JUMPSTARTING PHYLOGENETIC SEARCHES

by

Jesse L. Mecham

A thesis submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computer Science

Brigham Young University

April 2006

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a thesis submitted by

Jesse L. Mecham

This thesis has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	Mark J. Clement, Chair
Date	Quinn O. Snell
Date	Dennis Ng

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the thesis of Jesse L. Mecham in its final form and have found that (1) its format, citations, and bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

Date

Mark J. Clement
Chair, Graduate Committee

Accepted for the Department

Parris Egbert
Graduate Coordinator

Accepted for the College

Tony Martinez
Dean, College of Engineering and Technology

ABSTRACT

JUMPSTARTING PHYLOGENETIC SEARCHES

Jesse L. Mecham

Department of Computer Science

Master of Science

Phylogenetic analysis is a central tool in studies of comparative genomics. When a new region of DNA is isolated and sequenced, researchers are often forced to throw away months of computation on an existing phylogeny of homologous sequences in order to incorporate this new sequence. The previously constructed trees are often discarded, and the researcher begins the search again from scratch. The jumpstarting algorithm uses trees from the prior search as a starting point for a new phylogenetic search. This technique drastically decreases search time for large data sets. This kind of analysis is necessary as researchers analyze tree of life size data sets.

0.1 Thesis Statement

This thesis demonstrates that a search utilizing the jumpstarting algorithm will return better trees in less time than a search that chooses not to use jumpstarting.

ACKNOWLEDGMENTS

My many thanks go out to everyone involved in helping this project come to fruition. I would like to thank my advisor, Dr. Mark Clement, who helped keep the fire going even when I became discouraged. Most of all I would like to thank my loving and caring wife, Rachel. Her love and encouragement gave me the strenght to carry on.

Contents

List of Figures

Chapter 1

Introduction

Phylogenetic analysis has become an integral part of many biological research programs. These include such diverse areas as human epidemiology [1, 18], viral transmission [2], biogeography, and systematics. With the advent of high throughput sequencing, an increasingly large volume of sequence data are becoming available. Scientists should be able to take advantage of these data and also of the research that others have performed. For example, when a new virus is detected, it should be possible to estimate a phylogenetic tree (an evolutionary history) containing all related viruses and the unknown variant in order to answer questions such as:

- Where did this virus come from?
- When did this virus arrive in the human population?
- Which related species might have antibodies appropriate for testing in developing new treatments?
- Has this virus been genetically modified through natural or human induced recombinant technology?
- How is this virus evolving and what genetic changes occurred to allow it to successfully enter the human population?

Unfortunately, this kind of phylogenetic search is currently computationally infeasible. The time it takes to perform a complete search using maximum likelihood

exceeds several months with even a small number of sequences (on the order of 100-200). In the case of the SARS epidemic, and others like it, key information must be available in days or at most weeks in order for appropriate action to be taken. Much of the problem comes from the culture and software design for most phylogenetic software packages [17, 8, 7]. These packages require the user to start a search from scratch every time a new sequence is added to the search (this is exactly the situation when a new antigen is observed). The software packages also do not allow users to share partial trees that could speed up the phylogenetic search process. This creates a culture where investigators see little or no benefit to collaborate in phylogenetic research.

What if it were possible to utilize trees from previous phylogenetic searches as a starting point for future searches? The jumpstarting algorithm presented in this thesis allows researchers to use previously generated phylogenetic trees to create better start tree for future searches. By utilizing jumpstarting, it is possible to find better trees in less time than conducting a naive phylogenetic search.

Although jumpstarting may seem like an intuitive concept, there are many factors that must be considered if prior trees are actually going to be of benefit. Through investigating the influence of these factors, researchers can make correct decisions when utilizing jumpstarting to speed up the generation of phylogenetic trees. Such factors include:

- The number of taxa inserted into a new search. Researchers will often sequence a new specimen and want to incorporate this sequence data into an existing search. We have found that when a small number of sequences are inserted, the trees from prior searches are of great benefit to the search with new sequences. When too many sequences are added, however, the effectiveness of jumpstarting diminishes.

- Alignment of previous searches. When new sequences are added to a search, a new alignment must normally be performed to incorporate the new data. The alignment used to generate prior trees also influences the impact these prior

trees will have on the new search. Even when a poor alignment is used to generate prior trees, jumpstarting is able to gain some benefit from using these trees.

- Type of consensus used when merging trees from previous searches. When several overlapping trees are used from prior searches, it may be necessary to merge the trees to create a jumpstart tree with maximal overlap with new search sequences. Many trees may be available from prior searches with the same score. The way in which consensus is performed can greatly impact the efficacy of jumpstarting.

This thesis will demonstrate that through correct usage of jumpstarting, researchers can significantly reduce the time required to conduct a phylogenetic search without compromising the quality of the trees returned.

1.0.1 Phylogenetics

The branching pattern of ancestor/descendant relationships among species or their parts (e.g., genes) is a phylogeny. Researchers attempt to estimate these historical relationships by examining character evolution using a tree – a mathematical structure used to model the actual evolutionary history of species or their parts [6]. These inferred trees (historical branching relationships) can be represented as cladograms, where branch lengths are arbitrary and only the branching order is significant, or as phylograms, where the branch lengths are proportional to the amount of evolutionary change along the branch.

Phylogenies were historically used to classify organisms into natural evolutionary groups based on these ancestor/descendant relationships. Indeed, great effort is currently being spent on estimating the *Tree of Life* to quantify the biodiversity of our planet [3]. However, phylogenies have also spread in use as the utility of the evolutionary framework for numerous other disciplines becomes increasingly obvious [13]. For example, phylogenies are now being extensively used in the biomedical sciences including developmental biology, genomic biology, infectious disease, virology,

and human genetics.

Phylogenies have become essential tools in the study of the molecular epidemiology of disease agents. A prime example of the troubles encountered when the phylogenetic approach is ignored comes from the outbreak of the West Nile Virus in New York City. This virus was responsible for multiple deaths in New York, yet the Centers for Disease Control and Prevention (CDC) initially misdiagnosed the causative agent as St. Louis encephalitis due to their lack of an appropriate phylogenetic comparison [4]. The study of origins, spread, and diversity of pathogens are clearly evolutionary questions. Only after the serological evidence was coupled with strong phylogenetic evidence was the West Nile Virus correctly identified as the etiological agent responsible for the encephalitis outbreak in New York [9].

Phylogenetic estimation is accomplished by optimizing character change relative to some criterion over a tree. The tree for which the character data show the best optimization is the preferred tree. Two of the principle optimization criteria used by researchers are maximum parsimony and maximum likelihood. The parsimony criterion attempts to minimize the number of changes among a tree for shared-derived characters, while likelihood attempts to maximize the probability of change for all characters relative to some model of evolution. Each criterion has its own strengths and weaknesses. For example, maximum parsimony can incorporate insertion/deletion (indel) events and have asymmetric changes (e.g., a change from character A to character B is not the same as a change from character B to character A), whereas current implementations of maximum likelihood cannot accommodate these biological realities. Likewise, maximum likelihood can account for heterogeneity in evolutionary rates and multiple changes at the same character position, whereas maximum parsimony cannot. Thus there is, often times heated, discussion about appropriate methods to use to estimate phylogenetic relationships.

Another reason there is such debate about phylogenetic methods is that their performance varies depending upon the type of data used, the number of sequences involved, and the depth of the evolutionary relationships to be inferred. Exact searches, those that explore every possible tree topology for a given optimality criterion, are

only possible for a very small number of taxa (on the order of 20-30). This limited search is due to the rapidly increasing number of possible trees with a modest increase of taxa [5]. The total number of (unrooted, strictly bifurcating) trees for T taxa is shown in Equation 1.1.

$$B(T) = \prod_{i=3}^{T}(2i - 5) \tag{1.1}$$

So, for example, with only 50 sequences, there are $3x10^{74}$ possible trees. For the tree of life, there are estimated to be well over 10 million species, yet for 10 million sequences there are $5x10^{68,667,340}$ possible trees. Therefore, the phylogeny problem is a particularly tough one that is well suited for distributed technology (because one performs the same calculations over different, independent, tree topologies) such as web based systems that utilize distributed resources.

Phylogenetics has become an active field in and of itself [15]. It is an extremely exciting field where talents in mathematics, computer science, and biology can be brought together to work on the problem of inferring historical relationships. A survey of the recent literature in many of the biomedical fields will attest to the ever increasing applicability of phylogenetic analysis.

1.0.2 Related Work

There are a wide variety of programs available to researchers to conduct phylogenetic searches. There is no 'best' phylogenetic program out there currently. Each of the programs have their strengths and weaknesses. Each program implements a different strategy for finding the best phylogenetic trees based upon the investigator's criteria. Researchers often have their personal favorites based upon past experience and size and properties of the dataset they are investigating. The advantage presented by jumpstarting is the fact that it should be able to work in conjunction with just about every phylogenetic search program available. This is due to the fact that jumpstarting is not truly a search algorithm, but an algorithm designed as an add-on to enhance the performance of current phylogenetic search programs. Below are descriptions of the most popular phylogenetic programs.

PAUP*

Phylogenetic analysis using parsimony (PAUP*) has established itself as the industry standard over the course of the past fifteen years. It is a commercial product currently in version 4.0 [17]. While it does not handle large datasets as well as some of the newer competitors, researchers continue to use it because it is well understood and recognized. Research is continually being done to find ways to extend PAUP's functionality to handle larger datasets (see DCM3 below).

PSODA

An open-source phylogenetic program designed to include all of the basic functionality of PAUP* [16]. While PSODA is currently able to perform the basic functions needed in phylogenetic searching, it does not currently include many of the extensions that make PAUP* desirable. The major advantage of PSODA is that it is an open source product that performs as well as the commercial product PAUP*. While past open source phylogenetic projects (PHYLIP, Mesquite, etc.) have appealed to the community as a free alternative, their poor performance has turned many serious researchers away to commercial products. It is hoped that as PSODA gains greater recognition, the open source community will embrace it and add the extended functionality necessary to compete with the other other commercial products.

PHYLIP

The phylogeny inference package (PHYLIP) is a collection of open source C packages that has been in distribution since 1980 [7]. The most recent version (3.65) was released in August of 2005. PHYLIP performs most of the basics types of phylogenetic searches and is widely used due to its free, open source nature. However, its performance is considerably lower than that of the other programs described in this thesis. Furthermore, as the size of sequence datasets increases, the gap between PHYLIP and these other programs is increasing.

TNT

Tree analysis using new technology (TNT) is a relatively new commercial program that utilizes a combination of simulated annealing (along with the parsimony ratchet), divide-and-conquer, and genetic algorithms to reconstruct claudiograms [10, 8]. Not much has been published about the inner workings of the program, but it has already created a quite a stir within the community due to its ability to process large data sets. It is considered by some to be the best performer currently on the market.

DCM

This program has gained some attention in recent years as a way to analyze larger datasets. It was designed to work on datasets in the range of thousands of sequences and has been shown to be effective up to almost fourteen-thousand sequences [14]. Disk Covering Methods (DCM) is essentially a divide-and-conqueror strategy used for claudiogram reconstruction. While there are a variety of flavors of DCM (DCM1, DCM2, and the most recent Parallel-Recursive-Iterative DCM3), they all center around the following four phases:

1. Dividing or decomposing the dataset
2. Solving the smaller subproblems
3. Combining the subproblems
4. Refining the resulting tree

The difference between the versions of DCM have to do with the way in which the different subproblems are broken down and joined back together. DCM is not a search program in and of itself, but relies on another program underneath (PAUP*, TNT, etc.) to work on the localized subproblems.

1.0.3 Jumpstarting

Since the phylogenetic search space is so large, it is extremely important to create search heuristics that are as efficient as possible. The jumpstarting algorithm was

developed to take advantage of previous searches in order to speed up new phylogenetic computations. Additionally, emphasis was placed on recognizing the difficulties researchers face when conducting phylogenetic research and trying to minimize work lost. A common example of work lost occurs when a phylogenetic search is begun on a collection of organisms (fleas for example), yet must be restarted from scratch when sequence data from a new organism is made available. By allowing the researcher to continue searching from the point they left of originally, months of computational time already invested can be utilized.

The Jumpstarting algorithm consists of four main parts:

1. Selecting from available trees those that might contribute to a new search.
2. Conducting a phylogenetic search.
3. Storing the new trees in a way that they may be accessed in future phylogenetic searches.

While a simple algorithm at its core, it allows for flexibility in they ways it collects tree data and combines it into new starting trees. There are various ways to implement jumpstarting, one possible algorithmic implementation is given below:

1. Let $T = \{x \mid x \in \{$ taxa involved in the new search $\}\}$.
2. Query the data base for prior searches with the set of taxa S_i where at least one of the taxa in the prior search is the same as the new search ($x \in T$ and $x \in S_i$).
3. For each of these prior searches on taxa S_i, determine the intersection $I_i = T \cap S_i = \{x \mid x \notin T \text{ and } x \in S_i\}$.
4. Use the Newick parenthetical notation for the best tree from this maximal intersection as the base tree for the new search.
5. Add taxa $x \in T$ where $x \in S_i$ to all trees within N_i.

6. Begin a normal search with the trees from N_i.

Figure 1.1 provides a concrete example of the jumpstart algorithm. In this example, User A on Peer 1 has performed a search resulting in Tree X version 1.1. The following steps are included in the algorithm:

1. User B on Peer 2 prepares a set of taxa that will be used in a phylogenetic search and creates the data structure for Tree Y, Version1.1. A query is sent to peer machines to determine if searches have already been performed with some of these taxa.

2. Peer 1 has Tree X Version 1.1 which matches the criteria in the query. This tree is returned to Peer 2.

3. Peer 2 uses Tree X Version 1.1 combined with other local taxa to jumpstart a phylogenetic search.

4. After expending significant computational resources, User B generates Tree Y Version 1.2 which refines the relationships between taxa in Tree X as well as Tree Y. This version of the tree is entered into the database.

5. User A has a reference to Tree Y since a subtree of Tree X was used as a jumpstart point for Tree Y. When Tree Y Version 1.2 is generated Peer 1 can send a query for derivative trees of Tree X Version 1.1.

6. Peer 2 will return Tree Y Version 1.2. User 2 may decide that all of the relationships contained in Tree Y Version 1.2 should not be made public. In this case, the subtree containing only the nodes originally found in Tree X would be returned.

7. Peer 1 receives the refined relationships and can create Tree X version 1.2. This tree can be used for future searches.

In this example interaction, both User A and User B have benefited from the collaboration. The tree returned from Peer 2 can be used, or discarded depending on

the value that User A places on the results. User B has been able to cut months off of his search time because of the initial jumpstart tree he/she was able to derive from Tree X Version 1.1.

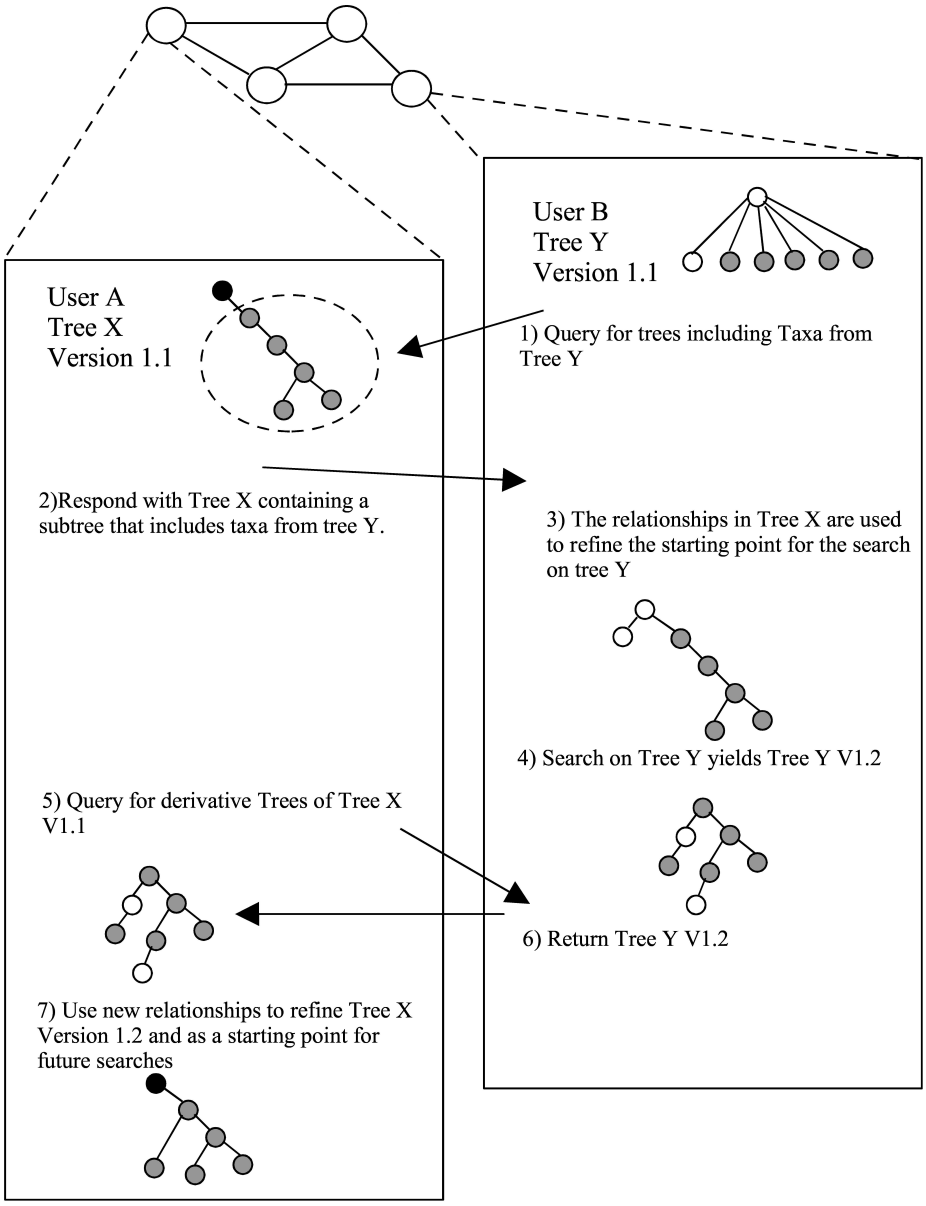

Figure 1.1: Example Jumpstart Interaction

Chapter 2

Methods

Typical phylogenetic studies involve the collection of data, sequence alignment and phylogenetic analysis. Jumpstarting a phylogenetic analysis can improve the results in multiple stages of this process. When a new sequence is added to a data set, a new alignment must normally be computed, and the old trees that were generated in phylogenetic analysis are not normally used in further analysis. Both alignment and phylogenetic analysis are time-consuming processes (some analyses have been known to take months). When a researcher adds a sequence to the analysis, all the previous time spent in computation is essentially thrown away because the researcher must start over again. This drastically slows the scientific process. Jumpstarting eliminates the wasted time by taking advantage of previous analyses.

2.0.4 Experimental_Setup

In order to analyse the effectiveness of the jumpstarting algorithm, an experiment was constructed an to simulate real phylogenetic analysis. A database was constructed to hold the phylogenetic trees returned from a search and to make them accessible to future queries. Four representative data sets were thoroughly analysed:

- Zilla: a 500 taxa dataset 759 nucleotide bases in length.
- Avian: a 921 taxa dataset 1120 nucleotide bases in length.
- Three Genes: a 567 taxa dataset 2153 nucleotide bases in length.
- HIV: a 397 taxa dataset 8583 nucleotide bases in length.

After selecting the data sets to run the experiment, the database was populated based on the following algorithm:

1. For each dataset $D_{original}$ consisting of t taxa, remove n random taxa to create a new dataset D_{small} consisting of *(t-n)* taxa.

2. Run PAUP* [17] on each D_{small} datasets for specified length of time, resulting in a set S_{small} of trees.

3. Insert S_{small} into the database along with all relevant information about how the trees were created.

4. Taking the original n taxa that were removed in step (A) and connect them to the base of each tree in S_small to create a new set of trees $S_original$.

5. Beginning a parsimony search by using the trees in $S_original$ as a starting point for the search.

At this point, the database has been populated and may now be queried by the jumpstarting algorithm. This is where the jumpstarting part of the experiment begins by:

1. Taking the original n taxa that were removed in step (A) and connecting them to the base of each tree in S_{small} to create a new set of trees $S_{original}$.

2. Beginning a PAUP* [17] TBR parsimony search by using the trees in $S_{original}$ as a starting point for the search.

The above mentioned algorithm was designed in order to simulate scenarios where jumpstarting would be beneficial to a researcher. Some possible scenarios might include a researcher who has been conducting a ten month phylogenetic analysis on various birds, but has just finished sequencing n new sequences that she wishes to insert into her avian dataset. Another scenario might be a researcher who has just sequenced n new HIV isolates and wants to know how these isolates are related to those already researched by colleagues at another institution. In each of these

scenarios the researcher is concerned with inserting a few new taxa into trees already constructed.

As the number of new taxa is increased, the jumpstarting algorithm should become less effective. It is important to know when jumpstarting should be abandoned and the researcher should just start a new full search. In order to answer this question, the jumpstarting algorithm will be run with *n = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 50, 60, 70, 80, 90, 100* for all datasets and *n = 200, 300, 400* additionally for the Avian and Zilla datasets (due to the fact that they contain a larger number of taxa). For our experiments, the Ratchet algorithm [12] was used in step (2) above in order to create the S_{small} set of trees. The ratchet will be allowed to run between 30 and 200 iterations and will be set to weight 30% of the matrix characters. This simulate's trees that would be present in a researcher's database after searching on a dataset for a significant amount of time.

In order to analyze the impact of alignment on jumpstarting, gaps will be removed from sequences in the dataset $D_{original}$, and the sequences will be realigned with CLUSTALW using the *fast* option [19]. This option creates a poorer alignment (which ultimately results in poorer trees). This experiment was performed to analyze jumpstarting's ability to overcome noise introduced by poor alignment. This is important in situations where the researcher may wish to incorporate trees into a jumpstarting search from an outside source, but may not be confident in the alignment used by outside researchers.

In order to analyse the impact of alignment on jumpstarting, gaps were removed from sequences in the data set $D_{original}$, and the sequences were realigned with CLUSTALW using the 'fast' option. This option creates a poorer alignment (which ultimately results in poorer trees). This experiment was performed to analyse the jumpstarting's ability to overcome noise introduced by poor alignment. This is important in situations where the researcher may wish to incorporate trees into a jumpstarting search from an outside source, but may not be confident in the alignment used by outside researchers.

Chapter 3

Results

While the primary goal of jumpstarting is to return lower scoring trees in less time, it is not sufficient to simply look at tree score over time. A variety of factors must be analyzed and studied when analyzing the effectivness of jumpstarting. The quality of the trees returned and the factors that directly influence the effectivness of jumpstarting must be scrutinized in order to gain a better understanding of the algorithm. The results are therfore divided into subsections as follows.

3.0.5 Tree Score vs. Time

In order to quantify the effectiveness of jumpstarting searches, we analysed the time to generate better tree using jumpstarting and compared these results with the time required to reach a better tree when starting from scratch. Better trees have shorter length values. In all the data sets analysed, jumpstarting TBR found better trees in less time than the equivalent regular TBR search. However, the time at which that score was found and the overall final score was found to be heavily dependent on a variety of factors:

- the number of taxa inserted
- the alignment quality
- the total number of taxa.

One of the greatest advantages of jumpstarting is its ability to find more optimal trees in substantially less time than a regular TBR search. In order to measure

this difference, execution times were recorded for each tree found in the experiments described in Section 3.1. A 'scratch' TBR parsimony search was also performed, and the time at which all trees were discovered was recorded. By comparing the results of each experimental jumpstart run with those obtained from the scratch experiment, we are able to analyse the performance benefits of jumpstarting over regular phylogenetic searches. All scratch data sets were allowed to run for a minimum of 24 hours beyond the point they found their lowest scoring tree. In all cases, the jumpstart data sets were run for less time than the corresponding scratch data set. All experiments were run separately on a 700 MHz SGI MIPS R16000 processor using 64 GB shared memory. The results of this analysis are shown in Figure 3.1. Each graph shows the results recorded from one data set. In order to make the plots more readable, results were grouped by the number of new taxa inserted (1-5, 5-10, 10-50, etc.). Each data point was then averaged with all other members of its group, and the results are plotted on the appropriate graph. The result of each respective 'scratch' search is also shown for comparison.

In experiments where one hundred or fewer taxa were inserted, jumpstarting returned significantly lower tree scores within the first 120 seconds than a regular TBR search was able to find over the course of the entire experiment. When ten or fewer taxa were inserted into the search, jumpstarting found a better tree in sixty seconds, than what a TBR scratch found during the entire experiment. One prime example of this is seen in the Three Genes experiment, where jumpstarting with the addition of 1-5 taxa returned a score of 44168 within sixty seconds, while the best tree using the TBR scratch method after 65 hours was returned a score of only 44179 (Figure 3.1). However, it should be noted that the ability of jumpstarting to quickly return low-scoring trees is inversely related to the number of new taxa inserted into the search. In other words, as the number of new taxa incorporated into a single jumpstart search increases, the effectiveness of jumpstarting will decrease. However, it is important to note that this result suggests that jumpstarting does, in fact, reduce the amount of time required to conduct a phylogenetic search and the final trees returned are better than those found through naive searching.

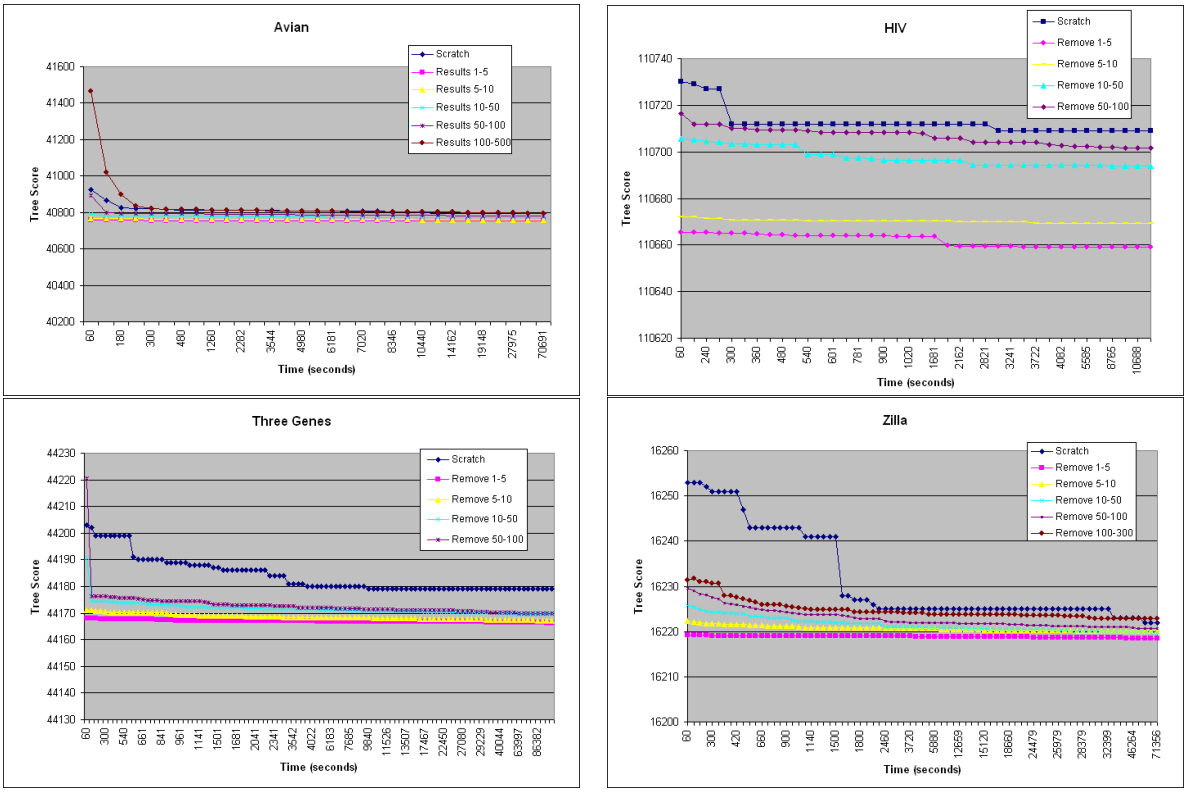

Figure 3.1: Each graph represents a different data set. All data series are grouped by the number of taxa inserted into the jumpstart search (1-5 taxa inserted, 5-10 taxa inserted, 10-50 taxa inserted, etc. as noted in legend of each graph). Additionally, a TBR search where no jumpstarting was used (scratch) is shown in each of the graphs. Note that for each data set, when 50 or fewer taxa were inserted, jumpstarting found better trees in significantly less time. However, it must be noted that as the number of taxa inserted increases, the effectiveness of jumpstarting decreases.

In order to understand why jumpstarting reaches its best tree score so quickly, it is helpful to observe the trends seen in the initial trees it returns. Below are the PAUP* tree scores from the first sixty seconds of a jumpstart search. The scores are grouped by the number of taxa added to the jumpstart tree (Figure 3.2). While an observable trend shows the initial jumpstart tree score increasing with the number of new taxa incorporated, it should be noted that jumpstarting consistently begins its search with more optimal trees in each case that has less than 50 taxa incorporated into the new search. This gives us a feel for how many new taxa a researcher can insert into a jumpstarting search and still see instant measurable benefits.

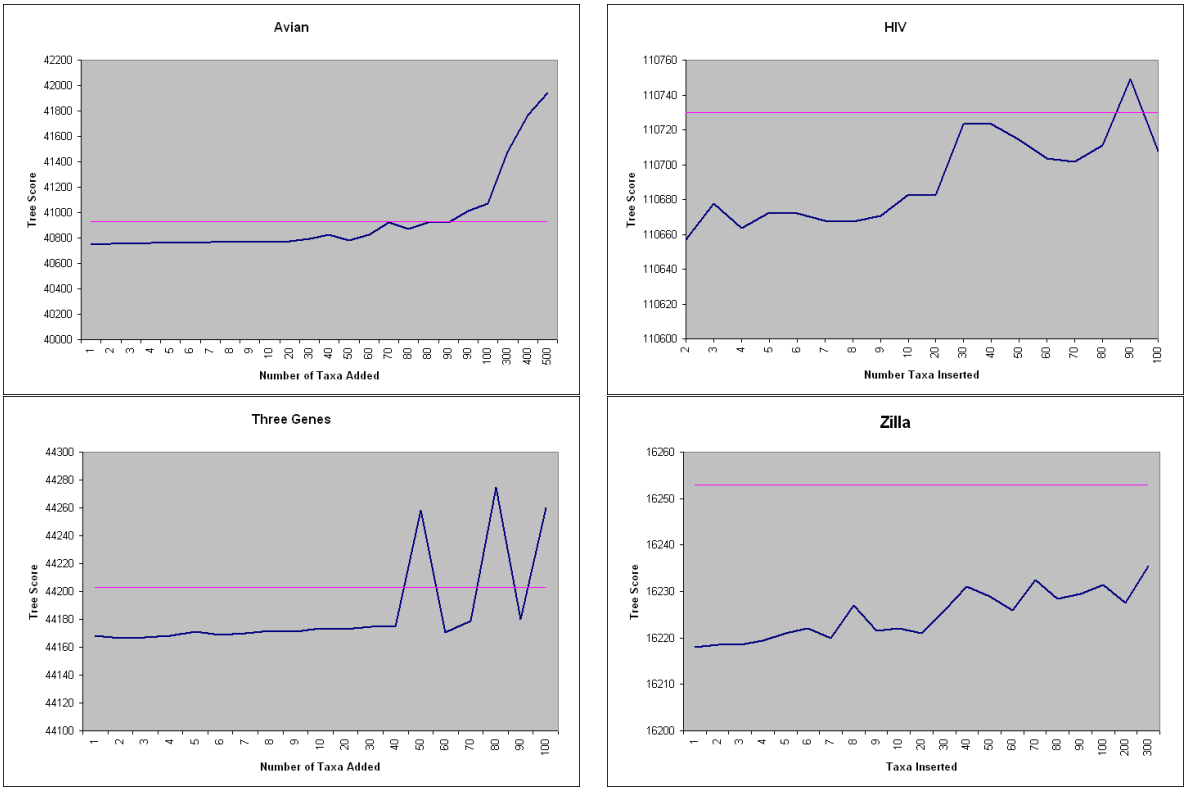

Figure 3.2: Each graph represents a different data set. The horizontal line in each data set represents the starting score of a TBR search, where no jumpstarting was used. The other line in each graph represents the average starting score (y-axis) for each jumpstarting experiment, grouped by number of taxa inserted into the jumpstart search (x-axis). Note the trend that as the number of taxa inserted increases, the average starting score increases as well.

It is possible to take these results one step further and compare the speed-up jumpstarting offers over a regular TBR search by measuring the time it takes for a TBR to find the worst/initial jumpstarting tree in a corresponding experiment (Figure 3.3). By observing the graph, it becomes apparent that as the number of taxa incorporated into the jumpstart search increases, the overall speed-up afforded by jumpstarting decreases. When five taxa were inserted, it took regular TBR over 2.5 hours to find the same tree that jumpstarting found in 60 seconds. When more than 100 taxa were inserted, however, jumpstarting returned worse starting trees than a regular TBR search did.

3.0.6 Tree Quality

Although a particular algorithm may be efficient at finding a sub-optimal tree quickly, the researcher is more concerned about the final tree score than the time it took to reach sub-optimal trees. In order to analyse the quality of trees found by the jumpstarting algorithm, we compared the scores of the best tree found by using the jumpstarting algorithm with the best tree scores found by a TBR search from scratch (Figure 3.3).

There is a general trend observed that the final tree score is inversely related to the number of taxa removed. As the number of taxa inserted into a jumpstart search increases, the average score of the best tree found during a jumpstarted phylogenetic search decreases. In all of our experiments, if fewer than thirty taxa were inserted into the search, jumpstarting found better trees than a regular TBR search.

There was a point at which the jumpstart search began to return worse trees than a regular TBR search, but it only occurred after a significant number of taxa had been added to the jumpstart search. The first occurrence of diminishing results occurred when the number of taxa inserted was greater than 6% of the total taxa included in the search (Zilla data set), while the average over all data sets was 11% of the total taxa included in the search. These results again suggest if the researcher has an understanding about the reasonable limitations imposed by jumpstarting, a jumpstarting search can return at least equally good trees in significantly less time.

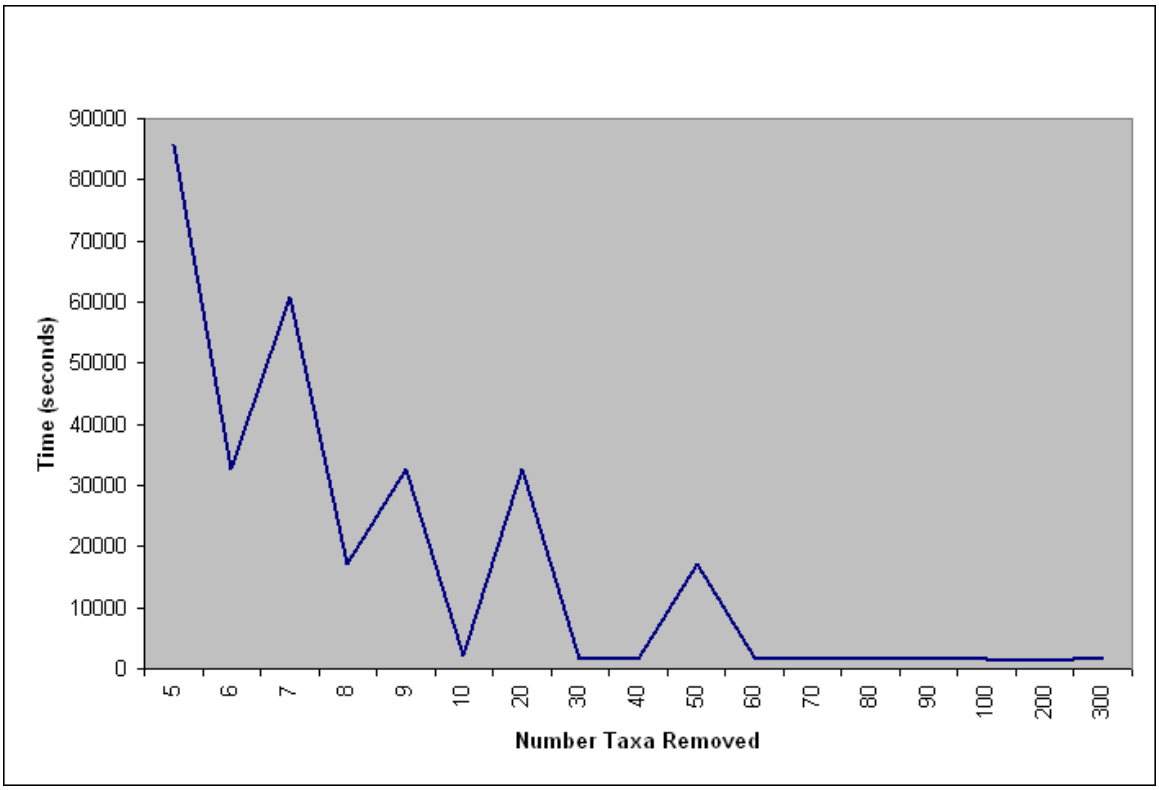

Figure 3.3: The above chart shows the average searching time it took TBR to find a tree with the same score as the initial search tree returned from the jumpstarting algorithm. The above chart shows the scores of the Zilla data set. The plotted coordinates represent the time (x-axis) it took for a regular TBR search to achieve the average score of the initial tree for each experiment grouped by the number of taxa inserted (y-axis). Since the regular TBR search never found a tree equal to the first jumpstarting tree when less than five taxa were inserted, only data points where five or more taxa were inserted are shown. Notice that as the number of taxa inserted increases, the time required for a regular TBR search to find an equally good tree decreases.

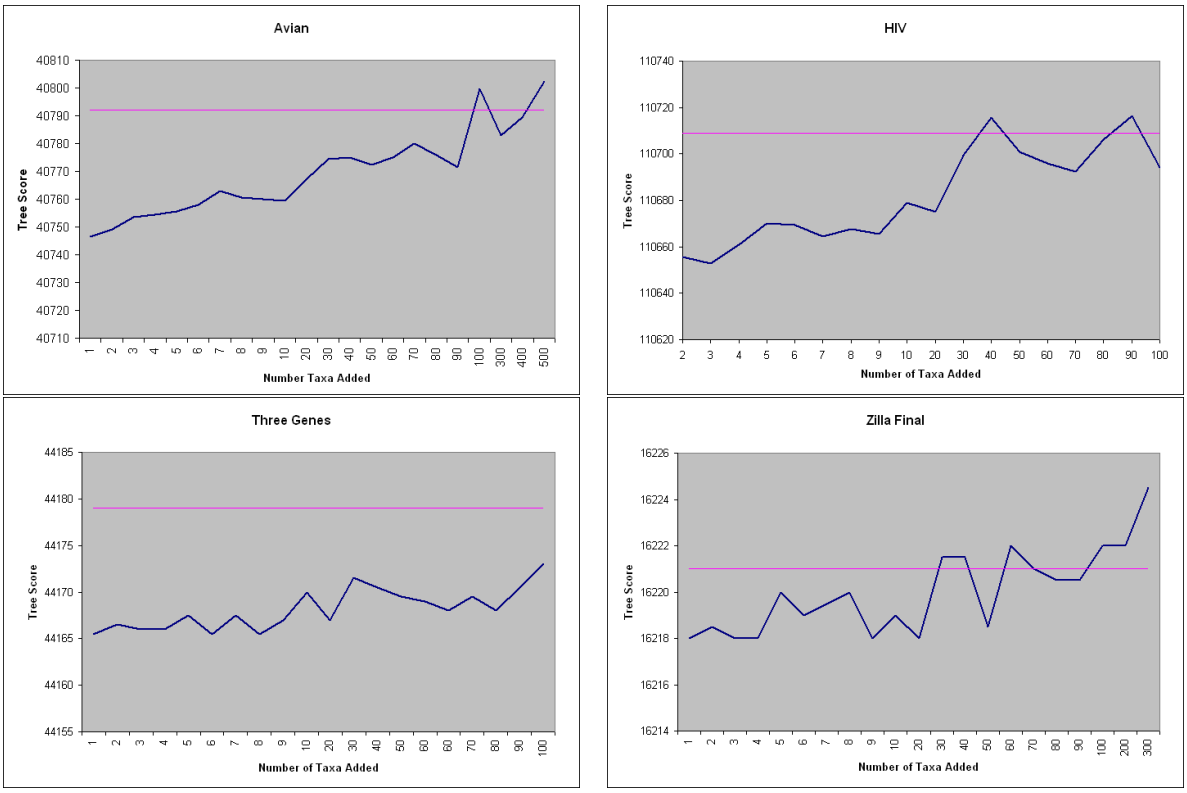

Figure 3.4: Each graph represents a different data set. The horizontal line in each data set represents the final tree score of a TBR search where no jumpstarting was used. The other line in each graph represents the average final score (y-axis) for each jumpstarting experiment, grouped by number of taxa inserted into the jumpstart search (x-axis). Note the trend that as the number of taxa inserted increases, the average starting score increases as well. It is also important to observe that jumpstarting always finds better trees, if twenty or fewer taxa are inserted.

3.0.7 Alignment Quality

When analysing the effects of alignment on jumpstarting, the decision was made to test trees constructed from a sub-optimal alignments in order to asses the ability of the algorithm to eliminate noise introduced by poor alignments. Each of the experiments was analysed based on the same metrics in other experiments: time to best tree and tree score (Figures 3.5-3.6).

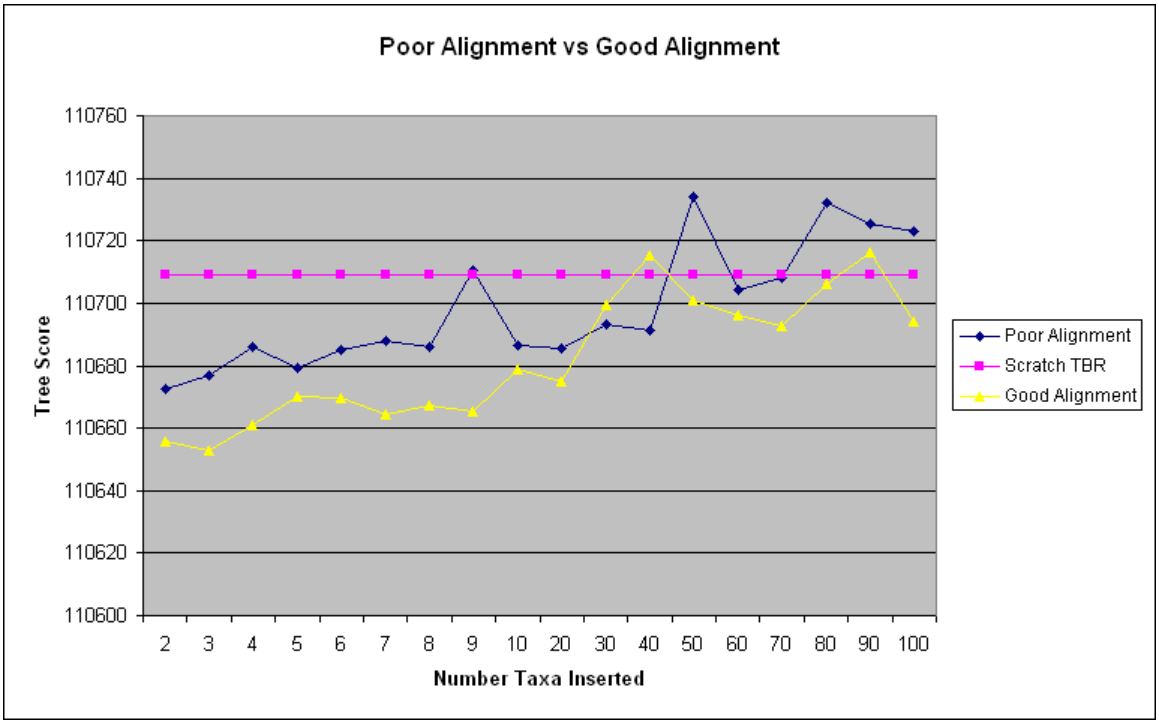

Figure 3.5: The horizontal line represents the final tree score of a PAUP* search where no jumpstarting was used. All final PAUP* searches (scratch, good and poor) were done using the same final alignment. The series labelled *poor* represents experiments where trees fed to the jumpstarting algorithm were created from a poor alignment. The poor alignment was created by using the fast option in CLUSTALW. The series labelled *good* represents experiments where trees fed to the jumpstart algorithm were created from a good alignment (using the slow option in CLUSTALW). Note that by using jumpstart trees that were created using a poor alignment, the effectiveness of the jumpstarting algorithm decreased, yet it still outperformed a TBR PAUP* search from scratch.

HIV was chosen as the example in this paper since it had the longest sequences

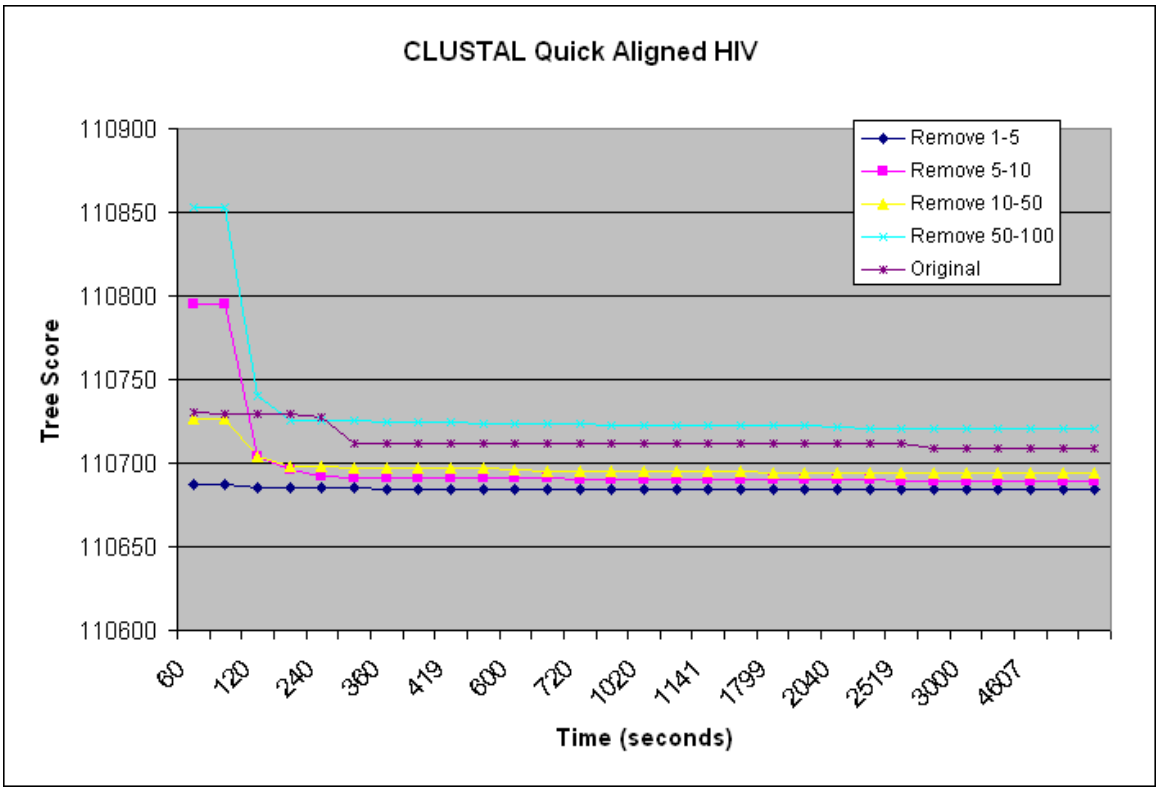

Figure 3.6: All data series are grouped by the number of taxa inserted into the jumpstart search (1-5 taxa inserted, 5-10 taxa inserted, 10-50 taxa inserted, etc. as noted in legend of each graph). Additionally, a PAUP search where no jumpstarting was used (original) is shown in each of the graphs. Note that for each data set, when 50 or fewer taxa were inserted, jumpstarting found better trees in less time. However, it must be noted that as the number of taxa inserted increases, the effectiveness of jumpstarting decreases.

and was most sensitive to alignment errors. In order to compare the relative change in jumpstarting performance, the results of the poorly aligned dataset were compared with the results of a good alignment (Figure 3.5). While the results show that jumpstarting with poorly aligned trees outperformed a 'scratch' search, the jumpstarting performance was decreased by approximately 30% when compared with trees constructed from good alignments. While there is a definite decrease in jumpstarting performance when a poor alignment is used to create the dataset, jumpstarting shows that it still has the robustness to outperform a regular TBR search even when the data used to feed it is not correctly aligned.

3.0.8 Consensus Algorithms

Jumpstarting is advantageous when adding sequences to an existing analysis. However, this is not the only use for jumpstarting. More typically, a researcher may request data from the database and wish to begin computation by taking advantage of these data. A user can request all trees that contain certain taxa or sequences from the database. However, these trees may also contain extraneous taxa. One approach is to simply strip out the extraneous samples and use the resulting trees as a starting point. A consensus tree could also be used.

Figure 3.7 demonstrates the different jumpstarting possibilities available. A researcher may choose to start computation based on one of the most optimal (Smallest) trees returned from the query. Optionally, a consensus tree may be created and used for jumpstarting the new search (Strict, 50%). Preliminary studies show that creating a majority rule consensus tree (50%) from the collection of most optimal trees returned by the jumpstart system seems to be the best option. Figure 3.7 shows that this method found the most optimal tree in 17 minutes, whereas the other choices took at least 57 hours. Increases in performance such as this are vital for advances and to apply the power of the phylogenetic approaches to studies in biomedical research.

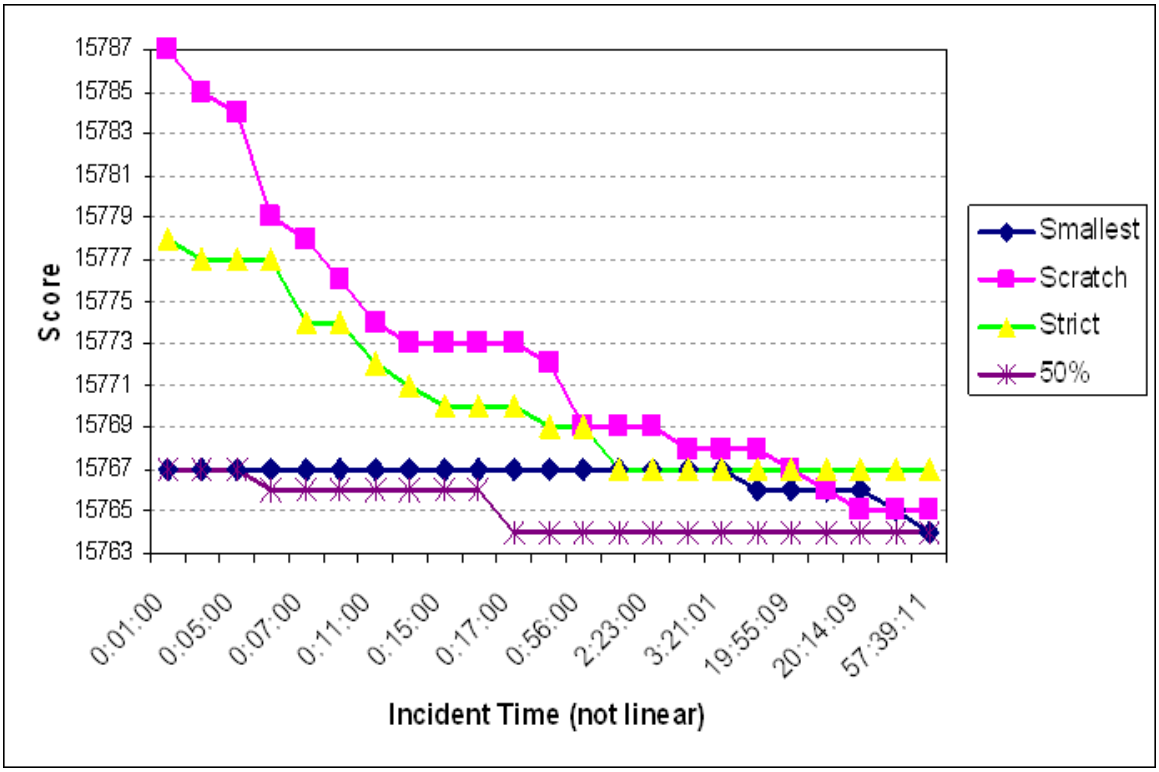

Figure 3.7: The graph illustrates the effect of various schemes for incorporating trees into a single jumpstarting set. Strict consensus tends to eliminate too much of the inherent structure of the jumpstarting tree, giving diminished results. However, using 50% consensus tends to show the most promising results in our experiments.

Chapter 4

Conclusion

Jumpstarting is effective at improving the ability of researchers to quickly generate phylogenies. When a new epidemic strikes, it is often important to determine the relationship between the current organism and others that have been successfully treated previously. This process can take a prohibitively long period of time with current algorithms. The jumpstarting algorithm can generate superior phylogenetic relationships much more quickly than existing algorithms. These relationships can be used to make informed decisions in epidemiology and other areas.

Through the course of this experiment, there were a variety of important factors that became apparent. These factors concist of:

- The number of taxa inserted into a new search. When an average of 30 or fewer taxa is added to a search, jumpstarting significantly outperforms searches started from scratch. The exact number is dependent on the data set and future research will define ways to define this threshold.

- Alignment of previous searches. While there is a decrease in jumpstarting performance when poor alignments are used for prior trees, jumpstarting still outperforms searches started from scratch. Future work will investigate which trees to incorporate into a search based on alignment differences.

- Type of consensus used when merging trees from previous searches. The 50% consensus shows the greatest improvement out of all consensus methods investigated. Future research will investigate additional consensus methods and their effect on jumpstarting.

Altering any one of these factors can drastically affect the jumpstarting's ability to return optimal trees in minimal time. However, even in a world of imperfect data, jumpstarting can be a powerful tool used by researchers to decrease the time required to conduct phylogenetic searches and improve the optimality of the trees they produce.

The results presented in this thesis have show strong evidence that jumpstarting is an effective algorithm for reducing search times while improving tree quality in phylogenetic analysis. It can be combined with various search algorithms to deal with important medical problems. Future work will investigate ways of mining the database and combining existing trees to provide the best starting point for future searches. It is hoped that these results will encourage researchers to begin collecting the trees that they have been discarding thus far, and help in developing a community which not only shares proteomic and nucleotide data, but also feels encouraged to share evolutionary data to facilitate the efforts of their colleagues.

Appendix A

Initial Results

Included below is an initial result obtained while developing the jumpstarting algorithm.

A.1 Two Thousand Dataset

The following result was one of the first experiments run using the jumpstarting algorithm. Figure A.1 demonstrates that while both jumpstarting and regular phylogenetic searches reach the same score eventually, the jumpstarting algorithm finds the best tree score in a matter of minutes, while the non-jumpstarted search takes almost a day.

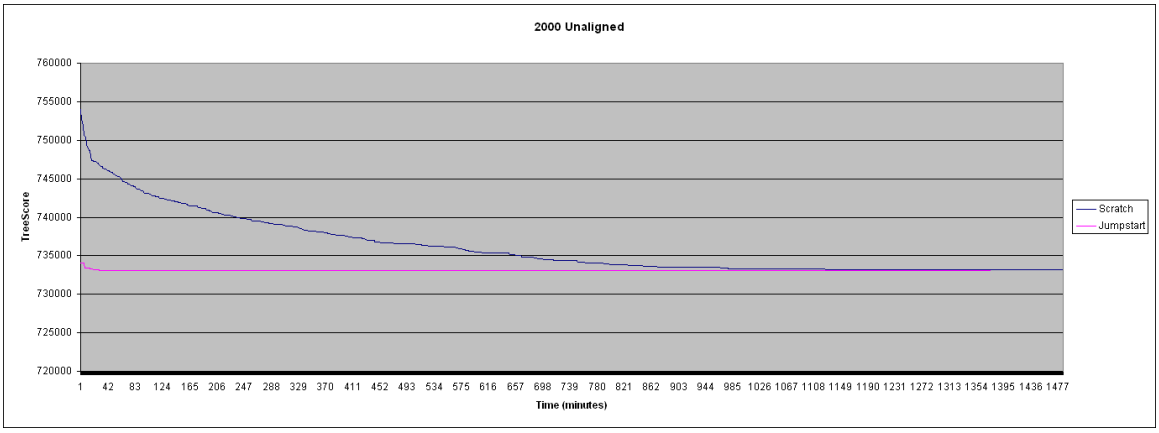

Figure A.1: The above graph shows the score of the best tree found while conducting a PAUP phylogenetic search with (top line) and without (bottom line) the jumpstarting algorithm. Both experiments were run on a set of two thousand unaligned sequences and were run on the same computer.

Appendix B

Implementation

The JumpstartSearch program was designed and written in order to implement and test the jumpstart algorithm. It was writen in modular PERL and utilizes BioPERL modules. The current implementation is configured to run either from the command line or through a web interface. All trees used in the jumpstarting algorithm are pulled from a PostgreSQL datbase named TreeBank. A CVS repository was created at the Computational Sciences Laboratory (CSL) at Brigham Young University which holds the JumpstartSearch program. The following sections describe the PERL program and the TreeBank database.

B.1 JumpstartSearch

The JumpstartSearch program consists of a main control program (jumpstartSearch.pl) which the user can call from the command line or place in a web-accessable directory and interface as cgi script. There are a collection of essential modules and files which jumpstartSearch.pl calls to accomplish various parts of the algorithm as well as some helper modules included in the package. The essential modules and files include the following:

- DatbaseInterface.pm
- SequenceControler.pm
- TreeControler.pm
- TreeBankControler.pm

- SettingsControler.pm
- Ratchet.pm
- settings.txt

Optional modules are as follows:

- EmailInterface.pm
- populateDB.pl
- uploadtree.cgi

As part of the instalation process, the user must also specify a collection of settings found in settings.txt in order to dictate properties of the search program. Further explanation can be found in the PERLDOC documentation located in the CVS repository as noted above.

B.2 TreeBank Schema

In order to store the trees used in the jumpstarting program, the TreeBank database was designed. It drew from many of the concepts presented by Nakhleh [11]. The schema given here was designed for PostgreSQL and can be found in the distributed JumpstartSearch package.

```
--#COMMENT Dropping tables

DROP TABLE sequence;
DROP TABLE score;
DROP TABLE scoretype;
DROP TABLE treeends;
DROP TABLE tree;
DROP TABLE spanning;
```

```
DROP TABLE edge;
DROP TABLE region;

DROP TABLE users;
DROP TABLE groups;
DROP TABLE usr_grp_map;


DROP SEQUENCE tree_counter;
DROP SEQUENCE sequence_counter;

--#COMMENT This is the sequence generator so we can auto-increment in Postgres
CREATE SEQUENCE tree_counter START 1;
CREATE SEQUENCE sequence_counter START 1;


--#COMMENT Creating tables
CREATE TABLE users (
    id              BIGINT NOT NULL PRIMARY KEY,
    name            VARCHAR(128),
    location        VARCHAR(128),
    phone           VARCHAR(30)
);

CREATE TABLE groups (
    id              BIGINT NOT NULL PRIMARY KEY,
    name            VARCHAR(128)
);

CREATE TABLE usr_grp_map (
```

```
    id              BIGINT references users(id),
    groupid         BIGINT references groups(id)
);

CREATE TABLE tree (
    id              BIGINT NOT NULL PRIMARY KEY,
    --        userid          BIGINT references users(id),
    --        groupid         BIGINT references groups(id),
    description     VARCHAR(32000),
    newick          BYTEA
);

CREATE TABLE sequence (
    id              BIGINT NOT NULL PRIMARY KEY,
    --        userid          BIGINT references users(id),
    --        groupid         BIGINT references groups(id),
    treeid          BIGINT references tree(id),
    name            VARCHAR(512),
    hash            VARCHAR(128),
    sequence        BYTEA,
    genebankid      VARCHAR(128)
);

CREATE TABLE region (
    id              BIGINT NOT NULL PRIMARY KEY,
    seqid           BIGINT references sequence(id),
    startIndex      BIGINT,
    endIndex        BIGINT,
    coding          SMALLINT,
    lock            SMALLINT,
```

```
    characters      VARCHAR(32600)
);

CREATE TABLE edge (
    id              BIGINT NOT NULL PRIMARY KEY,
    treeid          BIGINT references tree(id),
    v1              BIGINT,
    v2              BIGINT,
    weight          FLOAT,
    steering        FLOAT
);

CREATE TABLE treeends (
    seqid           BIGINT references sequence(id),
    edgeid          BIGINT references edge(id),
    treeid          BIGINT references tree(id)
);

CREATE TABLE scoretype (
    type            VARCHAR(256) NOT NULL PRIMARY KEY
);

CREATE TABLE score (
    id              BIGINT NOT NULL PRIMARY KEY,
    treeid          BIGINT references tree(id),
    type            VARCHAR(256) references scoretype(type),
    score           FLOAT8
);

CREATE TABLE spanning (
```

```
    id              BIGINT NOT NULL PRIMARY KEY,
    treeid          BIGINT references tree(id),
    v1              BIGINT,
    v2              BIGINT,
    weight          FLOAT8,
    steering        FLOAT8
);
```

Bibliography

[1] Andrew G. Clark, Kenneth M. Weiss, Deborah A. Nickrson, Scott L. Taylor, Anne Buchanan, Jari Stengard, Veikko Salomaa, Erkki Vartiainen, Markus Perola, Eric Boerwinkle, and Charles F. Sing. Haplotype structure and population genetic inferences from nucleotide-sequence variation in human lipoprotein lipase. *Am. J. Hum. Genet.*, 63:595–612, Aug 1998.

[2] Keith A. Crandall. Multiple interspecies transmissions of human and simian t-cell leukemia/lymphoma virus type i sequences. *Mol Biol Evol.*, 13:115–31, Jan 1996.

[3] Keith A. Crandall and Jennifer Buhay. Genomic databases and the tree of life. *Science*, 306:1144–1145, Nov 2004.

[4] Martin Enserink. Groups race to sequence and identify new york virus. *Science*, 286:206–207, Oct 1999.

[5] Joseph Felsenstein. The number of evolutionary trees. *Systematic Zoology*, 27:27–33, 1978.

[6] Joseph Felsenstein. *Infering Phylogenies*. Sinauer Associates, Sunderland, MA, first edition, 2004.

[7] Joseph Felsenstein. Phylip 3.65, May 2005.

[8] Pablo A. Goloboff, Steve Farris, and Kevin Nixon. Tnt: Tree analysis using new technology, May 2005.

[9] R. S. Lanciotti, J. T. Roehrig, V. Deubel, J. Smith, M. Parker, K. Steele, B. Crise, K. E. Volpe, M. B. Crabtree, J. H. Scherret, R. A. Hall, J. S. MacKenzie, C. B.

Cropp, B. Panigrahy, E. Ostlund, B. Schmitt, M. Malkinson, C. Banet, J. Weissman, N. Komar, H. M. Savage, W. Stone, T. McNamara, and D. J. Gubler. Origin of the west nile virus responsible for an outbreak of encephalitis in the northeastern united states. *Science*, 286:2333–2337, Dec 2000.

[10] R. Meier and F. Ali. The newest kid on the parsimony block: Tnt (tree analysis using new technology). *Systematic Entomology*, 30:179–182, 2005.

[11] Luay Nakhleh, Daniel Miranker, Francois Barbancon, William H. Piel, and Michael Donoghue. Requirements of phylogenetic databases. *Proceedings of the Third IEEE Symposium on BioInformatics and BioEngineering (BIBE'03)*, May 2003.

[12] Kevin C. Nixon. The parsimony ratchet: a new method for rapid parsimony analysis and broad sampling of tree islands in large data sets. In *Program of the 17th meeting of the Willi Hennig Soc*, Sao Paulo, Brazil, 1998. Willi Henning Society.

[13] Mark Pagel. Inferring the historical patterns of biological evolution. *Nature*, 401:877–884, Oct 1999.

[14] Usman W. Roshan, Benard M. E. Moret, Tandy Warnow, and Tiffani L. Williams. Rec-i-dcm3: A fast algorithmic technique for reconstructing large phylogenetic trees. *Computational Systems Bioinformatics Conference, 2004. CSB 2004. Proceedings. 2004 IEEE*, pages 98–109, Aug 2004.

[15] Charles Semple and Mike Steel. *Phylogenetics*. Oxford University Press, Oxford, NY, first edition, 2003.

[16] Daniel Sneddon, Quinn Snell, and Kevin Tewk. Psoda, March 2006.

[17] David L. Swofford. Paup: Phylogenetic analysis using parsimony, May 2005.

[18] Alan R. Templeton, Tylor Maxwell, David Posada, Jari H. Stengard, Eric Boerwinkle, and Charles F. Sing. Tree scanning: a method for using haplotype trees in phenotype/genotype association studies. *Genetics*, 169:441–53, Sep 2004.

[19] Julie D. Thompson, Desmond G. Higgins, and Toby J. GIbson. Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22:4467–4680, 1994.