Clearcut: A fast implementation of relaxed neighbor joining
Luke Sheneman\textsuperscript{1,*}, Jason Evans\textsuperscript{1} and James A. Foster\textsuperscript{1}
\textsuperscript{1}Department of Biological Sciences, University of Idaho, Moscow, Idaho, USA

Associate Editor: Keith A Crandall

\section{INTRODUCTION}

Scientists need to infer increasingly large phylogenies. Neighbor joining (NJ) (Saitou and Nei, 1987; Studier and Keppler, 1988) is a popular phylogeny construction algorithm which clusters taxa according to estimated pairwise evolutionary distances. While NJ is largely considered to be a fast algorithm, it cannot efficiently reconstruct extremely large phylogenies. Relaxed neighbor joining (RNJ) (Evans \textit{et al.}, 2006) is a very fast variation of NJ which scales better to larger datasets. Both RNJ and NJ share the desirable theoretical property of recovering the true tree if the distance matrix is purely additive (Waterman \textit{et al.}, 1977). In the more common case where distances are non-additive, RNJ produces results with negligible differences from those produced by NJ (Evans \textit{et al.}, 2006).

Specifically, NJ requires time in $O(N^3)$ for inputs with $N$ taxa (Studier and Keppler, 1988). RNJ requires approximately $N^3 \log N$ time for typical inputs, though in rare worst case scenarios it degenerates to the same asymptotic runtime as NJ. Thus, RNJ allows users to process larger inputs in less time than NJ, or to bootstrap more trees in the same amount of time.

As the name implies, neighbor joining works by starting with a star-shaped tree and iteratively joining “neighboring” nodes until a bifurcating tree is constructed. At each step, traditional NJ searches the entire distance matrix and identifies and joins the pair of nodes with the global minimum transformed distance. In contrast, RNJ opportunistically joins any two neighboring nodes immediately after it is determined that the nodes are closer to each other than any other node in the distance matrix. It is not required that the candidate nodes be the closest of all nodes remaining in the matrix. In this sense, our algorithm \textit{relaxes} the requirement of exhaustively searching the distance matrix at each step to find the closest two nodes to join.

This article announces the availability of Clearcut, which implements both RNJ and a highly optimized version of NJ.

\section{METHODS}

Clearcut is a small C program that compiles and runs under most UNIX variants, and has been explicitly tested on Linux, FreeBSD, MacOS X, and Solaris. It is entirely a text-based program and takes all arguments on the command-line. The source code for Clearcut is freely distributed under the BSD license.

Clearcut implements \textit{both} relaxed and traditional neighbor joining. It is capable of taking input either in the form of a pre-computed pairwise distance matrix or a set of aligned sequences in FASTA format. When presented with an alignment, Clearcut will compute pairwise distances by first determining the percent identity between all sequence pairs. Optionally, compensation for multiple hits is possible by applying either a Jukes-Cantor correction (Jukes and Cantor, 1969) or Kimura correction (Kimura, 1980) to the pairwise distances. These optional distance corrections can be applied to either DNA or amino acid sequences.

Both NJ and RNJ are sensitive to the order in which distances are input and the order in which nodes are joined. Command-line options allow Clearcut to randomly reorder taxa to mitigate stochastic bias resulting from the original order in which taxa are presented in the input. A similar Clearcut option controls whether attempts to join nodes are done randomly or in a strictly deterministic order. Attempting to join randomly selected nodes can reduce systematic bias in some cases, while it is faster to attempt to join nodes in a deterministic order.

Since RNJ is a non-deterministic algorithm, Clearcut optionally allows the user to quickly generate any number of distinct, equally valid RNJ trees from the same non-additive distance matrix.
3 RESULTS

We compared Clearcut to several popular traditional NJ implementations including PHYLIP Neighbor (Felsenstein, 2004), QuickTree (Howe et al., 2002) and QuickJoin (Mailund et al., 2004). Our comparison used both simulated sequences and biologically-derived sequences.

For the simulated dataset, we artificially constructed trees of different sizes which were representative of the two extreme tree shapes: maximally deep (pectinate) and maximally shallow (perfect). We stochastically assigned gamma-distributed branch lengths to each branch and then used the simulated tree to construct a purely additive distance matrix.

For the biological sequences, we constructed datasets of various sizes by sampling aligned bacterial rRNA sequences without replacement from RDP-II, the Ribosomal Database Project (Cole et al.). We then used Clearcut itself to generate the distance matrices.

Compared to existing neighbor joining programs, Clearcut’s RNJ implementation reconstructed phylogenies in a fraction of the time for all tested tree shapes and sizes as shown in Figures 1 and 2. Clearcut outperformed other implementations by as much as two or three orders of magnitude. Quickjoin, the second fastest NJ implementation, was unable to handle our largest inputs due to its extremely large memory requirements.

Due to rigorous implementation optimizations, especially with respect to cache locality, even Clearcut’s traditional NJ implementation is extremely fast.

4 FUTURE ENHANCEMENTS

Future versions of Clearcut will allow users to bootstrap RNJ trees by sampling with replacement from the provided distance matrix. Clearcut will then construct a majority-rule consensus tree with nodal-support values. The labeled consensus tree will be output in Graphviz (Ellson et al., 2003) format.

Future versions of Clearcut will initially compile into a C library before linking into an executable front-end. This will allow Clearcut to be directly embedded and used inside other programs.

ACKNOWLEDGEMENTS

Sheneman was partially funded by NIH P20 RR16454 from the INBRE Program of the National Center for Research Resources. Evans and Foster were partially funded by NIH NCRR 1P20 RR16448.

REFERENCES