How well does HPA perform against existing phylogenetic estimation methods?

(4,270 Words)

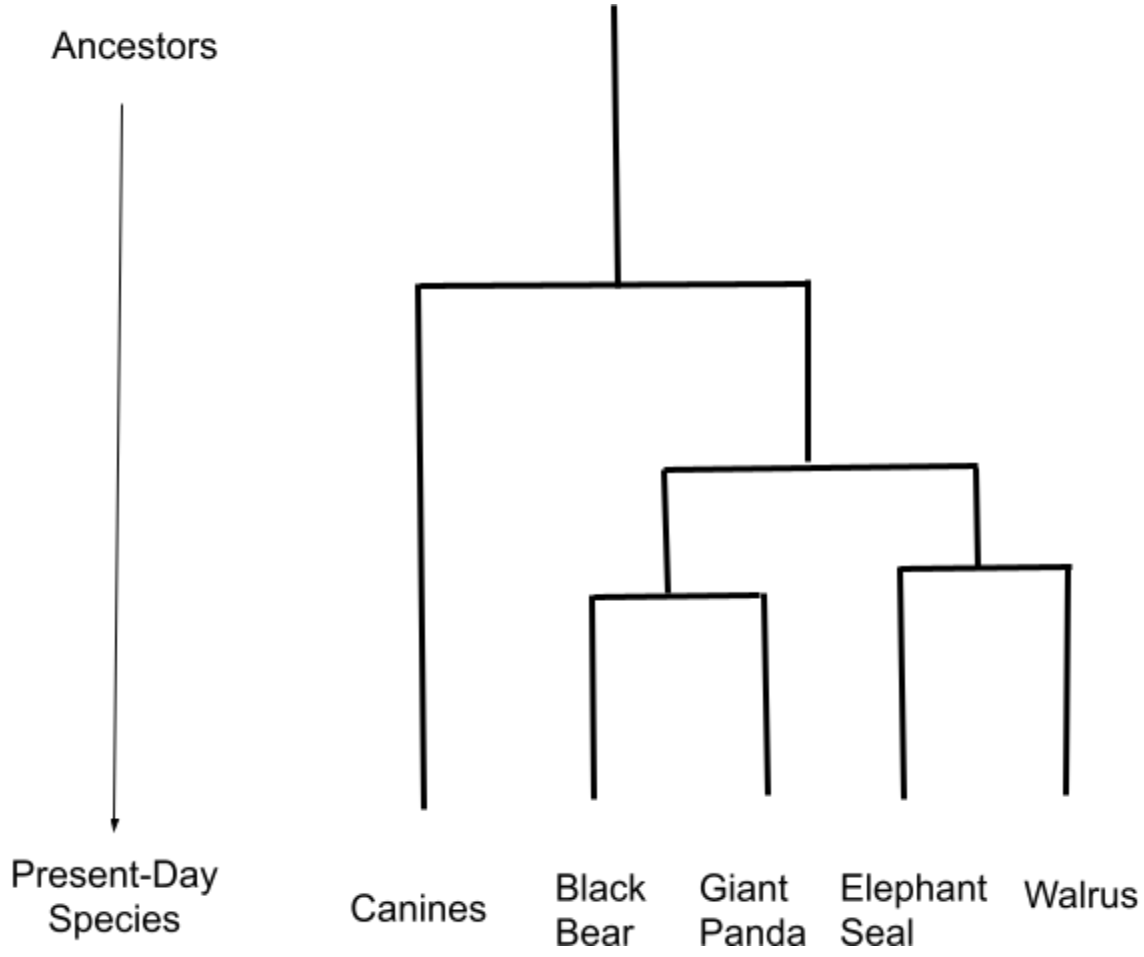
Introduction

Behind all life on Earth is a history of billions of years of adaptation and survival in a messy branching (and sometimes reconnecting) bush that eventually led to what lives today. While we can be quite sure that at some point in our evolutionary past humans looked like fish, we have to use indirect techniques to try to reconstruct, or estimate, what the tree looked like that led to us. Before the advent of modern genetic techniques, morphology was the dominant technique to try to estimate our evolutionary past. For example, we share a more recent common ancestor with the chimpanzee than we do with birds, because we both have hands, big brains, do not have wings or feathers, and so on. This can be complicated because some structures evolve repeatedly; bats have wings, but are more closely related to us than birds. For older creatures, morphology must be discovered through the fossil record. With the advent of DNA sequencing techniques and computers, algorithmic analysis of genetic information has become a very powerful way to estimate evolutionary history. This paper concerns one such new computer algorithm, called HPA (Hierarchical Partition Analysis), created at the University of Nevada, Reno by professor Guy Hoelzer and his group. We will be evaluating the HPA algorithm against other existing algorithms, in simulated evolution problems, to see how HPA compares and how it might be improved.

The basics of these computer techniques is not difficult to understand. The gradual change of organisms over time is often due to reproduction with mutation in the face of varying environmental fitness. The offspring of two organisms is never an exact replica of either one, nor even a perfect mixture of both. Traits emerge in complex ways from genes, the DNA which is the blueprint of all organisms. Offspring have characteristics that vary from both parents often

due to a process called mutation, which is an error in DNA replication. This error is one method that creates new characteristics, not observed in either parent (Herron et.al., 2015). While fundamentally it is a mistake of the reproductive system, it is one way species adapt.

Each successive generation is slightly different from the parents. The generations that have desirable traits will survive, those with disadvantageous traits will die. This is the process of natural selection. After thousands of successive generations, the final result will be very different from the original species. This study of evolutionary history is phylogeny (Berkeley, n.d.). Understanding this process of natural selection reflects the origin of species. There are many methods to track this change over time. For the majority of species, fossil records are used to trace the process. A tree structure is created to represent the consecutive generations, their interbreeding, and the eventual result. Below is an example of such a tree.



While fossil records represent the majority of our understanding of phylogenetic history, there are certain cases where more complex methods are necessary. With the advance of techniques that let humans read the makeup of DNA, new methods have developed to trace how organisms differ over time from one another. Modern computational tools can be used to estimate a hierarchical structural relationship between organisms using the information that DNA provides. Given a set of genetic information from a series of species, it is possible to reconstruct the evolutionary tree of the species. This is an important tool because it provides inference into how species have developed today. Computer approaches exist that can estimate the evolutionary relationship between a set of different organisms (Herron et.al., 2015). For example, given DNA

from a black bear, giant panda, elephant seal, and a walrus, it can say "the Black Bear and the Giant Panda share a most recent common ancestor (MRCA) more recently than an Elephant Seal and a Walrus, Elephant Seal and Walrus are more closely related (closer MRCA) than a Elephant Seal and Black Bear." These algorithms use different approaches to solve the same problem. I will be performing a comparison of these algorithms to determine the most effective method of synthesizing genetic information. Also, I will examine whether a newly created method, HPA, is comparable to the others. The algorithms used in this study will be RAxML, FastTree and HPA. These programs are in current use by researchers all over the world.

RAxML "is a popular program for phylogenetic analyses of large datasets under maximum likelihood" (Stamatakis, 2014). The algorithm has been maintained and updated to deal with growing computational power and larger datasets. Maximum likelihood refers to the procedural method with which this algorithm functions. Stamatakis' article in the leading molecular biology journal *Bioinformatics*, identifies the growing importance of phylogenetic reconstruction in all areas of medical biology. The latest article updating this algorithm was in 2014 making it current and applicable for my study. RAxML is licensed under the GNU General Public License which means it is free for public use. I can modify, use, or even publish this algorithm, which is important for my study.

FastTree is another tool for construction of phylogenetic trees. It is intended to scale to larger datasets which will be tested in my study. The most recent article with updates on the capabilities of this software is in 2009 (Price, 2009). It identifies the purpose of the algorithm as understanding "taxonomy and for predicting structure and biological function." This algorithm is a widely used tool in building phylogenetic trees which makes it important to consider in my

study. Contrary to RAxML, FastTree uses a Neighbor-Joining approach before using maximum likelihood. It is only important for the sake of this study to recognize that this is a seperate algorithmic method. It is not needed to understand the in-depth mechanics of these approaches. This program is licensed under the GNU General Public License version 2 which also guarantees the freedom to share and change free software. Like the previous algorithm this means there are no restrictions in publication.

The final algorithm in my study is the Hierarchical Partition Analysis (HPA). This is published by Dr. Guy Hoelzer at the University of Nevada Reno, who is my mentor for this study. HPA is intended to handle large datasets quickly and accurately with no short cuts. It uses pattern recognition to identify hierarchies within the dataset. This is separate from the maximum likelihood and neighbor-joining approaches used by RAxML and FastTree. While the official version of the program has not been released, I was able to test prototype algorithm for the sake of my comparison. The paper and algorithm have not yet been released, which makes it a contemporary approach to constructing these trees. Because it is not released yet, it has not yet been compared to any other algorithms, at least any that have been largely publicized.

Literature Review

I have chosen three algorithms for my study: RAxML, FastTree, and HPA. I have identified these for my study because a comparison of these three has not been previously performed and these methods are both popular and widely respected tools currently used by researchers. Other research has performed comparisons between different phylogenetic estimation algorithms, but none has included HPA in the comparison since it is so new. According to *An Investigation of Phylogenetic Likelihood Methods*: “We analyze the

performance of likelihood-based approaches used to reconstruct phylogenetic trees" (Williams et al., 2003). This study is an examination of just maximum likelihood methods. This is what distinguishes my study because I examine a maximum likelihood, neighbor-joining, and the new HPA model. The algorithms compared in this paper are "fastDNAml, MrBayes, PAUP-ML, and TREEPUZZLE." This does not examine any of the programs I am addressing in my study, likely because it was published in 2003. This further distinguishes my investigation from this paper. However, Williams' paper is still valuable because it does provide a method of comparing these programs that I will apply to my study. They are measuring accuracy and speed of the four algorithms they have identified which are the two variables I am also measuring. In the study they also vary the amount of organisms, or taxa, which is something that I will also be varying in my study to determine which algorithms perform better on higher order operations. This paper is an important resource for methodology but is fundamentally very different from my study. The paper concluded that their "results clearly demonstrate that MrBayes is the best algorithm of the methods we studied."

In another paper titled *A Simple, Fast, and Accurate Algorithm to Estimate Large Phylogenies by Maximum Likelihood,* the authors do a comparison of "PAUP, fastDNAml, MrBayes and PHYML" (Guindon et al., 2003). While this does perform a comparison of different algorithms and methods, it does not use any of the contemporary programs that I am examining in my study. This comparison is fairly similar to the previous because it was also published in 2003 and was working with the best available programs of that time. This research found that PHYML "is not only much faster than the standard approach but also slightly better in terms of topological accuracy and likelihood maximization."

More recent comparisons have been performed in a *Comparison of the Accuracies of Several Phylogenetic Methods Using Protein and DNA Sequences*, published in the journal *Molecular Biology and Evolution* (Hall, 2005). This is different from the Williams and Moret study because it is examining "Neighbor Joining, Parsimony, and Maximum Likelihood" approaches. This is more similar to my experiment in that it examines multiple different algorithmic approaches for phylogenetic tree reconstruction. However, the methods for estimation in this paper were done with PAUP and MrBayes. These are another two algorithms that I will not be using in my study because this paper was published in 2005, and this was all that was available at that time. This is more similar to my study because it addresses different methods, but does not examine any of the modern programs that I will use in my study.

The most similar study that I have identified is published in *FastTree: Computing Large Minimum Evolution Trees with Profiles instead of a Distance Matrix* (Price et al., 2009). This paper claims to do a comparison of virtually all existing contemporary methods as of 2009. Many other methods have been developed since, making this paper less applicable today. What differentiates my study is that I am examining the new method HPA. Furthermore this paper does address the computational speed of the various algorithms, which is an important consideration for larger order operations. It also performs a variety of mathematical manipulations in an attempt to standardize and improve the performance of the results of the algorithm that they were publishing. They found that FastTree performed favorably against other existing methods of that time. A similar analysis is done in *RAxML-III: a fast program for maximum likelihood-based inference of large phylogenetic trees* (Stamatakis et al., 2005). This paper uses RAxML, which is one of the methods I am using in my study, but it only compares it to PHYML and MrBayes.

Both of these papers do a comparison of different algorithms, some of which I am using in my study. It concluded that RAxML was superior to other existing methods of the time. While, neither one of these test HPA, they do provide an insight on speed comparison of other algorithms.

It is clear that determining the most effective method for phylogenetic tree reconstruction is important, which is what I am basing my investigation from. However, there is no contemporary analysis of algorithms, especially HPA. Many but not all of these studies took speed into consideration, which is very important given the large amount of times large computations can take. This is what makes my study unique and a contribution to the field.

Methodology

My experiment will mirror the comparison of algorithms in *Comparison of the Accuracies of Several Phylogenetic Methods Using Protein and DNA Sequences* (Hall, 2004). What differentiates my method is the algorithms that I am comparing and where my data come from. I will however still be determining the accuracy of the methods as well as the speed which is an important consideration. My methodology is divided into five sections: the algorithms, the data, the hardware, the computation, and the comparison.

HPA, FastTree, and RAxML all use different algorithmic approaches. HPA uses a novel approach to phylogenetic estimation that is not like the other two methods examined in this experiment. This makes it unique in this study because it has not previously been compared against these existing methods. The purpose of my study is to determine whether this will compare favorably to the existing methods that have been in use for a longer period of time.

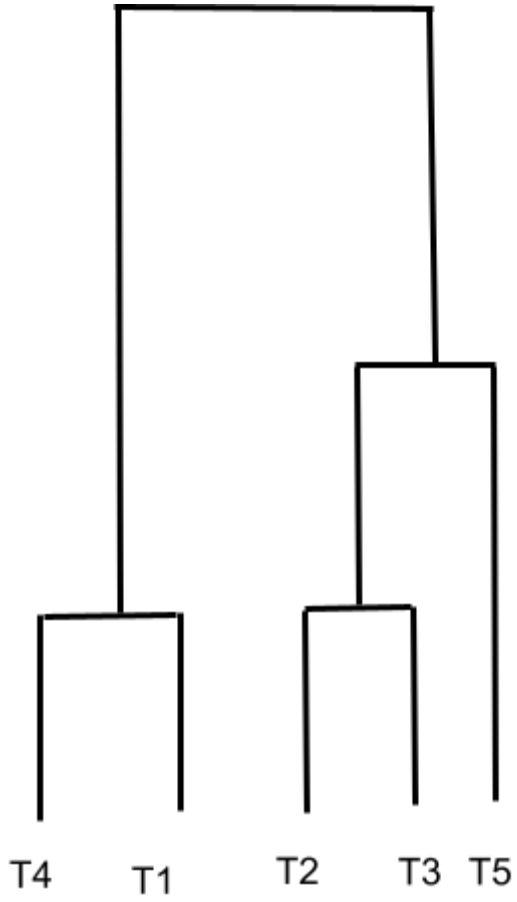
FastTree creates neighbor joining to construct phylogenetic trees from alignments of nucleotide or protein sequences. “Neighbor joining is the most popular method for constructing large phylogenies. Neighbor joining is also often used to generate an initial tree before searching for the maximum likelihood tree” (Price et.al., 2009). This is a fairly common method of estimation that is used by several other contemporary methods. FastTree is a well established program and thus is a good representative of neighbor joining methods.

RAxML uses a maximum likelihood approach for the construction phylogenetic trees. Maximum likelihood (ML) “is a popular method for inferring a phylogenetic tree of the evolutionary relationship of a set of taxa, from observed homologous aligned genetic sequences of the taxa. Generally, the computation of the ML tree is based on numerical methods” (Chor, 2006). There are also many other ML methods that I chose to omit because RAxML is a long established and representative method of this type.

These algorithms accept data in the form of a DNA sequence. This is a set of characters that is part of the genome of an individual that is taken as representative of the species. For this study, the amount of bases will be set to 10000 and not varied. In order to create variation in my study and test the performance of the three algorithms in comparison to each other, I am varying the amount of taxa. My experiment will be run with groups of 8, 16, 32, and 64 taxa repeated 20 times to reduce the effects of random variation. This will not only yield information on how the speed of these methods compare on higher order computations but also if the accuracy decreases when the number of taxa increases.

Genetic information will be created using Dendropy and Pyvolve. “DendroPy is a Python library for phylogenetic computing. It provides classes and functions for the simulation,

processing, and manipulation of phylogenetic trees and character matrices, and supports the reading and writing of phylogenetic data in a range of formats" (Holder, 2016). This is an example of the generated tree.



Pyvolve is a "module for simulating genetic data along a phylogeny using continuous-time Markov models of sequence evolution" (Spielman et.al., 2015). This will simulate evolution on this tree and yield a data set of DNA characters. This is an example of what the information will look like for one specific taxon.

```
>T1
AACATCTTCATTACCATGGCTCCTAGTGCGCCATGGGACAGACGACGTAAAGCTTCGGATGCGTCAATAC
TTTGTCGCTCGCCGTATTCAGACGGCGTAACCGTCTGTCGATAACTAGTAAGTGGCACCGAGAATGGGGA
TAATAACGGAGAACCTTGAGCGGAATATGGACACCTGTTCTTAATGACGGCCGTGATGACTCCTTAGAAC...
```

Dendropy will create the phylogenetic tree structure and Pyvolve will simulate evolution along that tree. This produces simulated genetic information, like what could be sampled from real organisms in a natural setting. Pyvolve provides this raw genetic information for every taxon in each experiment. This genetic information will be supplied to HPA, RAxML, and FastTree and the output will be recorded as well as the execution time. The true tree that was created with Dendropy and Pyvolve will then be compared against the results of the various algorithms to determine the accuracy of methods.

In order to test and gather the data in rapid succession while varying the number of taxa, I created a Python program that automates this. This will allow me to perform the experiment in the exact same conditions with the exact same data sets which is crucial to give a fair chance to each method. The complete program is available on https://github.com/C0L/PhyloTester.git. This program can be used to reproduce the results of this research or test other algorithms using this same framework. Various sections will be displayed within this paper to highlight some concepts that are important for this study. The following section was used to create the phylogenetic trees and simulate evolution and create the artificial tree using the prior mentioned Dendropy and Pyvolve.

```
def generateTree(tns, ntaxa, seqlen):
    #Construct the tree and save as newick file
    t = dendropy.simulate.treesim.birth_death_tree(birth_rate=1.0, death_rate=0,
taxon_namespace=tns, num_extant_tips=ntaxa)
    t.write(path='/tmp/pyvt', schema='newick', suppress_rooting=True,
suppress_internal_node_labels=True)

    #Set pyvolve data type
    m1 = pyvolve.Model("nucleotide")
    p1 = pyvolve.Partition(models=m1, size=seqlen)

    #Read tree from dendropy
    pot = pyvolve.read_tree(file='/tmp/pyvt')
```

```
#Simulate evolution with no save file
e1 = pyvolve.Evolver(tree=pot, partitions=p1)
e1(seqfile=None)

seqs=e1.get_sequences()

ds=dendropy.DnaCharacterMatrix.from_dict(seqs, taxon_namespace=tns)
ds.write(path="evolvedsequences.fasta", schema="fasta")
return t
```

The ability of these algorithms to analyze data depends significantly on the computational specifications of the machine that it is being run on. Universities have access to large scale computer networks capable of performing far more and faster computations which is what these algorithms are traditionally run on. I, however, will be running my experiment with the following computer specifications. The computer runs an Intel i7 2.8Ghz processor, in a Unix based operating system. It is important that all of the experiments will be run under these circumstances so it will be a comparison of algorithmic efficiency rather than computational power. It is not important to understand what these specifications mean, only that they were consistent throughout the trials, and they are important for reproducibility.

The computer language that each one of these algorithms is programmed in is an important consideration in execution. HPA is written in the computer language Python. Python is "an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development" (Python, n.d.). In many cases Python is not as efficient as other languages because it is a high-level, interpreted language designed for expressiveness not speed of execution. This is only important to mention because it may skew

the results to identify which programming language the algorithms are written in rather than what method is faster. HPA was executed from within my program with the following line.

```
call(["python", "hpa.py", "-f", "evolvedsequences.fasta", "-t", "HPAResults.newick"])
```

In contrast, RAxML is a compiled C program. This is usually faster than Python and comes with some certain efficiency bonuses. RAxML was compiled for my computer using GNU Compiler Collection (GCC) (Price, n.d). Compiling is the process of converting a program from the human readable source code to machine code that a computer can understand. It was compiled to run on a multicore system which means it will be faster and give it every opportunity to be as fast as possible. The command to call RAxML was executed with the following line.

```
call(["./RAxmlHPC-PTHREADS", "-m", "GTRCATX", "-V", "-s", "evolvedsequences.fasta",
"-p", "12345", "-n", "T1"])
```

FastTree is also a compiled C program. This was also compiled using GCC for my computer (Exelixis, n.d.). It also has multicore capability which should allow the algorithm to run faster in its trials. FastTree was executed with the following line.

```
call(["./FastTreeMP"], stdin=file("evolvedsequences.fasta"),
stdout=file("FastTreeResult.newick", "w"))
```

Each method will return its completed phylogenetic tree and a time of execution. Because genetic information was created with Dendropy and Pyvolve, the true tree is known. Calculating the Weighted Robinson Foulds distance will compare the simulated tree and the created trees from the algorithms. The Weighted Robinson Foulds distance takes into account how close each algorithm predicted the relationship between taxa was to the true value, as well as how long the evolutionary branches are in the evolutionary tree in comparison to the true values. It will return a similarity value, with lower numbers meaning the trees are more similar. The execution of this

comparison is outlined below, comparing the true tree (TRUETREE) to the algorithm's predicted tree (NEWTREE).

```
dendropy.calculate.treecompare.weighted_robinson_foulds_distance(TRUETREE, NEWTREE)
```

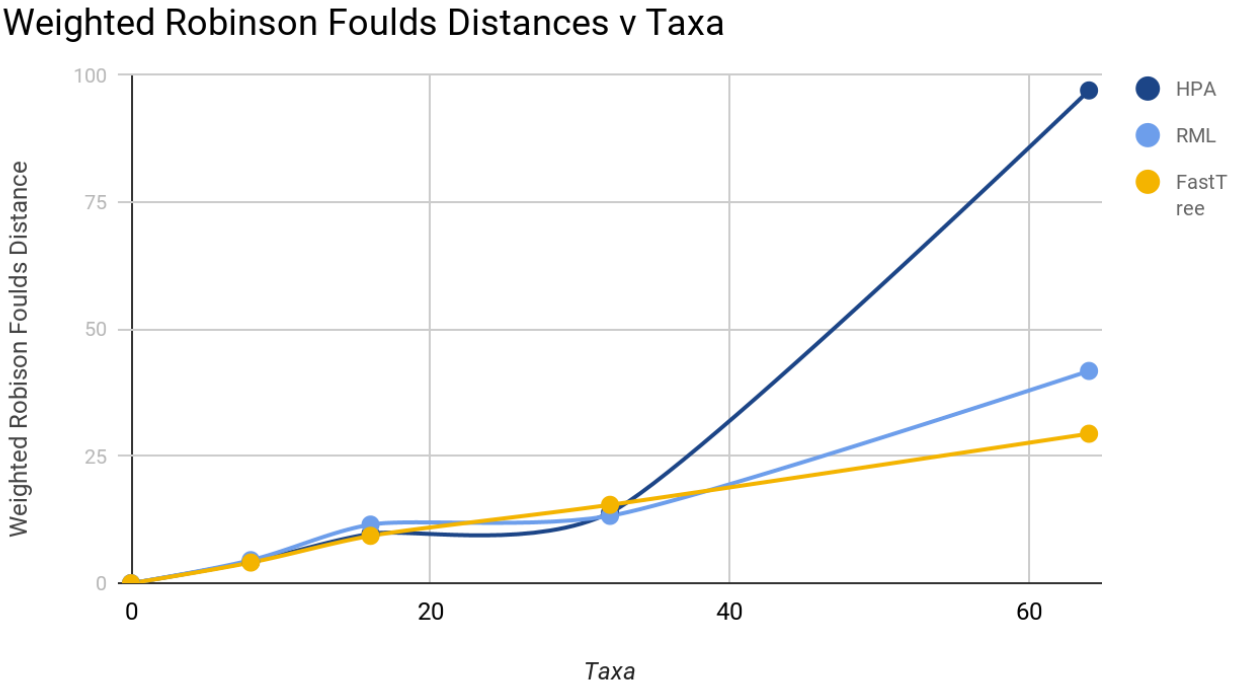
This will be performed for all different taxa amounts, and the time of execution will be recorded at all these values as well. This allows for a comparison between the accuracy of all methods as well as the speed of all methods. The results from this will be gathered and compared for each algorithm. The data was grouped based on algorithm and will be displayed graphically and in a table.

Results

The results of the weighted Robinson Foulds Distance are recorded below. Lower results mean the algorithm is more accurate.

	RAxML Accuracy	FastTree Accuracy	HPA Accuracy
8 taxa	4.55*	4.08	4.24
16 taxa	11.55	9.34	9.74
32 taxa	13.9	13.27	15.44
64 taxa	41.8	29.44	97.03
Average	17.95	14.03	31.61

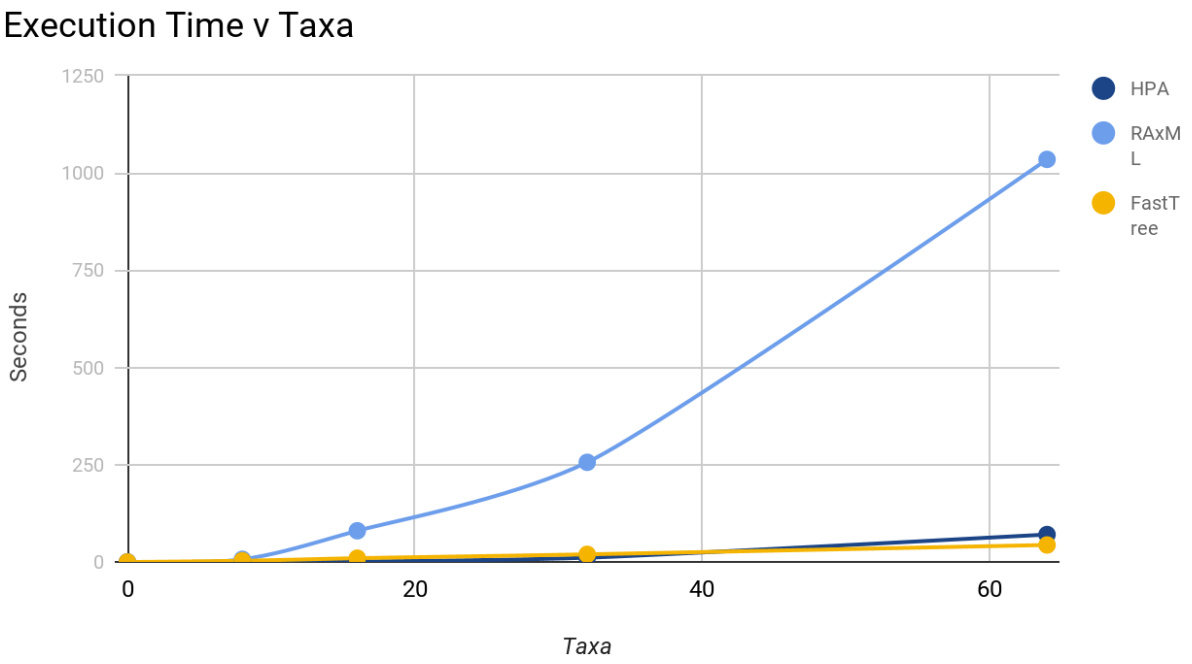
*All results are an average of 20 Weighted Robinson Foulds Distances rounded to nearest hundredths place



The results of the execution time are recorded below.

	RAxML Time (s)	FastTree Time (s)	HPA Time (s)
8 taxa	6.93*	3.23	.54
16 taxa	80.24	9.8	1.34
32 taxa	256.25	19.7	11.13
64 taxa	1034.89	43.79	70.60
Average	344.58	19.13	20.90

*All results are an average of 20 execution times in seconds rounded to nearest hundredths place



Conclusion

Beginning from 8 taxa, all three algorithms performed relatively similarly with roughly a Robinson Foulds Distance of 4. FastTree was the most effective in comparison but this was by a small margin it is not significantly different. HPA and RAxML were both roughly within .5 of FastTree which puts them only slightly behind on their accuracy. However, HPA performed far above both FastTree and RAxML in the speed of execution. RAxML followed behind FastTree by a significant margin being almost twice as slow.

At 16 taxa similar trends were observable. FastTree was still outputting more accurate results then HPA and RAxML. HPA was not far behind and remains reasonably close in this metric. However, RAxML began to fall behind, and became significantly separated from the other two methods. The execution times retained the same trend as 8 taxa, where HPA was significantly faster than the other two methods. However, RAxML's runtime exponentially

increased from the prior 8 taxa. FastTree still remained somewhat close to HPA but was several seconds slower.

At 32 taxa RAxML performed more similar to FastTree than in prior trials. HPA performed only slightly worse than the other two methods. HPA was less able to handle larger amounts of taxa while RAxML and FastTree still performed well. HPA still however had the best execution time and continued to widen the gap between it and FastTree. RAxML continued to be exponentially less quick in execution reaching times that would make it very difficult to run large scale operations with this algorithm. This is the last value in which all three methods performed similarly in accuracy.

At 64 taxa HPA dropped accuracy in comparison to the other methods. Its Weighted Robinson Foulds Distance became more then twice that of the other methods. This supports the hypothesis that HPA performs poorly on data with a large number of taxa. FastTree performed almost twice as well as RAxML in accuracy. HPA's execution time fell below FastTree's. RAxML, while only performing worse then FastTree in accuracy, had disproportionately poor execution time.

All these methods were run under the default settings. In order to take human analysis out of the equation no parameters were passed to the algorithms that might give them an advantage over their counterparts. Each algorithm has the ability to be largely customized to fit data but it would be unfair to add the prior human analysis needed to make this decision. This is, however, not how these methods are intended to be used and this may have had a larger effect on methods like RAxML. While this was the only fair way to compare these algorithms, it limits the results by the fact that each method was not specifically calibrated.

On average FastTree had both the more accurate results and faster runtimes. This appears to be the most effective method for phylogenetic reconstruction from my experiment. HPA is the second best, largely because of the enormous runtime that RAxML exhibited. It would be very difficult to use RAxML to work on any large data sets because its runtime is unreasonable to let complete in short time periods. However, it still did behave favorably over HPA in accuracy, but this comparison is not based entirely on speed or accuracy.

While HPA did not perform better than the two methods in accuracy, and was only faster then RAxML, it does have an important purpose through its ease of use. Furthermore, HPA is still under active development and therefore these results will not necessarily reflect the eventual released program. Because it was written in Python, integration and testing with it was very easy and efficient. RAxML and FastTree, written in C, were much more difficult to experiment on. This makes HPA still a useful tool, but did not perform as well as FastTree. Furthermore, HPA has not even seen an official release and therefore its final state could end up being very different and outperform these other methods.

References

Berkeley Understanding Evolution Team. (n.d.). Phylogenetic systematics, a.k.a. evolutionary trees. Retrieved January 12, 2018, from https://evolution.berkeley.edu/evolibrary/article/phylogenetics_01

Chor, B., Hendy, M., & Snir, S. (2005, November 30). Maximum Likelihood Jukes-Cantor Triplets: Analytic Solutions | Molecular Biology and Evolution | Oxford Academic. Retrieved March 19, 2018, from https://academic.oup.com/mbe/article/23/3/626/1110325

DendroPy Phylogenetic Computing Library. (n.d.). Retrieved March 19, 2018, from https://pythonhosted.org/DendroPy/index.html#acknowledgments

Elvers, Gunnar. "Building Phylogenetic Trees." Royal Institute of Technology, Dept. of Numerical Analysis and Computer Science, 2001.

Felenstein, J. (1981). Evolutionary Trees from DNA Sequences: A Maximum Likelihood Approach. Journal of Molecular Evolution, 17, 368-376. Retrieved November 15, 2017.

GB, Hall. (2004, December 08). Comparison of the Accuracies of Several Phylogenetic Methods Using Protein and DNA Sequences | Molecular Biology and Evolution | Oxford Academic. Retrieved March 19, 2018, from https://academic.oup.com/mbe/article/22/3/792/1076044

Guindon, S., & Gascuel, O. (2003). A Simple, Fast, and Accurate Algorithm to Estimate Large Phylogenies by Maximum Likelihood. Systematic Biology, 52(5), 696-704. doi:10.1080/10635150390235520

Hall, B. G. (2004, December 08). Comparison of the Accuracies of Several Phylogenetic

Methods Using Protein and DNA Sequences | Molecular Biology and Evolution | Oxford Academic. Retrieved January 06, 2018, from https://academic.oup.com/mbe/article/22/3/792/1076044

Hall, B. G. (2008, April). Simulating DNA coding sequence evolution with EvolveAGene 3. Retrieved March 19, 2018, from https://www.ncbi.nlm.nih.gov/pubmed/18192698

Herron, J. C., Freeman, S., Hodin, J., Miner, B., & Sidor, C. (2015). Evolutionary analysis. Harlow: Pearson.

Olsen, G. J., Matsuda, H., Hagstorm, R., & Overbeek, R. (1994). FastDNAml: a tool for construction of phylogenetic trees of DNA sequences using maximum likelihood. CABIOS, 10(1), 41-48. Retrieved October 18, 2017.

Price, M. N., Dehal, P. S., & Arkin, A. P. (2009). FastTree: computing large minimum evolution trees with profiles instead of a distance matrix. *Molecular biology and evolution*, *26*(7), 1641-1650.

Price, M. N., Dehal, P. S., & Arkin, A. P. (n.d.). FastTree: Neighbor-Joining with Profiles instead of a Distance Matrix. Retrieved March 19, 2018, from http://www.microbesonline.org/fasttree/FastTree-preprint.pdf

Price, M. (n.d.). FastTree. Retrieved March 19, 2018, from http://www.microbesonline.org/fasttree/

Spielman, S. J., & Wilke, C. O. (n.d.). Pyvolve: A Flexible Python Module for Simulating Sequences along Phylogenies. Retrieved March 19, 2018, from http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0139047

Stamatskis, A, et al. “RAxM-III: a Fast Program for Maximum Likelihood-Based Inference of

Large Phylogenetic Trees." *Bioinformatics*, vol. 21, no. 4, 2005, pp. 456–463., doi:10.1093/bioinformatics/bti191.

Sukumaran, J., & Holder, M. T. (2010, June 15). DendroPy: A Python library for phylogenetic computing. Retrieved March 19, 2018, from http://paperity.org/p/41723980/dendropy-a-python-library-for-phylogenetic-computing

Tatusov, R. L., D. A. Natale, I. V. Garkavtsev, T. A. Tatusova, U. T. Shankavaram, B. S. Rao, B. Kiryutin, M. Y. Galperin, N. D. Fedorova, and E. V. Koonin. 2001. The COG database: new developments in phylogenetic classification of proteins from complete genomes. Nucleic Acids Res. 29:22–8.

Williams, Tiffani, and Bernard Moret. 2018. "An Investigation of Phylogenetic Likelihood Methods." Accessed January 6. https://infoscience.epfl.ch/record/97879/files/bibe03.pdf.

What is Python? Executive Summary. (n.d.). Retrieved March 19, 2018, from https://www.python.org/doc/essays/blurb/

Zhang, J. (n.d.). Retrieved March 19, 2018, from https://sco.h-its.org/exelixis/web/software/RAxml/hands_on.html