previously aligned sequences. This is a form of error propagation, also known as "*once a gap, always a gap*". The guide tree has a direct qualitative impact on this error propagation, as the amount of error is heavily dependent on the order in which sequences are progressively aligned. Since guide tree construction algorithms such as neighbor-joining are greedy and starting-point dependent, they are easily trapped in local optima, often resulting in suboptimal multiple alignments. We hypothesize that an evolutionary algorithm is

better able to avoid entrapment in local optima and will therefore perform better than neighbor-joining in constructing good guide trees. Better guide trees result in better multiple sequence alignments.

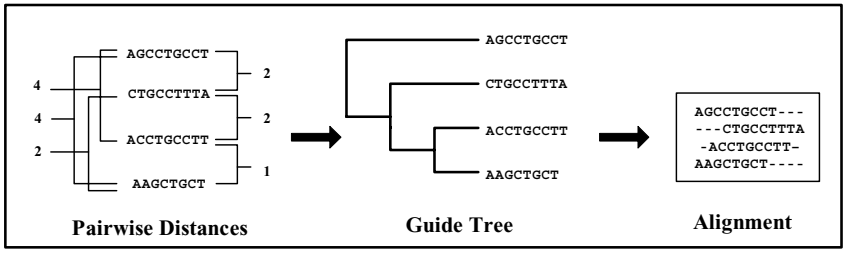

Fig. 2. In traditional progressive MSA, clustering algorithms use computed pairwise edit distances to construct a guide tree which clusters similar sequences prior to divergent sequences. A guide tree specifies an ordering of pairwise alignment operations which construct a complete multiple sequence alignment

Additionally, for large datasets, neighbor-joining has a prohibitive time complexity of $O(N^3)$, where N is the number of input sequences. As researchers apply MSA to larger and larger datasets, neighbor-joining scales poorly as N grows large. It is our hypothesis that an evolutionary computational approach to guide tree construction is more scalable than neighbor-joining.

2 Previous Work

Notredame and Higgins performed the seminal work in applying a genetic algorithm (GA) to MSA with a tool known as *Sequence Alignment by Genetic Algorithm* (SAGA) [6]. SAGA evolves a population of alignments using a complex set of 22 crossover and mutation operators in an attempt to gradually improve the fitness of the alignments in the population. Providing meaningful scores for sequence alignments can be somewhat problematic, and SAGA relies on a weighted sum-of-pairs approach [7] in which each pair of sequences in an alignment is compared and scored and then the scores from all of the pairwise alignments are summed to produce a representative score for the entire alignment.

Although SAGA produces high quality results which are comparable (or sometimes better) than other popular heuristic techniques, SAGA scales poorly [6] when aligning more than 20 sequences. SAGA applies a large and overly-complex litany of crossover and mutation operators, which are dynamically scheduled via a sophisticated adaptive, self-tuning mechanism. By contrast, the GA approach outlined herein uses only one form of crossover and one mutation operator, thus simplifying the implementation and analysis of the algorithm.

Thomsen et al. [8] developed an alignment post-processing program that uses a genetic algorithm to *improve* alignments constructed by algorithms such as CLUSTAL

V [9]. A population of alignments is initialized by randomly distributing gaps throughout the individual alignments yet seeding the population with a single alignment produced by CLUSTAL V. Assuming that this CLUSTAL-derived seed was of higher quality than the randomly generated seeds, any fitness improvement in the fittest individual is, by definition, an improvement over the output of CLUSTAL V. The authors aligned as many as 71 sequences with an average length of 100 residues and arguably demonstrated a 10% quality improvement over CLUSTAL V.

Related work has been done towards the application of genetic algorithms to the problem of evolving phylogenetic trees. Most notably, [10] and [11] used genetic algorithms to evolve trees which were optimized with respect to maximum likelihood. Additional previous work has been done by [12] in inferring phylogenetic trees with respect to maximum parsimony [13].

Notably, the manipulation of tree-based data structures with genetic algorithms has been widely explored in the *genetic programming* literature [14, 15].

3 Algorithm Implementation

We present EVALYN, a novel progressive multiple sequence alignment (MSA) program that utilizes a genetic algorithm (GA) to optimize guide trees. EVALYN starts with a steady-state population of randomly constructed binary trees and iteratively optimizes this population using a combination of selection, crossover, and mutation. Guide trees are rooted binary trees. Each node in the guide tree contains an alignment which in turn contains at least one sequence. Leaf nodes have no children and contain the original input sequences.

3.1 Crossover and Mutation Operators

In each iteration of the genetic algorithm, EVALYN selects two unique parents for crossover based on an exponential distribution of relative rank. This ensures that highly fit trees are selected for crossover far more often than unfit trees, yet all trees are viable crossover candidates. Similarly, EVALYN selects a single unfit guide tree to be replaced by the offspring of a crossover operation.

EVALYN's crossover operator is depicted in Fig. 3. We implement tree crossover in a way similar to that described in GAML [10], a genetic algorithm for phylogenetic inference. Both selected parents are copied and a randomly chosen *crossover point* (internal node) is selected in the first parent. The first parent is re-rooted at the crossover point, and the remainder of the tree above the crossover point is discarded. All leaf nodes which exist in this new, smaller tree are removed from the second parent, and the second parent is *collapsed* into a typical bifurcating tree. This collapsed second parent is then *attached* to the first parent at a randomly chosen *insertion point*.

With some small probability, EVALYN mutates the child tree by performing a same-tree branch swap. Conveniently, this is implemented by performing a crossover operation on two copies of the *same* child guide tree, effectively swapping branches within the same tree.

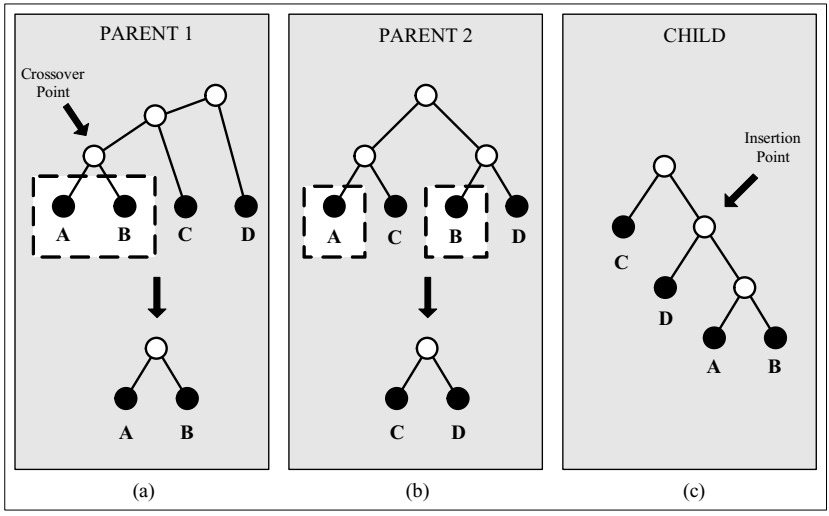

Fig. 3. Crossover is a three-step process. First, a copy of *PARENT 1* is rooted at a randomly selected crossover point and all nodes above this new root are discarded as shown in *(a)*. Next, all leaves are removed from a copy of *PARENT 2* which exist in the newly-rooted tree from *(a)*. As shown in *(b)*, leaves *A* and *B* are removed from *PARENT 2*, and the tree is collapsed to form a new bifurcating tree containing only leaves *C* and *D*. In *(c)*, the final child tree is constructed by combining the sub-trees from *(a)* and *(b)* at a randomly chosen insertion point

3.2 Measuring Guide Tree and Alignment Fitness

As shown in Fig. 4, the fitness of a particular guide tree is measured by performing progressive MSA in the order dictated by the guide tree and then scoring the resulting alignment.

We use the common sum-of-pairs score (SPS) as our scoring method as outlined in Fig. 5. In the case of protein, the evolutionary distance between every pair of residues in each column of the alignment is computed using a probabilistic residue substitution model such as PAM [16] or BLOSUM [17]. DNA is similarly handled using simple nucleotide substitution models which properly weight transitions and transversions. The SPS for the entire alignment is simply the sum of the SPS for each column in the alignment. Gaps are typically assigned large penalties, while substitutions are assigned smaller negative penalties or positive rewards. For our purposes, a higher SPS indicates a better alignment. Therefore, there is selective pressure favoring alignments with higher sum-of-pairs scores.

As in other MSA implementations, EVALYN implements *affine gap penalties*, in which the leading gap in a subsequence of contiguous gaps invokes significantly higher penalties than non-leading gaps. Affine gap penalties result in dense, contiguous gapped regions instead of sparsely distributed, isolated gaps. This affine gap model results in more biologically realistic alignments.

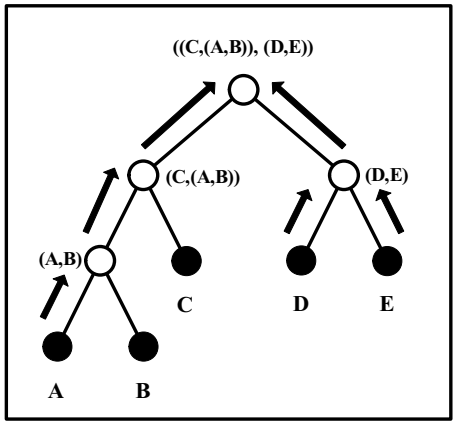

Fig. 4. Evaluating the fitness of a guide tree is accomplished by performing the progressive sequence alignment in the order dictated by the guide tree. EVALYN performs a depth-first traversal of the guide tree and aligns *A* and *B* first. Sequence *C* is then aligned to the alignment of *A* and *B* to form a 3-sequence alignment of sequences *A*, *B*, and *C*. The complete multiple sequence alignment of all sequences is performed by aligning the two alignments on either side of the root node of the guide tree. The sum-of-pairs score of the final alignment is the fitness of the alignment

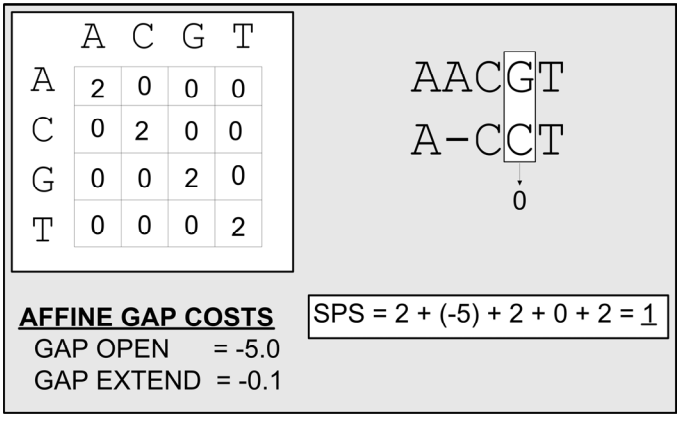

Fig. 5. In this toy example, a sum-of-pairs score (SPS) is computed for a simple pairwise alignment. A substitution matrix assigns points for nucleotide matches/mismatches and affine gap penalties are applied

4 Algorithm Analysis

We hypothesize that EVALYN has less computational complexity than neighbor-joining, and is therefore more scalable than CLUSTAL W as a function of the number

of input sequences. We briefly analyze the time complexity of EVALYN and compare it to the time complexity of the neighbor-joining algorithm used in CLUSTAL W to demonstrate support for this hypothesis. We show that with only regard to the number of input sequences, EVALYN is an $O(N)$ algorithm. By contrast, CLUSTAL W's neighbor-joining algorithm is $O(N^3)$.

EVALYN evaluates guide tree fitness by performing the progressive MSA in the order dictated by the guide tree and computing the sum-of-pairs score for the resulting alignment. At each step in the progressive alignment, we compute an optimal global pairwise alignment [1]. Each pairwise alignment has an $O(L^2)$ complexity, where L is the average length of the sequences or partial alignments being aligned. There are $N-1$ such pairwise alignments for every evaluation of a guide tree, resulting in an $O(N \times L^2)$ complexity, where N is the number of input sequences being aligned, and L is the average length of all sequences. This fitness evaluation happens once per iteration. With I iterations, EVALYN becomes an $O(I \times N \times L^2)$ algorithm. Finally, when evaluating the fitness of the initial, randomly generated population of size P, the fitness of each guide tree must be computed prior to iteration, resulting in an initial cost of $O(P \times N \times L^2)$. In typical usage, $I >> P$ and we can simplify our analysis to $O(I \times N \times L^2)$. Although EVALYN is an iterative algorithm and has large amounts of constant-time overhead in the form of the multiple I, it does have a linear time complexity with respect to the N input sequences.

Although CLUSTAL W is fast in the typical usage scenario, it performs very poorly as N grows very large (thousands of input sequences). CLUSTAL W uses neighbor-joining to construct guide trees, and neighbor-joining has been shown to possess a $O(N^3)$ time complexity [18].

The *practical* question remains as to whether or not EVALYN is capable of finding comparable or better guide trees in less time than CLUSTAL W when aligning extremely large numbers of sequences. For example, if N is very large, it may be the case that I must be similarly large in order for EVALYN to converge on guide trees which score better than those constructed via neighbor-joining. We've shown that EVALYN is $O(N)$, but how fast does I (or P) need to grow as a function of N in order to get good alignments? Future experiments will focus on characterizing this behavior.

5 Experimental Setup and Results

Our central hypothesis is that a genetic algorithm (GA) is capable of finding better guide trees than those which are constructed using traditional deterministic clustering algorithms such as neighbor-joining. To test this hypothesis, we compare EVALYN to the popular CLUSTAL W progressive MSA tool [19]. CLUSTAL W uses neighbor-joining to construct guide trees based on a computed pairwise distance matrix. Both EVALYN and CLUSTAL W compute the sum-of-pairs score for the final multiple sequence alignment, and this is used as an objective metric of alignment quality.

First, we simulated DNA sequences according to the Jukes-Cantor model [20] of sequence evolution in which transitions and transversions are equally probable. We

simulated the DNA sequences by producing a random template sequence of the desired length and then used this template sequence to generate related sequences with no more than 50% sequence divergence. In this way, we generated 10 independent sets of 50 sequences, all of which were 100 nucleotides in length. We performed 10 experimental runs of EVALYN on each of the 10 input datasets and averaged the results.

After generating our input sequences, CLUSTAL W was run with *default* parameters on each of the 10 input datasets and we recorded the final sum-of-pairs score of the output alignment. In 10 independent trials, EVALYN used the same 10 inputs and saved the best guide tree after 2500 iterations. In all cases, EVALYN used a population size of 500 guide trees, a mutation rate of 0.01, and ran for 2500 iterations as shown later in Table 1.

Where possible, EVALYN was parameterized identically to CLUSTAL W with respect to gap penalties and nucleotide substitution costs. The results from this experiment are shown in Fig. 6.

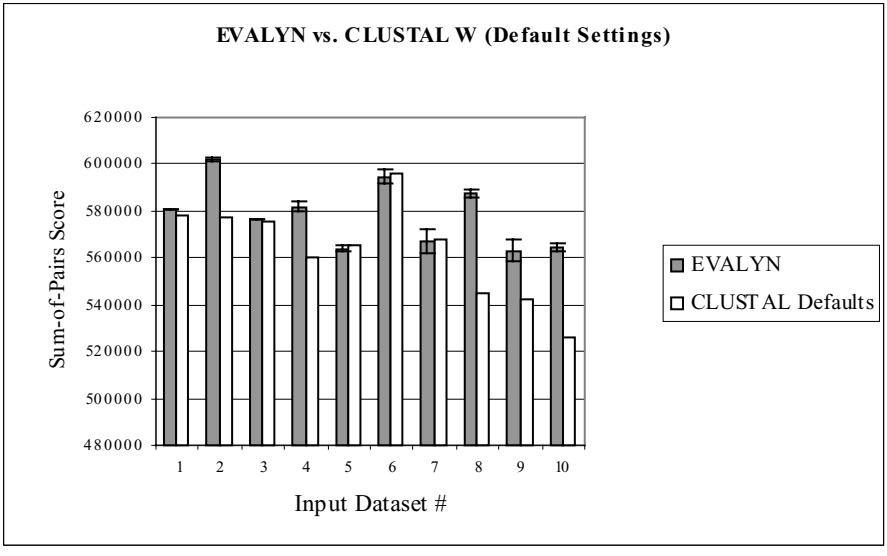

Fig. 6. CLUSTAL W run with default settings. In 70% of the runs, EVALYN produced better scoring alignments. The mean SPS and standard deviation across repeated trials is shown

We then invoked CLUSTAL W *again* for each dataset, but instead of generating its own guide trees, CLUSTAL W instead used the best guide trees produced *by EVALYN*. This novel technique removed any experimental error due to possible inconsistencies in alignment scoring between CLUSTAL W and EVALYN. We computed the mean and standard deviation of the sum-of-pairs scores across all 10 trials for each of the 10 inputs. We also calculated the standard deviation across EVALYN runs to assess the error and statistical significance of our results. Results indicate that EVALYN typically (70% of the time) outperforms CLUSTAL W when using CLUSTAL W with its default settings.

Additionally, EVALYN outperforms CLUSTAL W significantly and consistently when the two programs have identical parameterization as shown in Fig. 7. Tests of the statistical significance of all results were performed using a non-parametric Wilcoxon signed-rank test and showed that these results are statistically significant under that test.

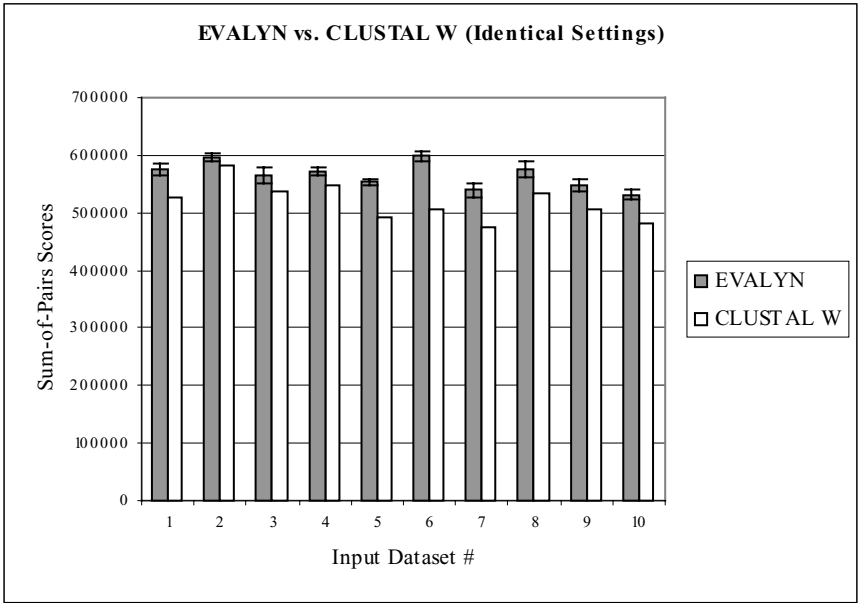

Fig. 7. CLUSTAL W and EVALYN were run with nearly identical parameterization (same substitution matrix, same gap penalties, etc.) Across all 10 input sets, EVALYN produced better guide trees which resulted in better alignments with higher sum-of-pairs scores

Table 1. Experimental configuration

Population Type	Steady-state
Population Size	500 guide trees
Crossover Rate	100% (steady-state population)
Mutation Rate	0.01
Iterations	2500
Selection Type	Rank-Based
Substitution Matrix	Matches = +1.9, Mismatches = 0
Gap Open Penalty	-15.0
Gap Extension Penalty	-6.66

By taking optimized guide trees produced by EVALYN and providing them as input to CLUSTAL W, we have found strong evidence to support our main hypothesis that guide trees evolved via a genetic algorithm produce better multiple sequence alignments than guide trees constructed using neighbor-joining.

6 Conclusions

Aligning multiple sequences of biological data is a critical aspect of modern biology. Since constructing optimal alignments is prohibitively time-intensive, popular heuristic MSA algorithms trade accuracy for speed in order to maximize their practical usefulness. We introduced EVALYN, a genetic algorithm for performing multiple sequence alignment. We demonstrated that EVALYN produces higher-scoring alignments than CLUSTAL W under the popular sum-of-pairs metric. In addition, we provided a strong analytical argument that an evolutionary computational approach to guide tree optimization as used in EVALYN is more scalable than traditional guide tree construction algorithms such as neighbor-joining.

By using a genetic algorithm to produce better guide trees, we achieved an important practical goal of producing a measurably better multiple sequence alignment program than CLUSTAL W, currently the most popular and actively used MSA program today.

7 Future Work

In future work, we will examine different metrics of alignment quality in order to show that EVALYN is able to produce biologically significant results. Although the sum-of-pairs score (SPS) for an alignment is commonly used, it has some inherent problems. First, alignments with the highest SPS are not always the most biologically significant or meaningful. Guide trees and multiple sequence alignments constructed under the SPS optimality criterion may in fact produce alignments which are meaningless when interpreted by a biologist. Finding ways of quantifying the level of biological significance of an alignment is an ongoing and active area of research. Toward this end, we intend to test EVALYN against the Benchmark Alignment dataBASE (BAliBASE) [21], which is a carefully designed set of protein alignments that were aligned and verified with elucidated or inferred structural and functional information. The BAliBASE alignments are composed of real protein sequences, and are intended to be a kind of "gold standard" by which alignment algorithms can be tested for their ability to recover biologically significant alignments. In addition to testing EVALYN against BAliBASE, we will develop new fitness functions to maximize meaningful biological signal in alignments. For example, future fitness functions may take into account predicted or elucidated secondary structure.

All phylogenetic analysis begins with multiple sequence alignment, which establishes the positional homology of the sequence data that is used for the basis of constructing and optimizing phylogenetic trees. Phylogenetic analysis is extremely dependent on the assumptions, biases, and accuracy of the initial multiple sequence alignment. State-of-the art work in phylogenetic inferencing attempts to address this by simultaneously optimizing *both* alignment *and* phylogeny. As guide trees serve as rough estimations of sequence phylogeny, EVALYN also simultaneously optimizes phylogeny and alignment. In effect, EVALYN currently performs phylogenetic tree optimization by using alignment sum-of-pairs scores as the optimality criterion. Along these lines, we will explore additional measures of guide tree fitness such as

parsimony [13] and the more statistically rigorous approach of maximum-likelihood [22].

Finally, empirically characterizing the scalability of EVALYN across different numbers, lengths, and types of input sequences is a key focus of future work. In this paper, we analyzed EVALYN to show that it has a linear time complexity with respect to the number of input sequences. However, the rate of solution convergence as a function of the number of input sequences is not yet well understood. Future experimentation will explore the relationship between population size, GA convergence properties, sequence divergence effects, and alignment quality.

Acknowledgments. The project described was supported by NIH Grant Number P20 RR16454 from the BRIN Program of the National Center for Research Resources. Experiments were run on the IBEST Beowulf cluster, which is funded in part by NSF EPS 00809035, NIH NCRR 1P20 RR16448 and NIH NCRR 1P20 RR16454. Foster was partially funded for this research by NIH NCRR 1P20 RR16448.

References

1. Needleman, S.B., Wunsch, C.D., *A general method applicable to the search for similarities in the amino acid sequences of two proteins.* Journal of Molecular Biology, 1970. **48**(3): p. 443-53.
2. Smith, T.F., Waterman, M.S., *Identification of Common Molecular Subsequences.* Journal of Molecular Biology, 1981. **48**: p. 443-453.
3. Just, W., *Computational Complexity of Multiple Sequence Alignment with SP-Score.* Journal of Computational Biology, 2001. **8**(6): p. 615-623.
4. Notredame, C., *Recent progresses in multiple sequence alignment: a survey.* Pharmacogenomics, 2002. **3**(1).
5. Saitou, N., Nei, M., *The neighbor-joining method: A new method for reconstructing phylogenetic trees.* Molecular Biology Evolution, 1987. **4**: p. 406-425.
6. Notredame, C., Higgins D.G., *SAGA: sequence alignment by genetic algorithm.* Nucleic Acids Research, 1996. **24**(8): p. 1515-1524.
7. Carillo, H., Lipman, D., *The multiple sequence alignment problem in biology.* SIAM Journal on Applied Mathematics, 1988. **48**(5): p. 1073-1082.
8. Thomsen, R., Fogel, G.B., Krink, T. *A Clustal alignment improver using evolutionary algorithms*. in *Congress on Evolutionary Computation (CEC)*. 2002. Honolulu, Hawaii.
9. Higgins, D.G., Bleasby A.J., Fuchs, R., *CLUSTAL V: improved software for multiple sequence alignment.* Comput Appl Biosci, 1992. **8**(2): p. 189-191.
10. Lewis, P.O., *A genetic algorithm for maximum-likelihood phylogeny inference using nucleotide sequence data.* Molecular Biology Evolution, 1998. **15**(3): p. 277-283.
11. Matsuda, H. *Protein phylogenetic inference using maximum likelihood with a genetic algorithm*. in *Pacific Symposium on Biocomputing*. 1996: World Scientific, London.
12. Congdon, C.B. *Gaphyl: An Evolutionary Algorithms Approach for the Study of Natural Evolution*. in *Genetic Evolutionary Computation Confernece (GECCO)*. 2002: Morgan Kaufmann.
13. Fitch, W.M., *Toward defining the course of evolution: Minimum change for a specific tree topology.* Systematic Zoology, 1971. **20**: p. 406-416.

14. Koza, J., *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. 1992: MIT Press.
15. Soule, T., Foster, J.A., *Effects of code growth and parsimony pressure on populations in genetic programming.* Evolutionary Computation, 1998. **6**(4): p. 293-309.
16. Dayhoff, M.O., Schwartz, R.M., and Orcutt, B.C., *A model of evolutionary change in proteins.*, in *Atlas of Protein Sequence and Structure*. 1978.
17. Henikoff, S., Henikoff, J.G. *Amino acid substitution matrices from protein blocks.* in *National Academy of Sciences of the USA*. 1992.
18. Howe, K., Bateman, A., Durbin, R., *QuickTree: building huge Neighbor-Joining trees of protein sequences.* Bioinformatics, 2002. **18**(11): p. 1546-1547.
19. Thompson, J.D., Higgins, D.G., Gibson, T.J., *CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice.* Nucleic Acids Research, 1994. **22**(22): p. 4673-4680.
20. Jukes, T.H., Cantor, C.R., *Evolution of protein molecules*, in *Mammalian Protein Metabolism*, H.N. Munro, Editor. 1969, Academic Press. p. 21-132.
21. Thompson, J.D., Plewniak, F., Poch, O., *BAliBASE: A benchmark alignments database for the evaluation of multiple sequence alignment programs.* Nucleic Acids Research, 1999. **27**(13): p. 2682-2690.
22. Felsenstein, J., *Evolutionary trees from DNA sequences: A maximum likelihood approach.* Journal of Molecular Evolution, 1981. **17**: p. 368-376.