An Algorithm for Constructing Hypothetical Evolutionary Trees Using Common Mutations Similarity Matrices

Peter Z. Revesz
Department of Computer Science and Engineering
University of Nebraska-Lincoln
Lincoln, NE 68588
1 571 201 5639
revesz@cse.unl.edu

ABSTRACT
In this paper, we introduce a new evolutionary tree algorithm that is based on common mutations similarity matrices instead of distance matrices.

Categories and Subject Descriptors
E.1 [**Data Structures**]: Graphs and Networks. F.2.2 [**Nonnumerical Algorithms and Problems**]: Computations on Discrete Structures.

General Terms
Algorithms, Measurement, Performance, Theory.

Keywords
Common mutations similarity matrix, data structure, distance matrix, evolutionary tree, phylogenetic tree.

1. INTRODUCTION
Constructing hypothetical evolutionary trees for a set of genetically related DNA strings, a set of proteins within a protein family, or other related identifiers is a particular clustering or classification problem that occurs frequently in computational biology in various forms [1, 2, 3, 4, 5, 6]. The reconstruction of the evolutionary trees is commonly based on some kind of *distance matrix* [1].

For example, consider the following seven DNA sequences, $S_1 \ldots S_7$, each with a length fifteen nucleotides displayed by groups of five nucleotides per column in table below:

S_1	AGCTA	CTAGT	AATCA
S_2	AGCTA	CGAGT	AATCA
S_3	ATCCA	CTAGT	ACACT
S_4	ATCCA	CTAGT	ATACT
S_5	CGGTA	TTTGT	AAGCT
S_6	CGGTT	CATCA	AATGC
S_7	AGGTA	CTTGA	AATCC

Let $S_i[k]$ denote the k^{th} nucleotide of S_i. The Hamming distance between two DNA sequences S_i and S_j each with length n, denoted $\delta(S_i,S_j)$, is defined as the number of corresponding nucleotide pairs that are different, that is, $\Sigma_{1 \le k \le n} S_i[k] \neq S_j[k]$.

BCB '13, September 22 - 25, 2013, Washington, DC, USA
ACM 978-1-4503-2434-2/13/09.

For example, $\delta(S_1,S_2) = 1$ because $S_1[7] = T$ while $S_2[7] = G$. For the above set of DNA sequences the following distance matrix, denoted D, can be generated using Hamming distances:

	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_1	0	1	5	5	6	9	4
S_2	1	0	6	6	7	8	5
S_3	5	6	0	1	8	13	8
S_4	5	6	1	0	8	13	8
S_5	6	7	8	8	0	8	5
S_6	9	8	13	13	8	0	5
S_7	4	5	8	5	5	5	0

Evolutionary tree construction algorithms generally start from such a matrix to recursive combine pairs of sequences (rows and columns) until only a single combined sequence remains. For example, the UPGMA (unweighted pair group method with arithmetic mean) [6] method would always search for the closest pairs to combine. When several pairs are equally distant, then an arbitrary choice is made. In this case, the closest pairs are S1 and S2 and S3 and S4 because $\delta(S_1,S_2) = 1$ and $\delta(S_3,S_4) = 1$. Combining the first pair gives:

	S_{12}	S_3	S_4	S_5	S_6	S_7
S_{12}	0	5.5	5.5	6.5	8.5	4.5
S_3	5.5	0	1	8	13	8
S_4	5.5	1	0	8	13	8
S_5	6.5	8	8	0	8	5
S_6	8.5	13	13	8	0	5
S_7	4.5	8	5	5	5	0

Then combining the second pair gives:

	S_{12}	S_{34}	S_5	S_6	S_7
S_{12}	0	5.5	6.5	8.5	4.5
S_{34}	5.5	0	8	13	6.5
S_5	6.5	8	0	8	5
S_6	8.5	13	8	0	5
S_7	4.5	6.5	5	5	0

Next S_{12} and S_7 would be combined because they are the closest pair of sequences according to the distance matrix. The neighbor-joining method is a more sophisticated and commonly used method that is also based on distance matrices [5].

2. EVOLUTIONARY TREE ALGORITHM

2.1 Common Mutations Similarity Matrix

As a departure from distance matrices, we introduce a common mutations similarity matrix. The motivation behind looking for common mutations is that in practice rare but shared features, such as rare mutations, are often provide useful marker of similarity among a set of closely related items. Moreover, if mutations are rare, then it may be more efficient to count their occurrences than finding the Hamming distances for long sequences. Assuming that the seven DNA sequences are related, we can find the most likely common ancestor sequence, denoted μ, as the mode of each column. If there is no most frequent nucleotide in each column, then we arbitrarily chose one of the most frequent.

S_1	AGCTA	CTAGT	AATCA
S_2	AGCTA	CGAGT	AATCA
S_3	ATCCA	CTAGT	ACACT
S_4	ATCCA	CTAGT	ATACT
S_5	CGGTA	TTTGT	AAGCT
S_6	CGGTT	CATCA	AATGC
S_7	AGGTA	CTTGA	AATCC
μ	AGCTA	CTAGT	AATCT

It can be assumed that in each sequence S_i those nucleotides that do not match the corresponding nucleotide in μ were mutated at some point during evolution. These nucleotides are underscored in the above table. Intuitively, the more common mutations two sequences Si and Sj share, the closer they are likely to be in an evolutionary tree. For the above set of sequences, the common mutations matrix is the following:

	S_1	S_2	S_3	S_4	S_5	S_6	S_7
S_1	0	1	0	0	0	0	0
S_2	1	0	0	0	0	0	0
S_3	0	0	0	3	0	0	0
S_4	0	0	3	0	0	0	0
S_5	0	0	0	0	0	3	2
S_6	0	0	0	0	3	0	4
S_7	0	0	0	0	2	4	0

According to the common similarity matric the closest pair of sequences is S_6 and S_7. Hence these will be merged. When we merge two sequences S_i and S_j, in the merged sequence the k^{th} element will be equal to the nucleotide in the two sequences if $S_i[k] = S_j[k]$ and will be equal μ[k] otherwise. Hence the matrix of sequences will be updated as follows:

S_1	AGCTA	CTAGT	AATCA
S_2	AGCTA	CGAGT	AATCA
S_3	ATCCA	CTAGT	ACACT
S_4	ATCCA	CTAGT	ATACT
S_5	CGGTA	TTTGT	AAGCT
S_{67}	AGGTA	CTTGA	AATCC
μ	AGCTA	CTAGT	AATCT

2.2 A New Evolutionary Tree Algorithm

Our new evolutionary tree algorithm based on common mutations similarity matrices CMSM is described as follows.

ALGORITHM CMSM($S_1 \ldots S_n$, n)
1 Form n clusters of sequences, each with a single sequence.
2 Find the putative common ancestor μ of the sequences.
3 Construct a graph T with a node for each n cluster and for μ.
4 **While** (there is more than one cluster)
5 Find the common mutations similarity matrix.
6 **If** (common mutations similarity matrix has non-0 entry)
7 Merge a closest pair S_i and S_j into a new cluster S_{ij}
8 and create a node for S_{ij}.
9 Connect the nodes for S_i and S_j with parent node S_{ij}.
10 **Else**
11 Connect the remaining clusters' nodes to parent μ.
11 **Return** T.
12 **Return** T.

2.3 Example

Let us consider now the algorithm for the example that we started above. After merging S6 and S7 into a cluster, in the next iteration of the while loop, the algorithm updates the common mutations matrix as follows:

	S_1	S_2	S_3	S_4	S_5	S_{67}
S_1	0	1	0	0	0	0
S_2	1	0	0	0	0	0
S_3	0	0	0	3	0	0
S_4	0	0	3	0	0	0
S_5	0	0	0	0	0	2
S_{67}	0	0	0	0	2	0

Next merging the closest pair, S_3 and S_4, yields:

S_1	AGCTA	CTAGT	AATCA
S_2	AGCTA	CGAGT	AATCA
S_{34}	ATCCA	CTAGT	AAACT
S_5	CGGTA	TTTGT	AAGCT
S_{67}	AGGTA	CTTGA	AATCC
μ	AGCTA	CTAGT	AATCT

The updated common mutations matrix will be:

	S_1	S_2	S_{34}	S_5	S_{67}
S_1	0	1	0	0	0
S_2	1	0	0	0	0
S_{34}	0	0	0	0	0
S_5	0	0	0	0	2
S_{67}	0	0	0	2	0

Next merging the closest pair, S_5 and S_{67}, yields:

S_1	AGCTA	CTAGT	AATCA
S_2	AGCTA	CGAGT	AATCA
S_{34}	ATCCA	CTAGT	AAACT
S_{567}	AGGTA	CTTGT	AATCT
μ	AGCTA	CTAGT	AATCT

The updated common mutations matrix will be:

	S_1	S_2	S_{34}	S_{567}
S_1	0	1	0	0
S_2	1	0	0	0
S_{34}	0	0	0	0
S_{567}	0	0	0	0

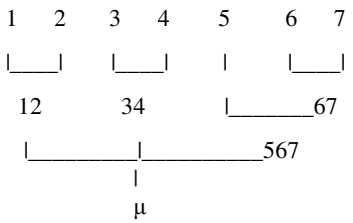
Next merging the closest pair, S_1 and S_2, yields:

S_{12}	AGCTA	CTAGT	AATCA
S_{34}	ATCCA	CTAGT	AAACT
S_{567}	AGGTA	CTTGT	AATCT
μ	AGCTA	CTAGT	AATCT

The updated common mutations matrix will be:

	S_{12}	S_{34}	S_{567}
S_{12}	0	0	0
S_{34}	0	0	0
S_{567}	0	0	0

Finally, these can be interpreted as being separate branches all descendent from the common ancestor sequence μ. Hence in the else clause, the new algorithm connects the remaining clusters's nodes to μ and generates the following hypothetical evolutionary tree:



Note that this tree is different from the tree that would be generated by the UPGMA algorithm.

2.4 Computational Complexity

The computational complexity of the algorithm can be kept low if we set up a data structure that keeps a list of all mutations for each cluster. Then when merging clusters, then we need to compute only the intersection of the lists associated with the two child clusters.

3. ACKNOWLEDGMENTS

Peter Z. Revesz is currently on leave from the University of Nebraska-Lincoln and is an AAAS Science & Technology Policy Fellow and serves as a Program Manager in the U.S. Air Force Office of Scientific Research (AFOSR), a basic research funding agency of the federal government. The current work was not supported by AFOSR, and the views and opinions expressed in this publication are those of the author and do not necessarily reflect the official policy or positionof the U.S. government.

4. REFERENCES

[1] Revesz, P. Z. 2010. *Introduction to Databases: From Biological to Spatio-Temporal*, Springer.

[2] Revesz, P. Z. and Assi, C. J.-L. 2013. Data mining the functional characterizations of proteins to predict their cancer relatedness. *International Journal of Biology and Biomedical Engineering*, 7 (1), 7-14.

[3] Revesz, P. Z. and Triplet, T. 2010. Classification integration and reclassification using constraint databases. *Artificial Intelligence in Medicine,* 49 (2), 79-91.

[4] Revesz P. Z. and Triplet, T. 2011.Temporal data classification using linear classifiers, *Information Systems*, 36 (1), 30-41.

[5] Saitou, N. and Nei, M. 1987. The neighbor-joining method: A new method for reconstructing phylogenetic trees. *Molecular Biological Evolution*, 4, 406-425.

[6] Sokal, R. R. and Michener, C. D. 1958. A statistical method for evaluating systematic relationships. *University of Kansas Science Bulletin*, 38, 1409-1438.