How To Grow a Phylogenetic Tree?

CS Project group

24th March 2005

Contents

Chapter 1

Preface

How to grow a phylogenetic tree? This sounds like the title of a botanical manual. This report is, however, far from that. This is because a phylogenetic tree isn't something botanical, but rather something artificial. We, a group of third-year mathematics students, were asked to construct so called phylogenetic trees.

Phylogenetic is a word which comes from the Greek words $\phi\upsilon\lambda o$ and $\gamma\eta\nu\varepsilon\sigma\eta\varsigma$, which mean *tribe* or *race* and *origin.* This indicates that the subject of phylogenetic trees is strongly related to that of evolution and DNA. The meaning of "phylogenetic tree" confirms this indication. A phylogenetic tree is a figure which resembles a tree showing the evolutionary interrelationships among various species or other entities that are believed to have a common ancestor.

In other words a phylogenetic tree is a classification of all living things according to their DNA. This is a very 'hot' topic in the field of bioinformatics. Therefore, a lot of universities and institutes have researched this particular subject for some time. Furthermore, there already exists a plethora of 'tree drawing' programmes on the web, such as *BLAST* and *CLUSTAL.*

Our assignment regarding those trees was to write such a computer programme, which compares two nucleotide (DNA) sequences of two different species, calculates their similarity and then draws two 'branches' according to their similarity. By inserting more species, more branches will be added and a whole tree 'grows' on your computer screen. Our group consisted of ten people and our time span was one semester to complete this programme.

After completing the programme we called it PhyloGen. Not a very original name but we found it adequate. In this report you will find our work, our problems and our findings.

Chapter 2

The biological and historical basis

2.1 The origin of the phylogenetic tree

In this chapter we are not looking for an answer to the question
How did life begin and what was the first species?, instead we will look at which point the theories about this subject started. When, where, and why did mankind start thinking about evolutionary theories. We will try to determine the origin and development of these theories in order to give us the background necessary to understand what we are dealing with here.

As far as we know the first people to start thinking about these matters were the Greeks, this happened more then 2300 years ago. Aristotle (384-322 BC) came up with a theory which ordered species from the lowest to the highest. This idea is known as the Scala Naturae, or The Great Chain of Being in his views species could never change and thus the rankings never could either. These days this is called the doctrine of fixed species. Aristotle's ideas have influenced thinkers around the world for hundreds of years.

A good example of this is the theory of Carolus Linnaus (1707-1778). He created a binomial system which standardized the methods of naming species. Every name would consist of two words, the first to indicate the group to which the specie belonged, and the second one would be the name for a particular specie. This made the names of species much easier to work with. For example the briar rose is known as the rosa canina in his system, whereas before him it was known as the Rosa sylvestris alba cum rubore, folio glabro. The basics of his system are still in use on this day.

New views only started to develop just after him, the first attempt to clarify the evolution of species was Jean Baptiste de Lamarck (1744-1829). He stated that species evolved, pursuing perfection, becoming more complex and better adapted to the environment. He explained his beliefs in two laws.

The first one stated that changes in the environment leads to a change in the needs of species, which then changes their behavior. According to Lamarck the

change in behavior leads to physical changes in the specie. His second law states that these changes are inheritable.
In his views there is no possibility for a specie and his ancestor to exist over a longer period of time. The ancestor is less perfect and will disappear from the face of the world.

Chapter 3

Growing the tree

3.1 The soil

How to grow a phylogenetic tree? Before we can start looking for the answers to this question we have to make some assumptions about the soil from which it will grow.

First of all, we will leave the choice of which type of soil the tree is to be grown from to the user of the program. In other words, we will not use a standard database; this has to be entered by the user.

This is because the goal of this project is not to make an accurate map of the animal history of the world, but rather to make a program which could be used for numerous scientific projects. From large scale projects like mapping the marine life in the Atlantic ocean to smaller scale projects such as comparing different types of native horses in Ireland.

Secondly, in a way this follows from our first point, we will make no choice in the type of alignment being used to compare two nucleotide sequences.

Instead we will leave the choice between local and global alignment once again to the user of the program. This is because there is no clear answer to the question which of the two is better.

Global alignment compares sequences over their entire length. It is more useful for sequences for which similarity is expected over the whole length. Also, with global alignment, all the data is used and not just small bits as is the case with local alignment.

Local alignment searches for segments in the sequences that match well. It is more useful for scanning databases and when you don't know whether or not similarities are shared over their entire length.

In the examples above local alignment would probably be more useful when studying the atlantic and global alignment would be the best choice when considering the horses.

3.2 Comparing 2 DNA sequences

Continuing with our question we will start with an easy scenario. For this purpose we select a database consisting of only two sequences and start working with that. The sequences consist of a unique nucleotid code. They are not required to share the same length or have any other requirements. In order to say anything useful about how close these two sequences are related we will use sequence alignment which returns an alignment score. This score is an indication of how close the two sequences are related. For this a scoring matrix has to be created. This matrix tells us how close a nucleotid is to each types of nucleotid, assigning a score to each one.

	A	C	T	G
A	1	-1	-1	-1
C	-1	1	-1	-1
G	-1	-1	1	-1
T	-1	-1	-1	1

Since there is no clear answer to what these values should be, we will not assign preset values to the matrix. Instead, this also is open for the user to decide. The next thing to do is to include a gap function. This function imposes a penalty on the alignment score for each gap, and a higher penalty for longer gaps. There are two possibilities for the gap function in *PhyloGen*: Linear and logarithmic. We will explain how sequence alignment works using global alignment on an example. Note that local alignment works in a similar way, the only difference is the function a zero is added to the maximum function, therefore we will not discuss it further.

Take the sequences s:=AGCCT and t:=CGAT. We will use the scoring matrix as above and assume a linear gap penalty. First a matrix is constructed out of these sequences and the entries $\mathrm{d}(1,k)$ and $\mathrm{d}(k,1)$ are the penalties for k gaps in a row. $\mathrm{score}(s(1),t(2))$ is the value returned by the scoring matrix when comparing elements $s(1)$ and $t(2)$

		A	G	C	C	T
	0	−1	−2	−3	−4	−5
C	−1					
G	−2					
A	−3					
T	−4					

The matrix is now calculated using the following formula:

$$\begin{aligned}(i,j) \quad = \max\{ \quad & d(i-1,j-1) + \text{score}(s(j-1), t(i-1)), \\ & d(i-1,j) + \text{gapscore} \\ & d(i,j-1) + \text{gapscore}\}\end{aligned}$$

The local alignment equivalent of this formula is

$$\begin{aligned}(i,j) \quad = \max\{ \quad & 0, d(i-1,j-1) + \text{score}(s(j-1), t(i-1)), \\ & d(i-1,j) + \text{gapscore} \\ & d(i,j-1) + \text{gapscore}\}\end{aligned}$$

Since we are using a linear gap function for our example all gap scores are equal to -1.
This results in the following matrix:

		A	G	C	C	T
	0	−1	−2	−3	−4	−5
C	−1	−1	−2	−1	−2	−3
G	−2	−2	0	−1	−2	−3
A	−3	−1	−1	−1	−2	−3
T	−4	−2	−2	−1	−2	−1

The result in the lower right corner of the matrix is the distance or similarity score. This is the indication of how close the sequences are to each other and is thus the result we were looking for.
This is enough to start comparing all strings in the database with each other. However if we choose to use a different system, we will also need to find out what exactly the optimal alignment is. For this we need to create the traceback matrix. This matrix tells us the optimal alignment between the two sequences. It shows us the origin of every matrix element d(i,j) (Which of the scores in the above formula returned the actual maximum). In this example we use d, u and l for diagonal, up and left and follow the directions starting in the lower right corner of the matrix.

Using this idea for our example we get:

		A	G	C	C	T
	0	$-$	$-$	$-$	$-$	$-$
C	$-$	d	$-$	$-$	$-$	$-$
G	$-$	$-$	d	l	$-$	$-$
A	$-$	$-$	$-$	d	d,l	$-$
T	$-$	$-$	$-$	$-$	$-$	d

Following the traceback we finally find the possibilities for the optimal alignment:

$$\begin{array}{r} AGCCT \\ CG-AT \end{array} \quad \text{and} \quad \begin{array}{r} AGCCT \\ CGA-T \end{array}$$

Chapter 4

Methods of phylogenetic tree construction

A set of taxa can be represented in a graph in several ways. This is done by different methods. In this chapter, we briefly outline the different methods and explain how the method works we applied in our programme.

4.1 Terminology

Phylogenies are usually presented in the form of a tree. Such a tree is a graph which is built from nodes and branches. Terminal nodes represent different taxonomic units, such as species, genes, populations etc. They are referred as Operational Taxonomic Units (OTUs). Internal nodes denote ancestors or divergence points. Branches describe the descent and ancestral relations between the nodes. Each branch has a certain evolutionary rate associated to it, defined by some measure of distance between the OTUs.
Then there is the remark that in the following example, as in modern scientific literature, we look at mimimum distances. In our programme, however, we use maximum distances. This is nothing more then a change in notation, the algorithm stays the same.

4.2 Overview of methods

There are two approaches to phylogenetic reconstructions. One approach uses evolutionary distances and such a method is called distance method. The other approach uses character state data, in our case the nucleotid sequences, and such method is referred as character state method. Commonly used methods are classified into three groups: distance matrix methods, parsimony methods and likelihood methods.

In a *distance matrix method*, the evolutionary distances are computed for all pairs of taxa. The phylogenetic tree is constructed by using an algorithm based on the distance values. Our method is of this type.
The *maximum parsimony method* uses character state data with the global optimality criterion: the smaller the number of evolutionary changes required by a tree, the better the tree. This algorithm costs a lot of time, as the time increases exponentially with the number of species that are compared. Moreover, it works well only if the species do not deviate too much from each other.
The *maximum likelihood method* uses both nucleotid sequences and distances and chooses the tree with the highest maximum likelihood value as the best tree. In maximum parsimony method, character states are used, and the shortest pathway leading to these character states is chosen as the best tree. In maximum likelihood method, one searches for the maximum likelihood (ML) value for the character state configurations among the sequences under study for each possible tree and chooses the one with the largest ML value as the preferred tree. This method looks like Maximum Parsimony, but it is possible to take into account that for example an A easier transforms into a G instead of a C. Besides, this method is also time-consuming.

4.3 Unweighted Pair Group Method with Arithmatic mean

This is how the algorithm we use is called, short: UPGMA. It is a simple method to construct trees. It assumes the *molecular clock* hypothesis. This hypothesis holds that the rate of evolution is constant across phylogenetic lineages. As told, the method uses a distance matrix. In stead of exploring all possible bifurcating trees, it constructs a tree that will be relatively close to the tree with a minimal number of mutations (which can be obtained by Maximum Parsimony methods). We chose this method because this method is fast. Besides, we would like to work with distance matrices, for simplicity.
The methods uses the following algorithm:

- Find the two nodes with the smallest distance and connect them with an internal node. The distance from each of the old nodes is defined as the half of the total distance between the old nodes.

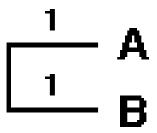
- Calculate the new distance matrix that includes the new node k, but excludes the two nodes i and j that it joins. In this new matrix rows and columns i and j are removed and a row and column k are inserted.

- If the distance matrix consists of a single element, stop. If the matrix doesn't consist of a single element start the algorithm again, viz. find the two nodes with ... etc..

We will explain the method in an example. Suppose we get the following distance matrix, of the species A to F:

	A	B	C	D	E	F
A	0	**2**	4	6	6	8
B	**2**	0	4	6	6	8
C	4	4	0	6	6	8
D	6	6	6	0	4	8
E	6	6	6	4	0	8
F	8	8	8	8	8	0

Now we cluster the pair of taxa with the smallest distance: A and B. The branching point is positioned at a distance of $2/2 = 1$. So we construct this subtree:



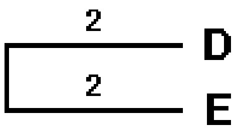
We now regard A,B as a new OTU and calculate the new distances:

$$\begin{aligned}
d_{\{AB\}C} &= \frac{d_{AC} + d_{BC}}{2} = 4 \\
d_{\{AB\}D} &= \frac{d_{AD} + d_{BD}}{2} = 6 \\
d_{\{AB\}E} &= \frac{d_{AE} + d_{BE}}{2} = 6 \\
d_{\{AB\}F} &= \frac{d_{AF} + d_{BF}}{2} = 8
\end{aligned}$$

So the distance matrix now becomes:

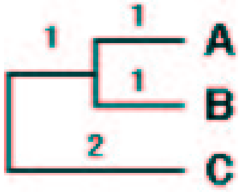
	A, B	C	D	E	F
A, B	0	4	6	6	8
C	4	0	6	6	8
D	6	6	0	**4**	8
E	6	6	**4**	0	8
F	8	8	8	8	0

Another subtree obtained:



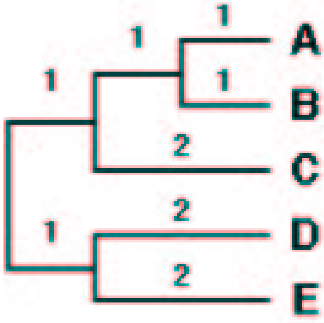
And again calculate the new matrix:

	A, B	C	D, E	F
A, B	0	**4**	6	8
C	**4**	0	6	8
D, E	6	6	0	8
F	8	8	8	0



Next step:

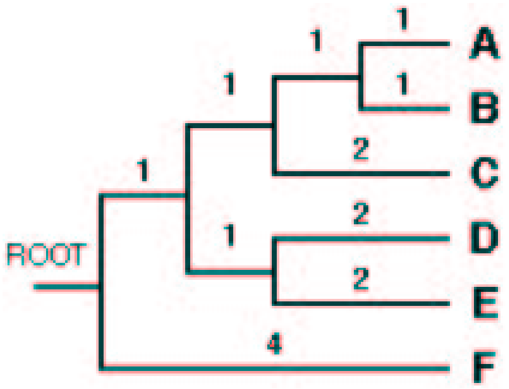
	A, B, C	D, E	F
A, B, C	0	**6**	8
D, E	**6**	0	8
F	8	8	0



And now the final step, clustering the last OTU F with the composite OTU.

	A, B, C, D, E	F
A, B, C	0	**8**
F	**8**	0

Although this method seems to lead to an unrooted tree, UPGMA assumes equal rates of mutation among all the branches. The root therefore, must have the same distance to all taxa. So the root of the tree is now positioned at distance $d_{\{ABCDE\}F} \ / \ 2 \ = \ 4$.
The final tree created by UPGMA is shown below:



4.4 Alternative algorithm

The previous way to draw a tree has some characteristics which you might not want to find in a program. It is very computationally expensive and it is a tree drawn backwards. In principle there is nothing wrong with this system, but if you, for instance, wanted to know if a whale is closer to a cow or a chicken, genetically speaking, this will not help you. Therefore we have added an experimental algorithm which works as follows. The first two sequences will become the first two branches of the tree like this:

```
   -----root-----
  |              |
 seq1           seq2
```

The root is not a sequence here but is more like a 'root of life'. Now a third sequence is compared to both sequence one and sequence two. Assume it looks the most like sequence two. Sequence one and two now form some sort of "class of lifeforms" represented by the mutual sequence (the traceback).

```
   -----root-----
  |              |
 seq1          -----
              |     |
             seq2  seq3
```

Now suppose a fourth sequence is to be added, we compare it to sequence one and the traceback of sequence two and three. If it looks most like sequence one, a subtree is created similar to the one above. If it resembles the traceback, then it is compared to sequences two and three and finally another subtree is made there:

```
  -----root-----
 |              |
seq1          -----
             |     |
           ----   seq3
          |    |
         seq2 seq4
```

This continues until all sequences are placed.
Obviously, considering the time we spent on this project, there are still quite a number of problems with this algorithm. For instance the order in which the sequences are used in the algorithm has a very big impact on what the tree looks like. Also, a traceback does not consist of merely one sequence and in the current layout a random traceback sequence is taken, but it would probably be more efficient to find a different way to do this.

Chapter 5

Test Results

5.1 Testing PhyloGen

Here is the output of the programme, there are three short sequences of only 8 symbols.
Sequence 0: Human CGAATACT
Sequence 1: Mouse GAATCGGA
Sequence 2: Rat GTGTCAGT

Global Alignment:

```
Compared seq 1 to 0: 4, corresponding fraction: 0.625
Found Alignment:
 (11) : -GAAT-C-GGA
 (11) : CGAATACT---
Compared seq 2 to 0: 6, corresponding fraction: 0.6875
Found Alignment:
 (10) : --AGCTGCAT
 (10) : CGAA-TAC-T
Compared seq 2 to 1: 5, corresponding fraction: 0.65625
Found Alignment:
 (10) : -AGCTG-CAT
 (10) : GAATCGG-A-
```

Local Alignment:

```
Compared seq 1 to 0: 5, corresponding fraction: 0.3125
Found Alignment:
 (6) : GAAT-C
 (6) : GAATAC
Compared seq 2 to 0: 8, corresponding fraction: 0.5
```

```
Found Alignment:
 (8) : AGCTGCAT
 (8) : AA-TAC-T
Compared seq 2 to 1: 6, corresponding fraction: 0.375
Found Alignment:
 (5) : AGCTG
 (5) : AATCG
```

You can, however, have sequences that are much bigger, here are three sequences of 100 symbols. You can make them much longer, but in this case that would be a waste of paper.
Sequence 0: Human
Sequence 1: Mouse
Sequence 2: Rat

Global Alignment:

```
Compared seq 1 to 0: 82, corresponding fraction: 0.705
Found Alignment:
 (125) : -GGTGT-CC--AGTACGT-GTGG---TAGCA-AATTT-A TTG-TGTAGT--GGA-
TTGCCAGG-T-ACC-GGT-AG--CCAGTGATGA--AAAG-GGACCCCCGA-CT-TT
 (125) : AGGTGTGCCCTG----GCAT-TT---ACG---GGACTGG---CATGGCTTGA-C-GGTGCA-TTTGGAC
TTGCTAGGCTTACCCG-TTAGGGCCAG-GG-AGTTAGAGTAG-CC---GAA-TACT
Compared seq 2 to 0: 94, corresponding fraction: 0.735
Found Alignment:
 (122) : GTGATCGATGACCCTGG--T--A-GGGATTGAC-TAGGACTC-CACA-CACCACTCT--CATCTACTA-
-CT-ACT-ACT-GA-CTC-GCG-A-CTCTACGGGTA-CTC-TCTCTAAC-CTC
 (122) : A-GGT-G-TG-CCCTGGCATTTACGGGACTGGCAT-GG-CTTG-ACGGTGC-ATT-TGG-ACTTGCTAG
GCTTACCCGTTAGGGC-CAG-GGAG-T-TA-GAGTAGC-CG-----AATACT-
Compared seq 2 to 1: 67, corresponding fraction: 0.6675
Found Alignment:
 (128) : G-TGATC-G----ATGAC-C----CTG---GTAG----GGATT-G---AC-TAG-G-ACT-CCACACAC
CACTCTCATCTACTACTACTA-CTGACTCGCGACTCTA-CGGGTACTCTCTCTAACCTC
 (128) : GGTG-TCCAGT-ACGTG-TGGTAGCAAATTTA TTGTGTAGTGGATTGCCA-G-GT
-AC-C--G---G-TA--GCCAG-TGA-T-G-AA----AA-GGG-ACCC-C-C-GACTTT
```

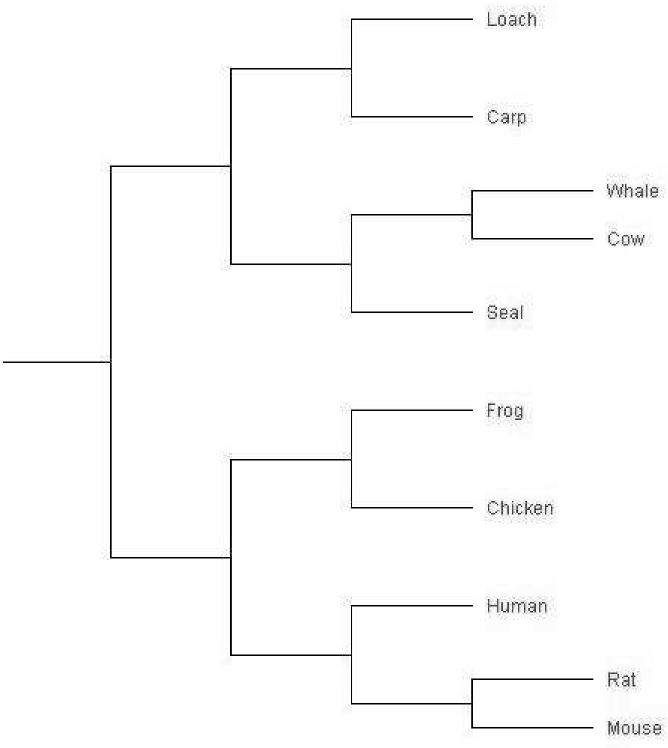
Local Alignment:

```
Compared seq 1 to 0: 83, corresponding fraction: 0.415
Found Alignment:
 (118) : GGTGT-CC--AGTACGT-GTGG---TAGCA-AATTT-A TTG-TGTAGT--GGA-T
TGCCAGG-T-ACC-GGT-AG--CCAGTG-A-T-GAAA-AG--GGAC-CC
 (118) : GGTGTGCCCTG----GCAT-TT---ACG---GGACTGG---CATGGCTTGA-C-GGTGCA-TTTGGACT
```

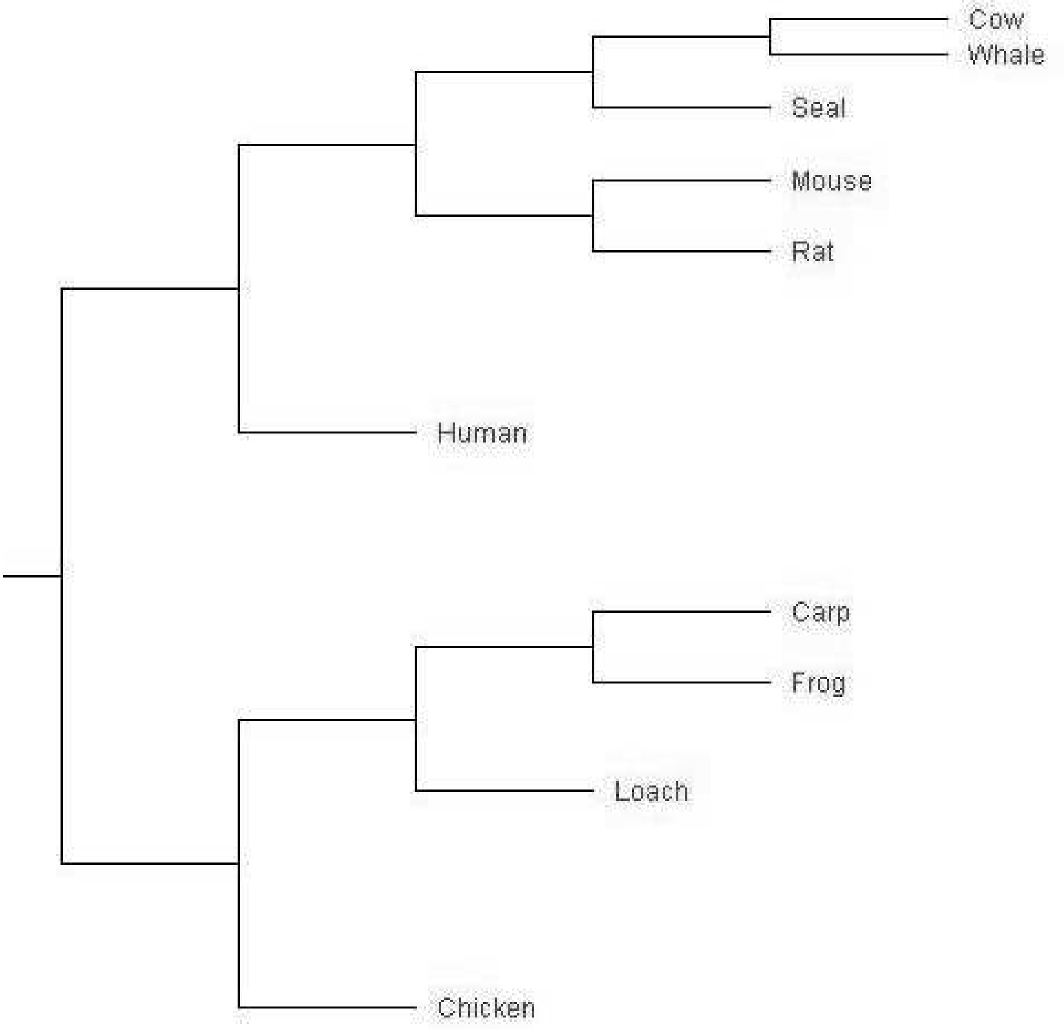
```
TGCTAGGCTTACCCG-TTAGGGCCAG-GGAGTTAGAGTAGCCGAATACT
Compared seq 2 to 0: 94, corresponding fraction: 0.47
Found Alignment:
 (114) : GATCGATGACCCTGG--T--A-GGGATTGAC-TAGGACTC-CACA-CACCACTCT--CATCTACTA--C
T-ACT-ACT-GA-CTC-GCG-A-CTCTACGGGTA-CTC---T-CT
 (114) : GGT-G-TG-CCCTGGCATTTACGGGACTGGCAT-GG-CTTG-ACGGTGC-ATT-TGG-ACTTGCTAGGC
TTACCCGTTAGGGC-CAG-GGAG-T-TA-GAGTAGC-CGAATACT
Compared seq 2 to 1: 68, corresponding fraction: 0.34
Found Alignment:
 (127) : GTGATC-G----ATGAC-C----CTG---GTAG----GGATT-G---AC-TAG-G-ACT-CCACACACC
ACTCTCATCTACTACTACTA-CTGACTCGCGACTCTA-CGGGTACTCTCTCTAACCTC
 (127) : GTG-TCCAGT-ACGTG-TGGTAGCAAATTTA TTGTGTAGTGGATTGCCA-G-GT-
AC-C--G---G-TA--GCCAG-TGA-T-G-AA----AA-GGG-ACCC-C-C-GACTTT
```

5.2 Trees

Our programme gives as final output a tree. Because we use two algorithms we have two trees, which are shown here: The tree according to the UPGMA algorithm:



The tree according to the alternative algorithm:



5.3 Running time analysis

In this section we will derive a theoretical answer on the question how much time it costs to get a phylogenetic tree with our program. I will give a global outline of the analysis; details are not taken into account. Computer dependent factors are not taken into account as well. That means that we will get an answer with use of the big-O notation. In fact, we count the number of primitive operations which are executed. Where a primitive operation is something like assigning a value to a variable, adding or multiplication of two numbers, indexing into an array, and so on.
The calculating time depends on two factors:

1. The number of inserted sequences (m).

2. The length of the sequences (n).

But what is the length of the sequences? They don't have to be all the same. Fortunately, this is not really important for us. We can choose n either as the length of the longest sequence or as the average of the length of the sequences. The choice of the length of the longest sequences gives somewhat pessimistic results, but it is really worst-case! However, our results do not depend on this choice. Our final result will be in terms of $\mathcal{O}(f(m,n))$, where f is a function of m and n.

5.3.1 UPGMA

First we consider the standard algorithm: UPGMA.
All sequences have to be aligned with all other ones. This means that the first sequence have to be aligned $m-1$ times, the second sequence still $m-2$ times, because he is already aligned with the first one, and so on. The last but one sequence has to be aligned with only one sequence. We get the sum:

$$\sum_{i=1}^{m-1} i = \frac{m(m-1)}{2} = \frac{m^2-m}{2}, \tag{5.1}$$

which is $\mathcal{O}(m^2)$.
Either global and local alignment is done by dynamic programming. This method fills a k by l matrix, with k and l the length of the sequences. This means that each alignment costs $\mathcal{O}(n^2)$ time. The time of calculating an entry of the matrix is not significant. Because all times that we align costs $\mathcal{O}(n^2)$, we get a total calculating time of $\mathcal{O}(m^2) \cdot \mathcal{O}(n^2) = \mathcal{O}(m^2n^2)$.
To make a tree with the UPGMA algorithm, each of the sequences has to be inserted in the tree. To know how this must be done, we use the UPGMA algorithm which updates the matrix everytime. This matrix becomes smaller and smaller. In really worst-case, the first time it costs m^2 of time, the second update costs $(m-1)^2$ and so on until the $(m-2)$th time, which costs $(m-(m-2))^2 = 4$. We get the sum:

$$\sum_{i=1}^{m-2} (m-i)^2, \tag{5.2}$$

which we can split in 3 terms:

$$\sum_{i=1}^{m-2} m^2 + \sum_{i=1}^{m-2} -2mi + \sum_{i=1}^{m-2} i^2. \tag{5.3}$$

Now we see that al this terms are $\mathcal{O}(m^3)$, therefore the calculating time of UPGMA is $\mathcal{O}(m^3)$.

Summarizing the above: aligning of all sequences costs $\mathcal{O}(m^2n^2)$ time, making the tree by UPGMA cost $\mathcal{O}(m^3)$ time. This results in a total running time of $\mathcal{O}(m^3+m^2n^2)$ with m the number of sequences and n the length of the sequences.

5.3.2 Alternative algorithm

Now we consider the other algorithm.
This algorithm does not align all sequences with all the other ones. The algorithm looks for the best location in the tree for the sequence. This is done by checking on an internal node, which child of that internal node is most similar to the current sequence. Then we do the same for that node, and so on, until we reach an external node. Checking which child is most similar to the sequence is done by local or global alignment. Consequently, we have to align sequences with the sequences on internal nodes, which are resulting optimal alignments found by the traceback method.
Align two sequences costs $\mathcal{O}(n^2)$, which we have already seen. We have to update the tree m times, but the first two are trivial, the only thing what have to do is align the sequences and execute the traceback method: $\mathcal{O}(n^2)$. The other $m-2$ times we have to update the tree. In really worst-case this result in $2(x-2)$ times aligning, where x is the number of sequences in the tree. We get the sum:

$$\sum_{x=2}^{m} 2(x-2), \tag{5.4}$$

which is $\mathcal{O}(m^2n^2)$.
We conclude with the remark that this algorithm is faster, but in practice you will only notice that when you want to align very many sequences. On the whole, we do not have the illusions that *PhyloGen* is as fast as would be possible. Optimizing the algorithms can result in a lower (worst-case) running time.

Chapter 6

Conclusions and future work

During our research in this project we found out that the subject of bioinformatics is vast. Therefore we think that our knowledge of the subject was too little and our time too scarce to discover the whole field.
That is also the main reason why we leave a lot of choices, concerning the programme, to the user. To make those choices beforehand would require a lot of research into very subtle differences, for which, as said, we had not the time or knowledge. This, however, does not mean that our programme is very superficial. On the contrary, our programme *PhyloGen* is a very useful tool for comparising nucleotid sequences.
When thinking of future work we walk on in the same direction. Our opinion is that there is a lot of research in sorting out the subtle differences mentioned above. In other words, make the chocies we did not make. If we take this a step further: to devise new algorithms and alignments that suit the users specific research. For example, by making use of parallel sequence alignments.
Another approach towards future work could be, what is being compared? We used the four characters used in nucleotid sequences. To compare sequences of amino acids, however, is a different and also computationally more tedious approach. We can go on in this fashion for some time.
Furthermore there is some work to do in the area of graphical interface and user-friendliness, i.e. a built-in help function. Some of the menus can use some further explanation. Moreover, the tree-output is centralized in such a way that when one half is occupied by only one specie and the other half with, say, twenty species, then the one species gets 50% of the screen and the other twenty 50% as well. The ideal situation would be that every species gets 4,7% i.e equally distributed along the screen. But these are just small modifications and are easier to accomplish than the research mentioned before.
On the whole we conclude that we have learned a lot about bioinformatics and phylogenetic trees in particular. It is an interesting cross discipline of biology, informatics and mathematics. Although none of us really had an interest in biology, we found out that this particular subject was worth our time and efforts

and is a useful subject for further study.
In the end, we are satisfied with the final programme and all the results surrounding it. We believe that we got the phylogenetic tree growing!

Chapter 7

References and authors

7.1 References

Books we used:
BLAST, An Essential Guide to the Basic Local Alignment Search Tool,
Ian Korf, Mark Yandell, Joseph Bedell. O'Reilly 2003.Sebastopol, CA, USA. .
The Phylogenetic Handbook, A Practical Approach to DNA and Protein Phylogeny,
Marco Salemi,Anne-Mieke Vandamme. Cambridge University Press 2003.

Web sites used for the history behind phylogenetic trees:

http://www.emc.maricopa.edu/faculty/farabee/BIOBK/BioBookEVOLI.html
http://www.talkorigins.org/faqs/faq-intro-to-biology.html
http://www.ncbi.nlm.nih.gov/About/primer/phylo.html
http://www.tolweb.org/tree/learn/concepts/classification.html
http://www.chelonia.org/Articles/genus_taxon.htm
http://www.mansfield.ohio-state.edu/~sabedon/biol3005.htm
http://www.ucmp.berkeley.edu/history/linnaeus.html

Web sites used for sequence alignment and algorithms:

http://www.cs.pdx.edu/~ps/CapStone03/SimilarityDiscussion.html
http://www.techfak.uni-bielefeld.de/bcd/Curric/PrwAli/prwali.html
http://www.inf.ethz.ch/~pvrohr/Courses/CompBio/2002/week6/DynProgTutorial.html

Web sites with various programmes more or less similar to ours:

http://iubio.bio.indiana.edu/treeapp/

Web sites used to find test data sets:

`http://evolution.genetics.washington.edu/book/datasets.html`
`http://workshop.molecularevolution.org/resources/fileformats/pir_dna.php`
`http://wavis.img.cas.cz/examples/dna.phy`
`ftp://ftp.ebi.ac.uk/pub/databases/embl/genomes/`
`http://images.tvnz.co.nz/news/graphics/monkey_dna.jpg`

7.2 Authors

The CS Project Group 2004-2005 are:

Robin Zeeman
Evert Hildebrand
Casper Zelissen
Tomas Waals
Jasper Valstar
Arjen Vermolen
Hanno Mulder
Albert-Jan Yzelman
Christian Poot

Coördinating teacher:

Rob Bisseling