Inferring Phylogenies from Protein Sequences by Parsimony, Distance, and Likelihood Methods

Joseph Felsenstein

Department of Genetics

University of Washington

Box 357360

Seattle, Washington 98195-7360

USA

E-mail: joe@genetics.washington.edu

(Send proofs to me at above address)

Running Head: Phylogenies from Protein Sequences

The first molecular sequences available were protein sequences, so it is not surprising that the first papers on inferring phylogenies from molecular sequences described methods designed for proteins. Eck and Dayhoff[1] described the first molecular parsimony method, with amino acids as the character states. Fitch and Margoliash[2] initiated distance matrix phylogeny methods with analysis of cytochrome sequences. Neyman[3] presented the first likelihood method for molecular sequences, using a model of symmetric change among all amino acids.

After a long period in which attention shifted to nucleotide sequences, attention is again being paid to models in which the amino acid sequences explicitly appear. This is not only because of the increased availability of protein sequence data, but because the conservation of amino acid sequence and protein structure allows us to bring more information to bear on ancient origins of lineages and of genes.

I will here briefly review the work on using protein sequence and structure to infer phylogeny, in the process describing some methods of my own.

Parsimony

Eck and Dayhoff's paper[1] did not describe their algorithms in enough detail to reproduce them, but it is apparent that the model of amino acid sequence evolution they used did not take the genetic code into account. It simply considered the amino acids as 20 states, with any change of state able to result in any of the other 19 amino acids. The realization that more information could be extracted by explicitly considering the code shortly led to more complex models. Fitch and Farris[4] gave an approximate algorithm to calculate for any set of amino acid sequences, on a given tree, how many nucleotide substitutions must, at a minimum, have occurred. As certain amino acid replacements would then require 2 or 3 base substitutions, this would differentially weight amino acid replacements. Moore[5,6] had already presented an exact, though more tedious, algorithm to count the minimum number of nucleotide substitutions needed, and he pointed out the approximate nature of Fitch and Farris's method[7].

These papers might have settled the matter for the parsimony criterion, except that they count as equally serious those nucleotide substitutions that do and do not change the amino acid. For example, we might have a Phenylalanine that is coded for by a UUU, which ultimately becomes a Glutamine that is coded for by a CAA. This requires three nuclotide substitutions. It is possible for one of these to be silent, as we can go from UUU (Phe) → CUU (Leu) → CUA (Leu) → CAA (Glu). Presumably the second of these changes will not be as improbable as the others, as it will not have to occur in the face of natural selection against change in the amino acid, or wait for a change of environment or genetic background that favors the amino acid replacement.

In the PROTPARS program of my PHYLIP package of phylogeny programs, I have introduced (in 1983) a parsimony method that attempts to reflect this. In the above sequence it counts only two changes, allowing the silent substitutions to take place without penalty. In effect the method uses the genetic code to designate which pairs of amino acids are adjacent, and allows change only among adjacent states. Sankoff[8] and Sankoff and Rosseau[9] have presented a generalized parsimony algorithm that allows us to count on a given tree topology how many changes of state are necessary, where we can use an arbitrary matrix of penalties for changes from one state to another. The PROTPARS algorithm is equivalent to Sankoff's algorithm, being quicker but less general.

The set of possible amino acid states in the PROTPARS algorithm has 23 members, these being the 20 amino acids plus the possibilities of a gap and a stop codon. Serine is counted not as one amino acid but as two, corresponding to the two "islands" of serine codons in the genetic code. These are {UCA, UCG, UCC, UCU} and {AGU, AGC}, which make serine the only amino acid whose codons fall into two groups that cannot be reached from each other by a single mutation. PROTPARS copes with this by regarding them as two amino acid states (ser1 and ser2) and treats an observation of "serine" as an ambiguity between these two.

Imagine that we know, for a node in the tree, the set of amino acid states that are possible at this node. If the node is a terminal (tip) species, these are just the observed amino acid, there being more than one if serine is observed or if any of asn, gln, or glx are observed. There is also the possibility that the amino acid is unknown, but known not to be a gap, and the possibility that the amino acid could be any one including a gap. More complex ambiguities are also possible and can arise in the process of reconstruction of the states at interior nodes in the tree. Any of these can be represented by designating the members of the set $\underline{S}_0$ of possible states.

Given the particular version of the genetic code that we are using, we can also precompute, for each amino acid $\underline{a}$, the set $\underline{N_a}$ of amino acid states that are one or fewer steps away. In PROTPARS, gaps are counted as being 3 steps away from all the amino acids and from stop codons. Having these precomputed sets allows us to take the sets $\underline{S_0}$ at the tips of the tree, and compute for them $\underline{S_1}$ and $\underline{S_2}$, the sets of amino acid states 1 or fewer steps away, and 2 or fewer steps away. In our program, all states, including gaps, are 3 or fewer steps away, so that we do not need a set $\underline{S_3}$. In PROTPARS the three sets $\underline{S_0}$, $\underline{S_1}$, and $\underline{S_2}$ are updated down the tree, and the number of steps needed for the tree counted, in the following way.

Imagine that there is an internal node in the tree with two descendants, and whose sets of possible states are the $\underline{L_i}$ and the $\underline{R_i}$. We are computing the sets $\underline{S_i}$ for the internal node. First, $\underline{L_0}$ and $\underline{R_0}$ are compared. If they are the same then the

$\underline{L_i}$ must be identical to the $\underline{R_i}$, and the $\underline{S_i}$ are simply set to be the $\underline{L_i}$, and no steps are counted. Otherwise, we compute the four sets

$$
\begin{aligned}
T_0 &= L_0 \cap R_0 \\
T_1 &= (L_1 \cap R_0) \cup (L_0 \cap R_1), \\
T_2 &= (L_2 \cap R_0) \cup (L_1 \cap R_1) \cup (L_0 \cap R_2), \\
T_3 &= R_0 \cup (L_2 \cap R_1) \cup (L_1 \cap R_2) \cup L_0.
\end{aligned}
\tag{1}
$$

They are computed one after the other. Their interpetation is straightforward. For example, $\underline{T_1}$ is the set of amino acid states that, if present at the internal node, requires one step to give rise to $\underline{L_0}$ and none to give rise to $\underline{R_0}$, or else one step to give rise to $\underline{R_0}$ and none to give rise to $\underline{L_0}$. Thus it is the set of states which, if present at the interior node, require one extra step in the subtree that is above that node. As soon as one of these, say $\underline{T_k}$, turns out to be nonempty, we know that a minimum of $\underline{k}$ more steps will be needed at this node, and that the set $\underline{S_0}$ for that node will be $\underline{T_k}$. $\underline{T_3}$ at least must be nonempty, as it contains the union of $\underline{R_0}$ and

$\underline{L_0}$.

Now, having found $\underline{S_0}$, all we need to do is to compute $\underline{S_1}$ and $\underline{S_2}$ for the internal node. The formulas for doing so are

$$S_k = \bigcup_{a \in S_{k-1}} N_a, \qquad k = 1, 2 \tag{2}$$

This of course does not need to be done if $\underline{L_0} = \underline{R_0}$, as the sets $\underline{L_1}$ and $\underline{L_2}$ (or $\underline{R_1}$ and $\underline{R_2}$) can then be used directly.

This method of calculation using sets is equivalent to having a vector of numbers, one for each amino acid state, which are 0, 1, 2, or 3. The Sankoff algorithm asks us to specify for each state the number of extra steps that would be required above that point in the tree if that state existed in that internal node. In our model the possible values for the number of extra steps are 0, 1, 2, and 3. The sets $\underline{S_i}$ are just the amino acids which would have the number of extra steps less than or equal to $\underline{i}$. The algorithm is then equivalent to the appropriate application of the Sankoff algorithm. It could probably be speeded up further, as most of the time the set $\underline{S_2}$ is the set of all amino acids, and that could be used as the basis for some further economies.

Figure 1 shows the sets that would be stored on a small sample tree for one amino acid position, and the counting of steps. At each node the three sets $\underline{S_0}$, $\underline{S_1}$, and $\underline{S_2}$ are shown, and at interior nodes the number of steps that are counted are also shown in circles. There are 4 different amino acids at the tips of the tree. If any amino acid could change to any other the tree would require only 3 steps, but in my protein parsimony model it requires 5.

Protein parsimony methods exactly equivalent to PROTPARS are also available in the programs PAUP and MacClade, using predefined matrices of costs of subtitution between amino acid states, with the costs being taken into account by the Sankoff algorithm.

Distances

Distance-matrix methods calculate for every pair of sequences an estimate of the branch length separating them, where branch length is the product of time and rate of evolution. That tree is then chosen that, by some criterion, makes the best prediction of these pairwise distances. For protein sequences we need to specify a probabilistic model of evolution. Jukes and Cantor[10] were the first to do this for protein sequences (see also Farris[11]). This model was highly oversimplified, as it had equal probabilities of change between all pairs of amino acids. Dayhoff and Eck[12]and Dayhoff et. al.[13] empirically tabulated probabilities of change between amino acids over short evolutionary times, producing a table of transition probabilities between amino acids. This model does not take explicit account of the genetic code, and is subject to errors from the limited sample size on which it was based. Nevertheless the genetic code should affect its transition probabilities, and so should the biochemical properties of the amino acids. A more recent empirical model of amino acid change is that of Jones et. al.[14]. They have also produced models for specific subclasses of proteins, that may be more useful in those contexts[15]. Other recent compilations of scoring matrices for evaluating the similarity of amino acid sequences[16,17] are not in the form of transition probability tables. For this reason they cannot be used to compute the branch length estimates that we require here.

A naive alternative to these empirical matrices is to divide the amino acids into a number of categories, based on their chemical properties. Suppose that we imagine mutations occurring in the genetic code table, with the starting points being codons generated at random from a given base composition. Now imagine single base substitutions. If these do not change the biochemical class of the amino acid, they are accepted; if they do, they are only accepted with probability p. We omit the stop codons from consideration: if either the starting point or the destination of a change is a stop codon, the change is not made. This model, once given the amino acid categories, the base frequencies and the probability $\underline{p}$, generates a transition probability table between all pairs of amino acids.

Version 3.5 of PHYLIP contains a program, PROTDIST, which computes distances based either on the PAM001 model[13] and the transition probability matrix generated by the categories model. It also can compute distances using the formula of Kimura[18] which bases the distance on the fraction of amino acids shared between the sequences, without regard to which amino acids they are. The categories model, as implemented in PROTDIST, can use several different genetic codes (the universal code and several kinds of mitochondrial code). Three categorizations of the amino acids are used, one the categories given by George et. al.[19] , one from a categorization in a "baby biochemistry" text, and one the opinion of a colleague. Interestingly, all three of these turn out to be subdivisions of one linear order of amino acids. We have found that a value of $\underline{p} = 0.45$ brings the ratio of between- to within-category change in the category model of George et. al.[19] close to that in the Dayhoff model. In the next release (4.0) of PHYLIP, we hope to expand the range of models by including the model of Jones et. al.[14], and allowing for a Gamma distribution of evolutionary rates among sites, in the manner of Jin and Nei[20] and Nei et. al.[21].

Given the evolutionary model, we use maximum likelihood estimation to compute the distances. In effect we are specifying a two-species tree, with but one branch, between the pair of species, and estimating that branch length by maximum likelihood. If we observe $\underline{n_{ij}}$ changes between amino acids $\underline{i}$ and $\underline{j}$, and if the model we are using has equilibrium frequency $\underline{f_i}$ for amino acid $\underline{i}$ and transition probability $\underline{P_{ij}}(\underline{t})$ over time $\underline{t}$, the expected fraction of sites which will have amino acid $\underline{i}$ in one species and $\underline{j}$ in the other is $\underline{f_i}\underline{P_{ij}}(\underline{t})$. The PAM001 matrix gives the conditional probabilities $\underline{P_{ij}}$, but they are not reversible. In order to make a reversible model that is as close as possible to PAM001, we have used instead

$$Q_{ij} = \left(f_i P_{ij} + f_j P_{ji}\right)/2. \tag{3}$$

This gives us symmetric joint probabilities of observing $\underline{i}$ and $\underline{j}$ in two closely related sequences. Suppose that the $\underline{\underline{\mathbf{M}}}$ are transition probabilities that would lead to the joint probabilities $\underline{\underline{\mathbf{Q}}}$, and that $\underline{\pi}$ is the vector of equilibrium frequencies which is implied by $\underline{\underline{\mathbf{M}}}$. We start out knowing $\underline{\underline{\mathbf{Q}}}$ but not $\underline{\underline{\mathbf{M}}}$ or $\underline{\pi}$. It is not hard to show that the eigenvalues of $\underline{\underline{\underline{\pi}'\mathbf{M}}}$ are the same as the eigenvalues of $\underline{\underline{\mathbf{Q}}}$, and the eigenvalues of $\underline{\underline{\mathbf{M}}}$ can also be directly derived from those of $\underline{\underline{\mathbf{Q}}}$. The eigenvalues and eigenvectors of $\underline{\underline{\mathbf{M}}}$ are computed in this way (they are precomputed in the PAM001 case and computed by the program in the categories cases).

From the eigenvalues and eigenvectors of $\underline{\underline{\mathbf{M}}}$ we can readily compute the transition probabilities $\underline{M_{ij}}(\underline{t})$, and their derivatives with respect to $\underline{t}$. The likelihood which we must maximize is

$$L = \prod_i \prod_j (\pi_i M_{ij}(t))^{n_{ij}} \tag{4}$$

The log-likelihood is maximized over values of $\underline{t}$ by Newton-Raphson iteration,

The resulting distance computation is not fast, but it seems adequate. However, it makes one assumption that is quite severe. All amino acid positions are assumed to change at the same rate. This is unrealistic. To some extent we can compensate for this by correcting the distances by using the approach of Jin and Nei[20]. However there is information that is being lost by doing this. We would like to be able to use the variation in an amino acid position in one part of the data set to infer whether that position allowed change to occur at a high rate, and thus to help us evaluate other parts of the same data set. But no distance matrix method can do this, as they consider only pairs of sequences.

Likelihood Methods

Neyman[3] and Kashyap and Subas[22] developed maximum likelihood methods for inferring phylogenies from protein data. They used the highly-oversimplified Jukes-Cantor[10] model of symmetric change among amino acids, and they could not handle more than 3 or 4 sequences in the tree in a reasonably exact way. I showed[23] how to make the likelihood computations practical for larger numbers of species. Likelihood methods for proteins have not been developed further until recently, because of the computational burden. Where nucleotide sequence likelihood methods use a 4×4 transition probability matrix, in protein models these must be either 20×20 or 64×64, and thus either 25 or 256 times as much computation. With increased speed of desktop and laboratory computers, developing a reasonable likelihood method for protein sequences has become more of a priority.

Adachi and Hasegawa[24] and Adachi et. al.[25] have developed such a method, using the Dayhoff PAM matrix[13] as the transition probability matrix among amino acid states, but without any direct use of the genetic code. Their program, which is similar to existing DNA likelihood programs but has some effort put into requiring fewer evaluations of the likelihood, is available in their MOLPHY package from their ftp site at `sunmh.ism.ac.jp`.

It is tempting to develop a method that takes the genetic code explicitly into account. In principle one could have 64 states, one for each codon, and regard the amino acids as ambiguous observations (for example, alanine would be regarded as an observation of “either TCA or TCG or TCC or TCT”). The computational difficulties would be severe. One could also hope to take into account both observed protein sequence and the underlying DNA sequence, which is often known. Hein[26] and Hein and Stovlbaek[27,28] have made a start on such models.

A more serious limitation of existing protein maximum likelihood models is that they assume that all positions change at the same expected rate. This assumption has been removed from nucleotide sequence likelihood models, using Hidden Markov Model techniques[29,30,31,32]. Its extension to proteins is straightforward and badly needed, but does promise to slow down the computer programs severalfold.

Structure, Alignment, and Phylogeny

Beyond any of these complications is the challenge of taking protein structure into account. Researchers on analysis of RNA sequences have found that there is a synergism between inferences of phylogeny, alignment, and structure. It is just beginning to become widely recognized that the same will be true with proteins, the advantages being probably greater. Structure-based Hidden Markov Models (HMMs) have been used to improve sequence alignment of proteins, although without taking phylogeny into account[33,34]. Three-dimensional protein structures can be used to infer phylogenies[35]. Structural context affects not only amino acid composition, but the substitution process itself[36]. When residues interact, there may result patterns of compensating substitutions. This has begun to be examined for proteins[37].

In RNAs, phylogenies and inferences of structure are increasingly important to each other. Patterns of compensating substitutions are strong, and have recently led to mathematical models of this substitution process[38,39]. One can imagine a unified process of inference for proteins and protein-coding regions that simultaneously infers phylogeny, alignment, secondary structure, and three-dimensional structure. The computational problems will be severe but many of the components needed are already being worked on.

Having coordinated our inferences of structure and evolutionary history, we will then be free to dream about function as well.

Acknowledgments

This work has been supported by grants from the National Science Foundation (DEB-9207558) and the National Institutes of Health (1 R01 GM 51929-01).

Literature Cited

1. R. V. Eck and M. O. Dayhoff, "Atlas of Protein Sequence and Structure 1966." Natl. Biomed. Res. Found., Silver Spring, Maryland, 1966.

2. W. M. Fitch, and E. Margoliash. Science 155, 279 (1967)

3. J. Neyman, in "Statistical Decision Theory and Related Topics." (S. S. Gupta and J. Yackel, eds.), p. 1. Academic Press, New York, 1971.

4. W. M. Fitch and J. S. Farris, J. Mol. Evol. 3, 263 (1974).

5. G. W. Moore, J. Barnabas, and M. Goodman, J. Theor. Biol. 38, 459 (1973).

6. G. W. Moore, J. Theor. Biol. 66, 95 (1977).

7. G. W. Moore, in "Genetic Distance," (J. F. Crow and C. Denniston, eds.), p. 105. Plenum Press, New York (1974).

8. D. Sankoff, SIAM J. Appl. Math. 28, 35 (1975).

9. D. Sankoff and P. Rousseau, Math. Progr. 9, 240 (1975).

10. T. H. Jukes and C. Cantor, in "Mammalian Protein Metabolism," (M. N. Munro, ed.), p. 21. Academic Press, New York (1969).

11. J. S. Farris, Am. Nat. 107, 531 (1973).

12. M. O. Dayhoff and R. V. Eck, "Atlas of Protein Sequence and Structure 1967-1968," Natl. Biomed. Res. Found, Silver Spring, Maryland, (1968).

13. M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt, in (Dayhoff, M.O. ed.) "Atlas of Protein Sequence and Structure, Vol 5, Suppl 3," p. 345. National

Biomedical Research Foundation, Washington, D.C. (1978).

14. D. T. Jones, W. R. Taylor, and J. M. Thornton, Comput. Appl. Biosci. 8, 275 (1992).

15. D. T. Jones, W. R. Taylor, and J. M. Thornton, FEBS Lett. 339, 269 (1994).

16. G. H. Gonnet, M. A. Cohen, S. A. Benner, Science 256, 1443 (1992).

17. S. Henikoff and J. G. Henikoff, Proc Natl Acad Sci USA 89, 10915 (1992).

18. M. Kimura, "The Neutral Theory of Molecular Evolution," Cambridge University Press, Cambridge (1983).

19. D. G. George, W. C. Barker, and L. T. Hunt, this series, vol. 183, p. 333 (1990).

20. L. Jin and M. Nei, Mol. Biol. Evol. 7, 82 (1990).

21. M. Nei, R. Chakraborty, and P. A. Fuerst, Proc Natl Acad Sci USA 73, 4164 (1976).

22. R. L. Kashyap and S. Subas, J. Theor. Biol. 47, 75 (1974).

23. J. Felsenstein, J. Mol. Evol. 17, 368 (1981).

24. J. Adachi and M. Hasegawa, Jpn. J. Genet. 67, 187 (1992).

25. J. Adachi, Y. Cao, and M. Hasegawa, J. Mol. Evol., 36, 270 (1993).

26. J. Hein, J Theor Biol. 167, 169 (1994). J

27. J. Hein and J. Stovlbaek, J. Mol. Evol. 38, 310 (1994).

28. J. Hein, and J. Stovlbaek, J. Mol. Evol. 40, 181 (1995).

29. J. Felsenstein and G. A. Churchill, Mol. Biol. Evol. in press, (1996).

30. Z. Yang, Mol. Biol. Evol. 10, 1396 (1994).

31. Z. Yang J. Mol. Evol. 39, 306 (1994).

32. Z. Yang Genetics 139, 993 (1995).

33. P. Baldi, Y. Chauvin, T. Hunkapiller, and M. A. McClure Proc. Natl. Acad. Sci. USA 91, 1059 (1994).

34. A. Krogh, M. Brown, I. S. Mian, K. Sjölander, and D. Haussler, J. Mol. Biol. 235, 1501 (1994).

35. M. S. Johnson, A. Sali, and T. L. Blundell, this series, vol. 183, p. 670 (1990).

36. J. Overington, D. Donnelly, M. S. Johnson, A. Sali, and T. L. Blundell, Protein Sci. 1, 216 (1992).

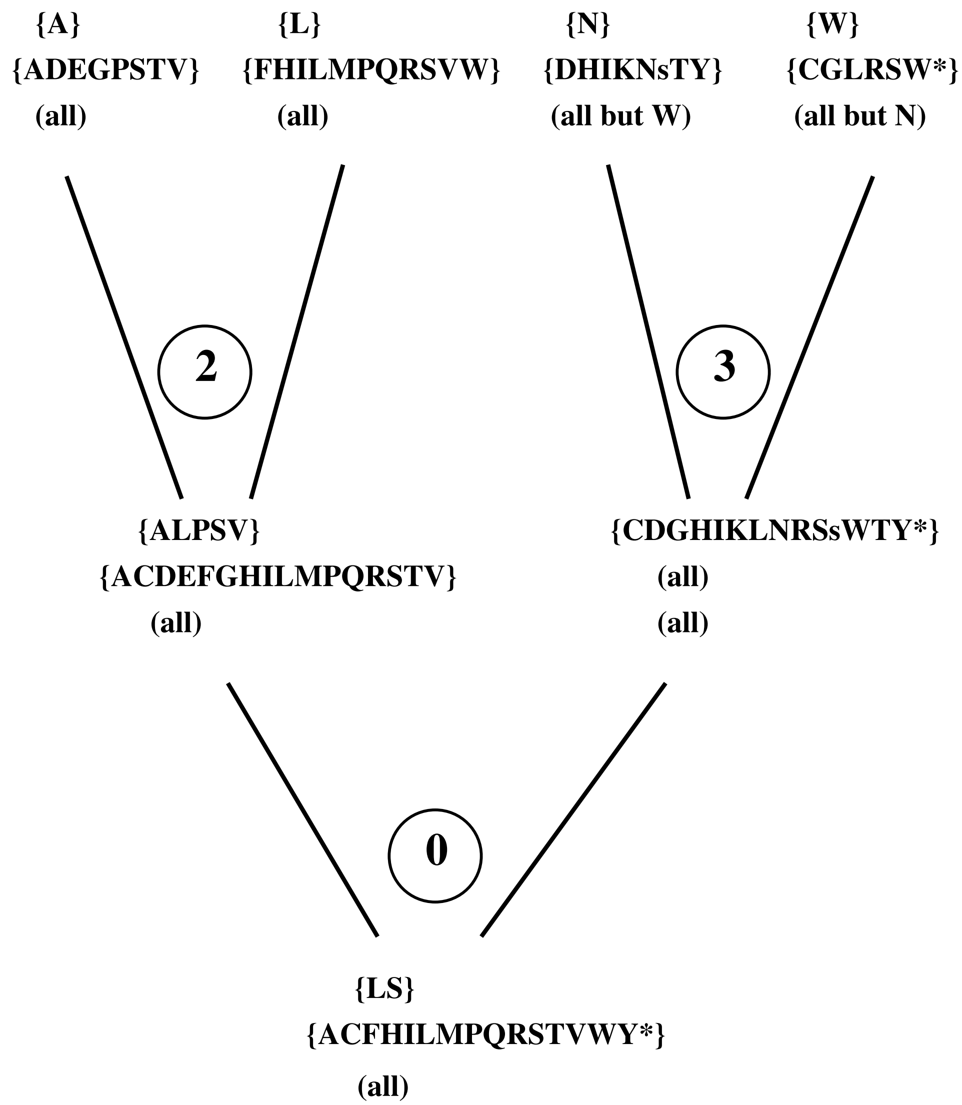
37. W. R. Taylor and K. Hatrick, Protein Eng. 7, 341 (1994).

38. E. R. M. Tillier, J. Mol. Evol. 39, 409 (1994).

39. E. R. M. Tillier and R. A. Collins, Mol. Biol. Evol. in press (1994).

Figure Captions

FIG. 1. A small tree with the calculation of the sets $\underline{S_0}$, $\underline{S_1}$, and $\underline{S_2}$ shown at each node, for a site where the tips have amino acid states alanine, leucine, asparagine, and trypthophane, respectively. The sets are shown as sets of one-letter amino acid representations. S and s are the two codon "islands" of serine, and "*" representd stop codons. The number of steps counted at each fork is shown in a circle.


{A}
{ADEGPSTV}
(all)
{L}
{FHILMPQRSVW}
(all)
{N}
{DHIKNsTY}
(all but W)
{W}
{CGLRSW*}
(all but N)
2
3
{ALPSV}
{ACDEFGHILMPQRSTV}
(all)
{CDGHIKLNRSsWTY*}
(all)
(all)
0
{LS}
{ACFHILMPQRSTVWY*}
(all)