Maximum-Likelihood Analysis Using TREE-PUZZLE

Maximum-likelihood (ML) analysis is a statistically well founded and well known method used in many scientific fields. Edwards and Cavalli-Sforza (1964) have proposed ML for phylogenetics, and Felsenstein (1981) made it applicable for molecular sequences. Although the computation time needed for ML analysis is large, recently, the usage of ML methods has substantially increased and has become an important component in molecular sequence analysis and phylogenetics. The ML approach is appealing because it incorporates explicit models of sequence evolution, and also allows statistical tests of evolutionary hypotheses (Page and Holmes, 1998; Felsenstein, 2004).

The TREE-PUZZLE software (Schmidt et al., 2002) applies the ML principle combined with a fast tree search algorithm called Quartet Puzzling to reconstruct phylogenetic trees from biological sequences. The Quartet Puzzling Algorithm uses quartets, i.e., groups of four sequences, to reconstruct large trees guided by the ML values of the quartet tree topologies (Fig. 6.6.1).

TREE-PUZZLE also offers other algorithmic features, such as Likelihood Mapping (Strimmer and von Haeseler, 1997), a method for visualizing the phylogenetic content of multiple sequence alignments. Likelihood Mapping can also be used to evaluate the quartet support for relationships among groups of sequences.

In addition, TREE-PUZZLE implements several other statistical methods to compare different tree topologies.

In this unit, an amino acid alignment is used to explain the main features of TREE-PUZZLE. The dataset comprises the elongation factors EF-Tu/1α and EF-G/2, two genes that duplicated before the split into the three domains of life, Eukaryota, Archaea, and Bacteria (see Table 6.6.1; similar datasets were first studied by Iwabe et al., 1989).

Although a protein example is used here, TREE-PUZZLE can analyze nucleotide and binary data (e.g., restriction digest data) as well.

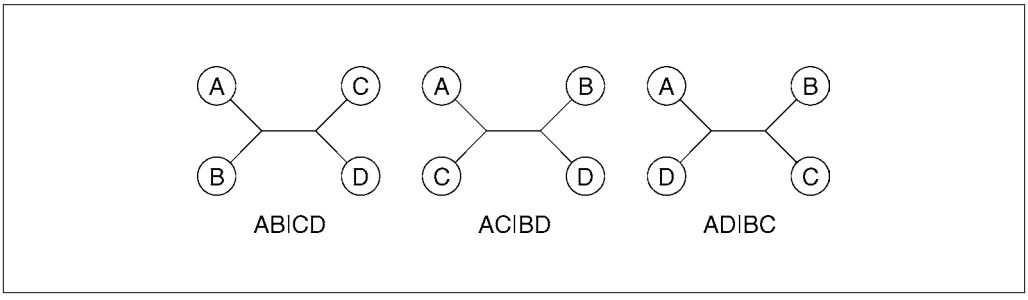

Figure 6.6.1 The three possible informative quartet tree topologies.

Contributed by Heiko A. Schmidt and Arndt von Haeseler
Current Protocols in Bioinformatics (2007) 6.6.1-6.6.23

RECONSTRUCT A PHYLOGENETIC TREE

The main use of TREE-PUZZLE is to reconstruct phylogenetic trees from sequences. The example shows how to use TREE-PUZZLE to construct a tree from amino acid sequences assuming Γ-distributed (Gamma-distributed) rates across sites (*UNIT 6.5*).

Necessary Resources

Hardware

TREE-PUZZLE runs on computers with MS Windows, Mac OS, and Unix/Linux operating systems, including workstation clusters and computers using parallel computing

Software

TREE-PUZZLE package (see Support Protocols 1 to 3 for information on how to obtain TREE-PUZZLE)

Files

Multiple Sequence Alignment file in standard PHYLIP format (see *APPENDIX 1B* and Figure A.1B.3 for a sample PHYLIP format file). The sample data set used here (`EF.phy`) is included with the TREE-PUZZLE software package.

1. Obtain and install TREE-PUZZLE (see Support Protocols 1 to 3).
2. Change to the `data` directory in the TREE-PUZZLE directory and start the program with the command `puzzle EF.phy`.

   Start `puzzle` *in a terminal, e.g., Command Prompt (for Windows), Terminal (for Mac OS X; see APPENDIX 1C), or xterm (for Unix/Linux; see APPENDIX 1C & APPENDIX 1D), using the command* `puzzle alignmentfile`*, where* `alignmentfile` *is the name of the file containing the alignment to be analyzed; the example here is* `EF.phy`*. If* `puzzle` *is invoked from a file manager or without a filename, it will search for a file called* `infile` *in the current directory. If* `infile` *does not exist, TREE-PUZZLE will ask for a filename. The* `alignmentfile` *has to be in the current working directory or the full path to its location must be given.*

   IMPORTANT NOTE: *When* `puzzle` *is started by a mouse click, e.g., from the desktop under Windows, Unix/Linux, and Mac OS v. 9.x or lower, the working directory is set to the one in which the executable is located. Under Mac OS X, however, the working directory of a "click-started" instance of* `puzzle` *is set to the user's home folder. When used from a terminal on the command line, the working directory is always the current directory. Thus, the input files should be copied to the working directory or their complete path has to be entered.*
3. Change the type of analysis to `tree reconstruction` (using the `b` key) and the tree search procedure to `quartet puzzling` (using the `k` key), if necessary (Fig. 6.6.2).
4. Adjust the outgroup to the sequence 22 `EFG_MYCGE` (using `o` and the number of the sequence).

   By default, the first sequence is used to root the resulting tree for output. However, the choice of root has no impact on the log-likelihood.

   Note that the natural root lies between EF-α/Tu and EF-2/G (Iwabe et al., 1989). Hence the output tree has to be re-rooted using a phylogeny viewer like TreeView (see UNIT 6.2 and Internet Resources below).

   For further discussion of selecting a tree root, see UNIT 6.1.

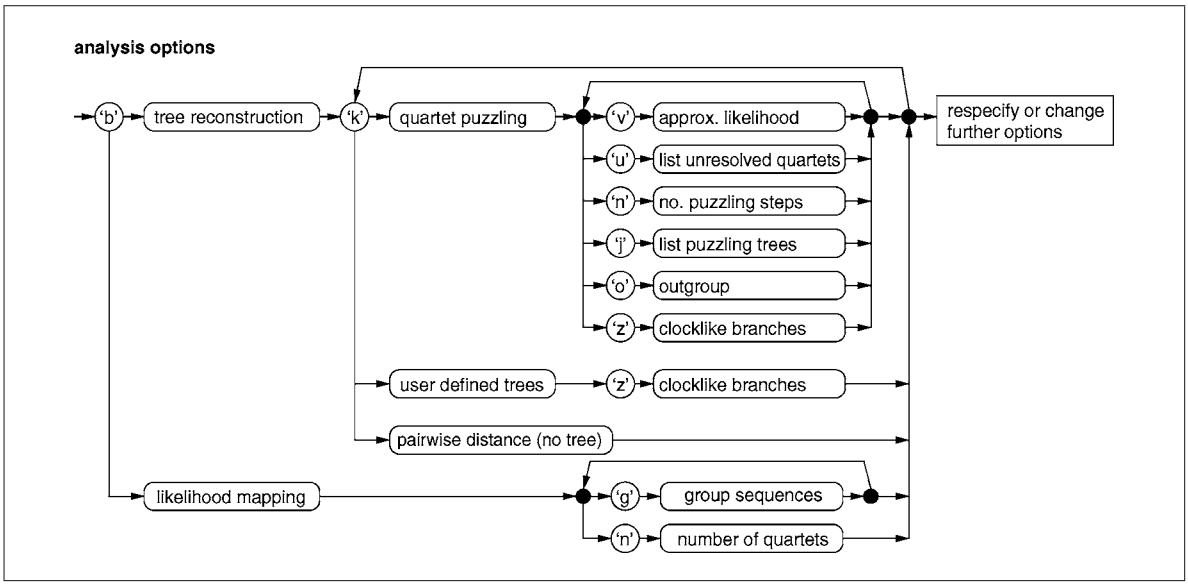

Figure 6.6.2 Flowchart of analysis type options in the TREE-PUZZLE menu. Options in TREE-PUZZLE are controlled by single letters. The flow chart shows the options that correspond to each letter. For example, entering the letter `b` toggles the analysis between tree reconstruction and likelihood mapping. Similarly, to choose among quartet puzzling, user-defined trees, or pairwise distance matrices, enter the letter `k` until the desired option is shown on the screen.

5. Choose parameter estimation to be performed approximately (with `e`) using neighbor-joining trees (with `x`).

   Parameters are estimated using tree topologies. These are either inferred by neighbor-joining or given as usertree (usertree evaluation; see Basic Protocol 3). With the `quartet samples + NJ` *option, the evolutionary parameters are estimated on random quartet samples; neighbor-joining trees are only used for rate parameters.* `Approximate` *estimation uses pairwise distances to fit the branch lengths of the tree topologies, while ML branch lengths are inferred in the* `exact` *estimation.*

Choose a model of evolution

6. Change the type of sequence data to amino acids (using `d`) if the automatically assigned type is not correct (Fig. 6.6.3).

   Using the character composition of the alignment, TREE-PUZZLE tries to figure out whether the type of data is nucleotide, protein, or binary data.

7. Choose an appropriate model of sequence evolution to analyze the dataset. For the example alignment, choose the VT model using `m` (Fig. 6.6.3).

   Several models for protein evolution are implemented in TREE-PUZZLE. While the models by Dayhoff et al. (1978) and Jones et al. (1992) are universal models created from different protein families, more specific models are available, e.g., the mtREV24 model by Adachi and Hasegawa (1996) for mitochondrial protein sequences. Also implemented are the VT (Müller and Vingron, 2000) and the WAG models (Whelan and Goldman, 2001), which are suited to analyze distantly related sequences. The BLOSUM62 matrix (Henikoff and Henikoff, 1992; UNIT 3.5) was designed for database searches and thus should be used with caution for the analysis of evolutionary relationships. For further evolutionary models, refer to the manual. TREE-PUZZLE tries to determine a suitable model by comparing the amino acid frequencies in various models with those of the dataset.

   For DNA (Fig. 6.6.4), the HKY (Hasegawa et al., 1985) and TN (Tamura and Nei, 1993) models are available. Those models can be restricted to simpler models like JC (Jukes and Cantor, 1969), K2P (Kimura, 1980), or F84 (Felsenstein, 1984) by setting substitution parameters accordingly. Also the general time-reversible model (GTR; Lanave et al., 1984; Tavare, 1986) is implemented, which can be confined to even more different models by explicitly setting the GTR parameters (refer to the manual and UNITS 6.4 & 6.5 for further

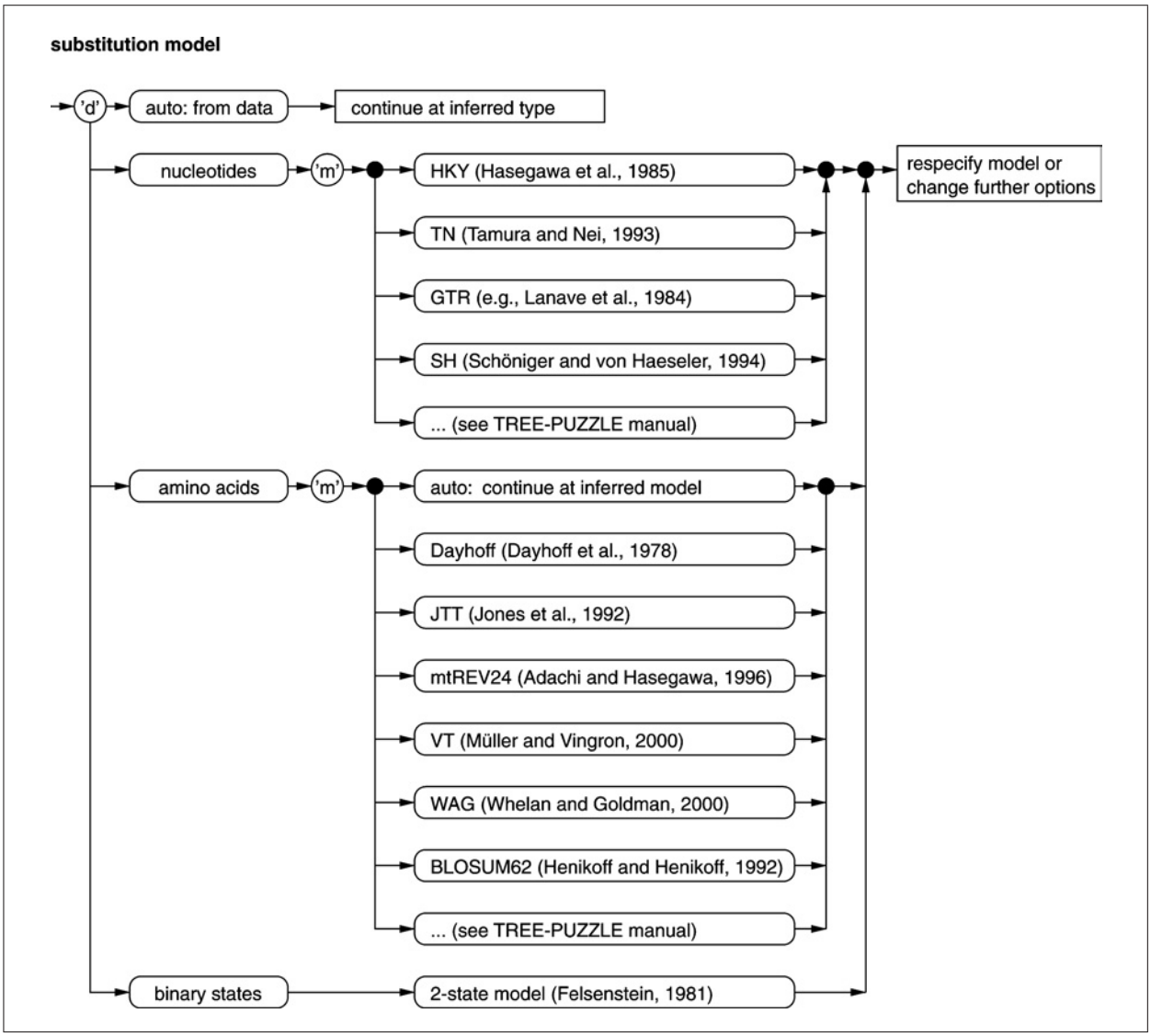

Figure 6.6.3 Flowchart of substitution model options in the TREE-PUZZLE menu.

details). Additionally, the SH nucleotide doublet model (Schöniger and von Haeseler, 1994) and a binary model based on the model of Felsenstein (1981) are implemented in TREE-PUZZLE.

8. Choose Γ-distributed rate heterogeneity by typing `w` (Fig. 6.6.5).

   It is known that positions in an alignment do not evolve with the same evolutionary rates, typically attributed to selective pressure or other functional constraints acting on positions of the sequence. In such cases, the assumption of rate heterogeneity can improve the estimation of the branch lengths.

   Three different models of rate heterogeneity are implemented in TREE-PUZZLE. Besides Γ-distributed rates, there is the two-rates model that assumes a fraction of the positions to be invariable and a mixed model that considers the variable sites to evolve according to a Γ distribution. The amount of rate heterogeneity of the Γ-distributed rates is described by the shape parameter α, where $\alpha < 1$ describes strong heterogeneity, while large values describe homogeneity (for more details, refer to Gu et al., 1995; Page and Holmes, 1998; Felsenstein, 2004; UNITS 6.4 & 6.5).

   If tree reconstructions with and without the assumption of rate heterogeneity construct different trees, those trees can be compared as described in Basic Protocol 3 to find out whether the resulting tree topologies are significantly different.

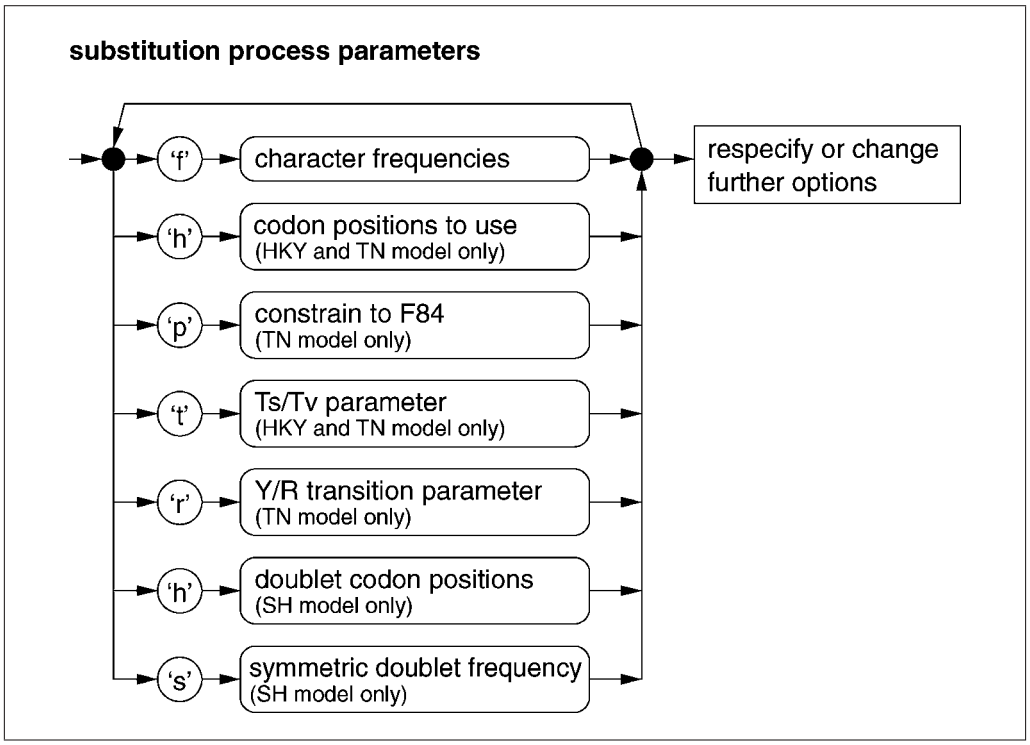

Figure 6.6.4 Flowchart of further substitution model parameters in the TREE-PUZZLE menu.

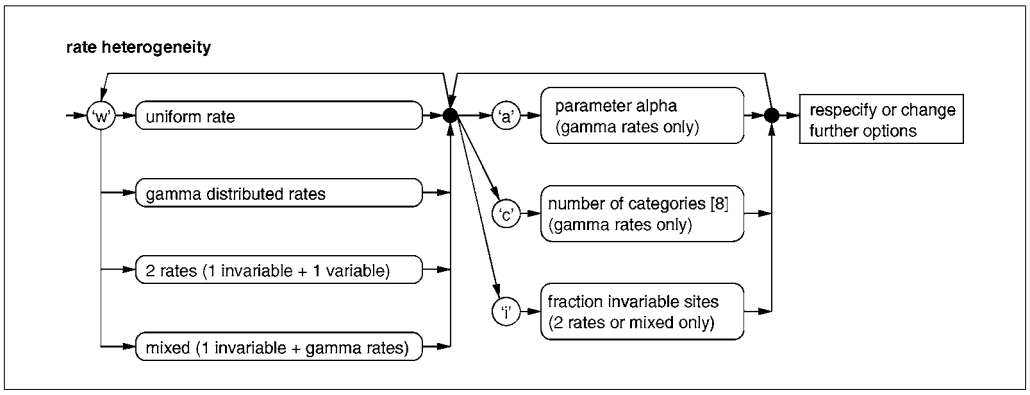

Figure 6.6.5 Flowchart of rate heterogeneity options in the TREE-PUZZLE menu.

9. Set the `list puzzling step trees` option to `unique topologies` with the `j` key, to make TREE-PUZZLE write all (unique) intermediate tree topologies to file (`EF.phy.ptorder`).

   When doing one's own analysis, it might be necessary to change other parameters. Many other parameters and options can be set manually. For instance, it is possible to specify the amino acid or nucleotide composition. Figures 6.6.2, 6.6.3, 6.6.4, 6.6.5, and 6.6.6 summarize all options currently available in TREE-PUZZLE. More details are given in the manual.

10. Start analysis by typing `y`.

    TREE-PUZZLE will now perform a tree reconstruction. During its run, it will indicate which steps are performed: first the missing parameters are estimated, then all possible quartet maximum-likelihood trees are computed, which are subsequently used to compute intermediate quartet puzzling trees. Finally, the likelihood and the branch lengths of the consensus tree are computed (Fig. 6.6.7).

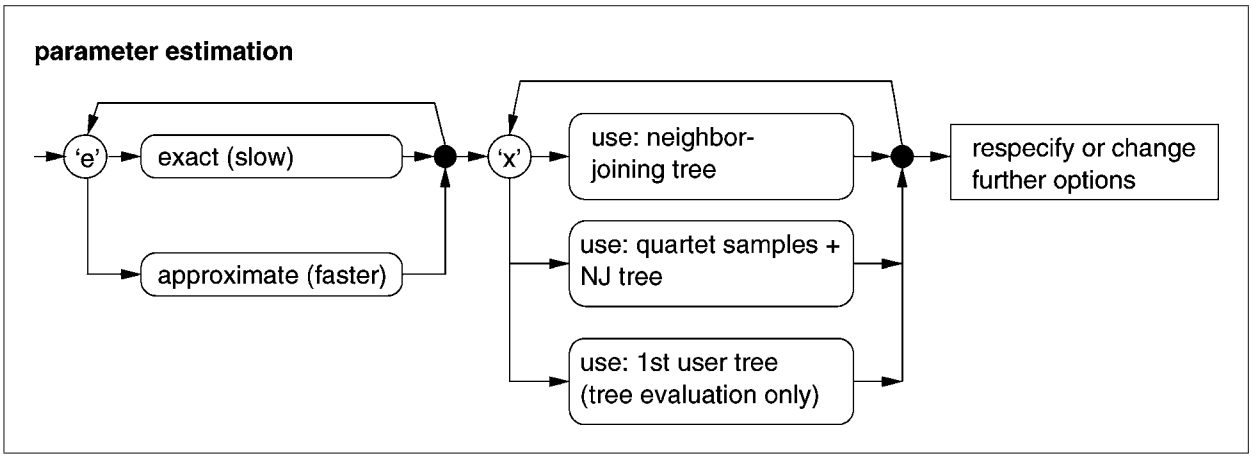

Figure 6.6.6 Flowchart of parameter estimation options in the TREE-PUZZLE menu.

```
GENERAL OPTIONS
 b                      Type of analysis?  Tree reconstruction
 k                 Tree search procedure?  Quartet puzzling
 v        Approximate quartet likelihood?  Yes
 u               List unresolved quartets?  No
 n               Number of puzzling steps?  1000
 j               List puzzling step trees?  Unique topologies
 o                    Display as outgroup?  EFG_MYCGE
 z      Compute clocklike branch lengths?  No
 e                    Parameter estimates?  Approximate (faster)
 x              Parameter estimation uses?  Neighborjoining tree
SUBSTITUTION PROCESS
 d           Type of sequence input data?  Auto: Amino acids
 m                 Model of substitution?  VT (MuellerVingron 2000)
 f               Amino acid frequencies?  Estimate from data set
RATE HETEROGENEITY
 w           Model of rate heterogeneity?  Gamma distributed rates
 a   Gamma distribution parameter alpha?  Estimate from data set
 c       Number of Gamma rate categories?  8

Quit [q], confirm [y], or change [menu] settings:
Optimizing missing rate heterogeneity parameters
Writing parameters to file EF.phy.puzzle
Writing pairwise distances to file EF.phy.dist
Computing quartet maximum likelihood trees
Computing quartet puzzling tree
Computing maximum likelihood branch lengths (without clock)

All results written to disk:
        Puzzle report file:          EF.phy.puzzle
        Likelihood distances:        EF.phy.dist
        Phylip tree file:            EF.phy.tree
        Unique puzzling step trees: EF.phy.ptorder
```

Figure 6.6.7 TREE-PUZZLE menu setting and screen output from tree reconstruction.

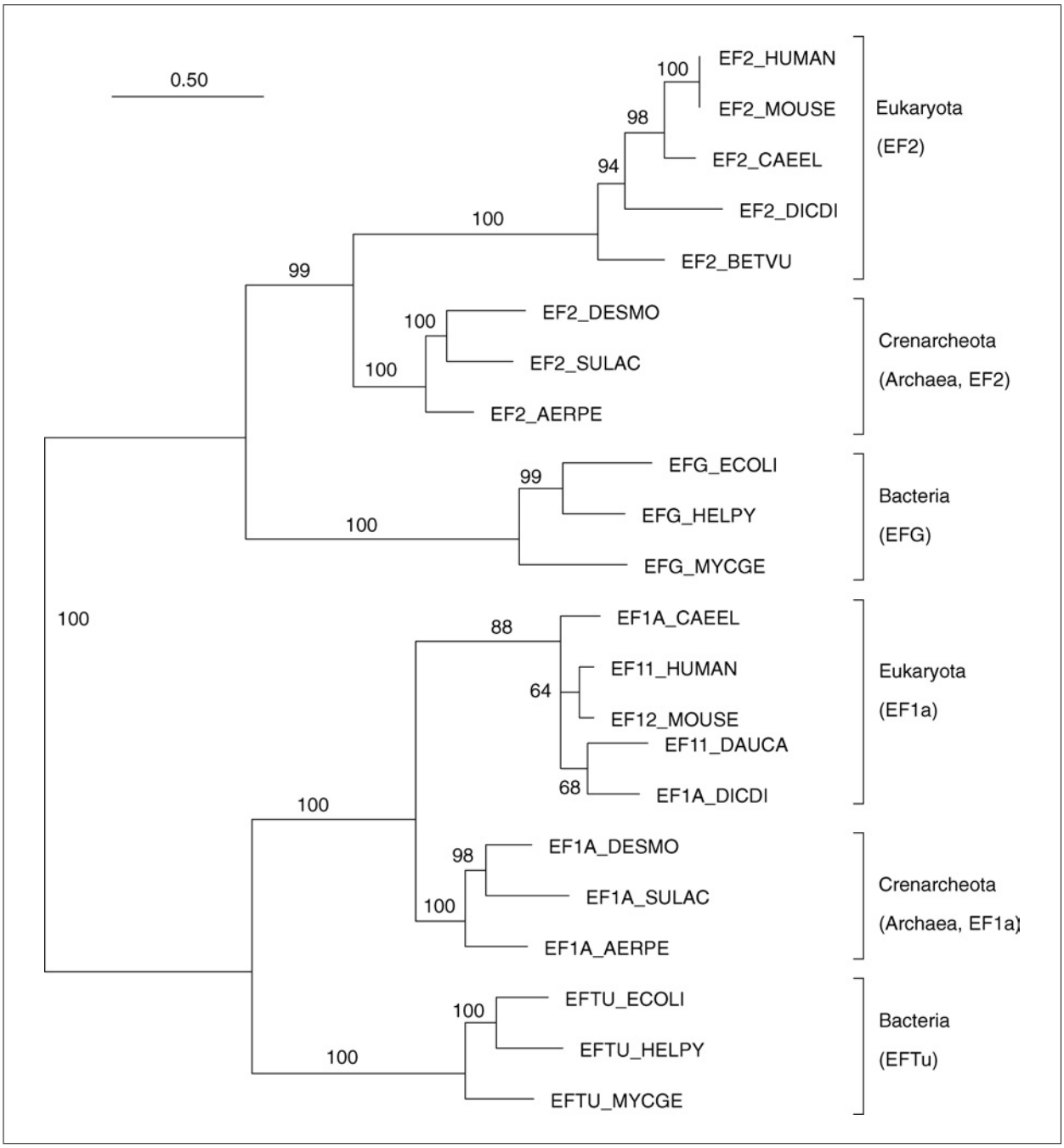

Figure 6.6.8 Phylogenetic tree reconstructed from the `EF.phy` dataset as described in Basic Protocol 1. The tree is rooted by the duplication event between EF-2/G and EF-1α/Tu.

Examine the results

11. Examine the puzzle report file. The report file is called `EF.phy.puzzle`.

    The puzzle report file presents details on the quality of the data as well as the reconstructed tree. Hence, it should be thoroughly examined (see Guidelines for Understanding Results below).

12. Examine the reconstructed tree by viewing the tree file `EF.phy.tree` (Fig. 6.6.8) using a tree-drawing program like TreeView or TreeTool (see *UNIT 6.2* and Internet Resources below).

    If a program cannot read the trees produced by TREE-PUZZLE, it may be necessary to remove the leading comment (bordered by square brackets). See Guidelines for Understanding Results for help understanding the tree file.

ANALYZE THE CONTENT OF PHYLOGENETIC INFORMATION AND THE QUARTET SUPPORT FOR THE RELATIONSHIP OF GROUPS OF SEQUENCES

Likelihood mapping provides the opportunity to either check the content of phylogenetic information in an alignment or to estimate the quartet support of relationships among groups of sequences. The former helps determine whether the data are suitable for phylogenetic analysis by measuring the resolution of the quartet topologies (i.e., trees of four sequences). It is recommended that this check be run especially on large datasets, to avoid spending days or maybe even weeks on phylogenetic analyses with data that contain little phylogenetic information. The latter method partitions a dataset into two to four clusters (i.e., groups of sequences). Likelihood mapping then visualizes which of the possible relationships between these clusters is most supported by the reconstructed quartet tree topologies (Fig. 6.6.1). This method is also useful for reducing the runtime, if the goal is to examine one special bipartition of a tree in a large dataset.

The EF data (Table 6.6.1) will serve as an example of both techniques. First, the suitability of the alignment for phylogenetic analysis is measured (step 4a), then the quartet support for the relationship of four subsets of the dataset is studied (step 4b) in more detail.

Table 6.6.1 Sequences and Their Accession Numbers Used in the Test Dataset (`EF.phy`)

Sequence type	Identifier	Accession no.	Species name
Bacterial EF-Tu	EFTU_ECOLI	P02990	*Escherichia coli*
	EFTU_HELPY	P56003	*Helicobacter pylori*
	EFTU_MYCGE	P13927	*Mycoplasma genitalium*
Crenarchaeotic EF-1α (Archaea)	EF1A_DESMO	P41203	*Desulfurococcus mobilis*
	EF1A_SULAC	P17196	*Sulfolobus acidocaldarius*
	EF1A_AERPE	Q9YAV0	*Aeropyrum pernix*
Eukaryotic EF-1α	EF11_HUMAN	P04720	*Homo sapiens*
	EF12_MOUSE	P27706	*Mus musculus*
	EF1A_CAEEL	P53013	*Caenorhabditis elegans*
	EF1A_DICDI	P18624	*Dictyostelium discoideum*
	EF11_DAUCA	P29521	*Daucus carota*
Crenarchaeotic EF-2 (Archaea)	EF2_DESMO	P33159	*Desulfurococcus mobilis*
	EF2_SULAC	P23112	*Sulfolobus acidocaldarius*
	EF2_AERPE	Q9YC19	*Aeropyrum pernix*
Eukaryotic EF-2	EF2_HUMAN	P13639	*Homo sapiens*
	EF2_MOUSE	P58252	*Mus musculus*
	EF2_CAEEL	P29691	*Caenorhabditis elegans*
	EF2_DICDI	P15112	*Dictyostelium discoideum*
	EF2_BETVU	O23755	*Beta vulgaris*
Bacterial EF-G	EFG_ECOLI	P02996	*Escherichia coli*
	EFG_HELPY	P56002	*Helicobacter pylori*
	EFG_MYCGE	P47335	*Mycoplasma genitalium*

Necessary Resources

Hardware

TREE-PUZZLE runs on computers with MS Windows, Mac OS, and Unix/Linux operating systems, including workstation clusters and computers using parallel computing

Software

TREE-PUZZLE package (see Support Protocols 1 to 3 for information on how to obtain TREE-PUZZLE)

Files

Multiple Sequence Alignment file in standard PHYLIP format (see *APPENDIX 1B* and Figure A.1B.3 for a sample PHYLIP format file). The sample data set used here (`EF.phy`) is included with the TREE-PUZZLE software download.

1. Obtain and install TREE-PUZZLE (see Support Protocols 1 to 3).
2. Change to the data directory in the TREE-PUZZLE directory and start `puzzle` with the command `puzzle EF.phy`.

   Start `puzzle` *in a terminal, e.g., Command Prompt (Windows), Terminal (Mac OS X; APPENDIX 1C), or xterm (for Unix/Linux; APPENDIX 1C & APPENDIX 1D) using the command* `puzzle alignmentfile`*, where* `alignmentfile` *is the name of the file containing the alignment to be analyzed; the example here is* `EF.phy`*. If* `puzzle` *is invoked from a file manager or without a filename, it will search for a file called* `infile` *in the current directory. If* `infile` *does not exist, TREE-PUZZLE will ask for a filename. The* `alignmentfile` *has to be in the current working directory or the full path to its location must be given.*

   See Basic Protocol 1, step 2, for working directory issues on various platforms.
3. Change the type of analysis to `Likelihood mapping` (using the `b` key).

4a. Leave the sequences ungrouped for a general likelihood mapping analysis to test the dataset.

4b. Group the sequences into four clusters (using `g`). Assign crenarchaeotic EF-2 to cluster `a`, bacterial EF-G to `b`, eukaryotic EF-2 to `c`, and all EF-1α/Tu sequences to cluster `d` (Table 6.6.1).

   To analyze the phylogenetic relationship among groups of sequences, define two to four disjoint sets of sequences from the alignment by assigning each sequence the name of the cluster `a` *through* `d` *(in the case of less than four clusters,* `c` *and/or* `d` *are not valid). Assigning* `x` *will exclude a sequence from the analysis. Each sequence must be labeled* `a`*,* `b`*, (*`c`*,* `d`*), or* `x`*.*

   A two-cluster analysis will check for the quartet support for bipartition into the two clusters, whereas a four-cluster analysis will infer the quartet support for any of the three possible relationships of the four clusters, namely (ab|cd), (ac|bd), or (ad|bc). where "|" denotes the inner branch that separates the groups (Fig. 6.6.1).

Choose a model of evolution (for more information, see Basic Protocol 1, steps 6 to 9)

5. Change the type of sequence data (using `d`) if the automatically assigned type is wrong.

   TREE-PUZZLE should have set the data type correctly to amino acids for the example.
6. Choose an appropriate model of evolution to analyze a dataset. For the example alignment, choose the VT model with the `m` key.
7. Choose Γ-distributed rate heterogeneity by typing `w`.

8. Change other parameters, if necessary. For this example, leave the parameters unchanged.

   The number of quartets used in the analysis can be set by the `n` *option. If the number of existing quartets is larger than the specified number, a random subset of all possible quartets is chosen by default, but the size of the sample is also adjustable.*

9. Start analysis by typing `y`.

   TREE-PUZZLE will now perform a likelihood-mapping analysis. During the run, it will indicate which steps are performed: first the missing parameters are estimated, then the likelihood-mapping analysis is performed, evaluating quartet maximum-likelihood trees. For large datasets, a random subset of quartets is analyzed (Fig. 6.6.9).

Examine the results

10. Examine the puzzle report file. The report file is called `EF.phy.puzzle`, if starting with the alignment file `EF.phy`.

    The puzzle report file presents the quality of the data as well as the results of the likelihood mapping. Hence, it should be thoroughly examined.

11. Examine the likelihood-mapping diagram (Figs. 6.6.10, 6.6.11, and 6.6.12), `EF.phy.eps`, using a PostScript browser like Ghostscript/Ghostview (see Internet Resources).

    See Guidelines for Understanding Results for help in interpreting these diagrams.

```
GENERAL OPTIONS
 b                      Type of analysis?  Likelihood mapping
 g           Group sequences in clusters?  No
 n                    Number of quartets?  7315 (all possible)
 e                   Parameter estimates?  Approximate (faster)
 x             Parameter estimation uses?  Neighborjoining tree
SUBSTITUTION PROCESS
 d           Type of sequence input data?  Auto: Amino acids
 m                 Model of substitution?  VT (Mueller-Vingron 2000)
 f                Amino acid frequencies?  Estimate from data set
RATE HETEROGENEITY
 w           Model of rate heterogeneity?  Gamma distributed rates
 a   Gamma distribution parameter alpha?  Estimate from data set
 c       Number of Gamma rate categories?  8

Quit [q], confirm [y], or change [menu] settings:
Optimizing missing rate heterogeneity parameters
Writing parameters to file EF.phy.puzzle
Writing pairwise distances to file EF.phy.dist
Performing likelihood mapping analysis

All results written to disk:
        Puzzle report file:          EF.phy.puzzle
        Likelihood distances:        EF.phy.dist
        Likelihood mapping diagram: EF.phy.eps
```

Figure 6.6.9 TREE-PUZZLE menu setting and screen output from likelihood-mapping analysis.

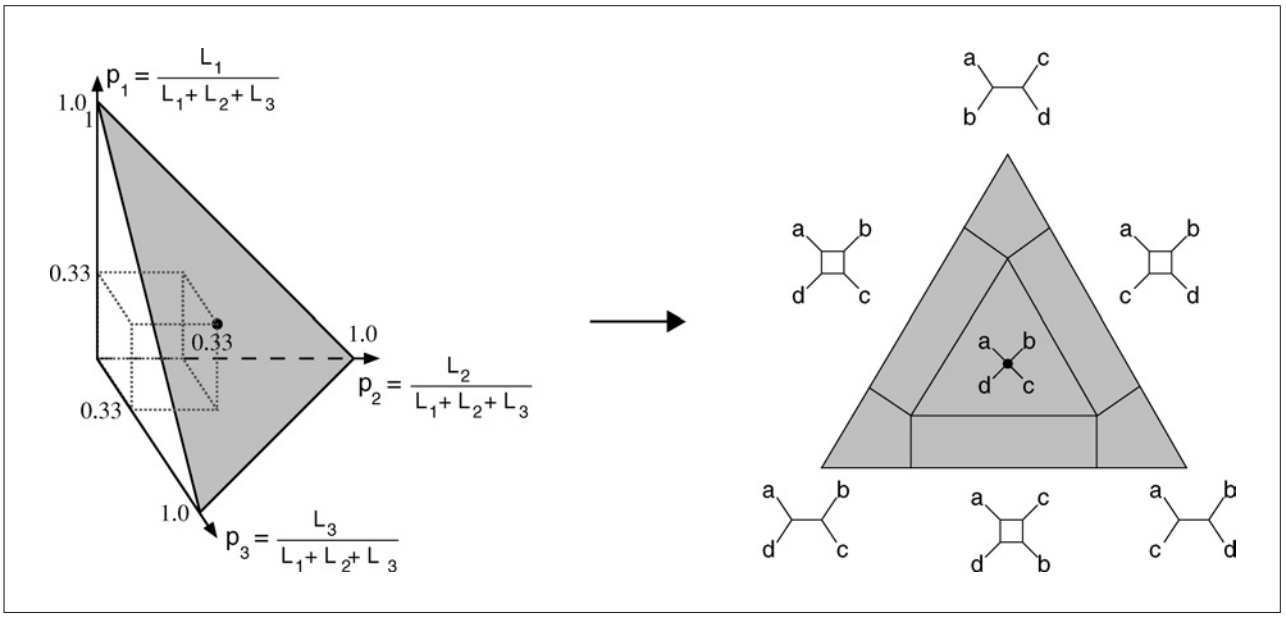

Figure 6.6.10 How likelihood weights are plotted in a likelihood-mapping diagram. Left side: likelihood weight plotted in a three-dimensional coordinate system. Right side: the simplex and its areas and the corresponding quartet topologies. The gray triangles are identical, only viewed from different angles.

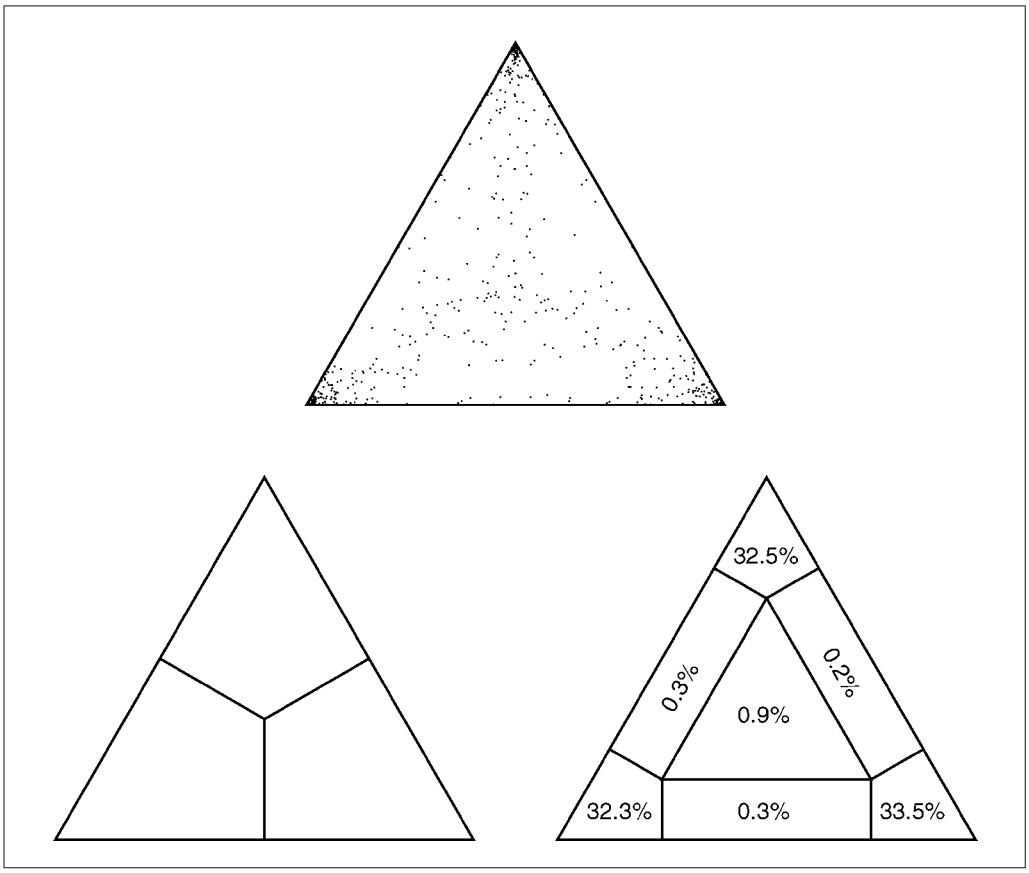

Figure 6.6.11 Likelihood-mapping diagram visualizing the phylogenetic content of the `EF.phy` dataset performed as described in Basic Protocol 2.

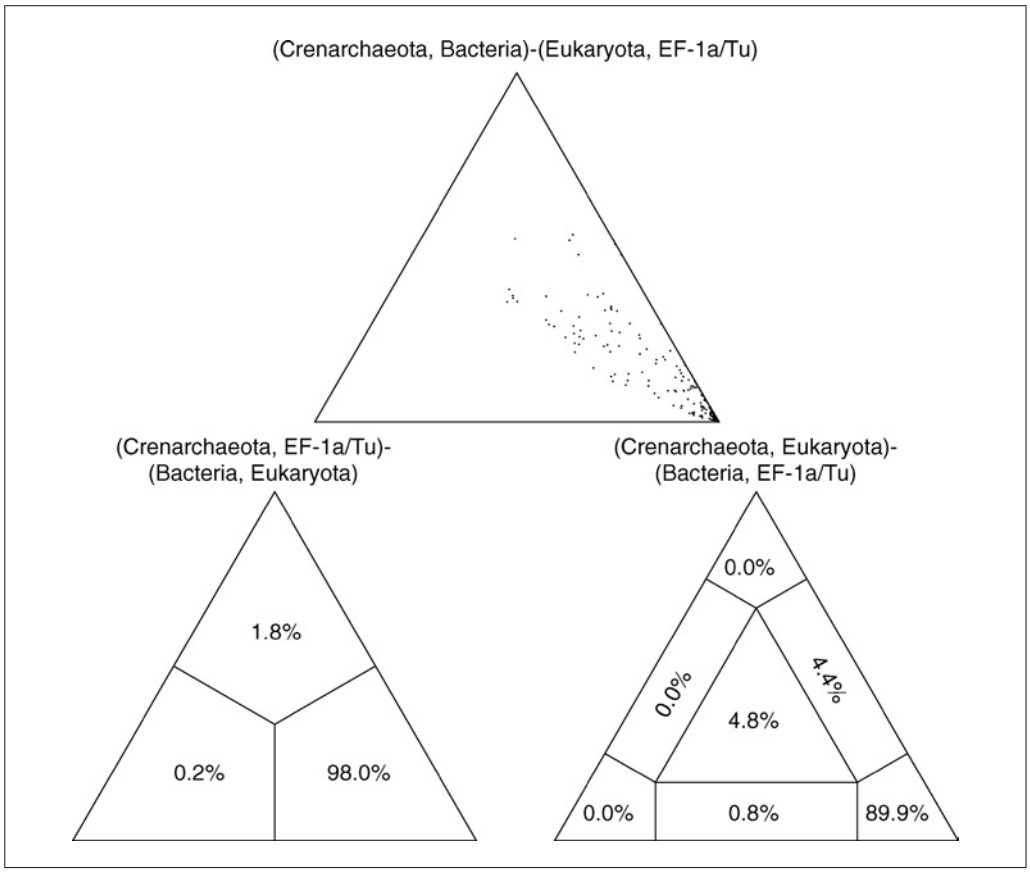

Figure 6.6.12 Likelihood-mapping diagram visualizing the support for a Crenarchaeota-Eukaryota sister group in the EF-2/G genes of the `EF.phy` dataset as described in Basic Protocol 2.

BASIC PROTOCOL 3

COMPARE TREE TOPOLOGIES

A third type of analysis implemented in TREE-PUZZLE is the likelihood-based comparison of two or more tree topologies using the tests suggested by Kishino and Hasegawa (1989), Shimodaira and Hasegawa (1999), and the so-called expected likelihood weights (Strimmer and Rambaut, 2002). These tests compare different trees to evaluate something like a confidence set of trees. The example used here is a dataset together with a set of trees with different branching patterns, comprising the tree reconstructed in Basic Protocol 1 and two trees with the different possible relationships of Crenarchaeota, Bacteria, and Eukaryota (Fig. 6.6.13).

Necessary Resources

Hardware

TREE-PUZZLE runs on computers with MS Windows, Mac OS, and Unix/Linux systems, including workstation clusters and computers using parallel computing

Software

TREE-PUZZLE package (see Support Protocols 1 to 3 for information on how to obtain TREE-PUZZLE)

Files

Multiple Sequence Alignment file in standard PHYLIP format (see *APPENDIX 1B* and Figure A.1B.3 for a sample PHYLIP format file). A tree file containing the

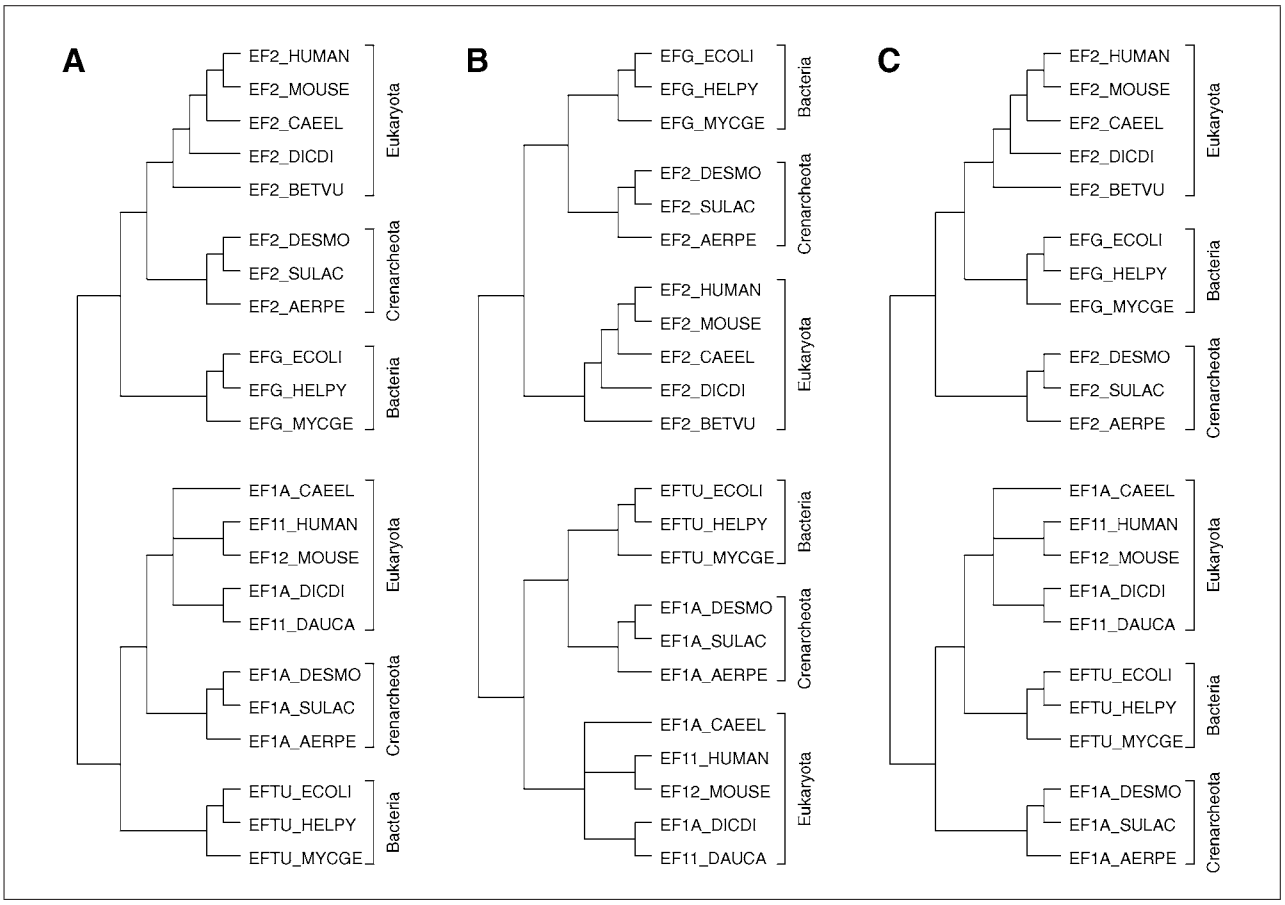

Figure 6.6.13 The three tree topologies used in the usertree comparison. (**A**) Tree 1: Eukaryota-Crenarchaeota sister groups, (**B**) Tree 2: Bacteria-Crenarchaeota sister groups, (**C**) Tree 3: Eukaryota-Bacteria sister groups. The tree topologies are used without branch lengths.

usertrees in Newick tree format as produced by many programs, e.g., PHYLIP or TREE-PUZZLE. (trees can span several lines and contain comments; for more information see *UNIT 6.2*). The sample data set used here is included with the TREE-PUZZLE software download.

1. Obtain and install TREE-PUZZLE (see Support Protocols 1 to 3).

2. Change to the `data` directory in the `TREE-PUZZLE` directory and start puzzle with the command `puzzle EF.phy EF.3trees`.

   Start `puzzle` *in a terminal, e.g., Command Prompt (for Windows), terminal (for Mac OS X; see APPENDIX 1C), or xterm (for Unix/Linux; see APPENDIX 1C & APPENDIX 1D) using the command* `puzzle alignmentfile usertreefile`, *where* `alignmentfile` *is the name of the file containing the alignment to be analyzed and* `usertreefile` *is the name of the file that contains the tree topologies for comparison. If* `puzzle` *is invoked from a desktop or from the command line without filename arguments, it will search for the files* `infile` *and* `intree` *in the current directory. If* `infile` *and/or* `intree` *does not exist, TREE-PUZZLE will ask for a filename. The* `alignmentfile` *and* `usertreefile` *have to be in the current working directory or the full paths to their respective locations must be given.*

   See Basic Protocol 1, step 2 for working directory issues on various platforms.

3. Change the type of analysis to `tree reconstruction` (using the `b` key) and the tree search procedure to `user defined trees` (using the `k` key), if necessary.

Choose a model of evolution (for more information, see Basic Protocol 1, steps 6 to 9)

4. Change the type of sequence data (using `d`) if the automatically assigned type is wrong. TREE-PUZZLE should have set the data type correctly to amino acids for the example.

5. Choose an appropriate model of evolution to analyze the dataset. For this example alignment, choose the VT model by selecting the `m` option.

6. Choose Γ-distributed rate heterogeneity by typing `w`.

7. Choose neighbor-joining (NJ) tree as the means for the parameter estimation, using the `x` key. Change other parameters, if necessary.

   For tree evaluation, TREE-PUZZLE uses the first usertree for the parameter estimation by default. This makes sense for the evaluation of single trees, but to test a set of trees, as in this example, an NJ tree should be used to estimate the parameters.

8. Start the analysis by typing `y`.

   TREE-PUZZLE will now evaluate and compare the tree topologies in the usertreefile (`EF.3trees`*). During its run, it will indicate which steps are performed: first, the missing parameters are estimated, then all trees in the usertreefile (*`EF.3trees`*) are evaluated and the results are written to the puzzle report file (Fig. 6.6.14).*

```
GENERAL OPTIONS
 b                      Type of analysis?  Tree reconstruction
 k                  Tree search procedure?  User defined trees
 z     Compute clocklike branch lengths?  No
 e                    Parameter estimates?  Approximate (faster)
 x              Parameter estimation uses?  Neighborjoining tree
SUBSTITUTION PROCESS
 d           Type of sequence input data?  Auto: Amino acids
 m                 Model of substitution?  VT (Mueller-Vingron 2000)
 f               Amino acid frequencies?  Estimate from data set
RATE HETEROGENEITY
 w           Model of rate heterogeneity?  Gamma distributed rates
 a   Gamma distribution parameter alpha?  Estimate from data set
 c      Number of Gamma rate categories?  8

Quit [q], confirm [y], or change [menu] settings:
Optimizing missing rate heterogeneity parameters
Writing parameters to file EF.3trees.puzzle
Writing pairwise distances to file EF.3trees.dist
Computing maximum likelihood branch lengths (without clock) for tree # 1
Computing maximum likelihood branch lengths (without clock) for tree # 2
Computing maximum likelihood branch lengths (without clock) for tree # 3
Performing single sided KH test.
Performing ELW test.
Performing SH test.

All results written to disk:
        Puzzle report file:          EF.3trees.puzzle
        Likelihood distances:        EF.3trees.dist
        Phylip tree file:            EF.3trees.tree
```

Figure 6.6.14 TREE-PUZZLE menu setting and screen output from usertree evaluation.

```
COMPARISON OF USER TREES (NO CLOCK)

Tree   log L   difference    S.E.     p1sKH      pSH        cELW       2sKH
-------------------------------------------------------------------------------
 1  18965.87      0.00  <---- best    1.0000 +   1.0000 +   0.9123 +    best
 2  18974.43      8.56         5.77    0.0690 +   0.0810 +   0.0846 +      +
 3  18977.24     11.37         4.96    0.0130     0.0130     0.0031        -

The columns show the results and pvalues of the following tests:
1sKH - one sided KH test based on pairwise SH tests (Shimodaira-Hasegawa
       2000, Goldman et al., 2001, Kishino-Hasegawa 1989)
SH   - Shimodaira-Hasegawa test (2000)
ELW  - Expected Likelihood Weight (Strimmer-Rambaut 2002)
2sKH - two sided Kishino-Hasegawa test (1989)

Plus signs denote the confidence sets. Minus signs denote significant
exclusion. All tests used 5% significance level. 1sKH, SH, and ELW
performed 1000 resamplings using the RELL method.
```

Figure 6.6.15 Results of the comparison of three trees from the `EF.3trees` dataset as described in Basic Protocol 3.

Examine the results

9. Examine the puzzle report file. The report file is called `EF.3trees.puzzle`.

   The puzzle report file presents the quality of the data as well as the results of the usertree evaluation (Fig. 6.6.15). Hence, it should be thoroughly examined. The file `EF.3trees.tree` *contains each tree from the usertreefile in NEWICK tree format with estimated branch lengths. The trees can be viewed with tree-drawing programs like TreeView or TreeTool (see UNIT 6.2 and Internet Resources). If a program cannot read such trees, it might be necessary to remove the leading comment (bordered by square brackets).*

   See Guidelines for Understanding Results for help in interpreting these files.

OBTAIN AND INSTALL TREE-PUZZLE FOR UNIX/LINUX AND Mac OS X

SUPPORT PROTOCOL 1

This protocol describes how to obtain and install TREE-PUZZLE for Unix/Linux operating systems, including Mac OS X.

Necessary Resources

Hardware

Unix/Linux system with TCP/IP Internet connection and a Web browser

Software

On Unix systems, an ANSI/ISO C compiler is needed, which is usually delivered with the operating system; otherwise, use the free GNU C compiler (*http://www.gnu.org*)

To use the parallel version of TREE-PUZZLE for supercomputers and workstation clusters, implementation of the MPI library (Message Passing Interface) is needed. There are several free implementations like LAM or MPICH (see *http://www.lam-mpi.org/mpi/implementations* for a list of implementations).

1. Download the current TREE-PUZZLE package for Unix from *http://www.tree-puzzle.de*. It has a name like `tree-puzzle-X.X.tar.gz`, where `X.X` should be the current version.

2. Unpack the package using:

```
gunzip tree-puzzle-X.X.tar.gz
tar -xvf tree-puzzle.X.X.tar
```

This should create a directory `tree-puzzle-X.X`. The subdirectories `doc` and `data` contain the manual and test data, respectively.

3. Change to the `tree-puzzle-X.X` directory.

4. Read the `INSTALL` file and the installation part of the manual carefully. Type the following commands to produce an executable:

```
./configure
./make
```

The command `configure` *will determine the system type, and whether all needed software is installed. The* `make` *command will then compile the executable. If* `configure` *finds an MPI library installed,* `make` *will automatically produce the parallel version (*`ppuzzle`*) as well.*

5. To install the executables, run the command:

```
make install
```

To complete this step, the user will probably need to be logged in as the root user.

This will install the executables `puzzle` *and* `ppuzzle` *(the parallel version).The programs will be installed to* `/usr/local/bin` *by default. If it is necessary to have the programs installed in another directory, change with* `configure` *(see the* `INSTALL` *file for more details) or copy the executables* `src/puzzle` *and/or* `src/ppuzzle` *to the desired location.*

SUPPORT PROTOCOL 2

OBTAIN AND INSTALL TREE-PUZZLE FOR Mac OS X

This protocol describes how to obtain and install TREE-PUZZLE for the Macintosh operating system, Mac OS X.

Necessary Resources

Hardware

Macintosh system with TCP/IP Internet connection running Mac OS X and a Web browser

1. Download the current TREE-PUZZLE package for Mac OS X from *http://www.tree-puzzle.de*. It has a name like `tree-puzzle-X.X.tar.gz`, where `X.X` should be the current version.

2. Unpack the package using a program like Stuffit (*http://www.stuffit.com*), which should belong to the Mac OS X release.

This should create a directory `tree-puzzle-X.X`*, which contains* `tree-puzzle-YYY` *in its* `src` *folder (*`YYY` *indicating the compiler used). The subdirectories* `doc` *and* `data` *contain the manual and test data, respectively.*

3. Copy the Mac OS X executable to the desired location and renamed to `puzzle` or `tree-puzzle`.

This location should be in the search `PATH` *variable (see APPENDIX 1B). For convenience, create a link on the Desktop.*

OBTAIN AND INSTALL TREE-PUZZLE FOR MS WINDOWS

This protocol describes how to obtain and install TREE-PUZZLE for Windows operating systems.

Necessary Resources

Hardware

Windows system with TCP/IP Internet connection and a Web browser

1. Download the current TREE-PUZZLE package for Windows from *http://www.tree-puzzle.de*. It is named `tree-puzzle-X.X.zip`, where `X.X` is the current version.
2. Unpack the package using a program such as WinZip (*http://www.winzip.com*).

   This should create a directory `tree-puzzle-X.X`. *The subdirectories* `doc` *and* `data` *contain the manual and test data, respectively. In the* `src` *directory, there is a Windows executable,* `puzzle-windows-YYY.exe`, *where* `YYY` *states the compiler used to prepare the executable.*
3. Copy the Windows executable to the desired location and rename it as `puzzle.exe` or `tree-puzzle.exe`.

   This location should be in the Windows search path. For convenience, create a link on the Windows Desktop.

GUIDELINES FOR UNDERSTANDING RESULTS

General Aspects

As one can imagine, the outcome of an analysis is highly dependent on the data quality. In an optimal case, the data provide perfect phylogenetic information and no inconsistencies, and hence the resulting tree will show the history of the sequences, which means that the data are perfectly tree-like. Unfortunately, convergent evolution, multiple substitutions, and other processes introduce noise. Thus, careful screening of the data is necessary. TREE-PUZZLE tries to determine if the dataset is suited for phylogenetic analysis.

After running an analysis with TREE-PUZZLE, check the puzzle report file, here called `EF.phy.puzzle` or `EF.3trees.puzzle`. TREE-PUZZLE measures several features of the dataset. In the `SEQUENCE ALIGNMENT` part, it shows the fraction of constant sites as well as how many different columns (site patterns) occur in the alignment. It also checks for identical sequences in the data. Identical sequences should be removed, because they increase computation time and provide no additional information about the phylogeny of the data.

TREE-PUZZLE also estimates the nucleotide composition or amino acid composition of the alignment. It tests if the composition of each sequence (e.g., amino acids or nucleotides) deviates significantly from the average composition. Also, the gaps and ambiguous characters, like `N` in nucleotide sequences and `X` in protein sequences, are counted for each sequence. If a sequence contains many gaps and ambiguous characters, there might not be enough informative characters left to ensure a reliable placement of this sequence in the reconstructed tree.

These features of the data, as well as the resolution of the quartets (in the `QUARTET STATISTICS` part) described below, will help one to find out which of the sequences might have caused inconsistencies in the analysis (see below).

Tree Reconstruction (see Basic Protocol 1)

To reconstruct phylogenies, TREE-PUZZLE applies a three-step algorithm called Quartet Puzzling (Strimmer and von Haeseler, 1996). In the first step, the maximum-likelihood step (ML step), all possible groups of four sequences, quartets, and their three different topologies (Fig. 6.6.1) are evaluated to create a set of quartet trees supported by the data. This step also determines whether two or even all three tree topologies are almost equally good, i.e., partly resolved or unresolved topologies, respectively (Strimmer et al., 1997). Fully resolved, partly resolved, and unresolved quartets are explained in more detail below (see Likelihood Mapping for Data Quality and Quartet Support of Clusters). In the puzzling step, the supported quartet tree topologies are combined into an overall tree. Since this step is dependent on the input order, it is performed many times for randomized input orders, thus producing a large number of so-called intermediate or puzzle trees. These trees and their frequency can be output to file using the `j` option, as explained in Basic Protocol 1 (see manual for more details; Figure 6.6.2). In the final consensus step, a consensus tree is computed from the intermediate trees, which is then used to infer maximum-likelihood branch lengths and the maximum-likelihood value for the tree, as described in Felsenstein (1981). The percentage of splits (i.e., bipartitions of the dataset induced by an internal edge in a tree) that occurred in the collection of intermediate trees is used as a reliability measure for the splits in the consensus tree. The higher these so-called support values, the more confidence one may place upon the according bipartition. However, never confuse support values with bootstrap values.

If a split does not occur in >50% of the intermediate trees, it is not included in the consensus tree (McMorris and Neumann, 1983). Thus, multifurcations are possible. There is a multifurcation in the eukaryotic EF/1α subtree in Figure 6.6.8.

In the puzzle report file (`EF.phy.puzzle`), all intermediate trees occurring more often than 5% are listed. In addition, the amount of fully resolved, partly resolved, and unresolved quartets for the entire dataset is shown. TREE-PUZZLE also outputs how frequently each sequence occurs in fully resolved, partly resolved, and unresolved quartets. This is another way of displaying phylogenetic information in the data (see Likelihood Mapping below) as well as in any of the sequences. If the reconstructed tree is highly unresolved, the unresolved quartets indicate whether the dataset was not suitable for tree reconstruction (overall fraction of unresolved quartets high) or if there are sequences that should be excluded because they introduce unresolved quartets. If the amount of unresolved quartet for a sequence is high, this sequence should be discarded from the dataset (see below for more details on unresolved quartets).

If the assumption of rate heterogeneity is applied, as in the example, then the report file also displays the site specific rates of each alignment site (`RATE HETEROGENEITY` section in the puzzle report file).

Likelihood Mapping for Data Quality and Quartet Support of Clusters (see Basic Protocol 2)

Likelihood mapping (Strimmer and von Haeseler, 1997) is based on likelihood values inferred for each of the three possible tree topologies for a quartet (Fig. 6.6.1). Every likelihood value is transferred into a weight (posterior probability) by dividing it by the sum of all three likelihoods (Strimmer et al., 1997). If one of the topologies has a higher likelihood than the others, its weight will be near 1.0 while the other weights are almost zero. If two quartet topologies have similar likelihoods, their weights will be ~0.5, i.e., it is difficult to decide which is the more advantageous topology (partly resolved quartet). For an unresolved quartet, each possible topology has a weight about one-third. The three likelihood weights for a quartet add up to 1.0 and can be plotted in a three-dimensional

coordinate system, one axis for each quartet topology. Each point falls into a triangular surface between (1.0, 0.0, 0.0), (0.0, 1.0, 0.0), and (0.0, 0.0, 1.0), as shown on the left side of Figure 6.6.10. Likelihood mapping plots the likelihood weights directly into such a triangle, also called simplex (Fig. 6.6.10, right side).

The likelihood mapping output (Figs. 6.6.12 and 6.6.14) comprises two different illustrations of the distribution of quartet weights in the simplex. One simplex is divided into three areas. Each area represents the region where a maximum-likelihood reconstruction would reconstruct the tree at the corner of the simplex. The second simplex is partitioned into seven regions. The central region represents the area of unresolved quartets. The three rectangles illustrate partly resolved quartets and the three trapezoids reflect fully resolved quartets, defined by the trees in the corner (Fig. 6.6.10). Each point represents the likelihood of a single quartet.

In an unrestricted likelihood mapping, all quartets are used for analysis, whereas in a grouped analysis, quartets are chosen according to the 2 to 4 assigned clusters:

4 clusters: (a,b,c,d) with $a \in A$, $b \in B$, $c \in C$, $d \in D$, where A, B, C, and D are the clusters
3 clusters: (a,a′,b,c) with $a,a' \in A$, $b \in B$, $c \in C$ $a \in A$, where A, B, and C are the clusters
2 clusters: (a,a′,b,b′) with $a,a' \in A$, $b,b' \in B$, where A and B are the clusters

The 4-cluster analysis is applied to evaluate the support for the phylogenetic relationships of four disjoint groups of sequences (cluster A, B, C, D). The 3-cluster analysis helps, inter alia, to elucidate the reliability of an outgroup by visualizing how well the representatives of a cluster A are separated from a sister group B joined with the outgroup C. If many points are visible in the lower corners of the triangle, the chosen outgroup might be poorly separated from the ingroup; if many points are visible in center of the triangle, its sequences show saturation of the phylogenetic signal. Finally, the 2-cluster analysis reveals how many quartets support the split induced by the two clusters A and B.

The results of the two likelihood mapping analyses for `EF.phy` are given in Figures 6.6.11 and 6.6.12. Figure 6.6.11 shows that the EF dataset is well suited for phylogenetic analysis with 98.3% fully resolved, 0.8% partly resolved, and only 0.9% unresolved quartets. A large percentage of unresolved quartets would indicate that the data are not appropriate for phylogenetic analysis.

The analysis of the branching pattern within the EF-2/G sequences (Fig. 6.6.12) shows a preference for a monophyly of Crenarchaeota and Eukaryota. A percentage of 89.9% of all admissible quartets support this monophyly strongly (lower right simplex) and 98.0% of all quartets would suggest this tree, if the maximum-likelihood values of the quartet trees are considered.

Comparison of Different Tree Topologies (see Basic Protocol 3)

As mentioned above, the ML framework allows one to test competing hypotheses. Several tests have been proposed to compare phylogenetic trees (for a review, see Goldman et al., 2000). Three of these tests are implemented in TREE-PUZZLE.

The most commonly used is the pairwise KH test (Kishino and Hasegawa, 1989). This test is frequently used to compare the best tree, according to its ML value, to the other trees in the set.

Shimodaira and Hasegawa (1999) proposed a nonparametric test that is applicable if the maximum-likelihood tree, i.e., the tree with the highest likelihood, is among the members

of the set of proposed trees. Note that in a typical application it is not guaranteed that the maximum likelihood tree is present. Contrary to the KH test, which is essentially a pairwise test, the SH test compares all candidate trees simultaneously.

More recently, Strimmer and Rambaut (2002) published a method to infer confidence sets from possibly misspecified trees based on expected likelihood weights (ELW).

Before interpreting the results of the tests for one's own data, the authors of this unit suggest that one carefully study the relevant literature and the limitations of each method. When performing tests on trees, make sure that these tests are applicable. Goldman et al. (2000) explain which tests are valid for a given type of dataset. According to Goldman et al. (2000), KH tests should not be applied if trees were constructed on the basis of the alignment that is then, in turn, used to compare the ML tree against the second and third best tree topology. The Shimodaira-Hasegawa test (1999) is a valid test if the best tree is in the test set and the test can be applied for a collection of trees. KH is a pairwise test, and can be used for testing whether a tree is significantly worse than the best tree. The SH test is typically more conservative. It also has a tendency to depend on the number of trees in the test set, i.e., the larger the test set, the larger the confidence set. For more details about topology testing, especially for KH and SH tests and their applicability, refer to Goldman et al. (2000).

Basic Protocol 3 tests the following trees:

Tree 1: Eukaryota-Crenarchaeota sister groups for EF-2/G and EF-1α/Tu (Fig. 6.6.13A)
Tree 2: Bacteria-Crenarchaeota sister groups for EF-2/G and EF-1α/Tu (Fig. 6.6.13B)
Tree 3: Eukaryota-Bacteria sister groups for EF-2/G and EF-1α/Tu (Fig. 6.6.13C).

The branching orders within the kingdoms are identical to Figure 6.6.8. The test results from the puzzle report file are given in Figure 6.6.15. All tests inferred "confidence sets" comprising trees 1 and 2. Note that tree 2, which groups together Bacteria and Crenarchaeota, got a lower likelihood, but is not significantly worse.

If all puzzling step trees occurring in Basic Protocol 1 are evaluated and tested together with the tree from Figure 6.6.8, the best tree found has a log-likelihood of −18958.52 compared to a log-likelihood of −18965.87 for the tree in Figure 6.6.8. The increase in likelihood is due to the fact that the best tree is fully resolved. This increase in the number of parameters (branches in the tree) leads to a higher likelihood. However, both statistical tests (KH and SH) indicate that the Figure 6.6.8 tree is not worse than the best tree. Incidentally, the best tree is the most frequent tree among all intermediate trees.

COMMENTARY

Background Information

Most of the background information needed to understand the results, as well as to interpret the data, are discussed in Guidelines for Understanding Results, above.

Programs that aim to reconstruct large phylogenetic trees have to contend with the enormous number of possible trees (Felsenstein, 1978). TREE-PUZZLE tries to cope with that problem by dividing the task into small fractions, the quartets (Strimmer and von Haeseler, 1996). For four sequences, only three informative topologies exist (Fig. 6.6.1) and the ML evaluation of each quartet is fast. Although there is still a large number of quartets to evaluate, this is often faster than computing likelihoods for a large number of large trees. From all quartet topologies, those chosen are the ones that are best supported by the data. TREE-PUZZLE takes into account that two or even all three topologies may be equally good (Strimmer et al., 1997). The set of quartet topologies is then "puzzled" together into so-called intermediate trees repeatedly with

different orders of taxa. The set of intermediate trees offers two important advantages. The frequency of bipartitions found in the intermediate trees gives a reliability measure for the internal branches in the final tree without the necessity of running a large number of initial analyses. In addition, this set of somehow biologically reasonable trees gives insight into the set of trees that is supported by the data (see Suggestions for Further Analysis).

The use of quartets also serves other purposes. The quartets are used to visualize the "tree-likeness" and subsequently the quality of the dataset for phylogenetic analysis. The number of unresolved quartets also helps to identify problematic sequences in the data sets.

Another advantage of TREE-PUZZLE is the broad variety of evolutionary models it implements. Besides DNA and binary sequence models, TREE-PUZZLE offers several general and specialized models to reconstruct phylogenies from amino acid sequences, which are supported only by a very limited number of phylogenetic software.

Critical Parameters

Number of sequences and length of the alignment

As previously mentioned, the example dataset contains 22 sequences, and thus the reliability of the reconstructed topology depends heavily on the selection of species, which is very small for the evolutionary span it covers. Several researchers (e.g., Hillis, 1996) suggest that an increased number of sequences will increase the accuracy of the reconstructed tree. Another crucial point that deserves attention is the length of the alignment. The authors' sample alignment is 915 amino acids long. If longer sequences were available, the accuracy of the reconstructed tree would increase, and the estimation of the parameters of evolution would be more precise.

Model selection

All phylogenetic methods rely on assumptions about the process of DNA or amino acid substitutions. The confidence one puts into a phylogenetic analysis depends on the goodness of fit, i.e., how appropriate is the model to describe the data? In a statistical framework, the goodness of fit is typically explored applying a likelihood ratio statistics. When the models are nested, (the null model is a special case of the alternative model), the differences in the log-likelihood between both models is typically χ^2-distributed (Posada and Crandall, 1998; *UNIT 6.5*). More recently, the Akaike Information Criterion and Bayesian approaches have been found to be better choices to compare and select an appropriate model of evolution (Posada and Buckley, 2004). To select the best model, a variety of programs are available, e.g., ModelTest (Posada and Crandall, 1998; Posada and Buckley, 2004; *UNIT 6.5*), which is applicable for DNA sequences. This program can be used to find the best model for a fixed tree topology. If, however, the tree topologies are different (the models are not nested), one needs to apply Monte Carlo simulations as suggested by Goldman (1993a,b).

Suggestions for Further Analysis

As previously noted, all methods for reconstructing large phylogenetic trees, i.e., trees with more than 10 to 15 taxa, try to contend with the enormous number of possible trees (Felsenstein, 1978) by heuristic search methods (Swofford et al., 1996). Because of this, none of these methods are guaranteed to find the overall best tree. Each method has its advantages and drawbacks, which influence the result in a way that is not fully understood. Hence, "the one and only" method to reconstruct trees is not available. The authors suggest applying different methods to reconstruct trees, including maximum-likelihood, maximum-parsimony, and distance-based methods. If the methods provide the same tree topologies, then one may have some confidence in the results. If all these methods produce different tree topologies, then one should interpret the data with great care and perform further analyses to find out what is going on (Sanderson and Shaffer, 2002).

In this context, TREE-PUZZLE can be used as a generator for data-driven plausible trees. For example, one can analyze the intermediate trees, which may be different from the consensus tree, to study the distribution of different trees, thereby obtaining an indication of noise in the data. Alternatively, one may use the set of intermediate trees to apply the tests outlined in the section about comparison of trees.

In conclusion, there is no one ideal method for phylogenetic analysis. Each dataset deserves its own careful analyses guided by the results of the rich collection of tree-building methods (Swofford et al., 1996). Finally, it is good to remember that, sometimes, a tree is simply not the best way to visualize the data.

Literature Cited

Adachi, J. and Hasegawa, M. 1996. Model of amino acid substitution in proteins encoded by mitochondrial DNA. *J. Mol. Evol.* 42:459-468.

Dayhoff, M.O., Schwartz, R.M., and Orcutt, B.C. 1978. A model of evolutionary change in proteins. *In* Atlas of Protein Sequence Structure, Vol. 5 (M.O. Dayhoff, ed.) pp. 345-352. National Biomedical Research Foundation, Washington, D.C.

Edwards, A.W.F. and Cavalli-Sforza, L.L. 1964. Reconstruction of evolutionary trees. *In* Phenetic and Phylogenetic Classification (V.H. Heywood and J. McNeill, eds.) pp. 67-76. Systematics Association, London.

Felsenstein, J. 1978. The number of evolutionary trees. *Syst. Zool.* 27:27-33.

Felsenstein, J. 1981. Evolutionary trees from DNA sequences: A maximum likelihood approach. *J. Mol. Evol.* 17:368-376.

Felsenstein, J. 1984. Distance methods for inferring phylogenies: A justification. *Evolution* 38:16-24.

Felsenstein, J. 2004. Inferring Phylogenies. Sinauer Associates, Sunderland, Mass.

Goldman, N. 1993a. Statistical tests of models of DNA substitution. *J. Mol. Evol.* 36:182-198.

Goldman, N. 1993b. Simple diagnostic statistical tests of models for DNA substitution. *J. Mol. Evol.* 37:650-661.

Goldman, N., Anderson, J.P., and Rodrigo, A.G. 2000. Likelihood-based tests of topologies in phylogenetics. *Syst. Biol.* 49:652-670.

Gu, X., Fu, Y.-X., and Li, W.-H. 1995. Maximum likelihood estimation of the heterogeneity of substitution rate among nucleotide sites. *Mol. Biol. Evol.* 12:546-557.

Hasegawa, M., Kishino, H., and Yano, T. 1985. Dating the human-ape split by a molecular clock of mitochondrial DNA. *J. Mol. Evol.* 22:160-174.

Henikoff, S. and Henikoff, J.G. 1992. Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. U.S.A.* 89:10915-10919.

Hillis, D.M. 1996. Inferring complex phylogenies. *Nature* 383:130-131.

Iwabe, N., Kuma, K.-I., Hasegawa, M., Osawa, S., and Miyata, T. 1989. Evolutionary relationship of Archaebacteria, Eubacteria, and Eukaryotes inferred from phylogenetic trees of duplicated genes. *Proc. Natl. Acad. Sci. U.S.A.* 86:9355-9359.

Jones, D.T., Taylor, W.R., and Thornton, J.M. 1992. The rapid generation of mutation data matrices from protein sequences. *Comput. Appl. Biosci.* 8:275-282.

Jukes, T.H. and Cantor, C.R. 1969. Evolution of protein molecules. *In* Mammalian Protein Metabolism (H.N. Munro, ed.). Academic Press, New York.

Kimura, M. 1980. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *J. Mol. Evol.* 16:111-120.

Kishino, H. and Hasegawa, M. 1989. Evolution of the maximum likelihood estimate of the evolutionary tree topologies from DNA sequence data, and the branching order in Hominoidea. *J. Mol. Evol.* 29:170-179.

Lanave, C., Preparata, G., Saccone, C., and Serio, G. 1984. A new method for calculating evolutionary substitution rates. *J. Mol. Evol.* 20:86-93.

McMorris, F.R. and Neumann, D.A. 1983. Consensus functions defined on trees. *Math. Soc. Sci.* 4:131-136.

Müller, T. and Vingron, M. 2000. Modeling amino acid replacement. *J. Comput. Biol.* 7:761-776.

Page, R.D. and Holmes, E.C. 1998. Molecular Evolution: A Phylogenetic Approach. Blackwell Science, Oxford.

Posada, D. and Crandall, K.A. 1998. MODELTEST: Testing the model of DNA substitution. *Bioinformatics* 14:817-818.

Posada, D. and Buckley, T. 2004. Model selection and model averaging in phylogenetics: Advantages of akaike information criterion and Bayesian approaches over likelihood ratio tests. *Syst. Biol.* 53:793-808.

Sanderson, M.J. and Shaffer, H.B. 2002. Troubleshooting molecular phylogenetic analyses. *Annu. Rev. Ecol. Syst.* 33:49-72.

Schmidt, H.A., Strimmer, K., Vingron, M., and von Haeseler, A. 2002. TREE-PUZZLE: Maximum likelihood phylogenetic analysis using quartets and parallel computing. *Bioinformatics* 18:502-504.

Schöniger, M. and von Haeseler, A. 1994. A stochastic model for the evolution of autocorrelated DNA sequences. *Mol. Phyl. Evol.* 3:240-247.

Shimodaira, H. and Hasegawa, M. 1999. Multiple comparisons of log-likelihoods with applications to phylogenetic inference. *Mol. Biol. Evol.* 16:1114-1116.

Strimmer, K. and von Haeseler, A. 1996. Quartet puzzling: A quartet maximum likelihood method for reconstructing tree topologies. *Mol. Biol. Evol.* 13:964-969.

Strimmer, K. and von Haeseler, A. 1997. Likelihood mapping: A simple method to visualize phylogenetic content of a sequence alignment. *Proc. Natl. Acad. Sci. U.S.A.* 94:6815-6819.

Strimmer, K. and Rambaut, A. 2002. Inferring confidence sets of possibly misspecified gene trees. *Proc. R. Soc. Lond. B* 269:137-142.

Strimmer, K., Goldman, N., and von Haeseler, A. 1997. Bayesian probabilities and quartet puzzling. *Mol. Biol. Evol.* 14:210-213.

Swofford, D.L., Olsen, G.J., Waddell, P.J., and Hillis, D.M. 1996. Phylogeny reconstruction. *In* Molecular Systematics, 2nd ed. (D.M. Hillis, C. Moritz, and B.K. Mable, eds.) pp. 407-514. Sinauer Associates, Sunderland, Mass.

Tamura, K. and Nei, M. 1993. Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. *Mol. Biol. Evol.* 10:512-526.

Tavare, S. 1986. Some probabilistic and statistical problems on the analysis of DNA sequences. *Lec. Math. Life Sci.* 17:57-86.

Whelan, S. and Goldman, N. 2001. A general empirical model of protein evolution derived from multiple protein families using a maximum likelihood approach. *Mol. Biol. Evol.* 18:691-699.

Key References

Felsenstein, 2004. See above.
A comprehensive textbook covering almost all areas of phylogenetic inference.

Goldman et al., 2000. See above.
A comprehensive review discussing tests for tree topologies and their applicability.

Page and Holmes, 1998. See above.
A well written textbook about phylogenetics and its applications.

Sanderson and Shaffer, 2002. See above.
A good review on problems in phylogeny reconstruction.

Strimmer and von Haeseler, 1996. See above.
An original publication of the Quartet Puzzling method.

Strimmer and von Haeseler, 1997. See above.
A more detailed description of Likelihood Mapping.

Swofford et al., 1996. See above.
An excellent introduction to the rich collection of phylogenetic methods.

Internet Resources

http://www.tree-puzzle.de
TREE-PUZZLE Web site.

http://rdp8.cme.msu.edu/download/programs/TreeTool/
TreeTool Web site (tree-drawing program).

http://taxonomy.zoology.gla.ac.uk/rod/treeview.html
TreeView Web site (tree-drawing program, see UNIT 6.2).

http://evolution.genetics.washington.edu/phylip/software.html
Joe Felsenstein's list of tree-reconstruction and -drawing programs.

http://www.ghostscript.com/
GhostScript Web page (PostScript viewer and converter).

Contributed by Heiko A. Schmidt and Arndt von Haeseler
Center for Integrative Bioinformatics Vienna (CIBIV), Max F. Perutz Laboratories (MFPL), Vienna, Austria
University of Vienna, Austria
Medical University Vienna, Austria
University of Veterinary Medicine, Vienna, Austria