Topic III: Parsimony Tree Search

Paul Lanctot
CSCI 5481
University of Minnesota

December 20, 2014

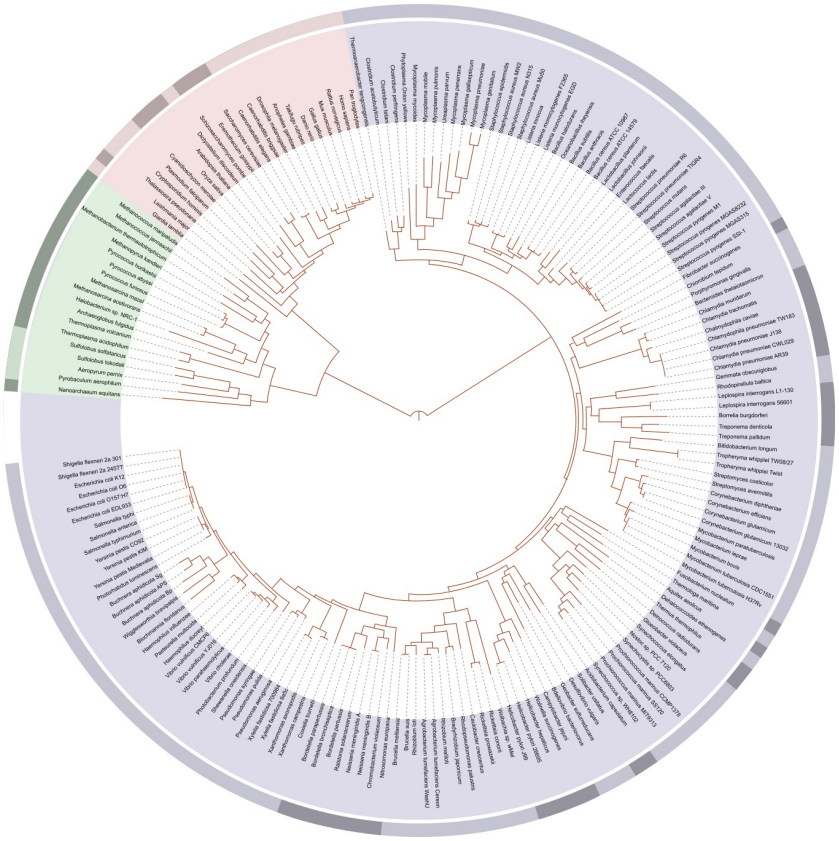


Figure 1: A phylogenetic tree of life, showing the relationship between species whose genomes had been sequenced as of 2006. The very center represents the last universal ancestor of all life on earth[1].

Contents

1 Topic Overview **3**
1.1 Tasks . . . 3
1.2 Tree Search . . . 3
1.3 Dataset . . . 3

2 Implementation **4**
2.1 Task One: Neighbor-Joining Algorithm . . . 4
2.2 Task Two: Sankoff's Algorithm . . . 4
2.3 Task Three: Tree Search Strategy . . . 5
2.3.1 Nearest-Neighbor Interchange . . . 6

3 Results **6**
3.1 Task One: Neighbor-Joining Algorithm . . . 6
3.2 Task Two: Sankoff's Algorithm . . . 7
3.3 Task Three: Tree Search Strategy . . . 8
3.3.1 Seed Family . . . 8
3.3.2 Full Family . . . 10

4 Analysis & Conclusion **11**
4.1 Nearest-Neighbor Interchange . . . 11
4.1.1 Benefits . . . 11
4.1.2 Drawbacks . . . 11

Click on any section in the Table to immediately go to that section.

1 Topic Overview

In this project we were asked to implement a tree search algorithm for parsimony analysis, which we would then use to analyze a multiple sequence alignment of a pfam protein family.

1.1 Tasks

1. Use the neighbor-joining algorithm to construct a phylogenetic tree of the sequences from the pfam multiple sequence alignment.

2. Apply your Sankoff's algorithm on the phylogenetic tree for parsimony analysis.

3. Implement some tree search strategies and perform the same parsimony analysis. Compare your results under different trees in the study.

1.2 Tree Search

There are several heuristic algorithms devoted to searching for an optimal tree structure in phylogenetics. The goal of these algorithms are to continuously rearrange a previously-generated phylogenetic tree in hopes of finding a tree with the lowest possible parsimony score (i.e. to find an optimal evolutionary tree for the input species). These algorithms can be applied to any set of data that can be arranged into a tree, but usually they are utilized in phylogenetics, notably in maximum parsimony and maximum likelihood searches of phylogenetic trees, where the ultimate goal is to identify the tree which best explains the evolutionary history of a set of species[3].

1.3 Dataset

A small test data of PIWI family was provided to test this algorithm, along with the original sequences of the species. The PIWI; originally P-element induced wimpy testis in Drosophila) class of genes was originally identified as encoding regulatory proteins responsible for maintaining incomplete differentiation in stem cells and maintaining the stability of cell division rates in germ line cells. PIWI proteins are highly conserved across evolutionary lineages and are present in both plants and animals. One of the major human homologues, whose upregulation is implicated in the formation of tumours such as seminomas, is called HIWI; other variants on the theme include the MIWI protein in mice[2].

PIWI family in pfam: `http://pfam.sanger.ac.uk/family/Piwi#tabview=tab0`

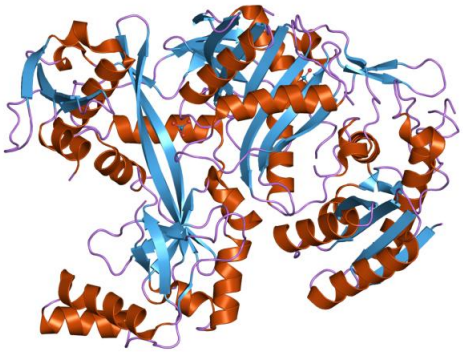


Figure 2: Structure of the Pyrococcus furiosus Argonaute protein (PIWI domain)[2].

2 Implementation

The following section briefly describes my implementation for each of the three tasks, along with the reasoning behind each method.

2.1 Task One: Neighbor-Joining Algorithm

For the first problem I implemented a method in Matlab that computes the pairwise distances of a multiple sequence alignment using the PAM250 matrix as the scoring matrix. After taking in a multiple sequence alignment, this program functions by ignoring the sites with gaps and then, according to the PAM250 matrix, computes scores for every possible pair of sequences in the multiple alignment. These scores correlate to a vector containing biological distances between each pair of sequences[4]. After the scores have been computed they are used to construct a phylogenetic tree for the multiple sequence alignment of the pfam family. This is done with the use of the neighbor-joining algorithm, which assumes equal variance and independence for the evolutionary distance estimates computed in the beginning.

2.2 Task Two: Sankoff's Algorithm

This task involved applying Sankoff's algorithm on the previously generated tree for parsimony analysis. In order to do accomplish this I converted my

Click on the citations throughout to be taken to the appropriate section in the bibliography.

C Cys	12																		
S Ser	0	2																	
T Thr	-2	1	3																
P Pro	-3	1	0	6															
A Ala	-2	1	1	1	2														
G Gly	-3	1	0	-1	1	5													
N Asn	-4	1	0	-1	0	0	2												
D Asp	-5	0	0	-1	0	1	2	4											
E Glu	-5	0	0	-1	0	0	1	3	4										
Q Gln	-5	-1	-1	0	0	-1	1	2	2	4									
H His	-3	-1	-1	0	-1	-2	2	1	1	3	6								
R Arg	-4	0	-1	0	-2	-3	0	-1	-1	1	2	6							
K Lys	-5	0	0	-1	-1	-2	1	0	0	1	0	3	5						
M Met	-5	-2	-1	-2	-1	-3	-2	-3	-2	-1	-2	0	0	6					
I Ile	-2	-1	0	-2	-1	-3	-2	-2	-2	-2	-2	-2	-2	2	5				
L Leu	-6	-3	-2	-3	-2	-4	-3	-4	-3	-2	-2	-3	-3	4	2	6			
V Val	-2	-1	0	-1	0	-1	-2	-2	-2	-2	-2	-2	-2	2	4	2	4		
F Phe	-4	-3	-3	-5	-5	-5	-4	-6	-5	-5	-2	-4	-5	0	1	2	-1	9	
Y Tyr	0	-3	-3	-5	-3	-5	-2	-4	-4	-4	0	-4	-4	-2	-1	-1	-2	7	10
W Trp	-8	-2	-5	-6	-6	-7	-4	-7	-7	-5	-3	2	-3	-4	-5	-2	-6	0	0	17
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W

Figure 3: The PAM250 matrix. The amino acid residues are indicated by their one-letter symbol[5].

Sankoff function from Homework 4 to be able process amino acids (previously it was designed only for nucleotides). The function performs Sankoff's algorithm on two phylogenetic nodes by looping through the 24 possible amino acids (includes the "B", "Z", "X", and gap symbols), while keeping track of the node's minimum parsimony at each position using the PAM250 scoring matrix and the node's current position. As this algorithm only functions for two input nodes, a separate method was created to operate on an entire tree, effectively pumping in the nodes of the entire tree into Sankoff's algorithm, computing where the lowest parsimony values lie at each node, and subsequently computing the parsimony score for the tree.

2.3 Task Three: Tree Search Strategy

The third and final task involved searching for the best possible evolutionary tree by finding the tree with the minimum parsimony value. The most basic of the tree search algorithms is the nearest-neighbor interchange (NNI) algorithm, which sets out to swap the branches of the four subtrees within the root tree. As there are three possible ways of doing this, each interchange creates two new trees (with the third tree being the orignal tree). NNI then searches the possible subsets recursively for each possible set of subtrees within the tree, making this a slow and rather inefficient strategy. Even

though I focused solely on the implementation and analysis of the NNI strategy, two other strategies are worth mentioning: subtree pruning and regrafting (SPR), and tree bisection and reconnection (TBR). SPR is a more wide-ranging search than NNI as it selects and removes a subtree from the main tree and reinserts it elsewhere on the root tree to create a new node. As for TBR, the method of rearranging focuses on detaching a subtree from the root tree at an interior node and then attempts all possible connections between branches of the two trees that were created[3].

2.3.1 Nearest-Neighbor Interchange

To start the search for a better tree, I implemented the nearest-neighbor interchange strategy. This functions by reordering the branches of the phylogenetic tree as mentioned previously, without dividing the clades or having crossing branches. Knowing that any internal branch of bifurcating tree has four neighboring nodes, each of which may represent an individual sequence or a subtree, the algorithm produces the other two topologies of the tree and examines them to see if they have more optimal values. After a new tree is generated, the current tree is checked to see if it has the minimum parsimony score, and kept track of if it does. This works down the branches until all N-3 internal branches are examined, thus generating 2(N-3) alternative trees. When the best of the new trees has been found it becomes the next starting tree. The search is continued until no further improvement in the tree has been obtained[6].

3 Results

The results discussed below are based on analysis from running the computations on a PIWI seeds family, containing 18 of the 2036 total sequences for the PIWI alignment. The entire alignment's evolutionary tree was produced as well, as shown in the "Full Family" section, but due to the large quantity of data it produced, proper analysis is more easily conducted using the smaller sample size of the seeds family.

3.1 Task One: Neighbor-Joining Algorithm

When computing the neighbor-joining algorithm on the PIWI seeds family, which contains 18 sequences, the resulting tree was produced:

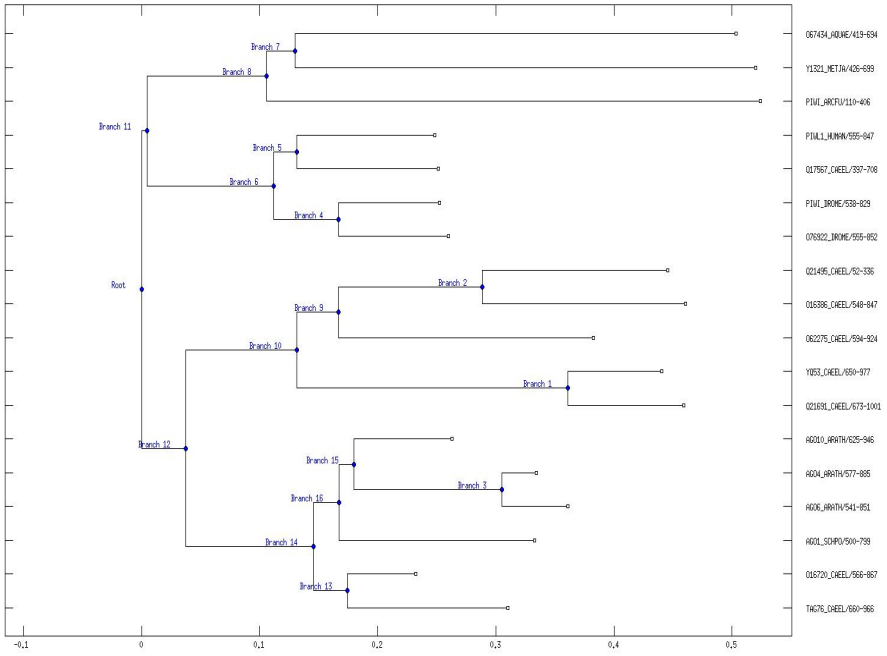


Figure 4: Initial Evolutionary Tree for PIWI seeds family (Phylogenetic_Trees(1)_seed.jpg).

As you can see, a proper bifurcating phylogenetic tree, with branch structure indicating evolutionary order, was produced during this step according to the evolutionary distances that were computed.

Next steps: computing parsimony scores, and searching for better trees.

3.2 Task Two: Sankoff's Algorithm

The parsimony value of this initial tree was scored in this step, according to the PAM250 matrix values, as,

$$\text{Tree 1 (Initial)} = -104667$$

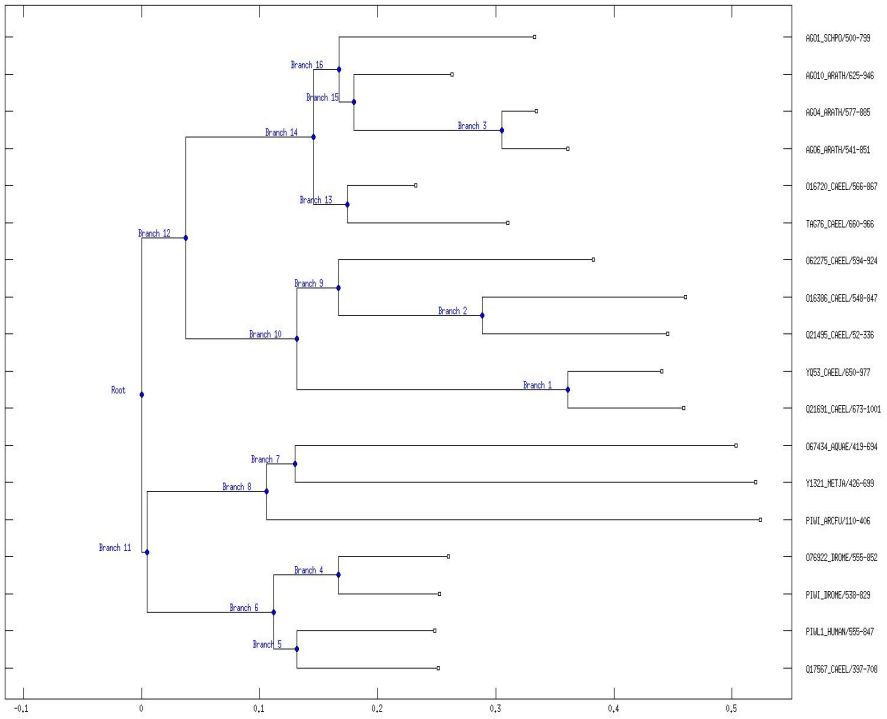
Alone, this value is rather arbitrary, however when computing parsimony scores for each newly generated tree, relatively kicks in, designating the lowest value as the most desirable, most accurate tree among the series of trees being processed as shown in the next step.

3.3 Task Three: Tree Search Strategy

The following two subsections show the results of computing the nearest-neighbor interchange strategy on trees created from two multiple sequence alignments: the PIWI seeds family (18 sequences), and the PIWI full family from NCBI (2036 sequences). More results, including Matlab trees, images, and parsimony scores are stored in the Extras directory within the rest of this submission.

3.3.1 Seed Family

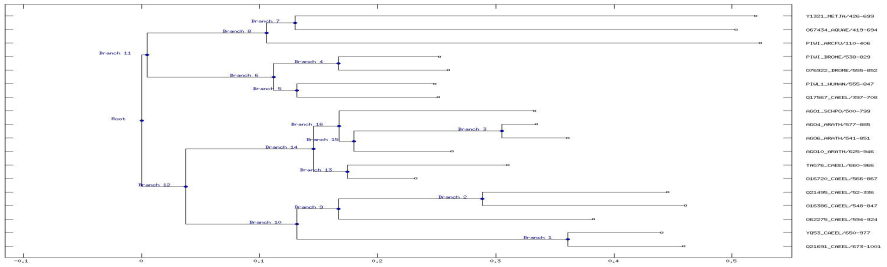
The tree below is the first resulting tree after swapping a branch on the initial tree displayed in section 3.1.



Tree 2 = −104672

Figure 5: Second Evolutionary Tree for PIWI seeds family (Phylogenetic_Trees(2)_seed.jpg).

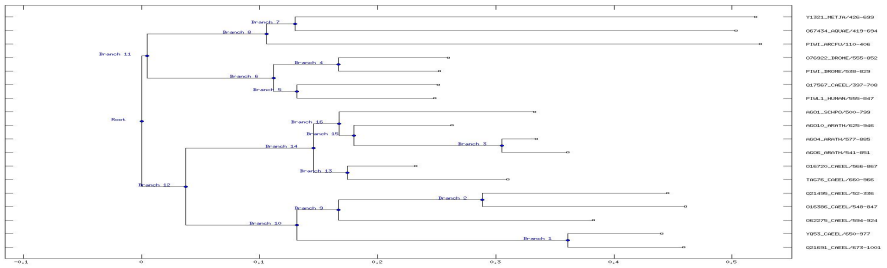
Here, the two major branches after the root node have been swapped from what they were in the initial tree (branches 11 & 12). Next, the third and final topology for this high-level branch was computed.



Tree 3 = −104677

Figure 6: Third Evolutionary Tree for PIWI seeds family (Phylogenetic_Trees(3)_seed.jpg).

In this one, branch 10 & branch 14 of the lower subroot were swapped. As can be seen from the parsimony values of the three trees, Tree 3 has the lowest-valued score, and therefore is the tree upon which the algorithm will choose to expand upon in the next iteration of tree generations. We are one step closer to the best evolutionary tree of the given species. Fast-forwarding a bit, we have come to the tree which computed the lowest score:



Tree 6 = −104718

Figure 7: Final Evolutionary Tree for PIWI seeds family (Phylogenetic_Trees(6)_seed.jpg).

Notice how this tree is not very much different from Tree 3, which we chose to expand upon in the second pass of the algorithm. Only one internal subtree has been swapped. This demonstrates a potential drawback of the nearest-neighbor interchange strategy where newly generated trees can get caught up in the local maxima of the initial tree from which it was iterating upon. While that is not necessarily the case here, and noting that we certainly do have a better tree than what was initially constructed, local maxima may be an issue to consider with this strategy.

3.3.2 Full Family

Not much analysis of the results will go into this section as there was a plethora of data to analyze, and the results of this strategy has more thoroughly been covered in the previous section with the seeds family, nevertheless, here are two trees of the full PIWI family that were constructed by NNI. The first is a radial view, showcasing the entire 2036 species and their evolutionary history (the actual tree is available in the extra files for further analysis). The second image demonstrates a zoomed in view of a subset of several leaf nodes within the tree.

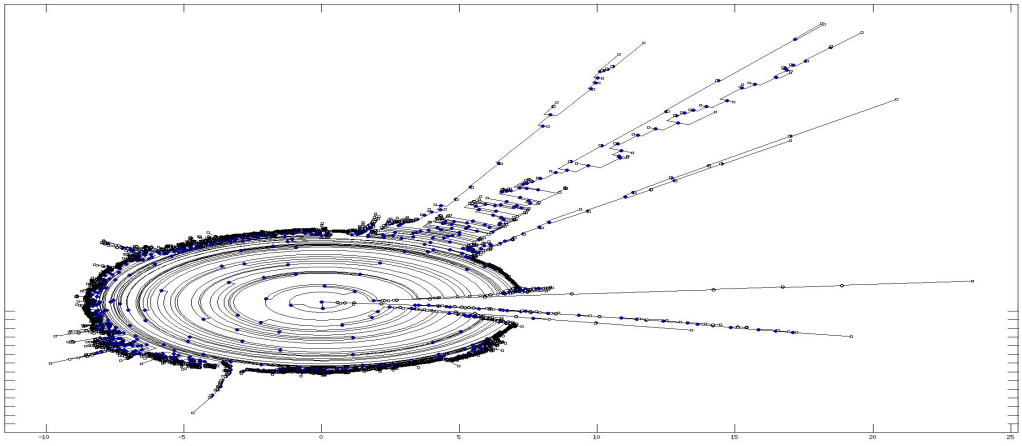


Figure 8: Radial Evolutionary Tree (Phylogenetic_Trees(1)_full_radial.jpg)

A more zoomed-in and square rendition of the final generated tree for the full PIWI family:

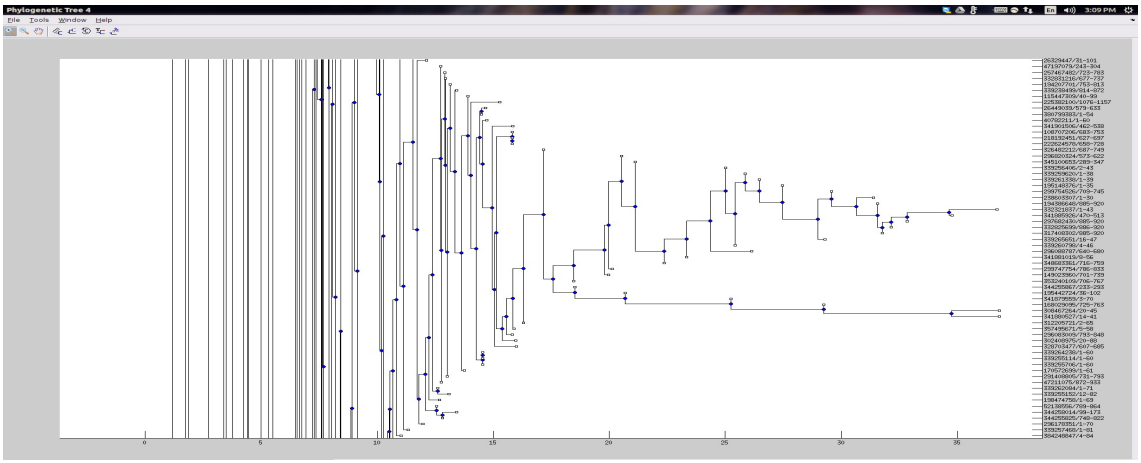


Figure 9: Evolutionary Tree (Phylogenetic Trees(4) full radial zoomed.jpg).

4 Analysis & Conclusion

Through the completion of the aforementioned tasks, several conclusions can be drawn in regards to the nearest-neighbor interchange tree search strategy. These conclusions involve several benefits with the use of this strategy as well the various drawbacks it entails.

4.1 Nearest-Neighbor Interchange

4.1.1 Benefits

Most notably, NNI is a simple and efficient strategy for progressively enhancing a phylogenetic tree. It takes an initial tree from any source, and improves upon it. This makes it a successful strategy in computing better evolutionary trees given various species, as it utilizes a basic branch swapping technique, examining all the potential possibilities for each branch, and chooses the best one. Then, iteratively, it examines all the potential possibilities of the chosen branch and effectively finds an even more accurate one, which it then again examines successively. Such an algorithm is bound to produce near-optimal trees, making it ultimately a useful tree search strategy.

4.1.2 Drawbacks

With that being said, the topic of local maxima is brought to mind after performing an NNI search. While the search promises, and does, find a better tree than what was initially given, it is not always an optimal tree.

The basis of choosing the best possible branch at each step is an effective one, but it lacks a random, scattered approach that is needed to find an optimal tree. Without this scattered approach, NNI quickly finds its best tree, without ever really expanding outside of the bounds of the initially constructed tree. Therefore, if the optimal tree lies outside these ranges, perhaps on an opposite field, of what was initially produced, it will never be found by NNI, and thus will need a different, more scattered search in order to be found.

References

[1] http://commons.wikimedia.org/wiki/File:Tree_of_life_SVG.svg

[2] http://pfam.xfam.org/family/Piwi#tabview=tab0

[3] http://en.wikipedia.org/wiki/Tree_rearrangement

[4] http://www.mathworks.com/help/bioinfo/ref/seqneighjoin.html

[5] http://what-when-how.com/wp-content/uploads/2011/05/tmp10945_thumb1.jpg

[6] Zvelebil, Marketa J., and Jeremy O. Baum. *"8."* *Understanding Bioinformatics*. New York: Garland Science, 2008. N. pag. Print.