PHYLOGENETIC DIVERGENCE TIME, ALGORITHMS FOR IMPROVED ACCURACY AND PERFORMANCE

A Dissertation

by

RALPH WALLACE CROSBY

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Tiffani L. Williams
Committee Members,	Nancy M. Amato
	Jianer Chen
	William J. Murphy
Head of Department,	Dilma Da Silva

August 2015

Major Subject: Computer Science and Engineering

Copyright 2015 Ralph Wallace Crosby

ABSTRACT

The inference of species divergence time is a key step in the study of phylogenetics. Methods have been available for the last ten years to perform the inference, but, there are two significant problems with these methods. First, the performance of the methods does not yet scale well to studies with hundreds of taxa and thousands of DNA base pairs. A study of 349 taxa was estimated to require over 9 months of processing time. Second, the accuracy of the inference process is subject to bias and variance in the specification of model parameters that is not completely understood. These parameters include both the topology of the phylogenetic tree and, more importantly for our purposes, the set of fossils used to calibrate the tree.

In this work, we present new algorithms and methods to improve the performance of the divergence time process. We demonstrate a new algorithm for the computation of phylogenetic likelihood and experimentally illustrate a 90% improvement in likelihood computation time on the aforementioned dataset of 349 taxa with over 60,000 DNA base pairs. Additionally we show a new algorithm for the computation of the Bayesian prior on node ages that is experimentally shown to reduce the time for this computation on the 349 taxa dataset by 99%.

Using our high performance methods, we present a novel new method for assessing the level of support for the ages inferred. This method utilizes a statistical jackknifing technique on the set of fossil calibrations producing a support value similar to the bootstrap used in phylogenetic inference.

Finally, we present efficient methods for divergence time inference on sets of trees based on our development of subtree sharing models. We show a 60% improvement in processing times on a dataset of 567 taxa with over 10,000 DNA base pairs.

DEDICATION

To my father, Dr. Edwin S. Crosby, for inspiring me to a lifelong quest for knowledge.

ACKNOWLEDGEMENTS

First and foremost I would like to thank my wife, Kathy, for her unwavering support in this process. I would also like to specially thank my daughter, Devon. My return to school came at a challenging point in her life and my absence during the week when I was in College Station made our relationship more complex.

I would like to give special thanks to my advisor, Dr. Tiffani Williams. A large part of my decision to attend Texas A&M University was based on my interview with Dr. Williams and it is apparent that my decision was the right one. She provided the appropriate guidance and support to make the challenging transition from industry back to school.

I would like to acknowledge Dr. Grant Brammer and Dr. Suzanne Matthews who preceded me in Dr. Williams lab and provided a sounding board for ideas as well as help navigating the intricacies of graduate school.

Finally, I would like to thank Dr. Laurence Moran at the University of Toronto. When I was trying to decide on my direction for graduate school Dr. Moran pointed me in the direction of Bioinformatics as a logical way of combining my experience in Computer Sciences with my interest in Biology.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1 INTRODUCTION

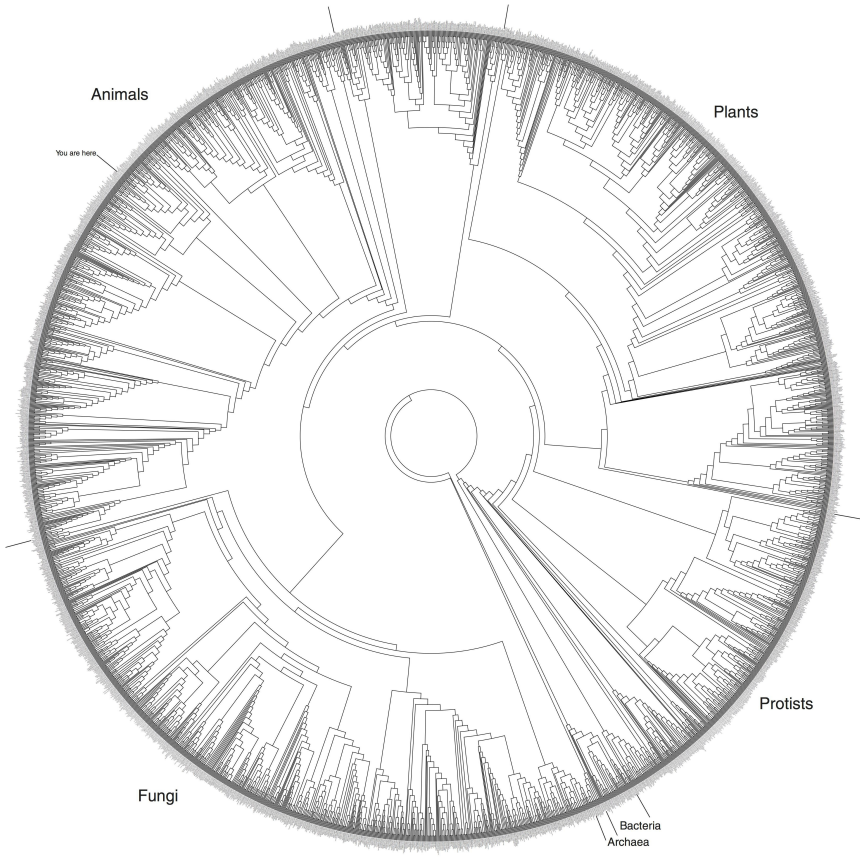


Figure 1.1: *The Tree of Life Represented as a Wheel.[48]*

"The affinities of all the beings of the same class have sometimes been represented by a great tree... "

- Charles Darwin 1859

Darwin envisioned the relationship between all the various species as a great tree with living species as the leaves and branches leading downward to extinct

ancestors. It's a big tree, a recent estimate[63] places the number of living species at 8.8 million ± 1.3 million. While the representation may vary (see Figure 1.1), the tree metaphor has become the core of modern phylogenetics.

As with physical trees, the lengths of the branches are significant to the age of the tree. Our research focus has been on the age of the branches, not the structure of the tree.

1.1 Phylogeny of the Squirrels

While projects such as the Open Tree of Life (http://opentreeoflife.org) seek to develop an all-encompassing tree, the vast majority of phylogenetic analysis focus on a particular branch of the tree. We will start this discussion by looking at the branch of the tree of life associated with the familiy Sciuridae; the Squirrels. This family includes 273 species in 50 genera found on all the continents except Australia and Antarctica. The phylogeny of a representative sample of the squirrels (65 taxa) is shown in Figure 1.2. We developed this phylogeny as part of the Quantitative Phylogenetics class (ENTO-606) and it closely matches the most recent published phylogeny of the squirrels. In this figure, the node labeled (a) represents the most recent common ancestor (MCRA) of the family. Evolution flows to the right and internal nodes such as node (b) indicate extinct species. Living species (*taxa*) are labeled and shown on the right of the figure. In order to properly root the tree, an additional group of closely related species, the Dormice (c), are included in the analysis. The length of the branches may indicate any number of things including an estimate of the amount of evolution that has occurred on the branch, the time that has transpired or other, statistical confidence, measures.

Contemporary, model based, methods of tree inference produce phylogenetic trees with branch lengths, b, that indicate the amount of evolution estimated to have oc-

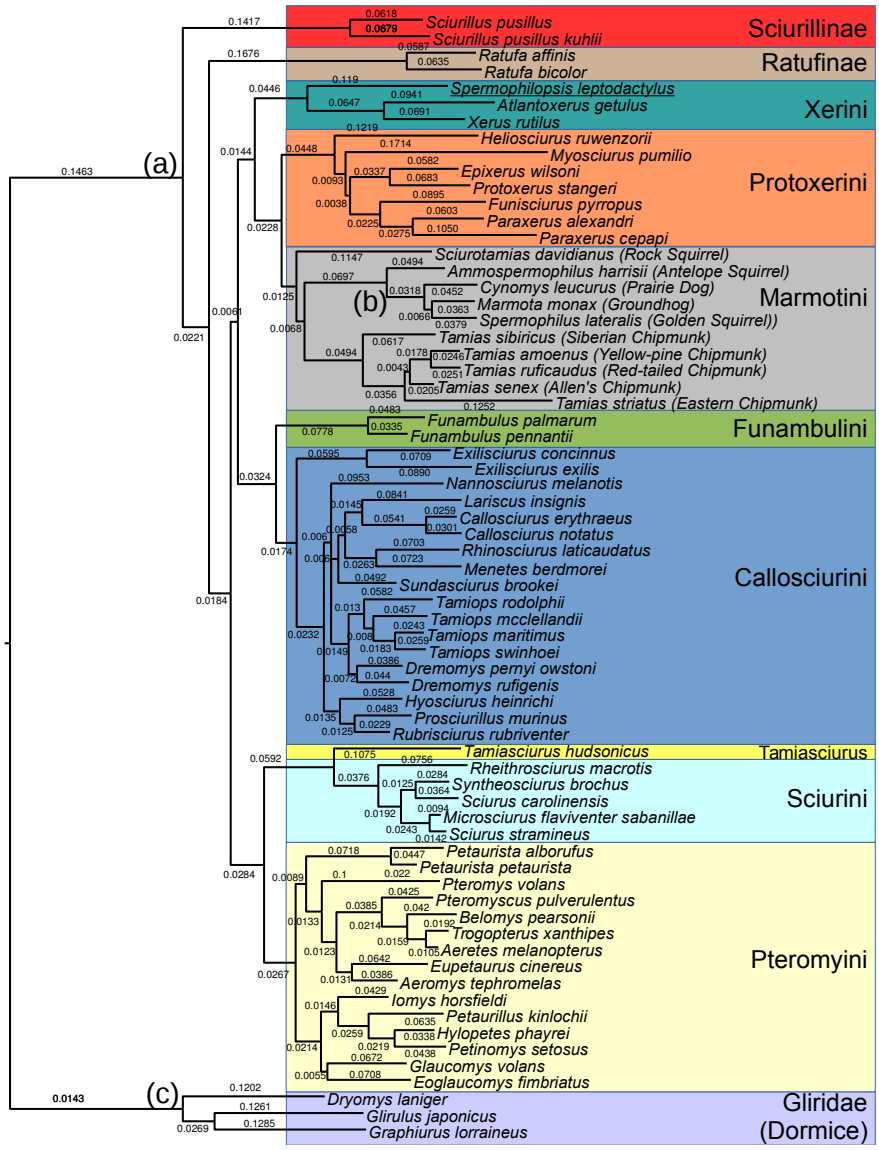


Figure 1.2: *The Phylogeny of the Squirrels.* In this figure the existing species are shown on the right. The node labelled (a) is the most recent common ancestor (MCRA) of the family. Branches flow to the right indicating the path of evolution and internal nodes (e.g. Node b) indicate extinct ancestors. To provide a root for the tree an additional group of species, the Dormice (c) are included as the outgroup.

curred along the branches. In the case of Figure 1.2 the values annotating the branches are estimates of the amount of evolution expressed as the probability of mutation for each of the characters (sites) in the DNA sequences for the species

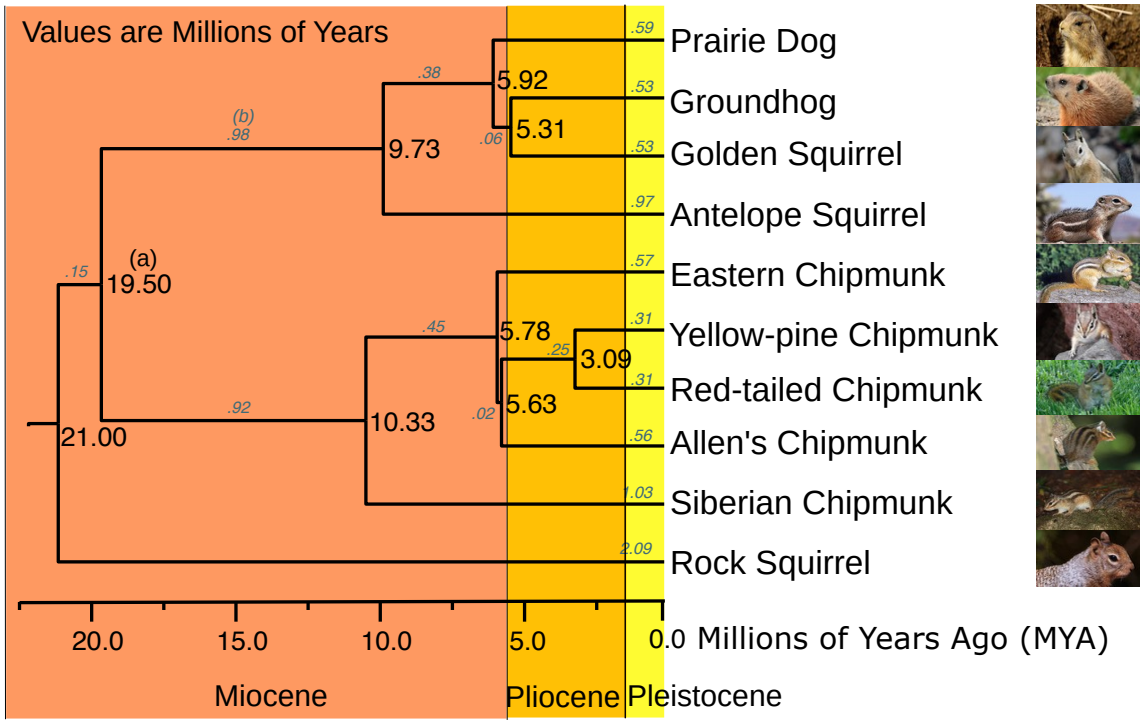


Figure 1.3: *Divergence Time of the Ground Squirrels.* The annotations on each branch refer to the length of the branch calibrated to millions of years. For example, the node marked with a star shows that the prairie dogs diverged from the groundhog/golden squirrel linage 5.92 million years ago. Geological era's are shown to allow for correlation of species divergence to geological events. Images used: Prairie Dog[1], Groundhog[15], Golden Squirrel[29], Antelope Squirrel[52], Eastern Chipmunk[38], Yellow-pine Chipmunk[57], Red-tailed Chipmunk[40], Allen's Chipmunk[71], Siberian Chipmunk[34], Rock Squirrel[80].

(taxa). While branch lengths are interesting, what we're really interested in is something that we can compare with other phylogeny's (and historical events); the dates, d, of the speciation events.

Were the rate, r, of evolution constant, r could simply be factored in and the times determined as $d = r/b$. But, since biology is never simple, the rate of evolution will vary.

Why do we care about the dates? Aren't branch lengths sufficient? We are

natural historians, we want to know when things occurred, not just some abstract information like branch lengths. Adding dates to historical events allows us to temporally connect events and draw further conclusions from the data.

Consider the ground squirrels as represented by the tribe Marmotini in Figure 1.3. This group represents a group of ground dwelling creatures distinct from the generally arboreal squirrels and includes the chipmunks, ground hogs and prairie dogs. The divergence times computed using the tree from Figure 1.2 are shown in the figure in units of Millions of Years. These times closely approximate the most recent published divergence time data for the family[96]. It is apparent from the figure that there was an significant increase in the species of ground squirrels during the late Miocene to early Pliocene eras. It is also known that there was a large increase in savannas and grasslands worldwide during the same period [16]. It is therefore possible to conclude an expansion in habitat fostered an expansion in ground dwelling animals like the Marmotini. The addition of divergence time data allowed for the correlation of these evolutionary events providing additional evidence in support of both events. Dates allow evolutionary events to be compared not only other evolutionary events (like the expansion of grasslands) but with geological and historical events like volcanic eruptions.

This begs the question of why divergence time isn't always done as part of phylogenetic analysis? An informal survey of recent phylogenetic studies by the authors showed that less than half of the studies that included more than 100 taxa also included the determination of divergence time. While the exact reasons in each case are unknown it is our hypothesis that there is a lack of confidence in the results of what is a computationally expensive process.

Further complicating the issue is the workflow. Divergence time is the last step in what has been a long process, the time from the start of sample collection to

the generation of the consensus tree can easily be years. But the divergence time inference itself can be a long process. A set of primate data[73] consisting of just 349 taxa was estimated to require a 9 month run time to compute the divergence time using the MCMCTree program[93]. Adding nearly a year to the time of publication is not reasonable in most cases. And this is for a single run. It is typically desirable to experiment with the parameters to the process and review the results of multiple runs. This is not practical given such long run times. For a researcher to spend days or even weeks doing a thorough divergence time analysis with multiple experiments seems reasonable, spending months to years does not.

Our research has been concerned with the lengths of the branches, not the topology of the tree. We assume a topology (or topologies) have been already been inferred. We introduce new algorithms for the computation of phylogenetic divergence time that significantly reduce the time required for the process. We further provide a new statistical measure to assess the quality of the ages produced.

The time required to compute the divergence time is a function of the number of taxa and the length of the DNA sequences used. The time complexity of the most common algorithms is $\mathcal{O}(lsn^2 \lg n)$ where n is the number of taxa, l is the number of genes and s is the number of sites in the DNA sequence for each gene. While this equation is polynomial, the values for both s and n can be large. In the example of the primate study above, the total number of sites under analysis, s, was approximately 61,000 over 79 genes with $n = 349$ taxa. But newer studies are further increasing both of these numbers. Dr. Murphy's lab is currently working with datasets of 500,000 base pairs and we have been in discussion with researchers interested in analyzing upwards of 30,000 taxa. It is also important to note that this computational complexity is per iteration. In the case of the squirrel analysis, 2 million iterations were used to achieve stationarity. In the case of the primates

dataset, 100,000 iterations were used to achieve stationarity.

1.2 Our Research

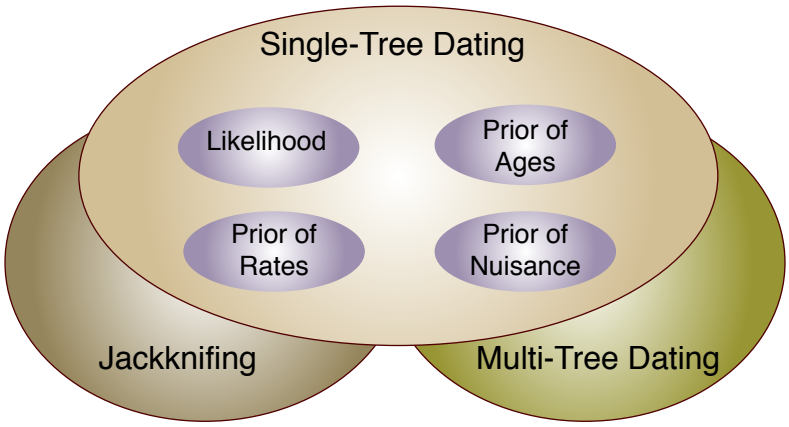


Figure 1.4: *Organization of Our Research.* We focus initially on algorithms to efficiently date single trees. In particular, new algorithms for the computation of the likelihood and the prior of ages as presented. We then use this core to show a new technique, calibration jackknifing, that allows for the generation of a new support measure on the ages of nodes. We end with a new set of algorithms for efficiently dating set of trees.

Our goal has been to improve the methods available for studying and understanding the evolutionary process. While there are robust methods that allow researchers to infer the structure of evolutionary trees, the methods that allow the inference of speciation dates are less mature. In particular, as the size of studies has increased the time required to perform divergence time inference has made the process prohibitive. As previously mentioned, run times of several months have been estimated for studies with only a few hundred species (taxa).

To this end, we have researched new, high performance, algorithms for the computation of phylogenetic divergence time that will allow for the inference of divergence time for much larger sets of taxa than had previously been possible.

As researchers have studied ever larger sets of taxa, the validity of the divergence time process itself has been called into question. We have developed a new measure to assess the phylogenetic support for the ages inferred.

Finally, divergence time inference is usually the last step in a long process. Typically a final topology has been determined prior to dating. We hypothesize that performing the dating process prior to determining the final tree will improve the quality of the mode.

We have developed a new divergence time framework, AncestralAge, and used it to host the new algorithms and methods developed.

- Efficient likelihood calculation.

  The likelihood of a tree and it's associated parameters (e.g. ancestral node ages) refers to the probability that a set of parameters were responsible for the set of taxa observed today.

  The computation of this value is typically the most expensive single component of divergence time inference (or any phylogenetic inference for that matter).

  We have developed a new algorithm, subtree site compression, for the computation of phylogenetic likelihood that reduces the time required for likelihood computation by over 90%.

- Efficient calculation of the prior on node ages.

  The use of a Bayesian framework for divergence time inference allows great flexibility in the specification of fossil calibrations. This information is incorporated into the statistical model in the form of a Bayesian prior on the ages of the ancestral nodes.

  We demonstrate a new algorithm for the computation of the prior of node ages

that reduces what had been a time complexity of $\mathcal{O}(n^2 \ln n)$ to best and worst case complexities of $\mathcal{O}(n)$ and $\mathcal{O}(n^2)$ respectively.

- Support measures for divergence time.

  The use of fossils as "calibrations" for the process is known to be error prone[39][64]. But there are few methods available to assess the "quality" of the fossil calibrations. It has been possible for researchers to build multiple sets of fossil calibrations to study the impact of difference sets of calibrations. But this has been a arduous, error-prone, manual process.

  We have developed a method of creating multiple copies of a tree with varying sets of calibrations using jackknifing[33]. With this method it becomes simple and efficient to request dating using varying sets of fossil calibrations. Using these jackknife replicates, we have developed a new support measure for the ages produced during divergence time inference. We compute this support measure for the aforementioned primates dataset and discuss it's significance.

- Efficient calculation of divergence time on multiple trees.

  We hypothesize that if it were feasible to date sets of trees prior to their consolidation into a single tree, bias resulting from the consensus process itself could be reduced. As with calibration jackknifing, it has always been possible to perform multiple divergence time runs on different trees. But, the performance of the process has scaled directly with the number of trees being dated leading to even more excessive run times.

  To facilitate research into the dating of multiple sets of trees, we have developed new models for the relationship between trees in a set. These models allow the divergence time process to leverage common subtrees improving efficiency and

increasing the number of trees that can be dated in parallel.

The next section in this document discusses aspects of phylogenetics that are important to divergence time inference. Following that section is a discussion of the existing methods and programs available for divergence time inference.

The remainder of the document follows the overall structure shown in Figure 1.4. Section 4 is the description and analysis of out new likelihood algorithms. Section 5 is the description and analysis of our new algorithm for the computation of the prior on ages. Section 6 introduces our calibration jackknifing and new support measure for ages. Section 7 provides detail on our algorithm and models for dating multiple trees efficiently.

We then show an experimental analysis of our algorithms performance in Section 8. We conclude in Section 9 with a summary of the work and a discussion of additional research that is enabled by the AncestralAge platform.

1.3 Terminology

A number of terms are used in the divergence time process. For the sake of clarity we define a number of the ones that will appear in this document.

Phylogenetic trees are undirected acycle graphs with labels on vertices of degree one only. Individual trees might or might not be rooted and, in most but not all cases, the final, published, tree will be rooted. For purposes of divergence time, inferred trees are considered to be rooted. The vertices of trees will be categorized as *leaves*, *inner nodes* and the *root*. A *leaf* in a tree refers to a labeled node of degree one (one incident edge). *Leaves* correspond to the taxa in a phylognentic tree. Vertices of degree three (three incident edges) will be referred to as *inner nodes* and, for our purposes, must be unlabeled. *Inner nodes* correspond to the extinct ancestors of existing taxa. The one vertex of degree two (two incident edges) will be referred to

as the *root* and also must be unlabeled. Phylogenetically, the *root* represents the most recent common ancestor (*MRCA*) of the taxa in the tree.

From a graph perspective, phylogenetic trees are undirected but common usage (that we will follow) considers a flow of evolution from the root out to the leaves corresponding to the passage of time.

There are two related trees that factor into divergence time; first a tree representing the evolution of the taxa being studied. This will be referred to as the *species tree*. For a given study, multiple partitions of the DNA data may be used. These partitions often are associated with particular genes. While the association of partitions to genes is not a requirement, for consistency we will use the term *gene tree* to refer to the tree associated with a single partition of the DNA data.

In statistics (and evolutionary biology) the term log refers to the natural logarithm (base e). In computer science the term log refers to the logarithm base 2. For consistency in this document we will use ln or log_e to refer to the natural logarithm and lg or log_2 to refer to the logarithm base 2.

In Markov Chain, Monte Carlo (MCMC) methods the fundamental unit is the *MCMC step*. In a step new values are in turn proposed for each of the model parameters and their impact evaluated. Specification of processing for MCMC methods is typically in terms of the number of *MCMC steps* to be taken.

The terms ages, dates and time are frequently confounded in the literature. We will use the term *age* to refer to a length of time from the present to the date when an event occurred. We will use the term *time* to refer to a duration. For example an edge in a tree has a duration and therefore a *time* associated with it. The term *branch length* refers to a distance and as such is the result of a rate r being applied for some period of time t. In phylogenetics, rates are commonly specified as the percentage of DNA bases that change in a unit of time. When multiplied by a time

value this generates a branch length that correlates to the number of DNA bases that have changed along the edge.

2 THE BUILDING BLOCKS OF DIVERGENCE TIME

The divergence time process uses methods and data in common with other portions of the phylogenetic workflow. All modern methods are based on the comparison of molecular information usually consisting of DNA sequences although other data such as amino acid or protein sequences may also be used. The DNA molecule is a long sequence of the four bases (adenine (A), cytosine (C), guanine (G) and thymine(T)). Each base bonds to another base on the opposing DNA strand. These bonded pairs are referred to as *base pairs* (BP). Each position on one of the two strands is referred to as a site.

2.1 Multiple Sequence Alignment

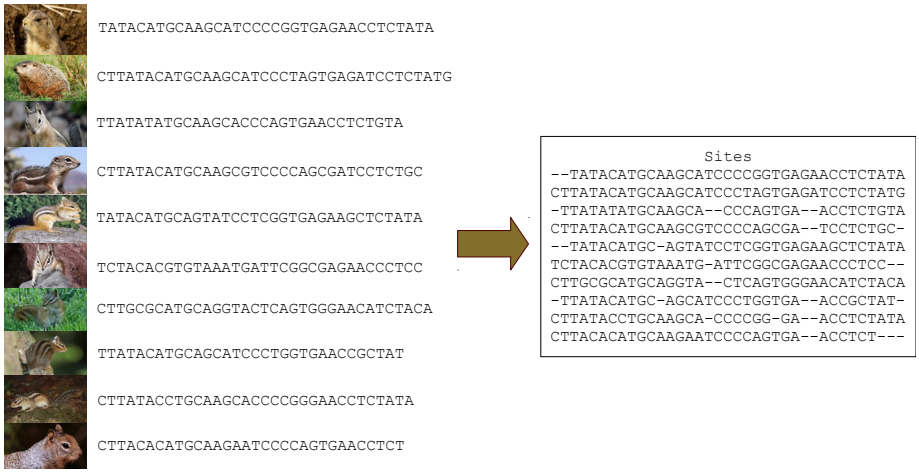


Figure 2.1: *Multiple Sequence Alignment.* Images used: Prairie Dog[1], Groundhog[15], Golden Squirrel[29], Antelope Squirrel[52], Eastern Chipmunk[38], Yellow-pine Chipmunk[57], Red-tailed Chipmunk[40], Allen's Chipmunk[71], Siberian Chipmunk[34], Rock Squirrel[80].

A fundamental input to the divergence time process is the set of DNA for the taxa of interest. It is comparisons between these sets of DNA that provide estimates of the relationships between the taxa and the amount of evolution that has occurred.

To compare the sequences across the set of taxa, it is necessary to align the sequences. As sites will not only have changed but have been inserted and deleted the alignment may contain holes or gaps as shown in Figure 2.1. The placement of these gaps is critical to the quality of the alignment.

This is a complex computational (and manual) process. While there are automated tools available for this step, typically there is some level of manual curation subsequent to the automated alignment. A considerable amount of research has been, and continues to be, focused on the alignment problem[69].

The result of the alignment process is referred to as the multiple sequence alignment (MSA). The MSA is a matrix where rows represent the taxa and the columns represent the sites occupied by the bases of the DNA (or gaps).

2.2 Phylogenetic Inference

Divergence time depends on the availability of a tree (or set of trees in our case) that best model the evolutionary relationships between the taxa. Additionally, the methods used for phylogenetic inference, particularly Bayesian methods, provide the core algorithms for divergence time.

A variety of methods are available for inferring a tree from the MSA. Early methods looked only at the sequence alignment and attempted to minimize the number of mutations required to convert one sequence into another. This method, maximum parsimony [79], assumes that any DNA base may change into any other DNA base at any time with an equal probability.

Contemporary methods take a model based approach with the addition of a

transition matrix that allows the researcher to vary the probabilities associated with mutation events. This transition probability matrix (TPM) defines an evolutionary model wherein the rows and columns of the model are the four DNA bases (A,C,G,T) and the cells of the matrix define the probability of a transition from the one base to another. The TPM, along with the MSA, is then used to "score" a tree. This score, referred to as the likelihood, is either optimized directly (maximum likelihood methods) or used as part of Bayesian inference.

2.2.1 Tree Space

To find the "best" tree, search methods are required since exhaustive search for all but the smallest trees ($\leq$ about 10 taxa) is not feasible as the number of possible binary trees associated with a set of taxa is $(2n-e)!!$. For even the modest 65 species of Squirrels included in Figure 1.2, the number of possible trees is an incomprehensible 1.6×10^{107}. When branch lengths are included the number of potential solutions becomes infinite. The problem of finding the optimal tree (including branch lengths) becomes one of searching an infinite "tree space" for the "best" solutions.

In phylogenetics, maximization of the likelihood score is commonly used as it provides good results with reasonable performance.[75]. Another approach is through the use of Bayesian inference. In the Bayesian model, the likelihood is just one component of the score. The Bayesian model allows the inclusion of additional, prior, knowledge into the calculations. This is particularly important for divergence time inference. In divergence time inference, additional information in the form of estimated dates for fossils, is used to "calibrate" the tree. Logically, these calibrations are in fact "prior information" and fit naturally into the Bayesian framework. There has been research into the use of maximum likelihood methods for divergence time inference [95]. But, to date, no methods have been developed that allow for the

inclusion of multiple, varied, calibration points in a maximum likelihood statistical framework.

We will therefore focus on Bayesian methods in the remainder of this document.

2.2.2 Likelihood

"Likelihood is the hypothetical probability that an event that has already occurred would yield a specific outcome. The concept differs from that of a probability in that a probability refers to the occurrence of future events, while a likelihood refers to past events with known outcomes."

- Wolfram Mathworld[88]

The term probability refers to chance that a set of parameters today will generate some specific event in the future. Likelihood defines the probability that a set of parameters (θ_L in Figure 2.2) in the past were responsible for the set of data observed in the present. In phylogenetic terms the parameters θ_L are the tree, the rates of evolution for each the branches and the dates for the inner nodes and root. The observed data are are the living taxa and their DNA sequences.

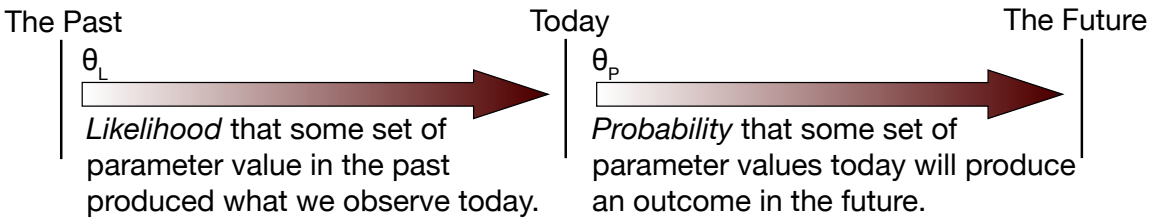


Figure 2.2: *Likelihood and Probability.* Likelihood refers to the probability that a set of parameters produced the data observed today. Probability refers a set of data and parameters today producing some outcome in the future.

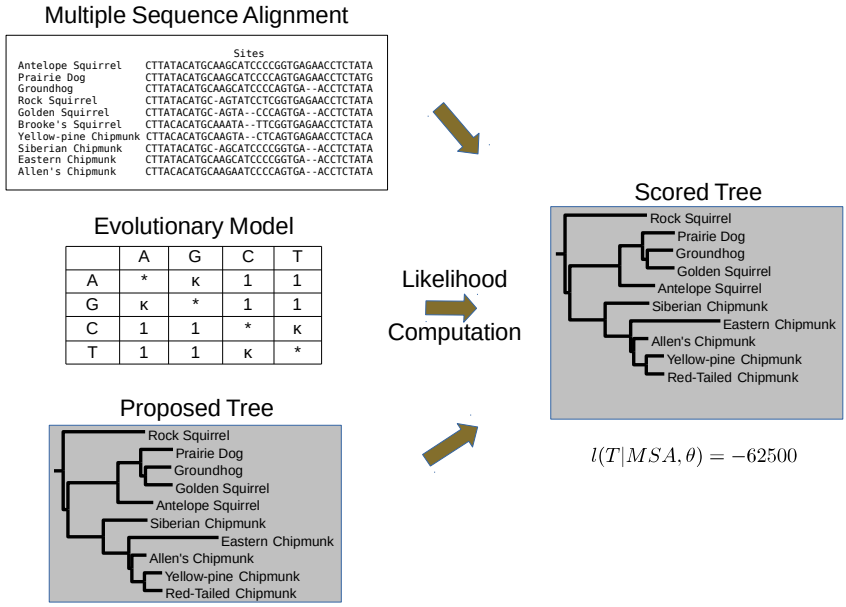


Figure 2.3: *Likelihood of a Phylogenetic Tree.* To determine the likelihood of a proposed tree, the evolutionary model is applied to the MSA. The log of the probabilities of each site in the MSA are summed to generate the final likelihood score.

A tree has a likelihood probability based on the evolutionary model as expressed in the TPMs and the MSA. Figure 2.3 illustrates the components of the likelihood model. The inference process needs to compute this value each time there is a change in the model parameters. In the case of phylogenetic inference, this would include topological and branch length changes. For divergence time inference, the topology is fixed but the branch lengths are computed as the product of rates over time and both the times and rates are model parameters.

The phylogenetic likelihood is computed as the sum of the log of the likelihoods of each of the n columns ($\mathbf{x}_h$) in the alignment given the evolutionary model parameters (θ).

$$l = \log(L) = \sum_{h=1}^{n} \log(f(\mathbf{x_h}|\theta)) \tag{2.1}$$

The probability function for a column, $f(\mathbf{x_h}|\theta)$, is the sum of the probabilities of of all possible values (A,C,G,T) for all of the taxa based on the proposed tree. Calculation of the likelihood is the typically the most computationally expensive component of any of the model based phylogenetic methods.

2.2.3 Bayesian Inference

Since the Bayesian model is currently the most prevalent statistical model for divergence time, we will focus on it's structure and application to divergence time.

In a Bayesian inference, the goal is to determine what is known as the posterior probability; the set of parameters that have the highest probability given the data, $P(\theta|D)$. Classical "frequentest" statistics optimize the probability of the data given the parameters, $P(D|\theta)$. This probability is the likelihood that the parameters specified gave rise to the data observed today. Bayes formula provides the conversion from the likelihood to to the posterior probability:

$$P(\theta|D)P(D) = P(D|\theta)P(\theta) \tag{2.2}$$

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)} \tag{2.3}$$

In Bayesian terms, the probability of the parameters, $P(\theta)$, is referred to as the prior and allows the model to include understanding about the values of the parameters independent of the data. The probability of the data in the denominator of the right hand side acts to normalize the values allowing the probabilities to integrate to 1.

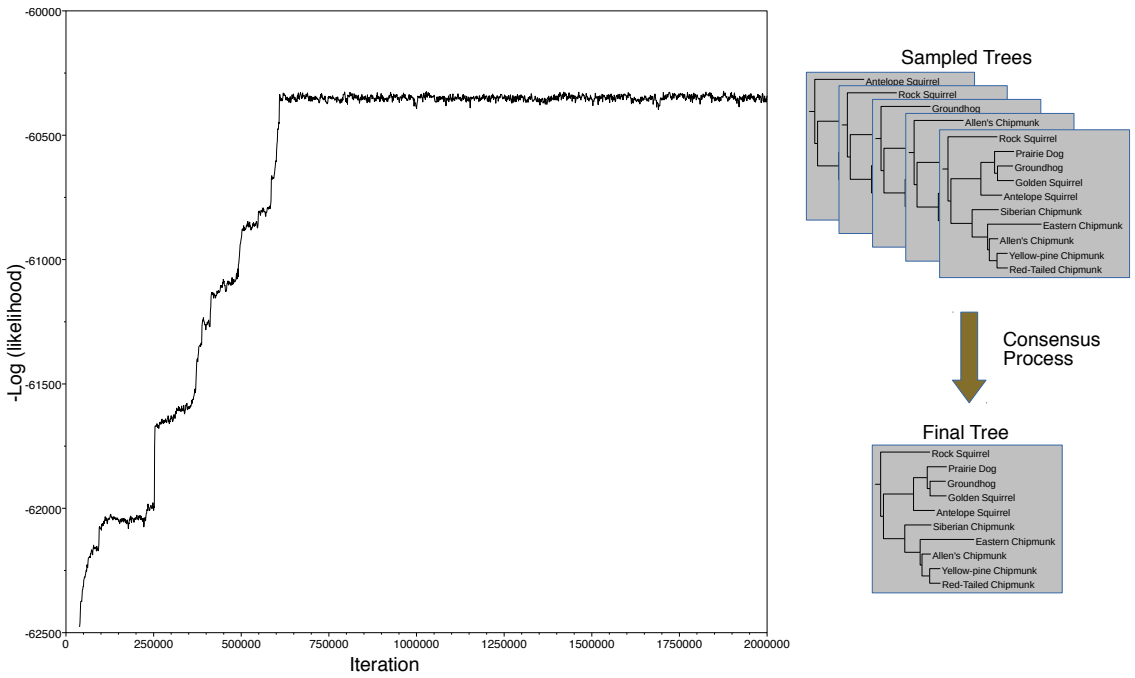


Figure 2.4: *Bayesian Inference of a Phylogenetic Tree.* The chart on the left shows the likelihood values for my squirrel data returned from the iterative process. The first portion of the plot up to step 600,000 reflects the algorithm seeking the true posterior distribution. This section is referred to as the "burnin" and is discarded. Trees are sampled on a regular basis (e.g. every thousandth tree). These sampled trees are then put through a consensus generation process to produce the final tree.

Typically, Bayesian methods depend on iterative techniques to determine the distribution of the posterior. The desired goal is to run the iterative process until it has converged on the "true" distribution of the posterior. In practice, achievement of a stationary process is used as a proxy for convergence. In Figure 2.4 the log of the likelihood is plotted over time. At around iteration (or step) 600,000 the values stabilize and the process is referred to as stationary. Thousands to millions of steps are generally required to reach stationarity. Trees are sampled at regular points during the iterative processing generating thousands to tens of thousands of samples. Since the samples provide a discrete approximation of the continuous

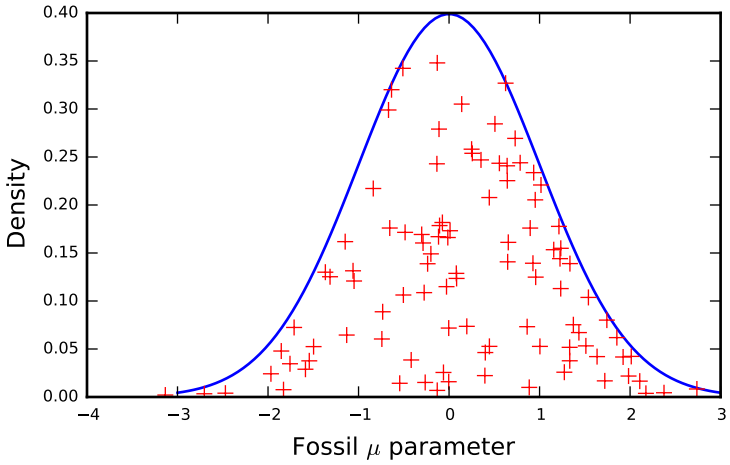


Figure 2.5: *Sampling from the Posterior Distribution.* To determine the probability density function (PDF) for some parameter μ the MCMC process randomly samples from a distribution proportional to the desired distribution "filling in" the area under the curve.

posterior distribution there is unlikely to be a single tree that represents the highest posterior probability (the maximum a posteriori or MAP).

Markov chain Monte Carlo (MCMC) methods provide a numerical method for evaluating the probability distribution function described by the posterior through sampling. The Markov chain portion of the name indicates that the steps in the process refer only to the current state, prior state is not considered in the computation of the next step. The Monte Carlo portion of the name refers to the use of random values in the computation of the steps of the process. The most prevalent sampling algorithm is that of Metropolis-Hastings[43]. The Metropolis-Hastings algorithm allows samples from a distribution (the posterior distribution in this case) to be drawn as long as the value of a function $g(x)$ can be computed that is proportional to the desired distribution. This eliminates the need to compute the normalization factor (the denominator in equation (3.1)). For each iteration, new parameter values are chosen, typically by drawing from a normal distribution given the current parameter

values. The new value of the proportional function, $g'(x)$ is compared to the prior iterations value $g(x)$. if $g'(x) > g(x)$ the new parameter values are accepted. If $g'(x) \leq g(x)$ the new parameter values are accepted with a small probability α specified by the user. Otherwise the new parameter values are rejected. By this means the area under the probability curve is filled in as shown in Figure 2.5.

In complex models with large numbers of free parameters, it is impractical to choose values for all the tree parameters at once and expect more than a very small percentage of the samples to be accepted. The use of Gibbs sampling, as first described by Stuart and Donald Geman in 1984[37], provides a solution to this problem at increased computational cost. Using Gibbs sampling, each MCMC iteration is further broken into propositions for each parameter. For each parameter in turn, a new value is drawn for the parameter and the posterior probability computed. Acceptance and rejection are the same as for the Metropolis-Hastings algorithm. There are two consequences of Gibbs sampling to note:

1. The likelihood is now recomputed after each parameter is drawn multiplying the number of likelihood computations by the number of parameters. This can have a significant performance impact.

2. There is a significant amount of correlation between likelihood values within the same iteration since not all parameters are sampled at the same time. The implication of this is that it may take some time for the Markov chain to reach the desired distribution, in essence overcoming the correlation induced by the Gibbs sampler. For this reason, it is usually necessary to discard the initial portion of the chain (the "burnin"). For example, in Figure 2.4 it can be seen that the likelihood starts out low and over the course of the first approximately 600,000 iterations continued to improve. It isn't until after 600,000 iterations

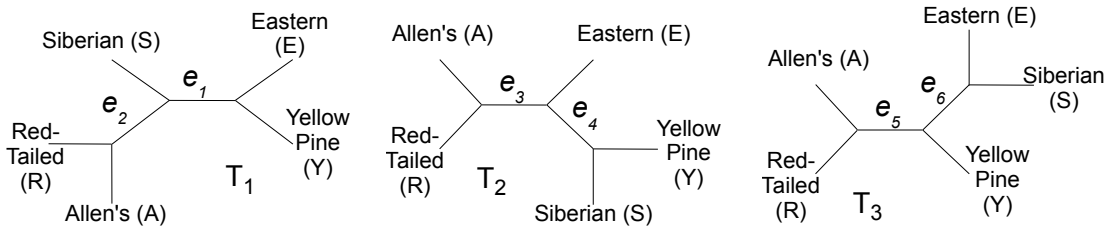


Figure 2.6: *Three Evolutionary Trees with a Common Edge.* While the overall topology of each of the trees is different, they all share a common edge, $RA|SEY$. This edge would appear in either a strict or majority consensus tree.

that the Markov chain achieves "stationarity" indicating that the process has stabilized on the posterior distribution.

2.3 Consensus Tree

Since evolution occurred one way, not many, the goal of phylogenetic inference is the production of a single tree that represents the best hypothesis of how evolution actually occurred. In order to determine the single tree that most closely approximates the MAP point, a method of consensus generation is used wherein the sampled trees are analyzed and consolidated into a single tree. A variety of methods are available for this consensus generation [11] but, the goal of all the methods is to produce the tree most informative of the set of trees being combined.

The most frequently used methods depend on the building a tree based on sets of edges in the sampled trees. Edges that appear in either all the trees (the strict consensus) or $> 50\%$ of the trees (the majority consensus) may be selected to appear in the final tree. In Figure 2.6 the edge separating the Red Tailed Squirrel and Allen's Squirrel from the other three taxa appears in all three trees. This edge would appear in either the strict or majority trees.

Branch length in the consensus will typically represent the mean or median values

for the branch. The variance of the branch lengths is not usually reported.

Consensus methods, by their nature, induce bias. One dataset we studied of 20,000 trees over 150 taxa was developed by the combination of two Bayesian runs. The consensus tree incorporated information from both runs but cluster analysis of the data[7] indicated that there were two distinct clusters of data. This clustering was lost in the consensus tree.

Divergence time inference is typically performed on the single, final, consensus tree developed. We hypothesize that this tree is therefore subject to significant bias as a consequence of the consensus process. We further hypothesize that were larger sets of trees selected prior to consensus dated independently, the impact of this bias on the dates would be reduced. A consensus tree would still be produced subsequent to dating but, the bias induced in the consensus process would not impact the dating.

3 THE ORIGIN OF ANCESTRAL AGE

The goal of this research was the development of new computational methods for divergence time inference. Out intent was not to investigate new statistical methods or models but, to provide new algorithms for computation using existing statistical models. We first discuss the origins of the statistical model then we discuss the model itself and finally the MCMCTree implementation of the model that first sparked our interest in divergence time.

Figure 3.1 illustrates the progression of MCMC research leading to modern divergence time methods. MCMC methods were first developed during the second world war as part of the atom bomb research. During the 1950's and 1960's research was focused on the theoretical aspects of MCMC methods as the computational overhead made the methods difficult to implement on any other than the largest computers then available.

By 1994 desktop computers had achieved sufficient performance to allow for practical experimentation into MCMC methods and their application to phylogenetics. In 1994, Yang produced the PAML package[91] which provided an MCMC method for inference of the phylogenetic tree. The first divergence time algorithm, Multidivtime, was developed by Thorne, *et al.* in 1998[86]. In 2001 Huelsenbeck and Ronquist developed the first version of the Mr. Bayes program for phylogenetic inference[49] which remains the most prevalent MCMC package for phylogenetic tree inference. In 2006, Yang and Rannala extended the PAML package with the development of the MCMCTree program[94] for divergence time inference and the following year Drummond *et al.* produced the *Beast* program[27] which also includes divergence time inference.

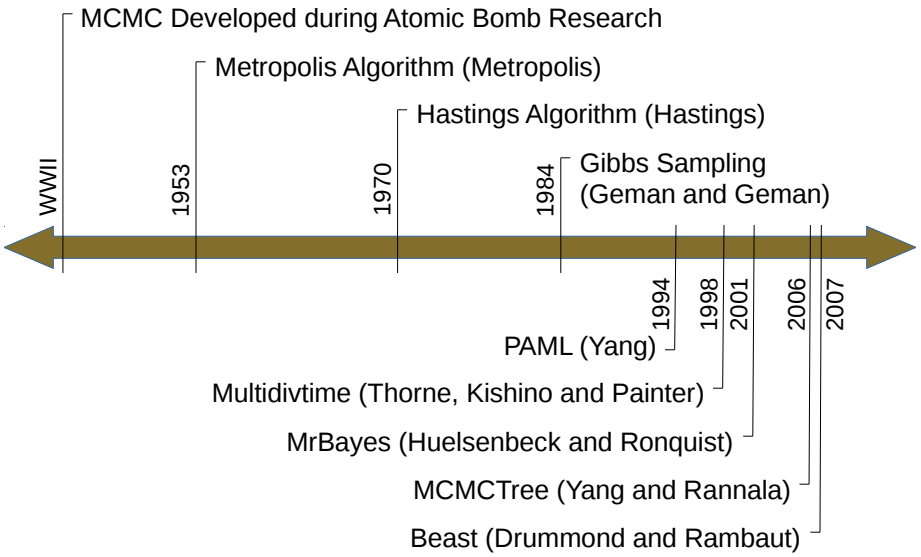


Figure 3.1: *Origins of Divergence Time Algorithms.* The early work on MCMC methods was largely theoretical due to limitations in computer hardware. With the availability of high speed desktop computers in the early 1990's, MCMC methods were applied first to phylogenetic inference and then to divergence time.

3.1 Divergence Time Inference

"All models are wrong, but some are useful"

- George Box 1987[6]

Equation (3.1) provides the general Bayesian formulation for the divergence time model as implemented in the MCMCTree program.

$$f\left(\mathbf{t},\mathbf{r},\theta|D\right)=\frac{f\left(D|\mathbf{t},\mathbf{r},\theta\right)f\left(\mathbf{r}|\mathbf{t},\theta\right)f\left(\mathbf{t}|\theta\right)f\left(\theta\right)}{f(D)} \tag{3.1}$$

The posterior probability of a set of times, rates ($\mathbf{t}$ and $\mathbf{r}$ respectively) and other, nuisance, parameters, θ, given the data is equal to the likelihood of the data D given the times, rates and parameters $f\left(D|\mathbf{r},\theta\right)$ multiplied by the priors for the rates, times and parameters. This is normalized by the marginal probability of the data,

$f(D)$.

The problem with the computation of this value becomes apparent when the probabilities are expanded, for example the probability of the data expands into a high dimension integral without a closed form solution

$$f(D) = \int_{\mathbf{t}} \int_{\mathbf{r}} \int_{\theta} f\left(D|\mathbf{t}, \mathbf{r}, \theta\right) f\left(\mathbf{t}, \mathbf{r}, \theta\right) d\mathbf{r}\ d\theta \tag{3.2}$$

Fossil calibrations are included through the prior on the times, $f\left(\mathbf{t}|\theta\right)$. If $\mathbf{t}_C$ is the set of calibrated nodes and $\mathbf{t}_{\bar{C}} = \mathbf{t} - \mathbf{t}_C$ is the set of nodes without calibrations, then the prior can be constructed as:

$$f(\mathbf{t}|\theta) = f(\mathbf{t}_C, \mathbf{t}_{\bar{C}}|\theta) \tag{3.3}$$

$$= f'(\mathbf{t}_{\bar{C}}|\mathbf{t}_C, \theta) f(\mathbf{t}_C|\theta) \tag{3.4}$$

where $f'()$ defines the conditional distribution of the non-calibration nodes based on the calibration nodes and parameters.

The probability of the data, $f(D)$ is computed by integrating the time, rates and parameters out of the likelihood.

$$f(D) = \int \int \int f\left(D|\mathbf{t}, \mathbf{r}, \theta\right)\ d\mathbf{r}\ d\mathbf{t}\ d\theta \tag{3.5}$$

This formulation is, in practice, intractable. To avoid this issue the Metropolis-Hastings algorithm[43] is employed. This algorithm makes use of the ratio between two values of the posterior; the original or old value $f(\mathbf{t}, \mathbf{r}, \theta|D)$ and a new value $f(\mathbf{t}', \mathbf{r}', \theta'|D)$ obtained by sampling new values for the parameters. This allows the denominator to drop out of what is known as the acceptance ratio (Equation (3.7)).

If this ratio is greater than zero the proposed new parameters are accepted. If the ratio is less than or equal to zero the proposal is still accepted with some probability p drawn from a uniform distribution $0 < p < 1$.

$$\frac{f(\mathbf{t}', \mathbf{r}', \theta'|D)}{f(\mathbf{t}, \mathbf{r}, \theta|D)} = \frac{f(D|\mathbf{t}', \mathbf{r}', \theta') f(\mathbf{r}'|\mathbf{t}', \theta') f(\mathbf{t}'|\theta') f(\theta') \frac{1}{f(D)}}{f(D|\mathbf{t}, \mathbf{r}, \theta) f(\mathbf{r}|\mathbf{t}, \theta) f(\mathbf{t}|\theta) f(\theta) \frac{1}{f(D)}} \tag{3.6}$$

$$= \frac{f(D|\mathbf{t}', \mathbf{r}', \theta') f(\mathbf{r}'|\mathbf{t}', \theta') f(\mathbf{t}'|\theta') f(\theta')}{f(D|\mathbf{t}, \mathbf{r}, \theta) f(\mathbf{r}|\mathbf{t}, \theta) f(\mathbf{t}|\theta) f(\theta)} \tag{3.7}$$

In practice, proposing new values for all parameters at once generated few accepted proposals. Geman and Geman demonstrated[37] that sampling individual parameters and computing the acceptance ratios after the change of individual parameters was statistically equivalent to Metropolis-Hastings and generated considerably more accepted proposals. This technique, known as Gibbs sampling, is used in all modern Bayesian phylogenetic programs including MCMCTree and AncestralAge.

Existing divergence time algorithms implement the statistical model using a complex set of inputs:

1. An aligned set of DNA sequences, the multiple sequence alignment (MSA) for the set of taxa.
2. A evolutionary model defining the transition probabilities associated with a DNA base mutating from one value to another.
3. A phylogenetic tree with estimates of the amount of evolution that has occurred along each branch.
4. A set of fossil calibration dates along with associated bounds and statistical distributions..

5. Statistical parameters defining the distributions associated with the rates of evolution along the branches.

There has been a significant amount of research into the sensitivity of the process to the specification of the MSA and evolutionary model[22][24][97]. Even so, there remain significant concerns as to how well supported the results of the process are[89].

The issues with fossil specification have also been well studied[50], but, with the high computational cost associated with divergence time, it was not feasible to run multiple experiments to determine the sensitivity of the model to different sets of calibrations.

3.1.1 Model Parameters

The statistical model exposes a large set of parameters that each must be estimated. For the model implemented in MCMCTree and AncestralAge this set includes the following:

- The ages of each ancestral node in the species tree. These are the principal parameters being inferred. In the case of the primates dataset, there are a total of 348 age parameters (including the root) for the 349 taxa in the study ($n-1$ inner and root nodes).

- The average rate of evolution along each branch of each gene tree. In reality the rate of evolution can vary throughout the existence of a species but it is typically modeled as an average across the branch. It is also well understood that, even within the same species, the rate can vary between genes hence the requirement for unique rates for each gene tree. In the primates dataset there are 21,584 rate parameters across the 79 genes in the study.

- There are a number of other parameters that, while important to the model, are not of any particular interest in the result. These are referred to as nuisance parameters and include the evolutionary model parameters (e.g. κ in the Kimura '80 model) and parameters that are used in the computation of the priors of the various parameters (e.g. the gamma distribution α and β values used to model the rates at the root of the gene trees). These parameters are associated with individual gene trees. For the primates dataset there are an additional 1027 nuisance parameters across the 79 genes.

It is important to note that for each parameter change, the acceptance ratio must be computed.

3.1.2 Components of the Model

To compute the acceptance ratio the four terms in the numerator of Equation (3.1) must each be computed. AncestralAge supports a set of algorithms for the computation of the terms using the mathematical models developed for the MCMCTree program[92].

3.1.2.0.1 Likelihood The likelihood can be considered the probability that the parameters as specified yielded the result seen (e.g. the sequence alignment of the existing taxa). For divergence time this calculation is across the entire tree and needs to consider the ages of each ancestral node, the rates of evolution across each branch and the parameters of each evolutionary model. In other words, all of the parameters of the model.

3.1.2.0.2 Prior on Ages It is through the use of the prior on ages that the fossil calibrations are included in the model. This is a natural and easily justified use of the Bayesian prior as the fossil record does comprise prior knowledge about the ages of the ancestors of the existing species.

The challenge in the design of this prior is to allow specification of the fossil calibrations to include uncertainty in the calibration values. In MCMCTree, Beast and now AncestralAge this is accomplished by the specification of parameters for each calibration[94]. These parameters are fixed, hyperparameters, in the model.

3.1.2.0.3 Prior on Evolutionary Rates In the simplest case, the rate of evolution will be constant, the "Molecular Clock" hypothesis. In this case, the prior for evolutionary rates would be a constant across all branches of a gene tree. The other extreme is to allow the rates on branches to vary independently. In this case, the prior is modeled as a gamma distributed random variable with specified shape (α) and scale (β).

But, it is reasonable to consider that there will be some relationship between the rate in a descendant and it's ancestors rate. This, correlated, model is handled in MCMCTree and AncestralAge using a prior where the rate for a branch is a random variable drawn from a gamma distribution with a mean specified as the rate for the branch's immediate ancestor[68].

All three of these models are supported in MCMCTree as well as AncestralAge.

3.1.2.0.4 Prior on Nuisance Parameters The priors on the other, nuisance, parameters depend on the type of parameter:

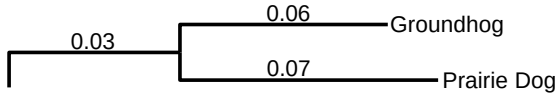
- The parameters of the evolutionary models (e.g. the α and β values in the Kimura '80 model) are each independently modeled as random variables drawn from gamma distributions with hyperparameters specified by the user.
- The character frequency parameters used in those evolutionary models that require them (e.g. the HKY 1985 model) are modeled using a Dirichet process prior.

	Sites
Prairie Dog	ATTACACATGCAAGTATCCCCTTCCCAGTGAGAATGCCC...
Groundhog	CTTACACATGCAAGCATCCCCGCCCCAGT--GAATGCCC...
Golden Squirrel	CTTATACATGC-AGCATCCCCGCCCCGGTGAGAATGCCC...
...	

(a) *Multiple Sequence Alignment.* In most likelihood algorithms any columns with gaps are ignored in the calculation.

	A	G	C	T
A	*	κ	1	1
G	κ	*	1	1
C	1	1	*	κ
T	1	1	κ	*

(b) *Evolutionary Model (K80).* The value of the parameters (κ in this case) are estimated from the data.



(c) *Tree with Branch Lengths.* The branch lengths represent the amount of evolution along the branch.

Figure 3.2: *Components of the Likelihood Model.*

- The evolutionary rates for each gene tree in the molecular clock model are modeled using a gamma prior with hyperparameters specified by the user.
- For the independent rate model, both the mean rate and it's variance are specified using gamma distributions with hyperparameters specified by the user.
- For the correlated rate model, the prior on the mean and variance of the rate at the root of the gene tree are specified using gamma distributions with hyperparameters specified by the user.

3.1.3 Likelihood

The MSA, evolutionary model and tree define the likelihood model in a very similar manner to that used for phylogenetic inference.

The MSA, an example of which is shown in Figure 3.2(a), provides a best estimate

of the relationship between the DNA sequences of the taxa. Since insertions and deletions of bases occur randomly (although not all survive) during the evolution of the sequences, the characters of one taxa's DNA sequence will not necessarily represent the same portion of a DNA sequence of another taxa. For this reason, the MSA is only an estimate of the relationship between the sequences since the exact set of insertions and deletions cannot be known. Gaps in the sequences introduced by insertions and deletions and typically treated as missing data although the statistical validity of this approach is the subject of ongoing research[87].

The evolutionary model defines the probabilities of one DNA character changing into another. Models vary from simple in the case of the Jukes-Cantor mode wherein all the probabilities are equal to the GTR (Generalized Time Reversible) model in which each individual probability is a separate parameter. As an example the K80 model (Kimura 1980) is shown in Figure 3.2(b). The four DNA bases (A,G,C,T) are also representative of two types of nitrogeneous bases, purines (A,G) and pyrimidines (C,T). In it's original formulation, this model used two parameters α and β. The α parameter represented the probability that a base will change into the same type of nitrogeneous base (e.g. purine to purine) and is referred to as the transition probability. The β parameter represented the probability that a base will change into the other type of nitrogeneous base (e.g. purine to pyrimidine) and is referred to as the transversion probability. The model is generally simplified as shown in Figure 3.2(b) with the transversion rate set to 1 and the transition/transversion ratio α/β referred to as κ.

The phylogenetic tree has already been discussed, but, it is important to note that branch lengths are a critical part of the model as shown in Figure 3.2(c). For divergence time inference, the factors leading to the branch lengths (age, rate) will be inferred but the branch lengths provide the best possible estimate for the ini-

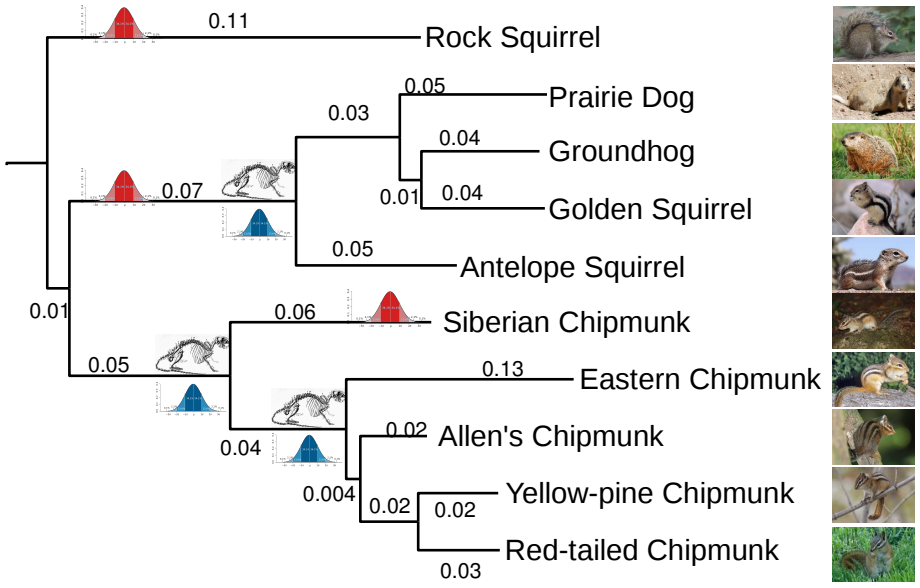


Figure 3.3: *Tree Calibration and Rate Estimation.* Fossils are used to "pin" nodes of the tree to branch points in the tree. Each fossil has a level of variability associated with it as shown by the distribution under the fossil. The distributions for the rates along the branches are also specified.

tial values. In MCMCTree branch lengths are not allowed in the input tree. In AncestralAge the branch lengths are used in the computation of the initial values.

3.1.4 Tree Calibration

In order to determine the date of the speciation events, it is necessary to provide calibration for the tree. This is provided through the use of fossils. For example, in Figure 3.3 three fossils are shown at different points in the tree for the Marmots. There are a number of issues with the use of fossils as calibration points:

1. The placement of the fossils in the tree is usually based on morphological characteristics as DNA is generally not available. It is well known that similar morphological characteristics may evolve independently. In fact, prior to DNA data being available, it was accepted that all the flying squirrels, tribe

Pteromyini, evolved their gliding ability through a common ancestor. But once molecular data was included it became apparent that the gliding ability evolved separately in the Americas and Eurasia from ancestors that did not have the capability. This complicates the placement of the fossil for a gliding squirrel.

2. Many organisms do not leave a fossil record. In particular many of the invertebrates do not have structures that would survive over geological time. This limits the set of taxa which can currently be dated.

3. The age of fossils is not exact. All modern radiometric methods have some variance associated with them. In Figure 3.3 this is represented by the distributions shown under the fossils. It may be possible to provide upper and/or lower limits, referred to as hard bounds" on the age of the fossil or it may be preferable to allow the age to vary out to the tails of a distribution.

4. A species exists for some period of time. There are existing species that have changed little over geological time. It is generally impossible to know if a fossil represents an early specimen of the species or a later one. In other words where on the branch the specimen represented by the fossil should be placed.

5. In early divergence time models fossils were always considered extinct ancestors of existing species. But, as shown in Figure 3.4 it is possible that the fossil represents an evolutionary "dead end" as is the case of Homo neanderthalensis.

3.1.5 Rate Correlation

The goal of divergence time inference is to determine the lengths of time associated with each branch of the tree. This is accomplished by first inferring the rates for each branch and then using these rates to infer the times for each branch. There are several models for the rates:

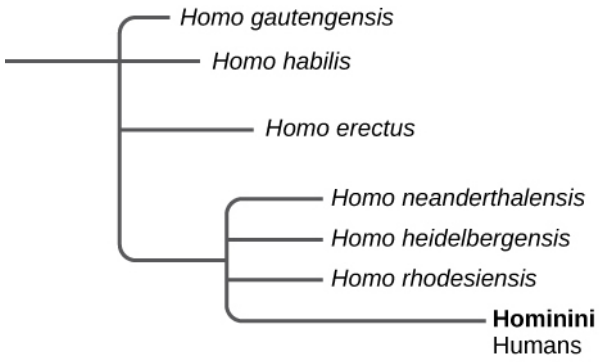


Figure 3.4: *Tips vs. Branches.* Not all fossils represent ancestors of existing species (the "branch" model). For example Homo neanderthalensis is not an ancestor of Homo sapiens but an extinct evolutionary "dead end".

1. The molecular clock hypothesis[98] postulates that the rate of evolutionary change is approximately constant over time and across lineages. In this model each branch of the tree will have the same rate. While this model may be usable in many cases the hypothesis has been shown to be violated by groups such as the mammals[59].

2. At the other extreme, if the rate of mutation can both vary over time and over different lineages each branch can have a completely independent rate. While it can be argued that this, the most complex, hypothesis would allow for any possible rate it also ignores any potential for correlation between rates. It is certainly reasonable to think that the rate of evolution didn't suddenly change when a new species.

3. Between these two extremes there are a number of possible hypothesis that allow for correlation between rates associated with different taxa. One of these models, implemented by both Yang[68] in the MCMCTree program and Drummond *et al.*[27] in the Beast Program, bases the rate for a branch on the rate

Beast V1.8	**MCMCTree V4.8**
Combined phylogenetic and divergence time inference. No option to use existing tree.	Divergence time inference only, tree provided by user.
Utilizes Beagle library for parallel (CPU/GPU) likelihood computations.	Serial implementation only.
Limited multi-gene support through the *Beast program.	Comprehensive multi-gene support.
Tip or branch dating both supported.	Tip or branch dating both supported.
Multiple statistical distributions for fossils (normal, lognormal, gamma).	Multiple statistical distributions for fossils (normal, t, gamma).
Java Application	C Application

Table 3.1: *Comparison Between Beast and MCMCTree.* Key differences are the ability of MCMCTree to accept a user-provided tree and the use of the parallel Beagle library by Beast.

of the branch's most recent ancestor.

3.2 Software for Divergence Time Inference

While the original Multidivtime program is still available, it is not actively under development. The two programs in most use today for divergence time inference are Beast[27] and MCMCTree[94] although other programs exist[45][84]. Both programs are widely used and actively developed by their original authors.

3.2.1 Beast and MCMCTree

A comparison of the Beast and MCMCTree shown in Table 3.1. The key difference is that Beast is intended to perform both phylogenetic inference and divergence time inference as a single step. This creates a issue in cases where the researcher has spent considerable time and effort producing a tree using other methods.

Beast supports a wide variety of options for the specification of the model[28][27]

including:

- Use of different evolutionary models for different genes even to the extent of allow a mix of DNA and other (i.e. Amino Acid) data within an analysis.
- Separate priors for each gene allowing different models to be used for different genes.
- A large number of different models including birth only (Yule) processes[77] and Birth-Death process models[94] for the prior on the trees (including the prior on ages).

It is possible to specify a starting tree for the MCMC process to Beast but the topology of the tree is considered a model parameter and perturbed throughout the process. It is also possible to define sets of taxa corresponding to subtrees (clades) that will be fixed throughout the process so it is at least theoretically possible to define all the clades in a tree to fix the tree topology. But it is possible, even likely, that this use of the clade functionality would destabilize the MCMC process leading to invalid results.

Specification of fossil calibrations to Beast is through the clade mechanism. The user specifies the members of the clade and the age for the MRCA of the clade. The distribution for the calibration is specified as a prior on the clade.

Beast implements the Beagle[2] library for parallel computation of the likelihood. Our own experiments on the squirrel data have shown an approximately 50% improvement in elapsed time for Beast when GPU support using the Beagle library is enabled. If, as is the case in most model based phylogenetic programs, the computation of the likelihood consumes 90% or better of the total time, an improvement of 50% seems modest. Improvements in other likelihood computations have been as

high as 500%[14] in general likelihood computations and as high as 300% for phylogenetic likelihood[21]. This causes one to question whether the implementation is the limitation or if the nature of the Beast model is limiting the scalability of the highly parallel GPU model.

Even though both Beast and MCMCTree are actively used, MCMCTree is intended for divergence time inference only. Therefore the statistical methods from MCMCTree have been used as the basis for the methods in AncestralAge.

3.2.2 Other Divergence Time Programs

The RelTime package[83][84], developed by Tamura *et al.* seeks to provide a simplified approach to divergence time by estimating the relative divergence times for a phylogeny using only the branch lengths without calibrations. A greedy algorithm processes from the root down to the leaves of a tree in a single pass estimating the evolutionary rates for each edge. Edges with statistically unsupportable rates are then revisited and re-estimated.

Of particular interest is the relationship between this algorithm and the algorithm used for setting the initial values in the AncestralAge platform. The initial rates in AncestralAge are also estimated using the branch lengths and then these values are used with the fossil calibrations to estimate the initial ages for the inner nodes and the root.

Heath *et al.* have developed the DPPDiv program [44][45][46] which implements Bayesian divergence time inference using a Dirichlet process prior. This prior is efficient to calculate but it is difficult to implement variable distributions on the fossil calibrations using this model.

3.3 The MCMCTree Program

The MCMCTree program performs Markov chain Monte Carlo inference using the Metropolis-Hastings algorithm and Gibbs sampling. A single Markov chain is implemented which limits the use of convergence diagnostics, most of which depend on the relationship between multiple Markov chains. It is for this reason the program's author recommends performing two runs of the program to see if it reaches stationarity at approximately the same set of divergence times.

The program supports two different methods of computing the likelihood, an approximated method and exact computation. The approximated algorithm depends on a predetermined set of gradient and second derivative values to allow for a Taylor series approximation of the likelihood value without the need to examine the DNA sequences. The gradient vector and second derivative (Hessian) matrix are computed in a step prior to the actual divergence time computation.

Inputs to MCMCTree include all the items previously specified in Section 3.1. A multiple sequence alignment for each loci to be processed is specified but not all taxa need to be represented in all the alignments.

A large number of evolutionary models are available with all the models available for computation using the approximated likelihood algorithm and a subset of the simpler (fewer free parameters) models available to the exact likelihood algorithm.

A tree must be provided and its topology will not be modified by MCMCTree. Branch lengths are not accepted and will be re-estimated by MCMCTree. We are of the opinion that this re-estimation increases the time required for MCMCTree to reach stationarity.

Fossil calibrations are specified as annotations to the input tree. The user is able to specify the statistical distribution (and it's parameters), upper and lower bounds

for the calibration and whether the bounds are soft (the tail of the distribution may extend past the bound) or hard (the tail of the distribution is limited at the bound. The fossil calibrations are used to define the prior for the times of each branch.

A statistical model is provided for the rates to be inferred along with Bayesian priors for the rates. The program supports three models for the rates:

1. A fully correlated rate model that follows the molecular clock hypothesis.

2. A fully independent rate model where each branch may have it's own independent rate.

3. An auto-correlated mode where each branch has a rate correlated to the rate of it's immediate ancestor.

The exact likelihood algorithm follows the pruning algorithm of Felsenstein[32]. It requires a loop over the DNA sequences each time the likelihood is computed and has a computational complexity of $\mathcal{O}(lbn^2 \lg n)$. Due to the cost of this algorithm the development of MCMCTree has focused on the approximated likelihood algorithm.

The approximated likelihood model is intended to reduce the computation of the likelihood to a Taylor series approximation using the first and second derivative values. The computation of the derivative values is accomplished through the use of one of the other programs in the PAML suite, BASEML. The BASEML program performs an MCMC computation on the likelihood function sufficient to generate the gradient vector and the Hessian matrix. This process is performed for each locus being processed and the results combined into a single file. Since the actual DNA sequences are not used as part of the MCMCTree approximated likelihood computation the complexity of the likelihood process is reduced to $\mathcal{O}(n^2 \log n)$. The main issue with the approximated model is it's dependence on the Taylor series

expansion. As long as the true rates for the branches follow the approximate value generated from the Taylor series the approximation is effective. But, when the rates diverge from the molecular clock model they no longer fit the approximation well and the accuracy of the model degrades. For this reason the approximated model tends to loose accuracy as the number of taxa increases or for groups of taxa that are known to be in violation of the molecular clock (e.g. the Mammals).

3.3.1 Software Engineering Issues in MCMCTree

While investigating the source of the 9 month MCMCTree run time estimate for the primates dataset it became apparent that there were a number of issues with the engineering of MCMCTree.

- MCMCTree was derived from the older BASEML and CODEML programs using the same global data structures implemented in those programs. These structures are not entirely appropriate to divergence time calculations and additional structures ave been added to accommodate divergence time with the consequence there are cases where one data field appears in multiple structures with the specific use dependent on when the code was written. All major data structures are arrays with, for example, parent and children node references in tree structures represented as indexes into the arrays instead of memory pointers. This approach creates overhead to perform index to address conversion as well as tying dynamic structures like trees to fixed arrays.

- During computation of the gradient vectors and Hessian matrices no information is provided to give the user an idea how long the process will take. While a detailed analysis has not been performed, experimental results indicate that the computational complexity of BASEML is likely to be at best quadratic and, in all likelihood, cubic or worse. For example, computation of the gradient and

Hessian matrix with 100 taxa and 10,000 base pairs was completed in less than 10 minutes. But an attempt to compute the gradient and Hessian for a 3400 taxa study with over 40,000 base pairs gave no indication of when it would complete after more than 24 hours of execution. Note, all experimental tests were executed on the Texas A&M Brazos supercomputer using 2.5Mhz, 8 core nodes each with 16GB of memory unless otherwise specified.

- Instead of using dynamic allocation, all of the data structures are of fixed, constant size. This limits the capacity of the program but also implies rigorous bounds checking on the structures. But, bounds checking was only partly implemented creating numerous opportunities for data to overrun the arrays and cause program crashes. In particular the array that was used to accumulate statistics would not have been sufficient for the samples from the primates 9 month study and the run would have generated a segmentation fault processing the statistics at the end of the run after a full 9 months of execution time.

- There are numerous dense blocks of code such as the following that are almost impenetrable even after automated reformatting.

```
1 if((ir+1)%max2(mcmc.sampfreq, mcmc.sampfreq*mcmc.nsample/1000)==0) {
2     if(com.np<nxpr[0]+nxpr[1]) { nxpr[0]=com.np; nxpr[1]=0; }
3     for(j=0; j<nxpr[0]; j++) printf(" %5.3f", mx[j]);
4     if(com.np>nxpr[0]+nxpr[1] && nxpr[1]) printf(" -");
5     for(j=0; j<nxpr[1]; j++) printf(" %5.3f", mx[com.np-nxpr[1]+j]);
6 }
```

- There is a single large file containing not only function declarations but their definitions as well that are included in all the PAML programs including MCMCTree. This module has numerous levels of conditional compilation making

determination of the exact code path generated in an particular program difficult.

All of these issues indicated to us that, instead of adding new capabilities to MCMCTree, a new framework should be developed.

3.3.2 Our Work with MCMCTree

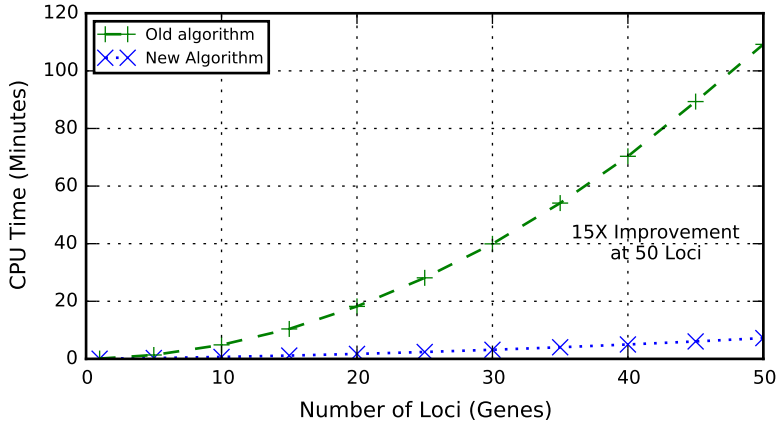
In our initial research into divergence time, we made a number of enhancements to MCMCTree to improve it's performance[20].

1. Performance of the approximated likelihood algorithm
2. Program reliability and robustness
3. Program restartability

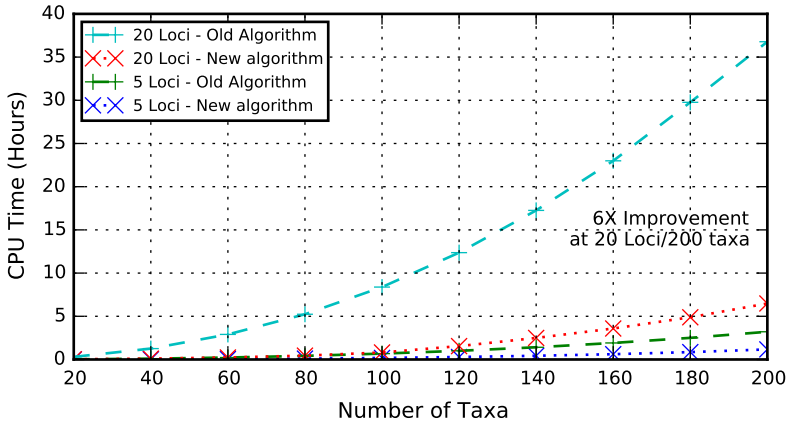
After implementing algorithmic enhancements in all three areas the primates dataset was successfully dated in 14 days, nearly 20 times faster than the original 9 month (270 day) estimate.

3.3.2.1 MCMCTree Approximated Likelihood Algorithm

Based on performance profiling of the code it was apparent that the approximated likelihood computation consumed over 99% of the CPU time required. Further, it was the likelihood computation for new proposed rates (as opposed to other parameters) that consumed the majority (95%) of the likelihood time. Analysis of the code indicated that for each variable proposition during Gibbs sampling, the entire likelihood tree was recomputed. But, detailed examination of the computations involved showed that for a given rate proposition the only values changed would be those associated with the immediate parent and immediate children of the node whose rate was being proposed.



(a) The number of loci is varied from 1 to 50 with a constant set of 20 taxa and a constant total of 1000 DNA base pairs.



(b) The number of taxa from 20 to 200 for two different numbers of loci (5 and 20). The total DNA sequence length was held constant at 1000.

Figure 3.5: *MCMCTree CPU times for Varying Numbers of Loci and Taxa.* Results are for synthetic data generated from random trees.

Based on this intuition I developed a new approximated likelihood algorithm that only recomputed the likelihood for the immediate parent and immediate children of the node being proposed and the adjusted the overall likelihood by the change in value for the nodes updated.

Figure 3.5(a) shows the results of the new algorithm compared to the Yang's old algorithm on a set of synthetic data. This data was generated by first randomly

creating a tree with the number of taxa required. The Seq-Gen program[66] was then used to generate synthetic DNA sequences of fixed length (1000 base pairs total) divided across varying numbers of loci. For Figure (a) the number of taxa was held constant at 20 and the number of loci varied from 1 to 50. Each run was allowed to continue for 100,000 MCMC steps. As can be seen the new algorithm performed up to 15 times faster than Yang's algorithm and the difference was continuing to increase. Also, the performance of the new algorithm was increasing in a linear fashion as would be expected since the likelihood computation was now independent of the number of loci.

Figure 3.5(b) shows the performance of the new algorithm compared to Yang's old algorithm for varying numbers of taxa at 5 and 20 loci on the same synthetic data. Once again the new algorithm significantly outperformed Yang's old algorithm with, in this case, a run time 6 times faster than the original at 20 loci and 200 taxa.

3.3.2.2 MCMCTree Reliability

In order to improve the reliability of the program comprehensive bounds checking on all the fixed arrays was added. At this point the design of the data structures was not changed but a future implementation should make use of dynamically allocation instead of fixed arrays. Data structure should be designed for the needs of the divergence time process, not just added to an existing set.

The statistics module was redesigned to use dynamic memory allowing for much larger study sizes than had previously been possible.

3.3.2.3 MCMCTree Restart

As discussed, after the implementation of the new approximated likelihood algorithm the run time for the primates dataset dropped to 14 days from the original 9 month estimate. But this is still a long time to run given the possiblity of an unre-

lated system crash losing several days of work. We had the opportunity to mentor an undergraduate research student at Vassar College in New York who undertook the project of developing a checkpoint/restart capability for MCMCTree. This capability saved the state of the program at a user-specified interval (time or MCMC steps) and could restart from any of the saved checkpoints.

3.4 Ancestral Age

As a result of our work with MCMCTree, it was apparent that there were significant opportunities for improvement in the implementation of the divergence time algorithms. Yang, *et al.* had focused their research into the approximation methods for likelihood. But, these accuracy of these methods degrades as the molecular clock is violated[26][25]. We developed the subtree site compressed likelihood algorithm discussed in the next chapter to allow for high performance computation of the exact likelihood.

The computation of the prior on ages in MCMCTree was also a brute force algorithm that did not consider either the structure of the data or the nature of the MCMC process itself. In chapter 5 we present our prior of ages algorithm that considers both the structure of the data and the implications of Gibbs sampling.

4 ALGORITHMS FOR COMPUTING THE LIKELIHOOD OF A TREE

In many, if not most, phylogenetic programs, the computation of the likelihood is the most expensive component of the inference process.

4.1 Motivation

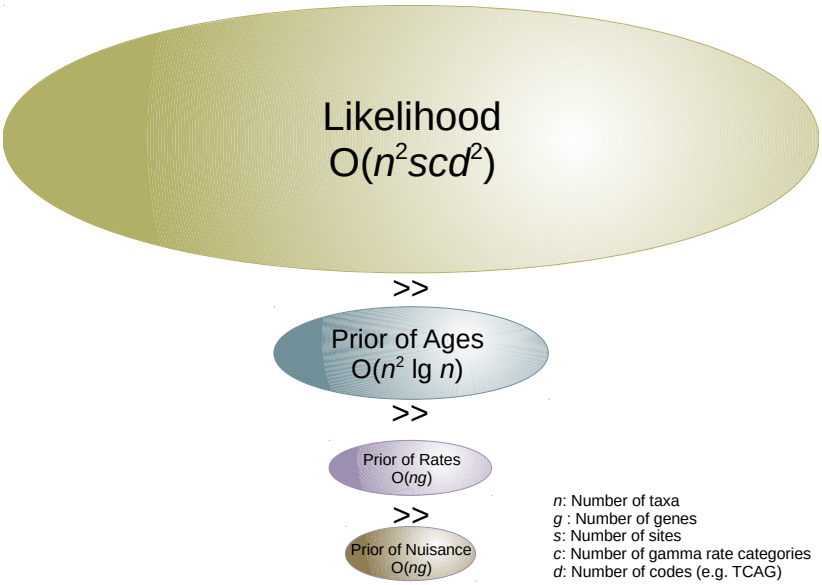


Figure 4.1: *Relative Costs of the Model Components.* This figure shows a ranking of the relative cost of each component of the computational model along with the computational complexity of the component. The costs are all relative to a single MCMC step as the cost per step does not vary significantly.

As shown in Figure 4.1, the likelihood consumes the majority of the CPU time in divergence time inference. In tests on MCMCTree in excess of 95% of the total CPU time is consumed by the exact likelihood computation.

This is in spite of the fact that the computational complexity of the likelihood calculation on a gene tree, $\mathcal{O}(n^2scd^s)$ (c is the number of gamma rate categories and

d is the number of code values), is polynomial. The problem is two-fold; first the values for the number of taxa, n, and number of sites, s, can be large and second, the number of times this calculation is run in a single MCMC step can be large.

For example, in the case of the primates dataset, n is 349 and s is 61,249. Other studies we have see include either larger numbers of taxa (over 4,000 in one case) or much larger sequence lengths (up to 500,000). As the cost of DNA sequencing decreases and the library of previously sequenced taxa[70][4] continues to increase the size of studies will continue to increase.

The second factor, the number of times the computation is run in an MCMC step, is a function of the number of parameters in the model. In the MCMCTree and AncestralAge models the number of parameters n_p can be computed as follows:

$$\begin{aligned} n_p = & (n_0 - 1) + && \text{Number of age parameters} \\ & \sum_{i=1}^{g} (n_i - 1) + && \text{Number of rate parameters} \\ & (m+4)l && \text{Number of nuisance parametes} \end{aligned} \tag{4.1}$$

where g is the number of independent genes, n_0 is the number of taxa in the species tree, n_i is the number of taxa included in gene i and m is the number of parameters in the evolutionary model ($0 \leq m \leq 9$).

As explained in section A, the full tree likelihood does not require computation for changes to an age or a rate parameter. In these cases only a path from the changed node to the root requires evaluation. The depth of a tree will vary depending on the topology, with the worst case depth being $n-1$ in the case of a so-called "caterpillar" tree and the best case being a balance tree with depth $\lg n$ so the average path length

will be half of the depth of the tree, $(n-1)/2$ or $\ln n/2$.

For the age parameters, each parameter proposal will require computation of the likelihood across all the genes. Assuming all taxa appear in all genes, this will require a total of g calculations. Each calculation will be along an average of half the path length to the root yielding a best case complexity for one calculation of:

$$C_{a1} = \frac{\lg n}{2} s_g c d^2 \tag{4.2}$$

where s_g is the number of sites in the gene. Across all the genes the number of sites will sum to s giving:

$$C_a = \frac{\lg n}{2} s g c^2 \tag{4.3}$$

as the computation complexity for the computation of a single age parameter.

So, for the all age parameters in a species tree, the computational complexity will cost per parameter time the number of parameters:

$$\begin{aligned} \mathcal{O}(\mathcal{L}_{ages}) &= (n-1)\frac{\lg(n)}{2} s c d^2 \\ &\simeq s c d^2 n \log n \end{aligned} \tag{4.4}$$

For rate parameters, the complexity will be similarly determined as the number of parameters times the cost per computation. Assuming again that every taxa appears in every gene, the number of calculations will equal $g(n-1)$.

$$\begin{aligned}\mathcal{O}(\mathcal{L}_{rates}) &= g(n-1)\frac{\log(n)}{2}s_g c d^2 \\ &\simeq g s_g c d^2 n \log n \\ &\simeq s c d^2 n \log n \end{aligned} \tag{4.5}$$

Finally, for the nuisance parameters the complexity will equal the number of parameters times the cost of a full likelihood computation at the gene:

$$\begin{aligned}\mathcal{O}(\mathcal{L}_{nuisance}) &= (m+4)g\frac{n}{2}scd^2 \\ &\simeq mgscd^2 n \end{aligned} \tag{4.6}$$

Therefore, for the best case the overall complexity of the likelihood computation for an MCMC step is:

$$\begin{aligned}\mathcal{O}(\mathcal{L}) &= \mathcal{O}(\mathcal{L}_{ages}) + \mathcal{O}(\mathcal{L}_{rates}) + \mathcal{O}(\mathcal{L}_{nuisance}) \\ &= scd^2 n \lg n + scd^2 n \lg n + mlscd^2 n \\ &\simeq scd^2 n \lg n \end{aligned} \tag{4.7}$$

In the worst case, the calculations are the same with the exception that the $\lg n/2$ path to the root is replaced with half the "caterpillar" distance $n/2$:

$$\begin{aligned}\mathcal{O}(\mathcal{L}) &= scd^2n^2 + scd^2n^2 + mgscd^2n \\ &\simeq scd^2n^2 \end{aligned} \tag{4.8}$$

It should be remembered that this computation is per-MCMC step and the number of steps is typically in the hundreds of thousands or better.

It should be apparent from this analysis that the primary focus for any improvements in performance or efficiently should be the likelihood computation. There have been a number of different approaches to reducing the cost of the likelihood computation including algorithmic improvements [76][32], approximation methods [25] and parallelization of the process [5][51][2][35]. For the most part these approaches have been focused on the problem in the context of phylogenetic inference wherein the topology of the tree is being inferred along with the edge lengths. In the context of divergence time inference where the tree topology is fixed, there has been far less research done[21].

In any optimization problem there are generally three methods for improving performance:

1. Reduce the time required for a particular operation.

2. Reduce the number of times an operation is performed.

3. Devise a method for approximating the results of the operation.

In the case of likelihood the computation of likelihood at a particular point in the tree is straightforward, our research has focused on reducing the number of times the basic computation is performed.

4.2 Likelihood Computation on a Tree

The likelihood of a species tree is computed as the produce of the likelihoods of each of the genes included in the analysis:

$$\mathcal{L} = \prod_{i=1}^{l} \mathcal{L}_i \tag{4.9}$$

This value is typically computed as the sum of the natural logarithms of the likelihoods at each gene. As the size of the tree increases, the likelihood values become very small. The use of the natural logarithm of the value provides protection from numeric over and under flows.

$$\ln \mathcal{L} = \sum_{i=1}^{l} \ln \mathcal{L}_i \tag{4.10}$$

Within a gene, the likelihood is computed as the product of the likelihood for each site. Once again, this is typically computed as the sum of the logarithms of the individual sites giving the species tree likelihood as:

$$\ln \mathcal{L} = \sum_{i=1}^{l} \sum_{j=1}^{s_i} \ln \mathcal{L}_{i,j} \tag{4.11}$$

where s_i is the number of site in gene i.

To compute the likelihood at a site in the alignment, two components are required:

1. The alignment values.

   The values for the associated taxa from the column in the multiple sequence alignment corresponding to the site of interest. For example, in Figure 4.2 at some site i the Golden Squirrel has a "T" in the alignment and the Groundhog has a "C".

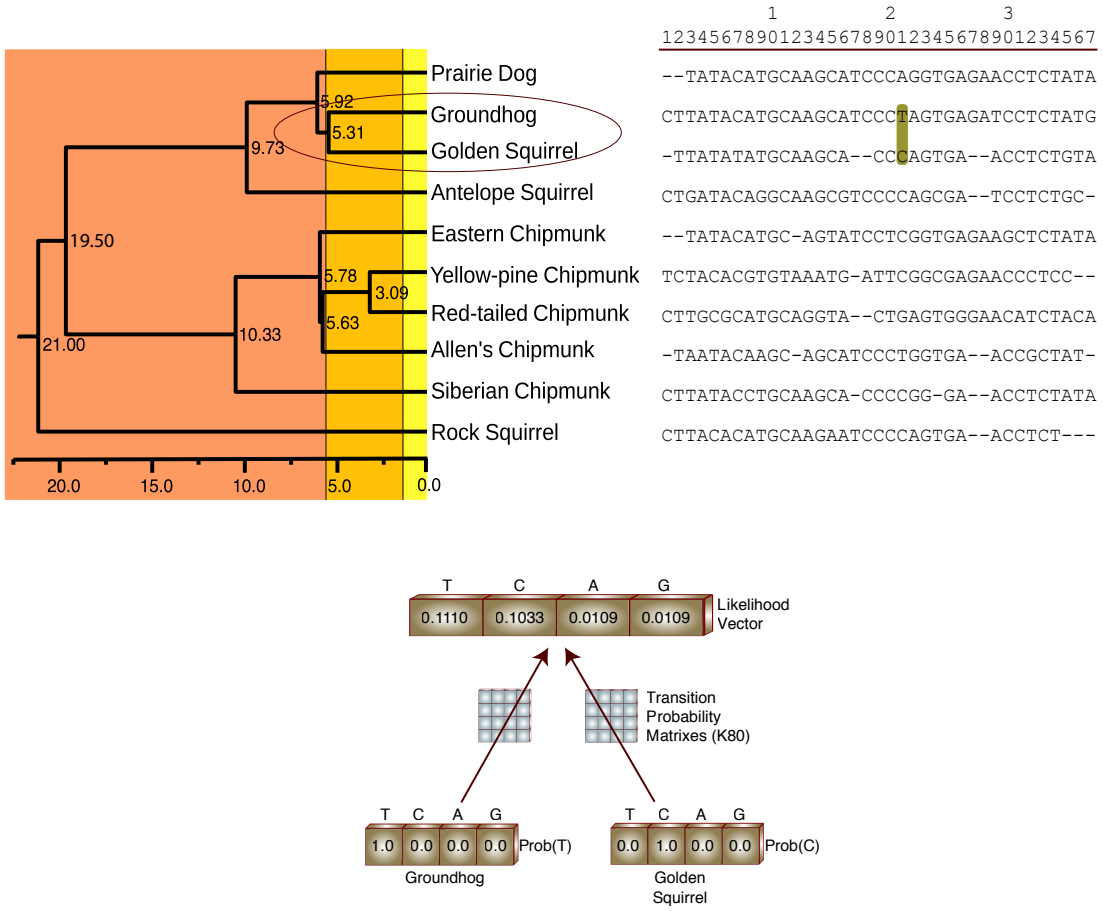


Figure 4.2: *Likelihood for a Pair of Leaf Nodes.* At some site i, the Golden Squirrel has a "T" in it's DNA sequence and the Groundhog has a "C". The likelihood that the ancestor of these creatures had the various DNA codes is computed using the code values and the TPMs.

2. A transition probability matrix (TPM).

   The TPM defines the probabilities associated with a site changing from one code value to another. The matrix is an expression of the evolutionary model for a specified edge length. For example, in the Kimura 1980 model the α and β parameters define the rates at which two different types of changes, transitions and transversions, occur. The actual probabilities of these events occurring is dependent on the length of the edge. A longer edge will have a

greater probability of a change occurring than a shorter edge.

The relationship between the TPM and the rate matrix is defined by:

$$P = e^{Qb} \tag{4.12}$$

where P is the TPM, Q is the rate matrix and b is the edge length.

For our purposes the edge length is defined as the rate of evolution over the time associated with the edge, $r \times t$. The important implication of this is that any time the rate or time changes for a edge, the TPM needs to be computed.

The result of this computation is a vector of length c (4 for DNA). We will refer to this as the likelihood vector.

In Figure 4.2 the likelihood vector for the MRCA of the Golden Squirrel and the Groundhog is being computed given the DNA codes at the site for the taxa and the TPMs computed for the individual edges leading to the MRCA. At the MRCA, the vector is computed such that each entry in the table corresponds to the likelihood that the ancestor had the associated DNA code as it's value. For example, in this case the likelihood that the MRCA had a "T" at the site is 0.1110.

To compute the overall likelihood at a site, the vectors are computed for each ancestral node in a depth first traversal with the vector at the root of the tree giving the likelihoods of each of the code values at the site. This is shown in Figure 4.3 across a set of five of the Marmots. Each edge will have a unique TPM corresponding to it's rate and time values. At the root of the tree, the product of the likelihoods for each of the codes (actually the sum of the logs of the likelihoods) is computed and returned as the likelihood for the overall site.

To compute the value at a specific ancestral node, a series of products and sums

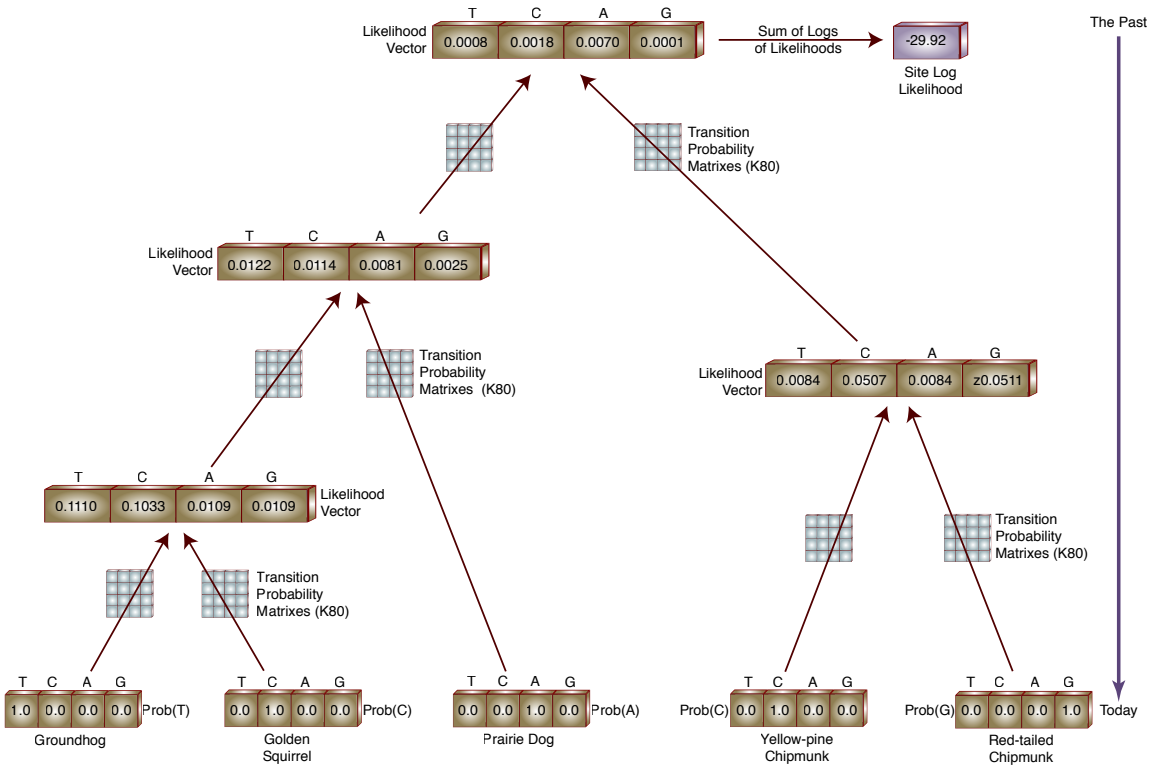


Figure 4.3: *Likelihood for a Site.* Values are computed bottom up in the tree with the value for the overall site being the product (or sum of the logs) of the values for the DNA codes.

is computed. Figure 4.4 demonstrates the computation at the MRCA of the Golden Squirrel, Groundhog and the Prairie Dog. For a particular code value in the likelihood vector for the MRCA ("C" in this example), the product of the likelihoods for the two children are computed. In the case that the child is another inner node, the specific value in the likelihood vector is computed as the sum of the likelihood that any of the codes in the child's likelihood vector would change into the code of interest. For example, in this case, the likelihood that the child has a "T" at the site is multiplied by the probability of a "T" changing into "C", the code of interest. This is summed with the values for a "C", "A" and "G" respectively changing into a "C".

If the child is an existing species, the probability is simply pulled from the TPM.

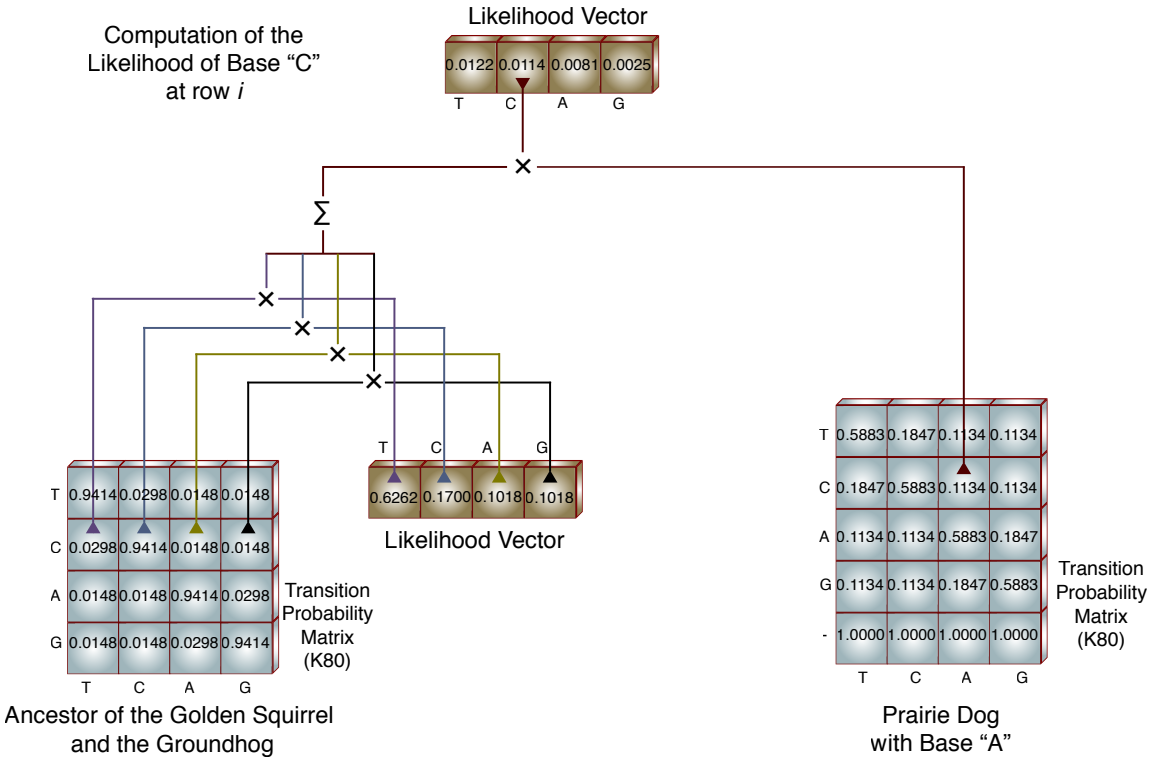


Figure 4.4: *Likelihood at an Inner Node.* For a given position in a likelihood vector, the likelihood is computed as the product of the right child's likelihood and the left child's TPM value.

In this case the Prairie Dog has an "A" in at the site. The value used to compute the MRCA's "C" likelihood is the probability that the Prairie Dog's "A" would change into a "C".

In terms of operations performed, each position in a likelihood vector will require a minimum of one multiplication (for the product of the children's likelihoods). Additionally, for each child that represents an inner node, and additional 4 multiplications (assuming DNA) and three additions are required. Therefore the total operations required to compute a likelihood vector at a site (still assuming DNA) is either 4 when the children are both leaf nodes, $4 \times (1 + 2 \times (4 + 3)) = 60$ when the children are both inner nodes and $4 \times (1 + (4 + 3)) = 32$ when one child is a leaf and

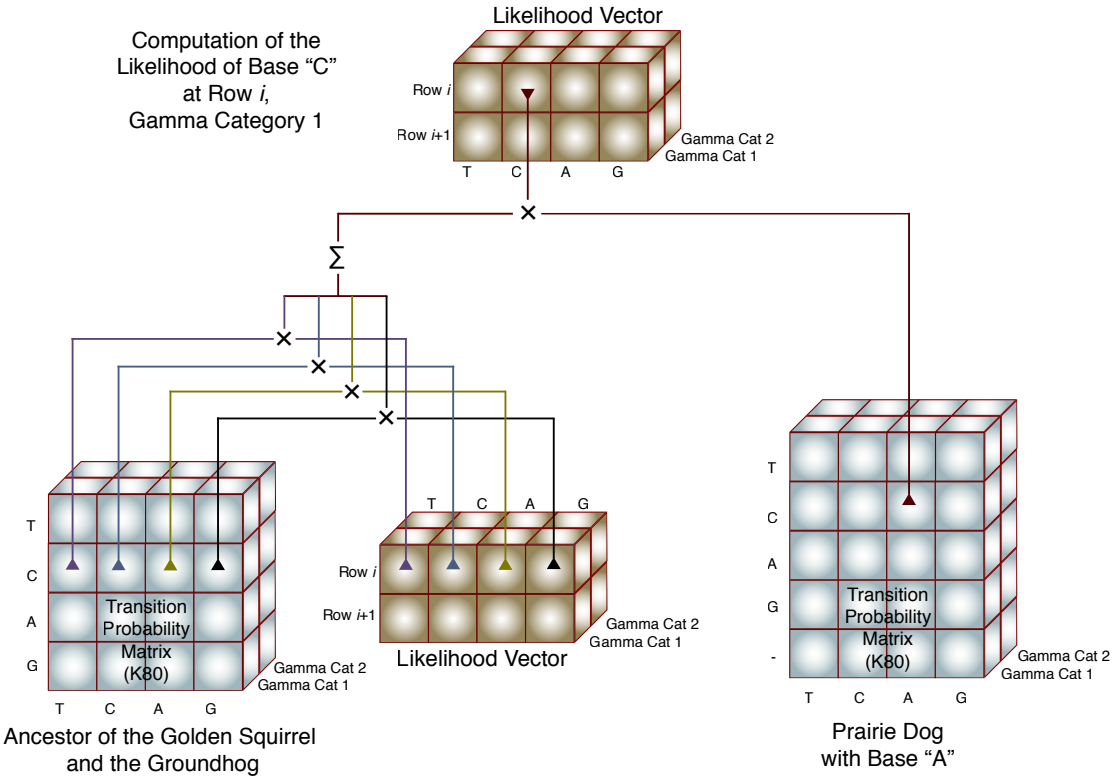


Figure 4.5: *Dimensions of the Likelihood Computation.* Gamma rate categories are shown adding an additional dimension to the likelihood vectors and TPMs.

the other an inner node.

To complete the discussion, there is one additional factor to consider. Typically, it is modeled that the rate of evolution may vary within a DNA sequence. This is usually modeled as a gamma distribution. For a true continuous model, it would be necessary to integrate the likelihoods across the gamma distribution at every site. Research has shown[95] that the use of a discrete approximation to the continuous distribution produces acceptably similar results. This is accomplished by dividing the gamma probability distribution into categories of equal probability. The mean value for each of the categories is then used as an approximation to the value for the entire category.

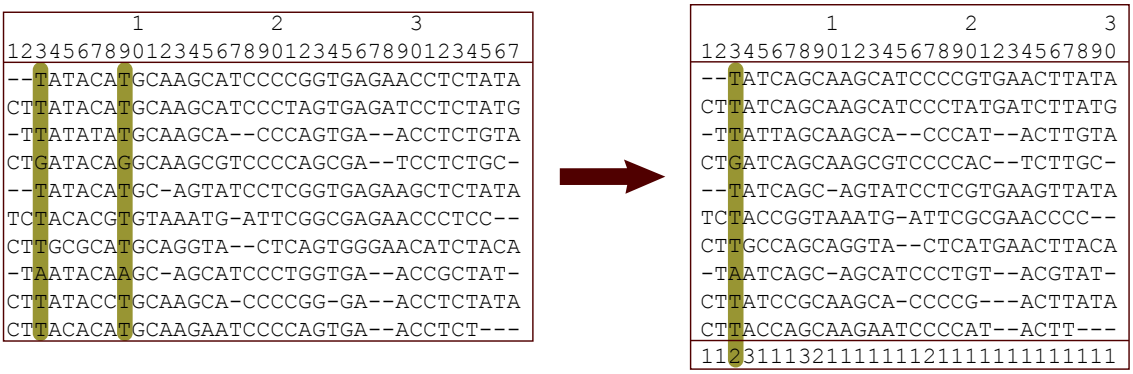


Figure 4.6: *Full Column Compression.* Columns 3 and 9 have identical values for all taxa and therefore are compressed into a single column with a count of 2.

In Figure 4.5 the likelihood vectors and TPMs have been extended with an additional dimension to accommodate the discrete gamma categories.

4.3 Subtree Site Compression

Most phylogenetic and divergence time programs support a simple compression technique wherein the sites in the alignment are examined and duplicate sites compressed. Each site in the alignment has a counter added to indicate the number of copies of the site found. During the summation of the logs of the site likelihoods, the log value is multiplied by the counter to generate the equivalent of repeatedly computing the same value for the multiple copies of the site. In Figure 4.6 sites 3 and 9, for example, have identical values for every taxa and there the counter for that pattern is set to 2.

This technique provides some improvement in the overall performance. For example, the 61,249 sites in the 79 genes included in the primates dataset compress to 32,789 unique sites (47% compression). The problem is that as the number of taxa in an alignment increases, the probability of finding duplicate sites decreases. In the primates dataset the highest compression ratios appeared in the genes with

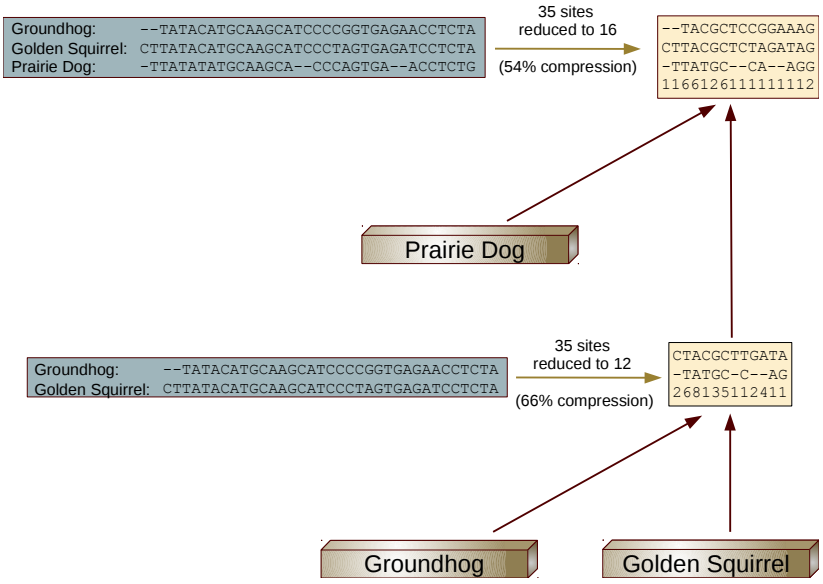


Figure 4.7: *Subtree Site Compression.* At the lowest level the MSA for the Groundhog and the Golden Squirrel compresses to just 12 sites. At the next level the compression is to 16 sites.

the fewest taxa.

The advantage to this technique is that is it independent of the tree topology. This type of compression has been applied in virtually all phylogenetic and divergence time programs. Our insight was that if the topology of the tree is fixed, as is the case in divergence time inference, there is a similar approach could be taken to the compression of subtrees.

Consider an alignment containing only a pair of the taxa from a larger tree. If two different sites in this two taxa alignment have the identical values, they will produce the same likelihood vectors since the edge lengths (and therefore the TPMs) will be the same. Therefore there is no need to repeat the computation for any subsequent site containing the same pattern for those two taxa. This approach, subtree site compression, can be applied to every subtree in the gene.

As an example consider two of the taxa from our Marmots study, the Golden

Squirrel and the Groundhog. In Figure 4.7 just the two sequences for these species are shown and the results with the results of applying subtree site compression. The 35 sites in the alignment compress to just 12 unique combinations, a 66% reduction. Going up another level in the tree, if the Prairie Dog is added to the alignment and subtree compression performed again, the 35 sites are reduced to 16 sites, a 54% reduction.

Using subtree site compression, it can be seen that the maximum number of combinations appearing in the compressed set at any inner node will be $\min(5^n, s)$ where n is the number of leaves in the subtree and s is the length of the original alignment. Obviously 5^n will quickly exceed even large numbers of sites but, every inner node whose children are both leaves will have a maximum of $(c+1)^2$ where c is the number of codes (4 in the case of DNA). One more than the number of codes is used to allow for the "unknown" or missing data code. In a balanced tree $n/2$ of the $n-1$ inner nodes will satisfy this condition.

The algorithm for subtree site compression adds two addition entities at each inner node in the tree; a hash table and a site lookup table. The hash table is used to determine whether, as the subtree alignment is scanned, the code combination has been seen before. The site lookup table is used to index into the likelihood vectors and TPMs for the descendants of the node.

At a given inner node in the tree, the sites corresponding to an alignment of only those leaves that are descendants of the node are considered, one at a time. The concatenation of the code values for the leaves at the site is used as the key into the hash table. If the key already exists in the hash table, no further processing is done. If the key does not exist in the hash table it is added and an entry is appended to the site lookup table. This index of the new entry in the site lookup table is set as the value pointed to by the hash table entry.

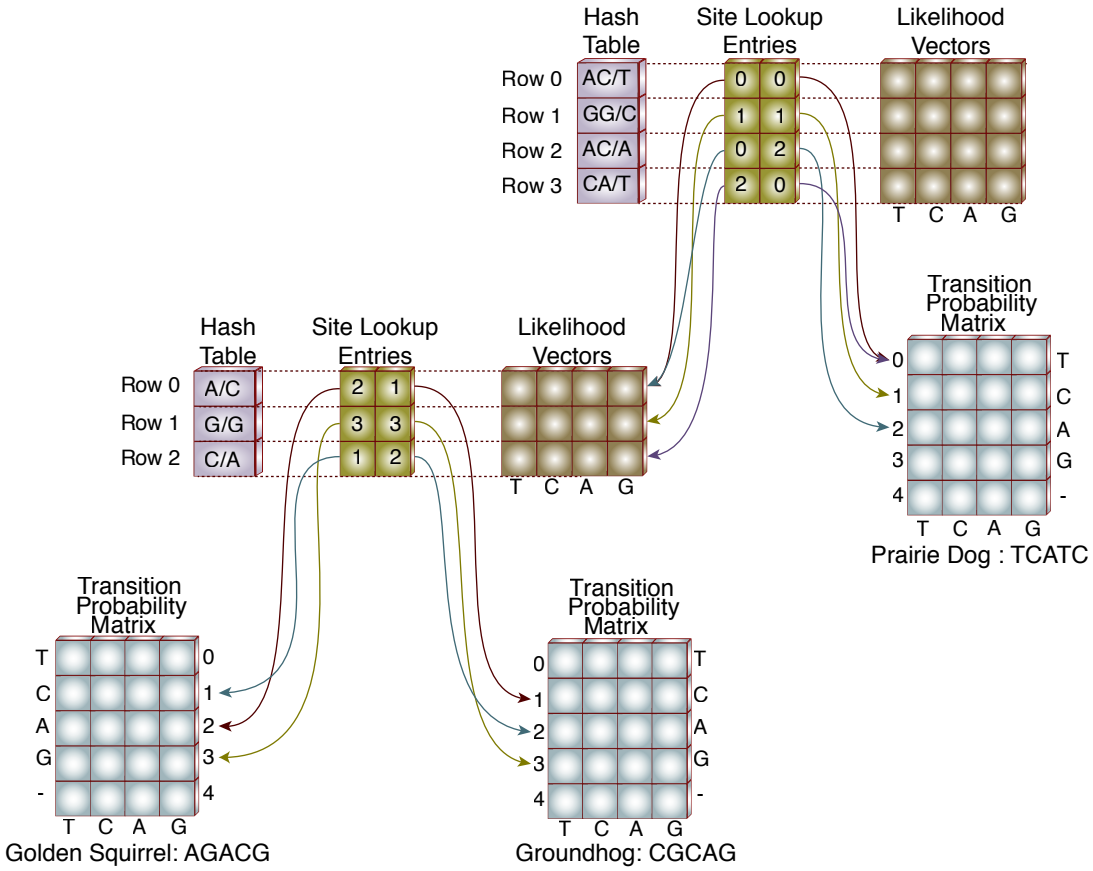


Figure 4.8: *Subtree Site Compression Algorithm.* The structures supporting the likelihood calculation for a set of three species is shown. At the lower level, the ancestor of the Golden Squirrel and the Groundhog has three unique site combinations shown as three entries in the hash, site lookup and likelihood vectors. The ancestor of all three species has four unique combinations in it's alignment. The entries in the site lookup table pointing to the descendant of the Golden Squirrel and the Groundhog contain indicies into the likelihood vectors for the descendant.

The site lookup table entry contains two fields, one for each of the descendants of the node. These fields provided the indices into the descendants likelihood vectors or TPMs. To compute the likelihood for an alignment position on an inner node, the site lookup table entries for the position are used to get the index into the descendants likelihood vector (if an inner node) or the descendants TPM (if a leaf node). If the descendant is a leaf node, the index points to the row in the descendants TPM

corresponding to the value of the site in the leaf. If the descendant is an inner node, a key is constructed containing the site values for only those leaves that are under the descendant. This key is then used to access the descendant node's hash table and retrieve the index value in the descendants likelihood table.

At the root node there is one additional field in the site lookup table, the repeat count as in the original site compression algorithm.

As an example, if we start at the Golden Squirrel and Groundhog in Figure 4.8 their ancestor will have a hash table, site lookup table and likelihood vector. An alignment of five columns of the sequences from each of the taxa is shown. Out of these five site, there are only three unique combinations, AC, GG and CA; therefore the hash table will only contain those three entries. The first row in the site lookup table will point to the "A" row in the TPM for the Golden Squirrel and the "C" row in the TPM for the Groundhog. Similarly the second and third rows in the site lookup table will point to the appropriate rows in the descendants TPMs.

As the next level up in the tree, the ancestor of three species, the Golden Squirrel, the Groundhog and the Prairie Dog is shown. In this case, there four entries in the hash table for the level corresponding to the four unique values appearing the alignment of the three species. The first site lookup table row contains an index into the likelihood vector for the Golden Squirrel and Groundhog's descendant vector corresponding to that portion of the key associated with the Golden Squirrel and Groundhog (AC). The other half of the first site lookup table row contains the index of the "T" row in the Prairie Dog's TPM.

4.4 Analysis

At any inner node in a tree the subtree site compression is limited by two factors; first the total length of the sequence alignment and second the number of leaves in

(a) Balanced tree with 4 leaves

(b) Unbalanced tree with 4 leaves

Figure 4.9: *Impact of Tree Topology on Subtree Site Compression.* In the balanced tree on the left, each of the nodes b_1 and b_2 will have a maximum of $5^2 = 25$ rows in their likelihood vectors and the root, b_3 will have a maximum of $5^4 = 625$. In the unbalanced tree on the right, the lowest inner node u_1 and the root u_3 will have the same maximums as the balanced tree ($5^2 = 25$ and $5^4 = 625$), but the other inner node u_2 will have a maximum of $5^3 = 125$ rows in it's likelihood vector.

the subtree. In the worst case, the number of rows in the likelihood vector at a subtree site compressed node will be the number of codes plus one (to account for the "unknown" or "missing" code value) taken to the power of the number of leaves. For example, in the case of an inner node whose children are both leaves using the 4 DNA codes, the most rows that will exist in the likelihood vector is $(4 + 1)^2 = 25$ (see Figure 4.9).

The maximum number of rows in any given likelihood vector for a DNA coded alignment is therefore the minimum of the sequence alignment length, s, and 5^n where n is the number of taxa in the subtree.

Given the exponential growth in the maximum number (5^n) associated with the leaf count, the maximum will quickly become limited by the sequence alignment length with performance no worse than the existing site compression method. But, in a balanced tree, $n/2$ out of the $n - 1$ total inner nodes will have two children and in these cases the leaf count exponent will, in all probability, be significantly

	Site Compression	Subtree Site Compression
Total Aligned Length	61,249	61,249
Total Likelihood Computations	4,427,618	438,138
Floating Point Operations/Computation	32	32
Total floating point operations	141,683,776	14,020,416
Estimated time on current hardware	0.109 sec	0.011 sec

Table 4.1: *Comparison Between Site and Subtree Site Compression.* The time required to compute the likelihood with the commonly used site compression algorithm and out subtree site compress is shown. Time estimates were computed on a 3.0GHz Intel Core i7 based system.

smaller than the sequence alignment length. Using DNA again, even with three or four leaves there is a good chance that the maximum for the leaf exponential would be less than the maximum for a typical sequence alignment.

At the other extreme, in a completely unbalanced "caterpillar" tree, the sequence length would quickly dominate the worst case and the impact of subtree site compression would be limited to the lowest one (5^2), two (5^3) or possibly three (5^4) nodes.

Using the primates data, we will compare the performance of current existing site compression with our subtree site compression algorithm.

As shown in Table 4.1, there are a total 10,792 inner nodes in the 79 genes with a total sequence length of 61,249 sites compressed to 32,789 unique sites. Factoring in the individual sequence length for each gene, a total of 4,427,618 site likelihood calculations will be performed to fully compute the likelihood for the species tree. With an average of 32 floating point operations (flops) per site likelihood calculation, a total of 141,683,776 flops will be performed to compute the likelihood for the tree.

The number of leaf edge calculations (n) will be roughly equal to the number of inner node edge calculations, $n-2$, so the average flops to compute the likelihood at a particular node and site will be considered the average of the leaf (4) and inner node

(60) values; 32. Using this value the calculation of likelihood across the entire tree will require 141,683,776 flops. Running on a modern desktop machine (3.0GHz Intel Core i7) this has been experimentally shown to require, on average, 0.109 seconds.

For the subtree site algorithm, the 10,792 inner nodes in the 79 genes compressed to a total of 438,138 positions in the various likelihood vectors for this same number of site likelihood computations. Using the same average flops for a calculation the total flops to compute the tree likelihoods is only 14,020,416. Experimentally, the time required for this computation using the same hardware as before is 0.011 seconds, a 90.1% reduction over the site compression alone.

4.5 CPU and GPU Algorithms

The initial implementation of the subtree compressed likelihood algorithm was a serial, CPU based, algorithm. We hypothesized that a GPU algorithm could provide even better performance than the CPU algorithm. To test this hypothesis we developed a GPU implementation of the algorithm. In this implementation we endeavored to keep all the likelihood related data on the GPU as data transfer from memory to the GPU memory is a known bottleneck in GPU processing. We therefore implemented the calculation of the TPMs and the likelihood vectors on the GPU with only the final likelihood value for a gene tree being retrieved from GPU memory.

Performance testing of the version indicated that the performance of the GPU version was on the order of 5x slower than the CPU version. Subsequent analysis indicated that the problem was that the compression allowed by the subtree site compression algorithm reduced the width of the likelihood vectors computed on the GPU such that any gain from parallel processing of these sets was overshadowed by the cost of initiating the computation on the GPU.

We have considered the possibility of computing rate parameter proposals for the

entire set of genes in parallel. Looking at the statistical model, the computation of the acceptance ratio for a rate parameter proposal is dependent only on values in the gene tree. Therefore it should be possible to compute rate parameters across all the gene trees in parallel. At present no implementation exists to allow this approach to be tested.

4.6 Summary

Our subtree site compression algorithm has been shown to significantly reduce the computational requirement for likelihood calculation. While the cost of an individual likelihood calculation has not changed, we have been able to reduce considerably the number of times the calculation is run.

While the most benefit will come when the tree is balanced, there is some benefit in even the most unbalanced cases. In practice the biological datasets we have worked with are neither completely balanced nor completely unbalanced. In practice, as experimental results have shown on the primates dataset, there can be a considerable improvement (90% in this case) in the efficiency of the likelihood calculation.

5 ALGORITHM FOR COMPUTING THE PRIOR PROBABILITY OF AGES

The Bayesian prior on the ages of the nodes in the tree ties the fossil calibrations into the statistical model. There has been some discussion of the use of other non-Bayesian methods for the computation of divergence time[26] but incorporation of fossil data into these models has not been successfully accomplished. The use of a Bayesian prior for fossil calibrations is a natural and easy to comprehend method of incorporating the calibration information in the model. The fossils are actually prior knowledge about the model.

5.1 Motivation

New values for the prior are only required when a new node age is proposed and therefore only computed $n-2$ times for each MCMC iteration. The computational complexity per MCMC step not excessive either, $\mathcal{O}(n^2 \log n)$. The problem is that the constant multiplier for the computation is large. A large computation is required for each node in the tree. In the MCMCTree the computation of the likelihood consumes approximately 93% of the total time, the next largest component of the time is the prior of ages which requires approximately 5% of the total time. In AncestralAge with our high performance likelihood algorithm the portion of the total time consumed by the prior of ages is 31%, still less than the likelihood but far larger than the 13% required for the priors of the rates and nuisance parameters combined.

In the case of likelihood our approach to optimization was to reduce the number of times a simple calculation was performed. In MCMCTree the approach was to add an approximated likelihood computation. For the prior of ages, since the computation is performed relatively few times, the approach has been to reduce the cost of an

individual computation of the prior.

5.2 Statistical Model

The challenge in the design of the prior is to include the fossil calibrations in such a fashion that uncertainties in the fossil dates can be expressed in the model. To this end both MCMCTree and Beast implement priors based on a birth-death process model with species sampling (BDS) [54][67][90].

Other models have also been used. Beast also implements a prior based of a birth only "Yule" model[47] that is a simplification of the birth-death model. Probably the most studied alternative is the use of Dirchlet process prior (DPP) originally devised by Kishino, Thorne and Bruno[56] and most recently applied in the DPPDiv program by Heath[45]. The key issue with the DPP model is the inability to specify arbitrary statistical distributions for the fossil calibrations. In DPP models the path from each leaf to the root is broken into segments modeled using the multivariate Dirichlet distribution. This implies homogeneity in the specification of the calibration points. But if the information about the calibration points varies, as it will given the variability of fossil dating methods, this prior loses statistical power as the variation among fossils increases.

The BDS model is parameterized using the following three parameters:

- The birth rate per linage λ.
- The death rate per linage ν.
- The sampling percentage ρ.

Using notation similar to that of Yang[94] the process is conditioned on the number of taxa n and the age of the root t_r. Non root nodes $\mathbf{t}_{\bar{r}}$ are grouped into two categories with c nodes that have fossil calibrations $\mathbf{t}_c$ and $n-2-c$ nodes without

fossil calibrations $\mathbf{t}_{\bar{c}}$. The prior is then expressed as the joint density of the root node, the calibration nodes and the non-calibrations codes conditioned on the calibration nodes, $f(\mathbf{t}) = f(t_1)f(\mathbf{t})f(\mathbf{t}_{\bar{c}}|\mathbf{t}_c)$.

If an estimation of the root age is available either through a calibration or another estimate, it can be included in the calculation of the calibration node density. If an estimate is not available for the root, it's density can be computed from the BDS process using the number of existing species and the parameters of the BDS model. Specifics of this calculation can be found in in Yang, *et al.*'s paper[94].

5.2.1 Calculation of the Calibration Node Density

The density of the calibration nodes is calculated as the product of the densities of the individual nodes:

$$f(\mathbf{t}_C) = \prod_{i=0}^{x} p_i(t_i) \tag{5.1}$$

where $p_i(t_i)$ is the probability of age t_i according to the probability density function (PDF) associated with the calibration.

Since the individual node calculations are independent, each calculation can be specified using distinct distributions and parameters. Similar to the support in MCMCTree the AncestralAge platform supports a number of specifications for the individual priors:

1. Lower bounds where the node time t will be greater than the bound specified t_L, with a small probability p_L (default = 0.025) that the node time will be less than the bound. Since the upper age is unspecified, the area beyond t_L is modeled as an improper uniform prior. The area before t_L is modeled with a

sharp decay using a factor $\theta = \log(0.1)/\log(0.9)$.

$$f(t) = \begin{cases} p_L \frac{\theta}{t_L} \left(\frac{t}{t_L}\right)^{\theta-1} & t < t_L \\ p_L \frac{\theta}{t_L} & t \geq t_L \end{cases} \tag{5.2}$$

2. Upper bounds where the node time t will be less than the bound specified t_U, with a small probability p_U (default = 0.025) that the node time will be greater than the bound. Since the lower age is zero, the area below t_U is modeled as an proper uniform prior. The area beyond t_U is modeled with a sharp decay using a factor $\theta = (1 - p_L)/(p_L t_U)$ to allow $f(t)$ to be continuous.

$$f(t) = \begin{cases} \frac{1-P_U}{t_U} & t < t_U \\ p_U \theta \mathrm{e}^{-\theta(t-t_U)} & t \geq t_U \end{cases} \tag{5.3}$$

3. Both upper and lower bounds where the node time t will be within the specified bounds ($t_L < t \leq t_U$) with, once again, a small probability p_B that t will be outside of either of the bounds. A default of 0.025 is used as the default that t will be less than t_L and similarly that t will exceed t_U. The area between the bounds is modeled as a proper uniform prior. The area below $t_L$is modeled using a power decay with a factor $\theta_L = (1-p_B)/(p_B(t_U - t_K))$. The area above t_U is modeled with an exponential decay with a factor $\theta_U = (1-2p_B)$ $(p_B(t_U - t_K))$ that .

$$f(t) = \begin{cases} p_B \frac{\theta_L}{t_L} \left(\frac{t}{t_L}\right)^{\theta-1} & 0 < t < t_L \\ \frac{1-2p_B}{t_U - t_L} & t_L < t \leq t_U \\ p_B \theta_U \mathrm{e}^{-\theta_U (t-t_U)} & t > t_U \end{cases} \tag{5.4}$$

4. A gamma prior where the density is specified with the common gamma α and β parameterization.

$$f(t) = \frac{\beta^\alpha}{\Gamma(\alpha)} t^{\alpha-1} \mathrm{e}^{-\beta t} \tag{5.5}$$

5. A normal prior where the density is specified with the usual μ and σ parameterization.

$$f(t) = \frac{1}{\sqrt{2\pi\sigma}} \mathrm{e}^{-\frac{(t-\sigma)^2}{2\sigma^2}} \tag{5.6}$$

5.2.2 Calculation of the Non-Calibration Node Density

The conditional density of the non-calibration nodes $f(\mathbf{t}_{\bar{c}}|\mathbf{t}_c)$ is derived using the theory of order statistics[18]. As shown by Yang[90], the speciation times under the BDS model are order statistics from the kernel:

$$g(t) = \frac{\lambda p_1(t)}{v_{t_1}} \tag{5.7}$$

where $p_1(t)$ is the probability that the linage starting at time t will leave exactly

one descendant in a sample of size ρn.

$$p_1(t) = \frac{1}{\rho} P(0,t)^2 \mathrm{e}^{(\mu-\lambda)t} \tag{5.8}$$

$P(0,t)$ is the probability that a linage starting at time t will leave at least one descendant in a sample of size ρn.

$$P(0,t) = \frac{\rho(\lambda-\mu)}{\rho\lambda + [\lambda(1-\rho)-\mu]\mathrm{e}^{(\mu-\lambda)t}} \tag{5.9}$$

and

$$v_{t_1} = 1 = \frac{1}{\rho} P(0,t_r)\mathrm{e}^{(\mu-\lambda)t_r} \tag{5.10}$$

normalizes the density using the root age.

It is useful to note that in the case where $\mu = \lambda$ these calculations simplify considerably.

Using this kernel the conditional distribution of the non-calibration nodes can be computed as

$$f(\mathbf{t}_{\bar{c}}|\mathbf{t}_c) = \frac{f_{BDS}(\mathbf{t}_c, \mathbf{t}_{\bar{c}}|t_R, n)}{f_{BDS}(\mathbf{t}_c|t_R, n)}. \tag{5.11}$$

The joint density of the node ages under the BDS model is computed as

$$f_{BDS}(\mathbf{t}_c, \mathbf{t}_{\bar{c}}|t_R, n) = (n-2)! \prod_{i=1}^{n-2} g(t_i) \tag{5.12}$$

To determine the marginal density of the calibration nodes $f_{BDS}(\mathbf{t}_C)$, the cumulative density function (CDF) of the kernel is used:

$$G(t) = \int_0^t g(x)dx \tag{5.13}$$

Therefore the marginal density of the calibration nodes given the root and number of nodes is

$$f_{BDS}(\mathbf{t}_c|t_R, n) = \frac{(n-2)!}{\prod_1^c h(i)} \prod_1^C G'(i) \tag{5.14}$$

where

$$h(i) = \begin{cases} (\mathbf{R}_i - 1)! & i = 0 \\ (\mathbf{R}_i - \mathbf{R}_{i-1} - 1)! & 0 < i < c \\ (n - 2 - \mathbf{R}_{i-1})! & i = c \end{cases} \tag{5.15}$$

and

$$G'(i) = \begin{cases} G(t_i)^{\mathbf{R}_i - 1} & i = 0 \\ (G(t_i) - G(t_{i-1}))^{\mathbf{R}_i - \mathbf{R}_{i-1} - 1} & 0 < i < c \\ (1 - G(t_{i-1}))^{n-2-\mathbf{R}_i} & i = c \end{cases} \tag{5.16}$$

where $\mathbf{R}$ defines a list containing the rankings of the ages of all c calibration nodes among the $n-2$ node ages.

By expanding equation (5.11) and canceling terms, the conditional density of the non-calibration nodes given the calibration nodes can then be calculated as

$$f\left(\mathbf{t}_{\bar{c}}|\mathbf{t}_c\right) = \prod_{i=1, i\notin c}^{s-2} g(t_i) \frac{\prod_{i=0}^c h(i)}{\prod_{i=0}^c G'(i)} \tag{5.17}$$

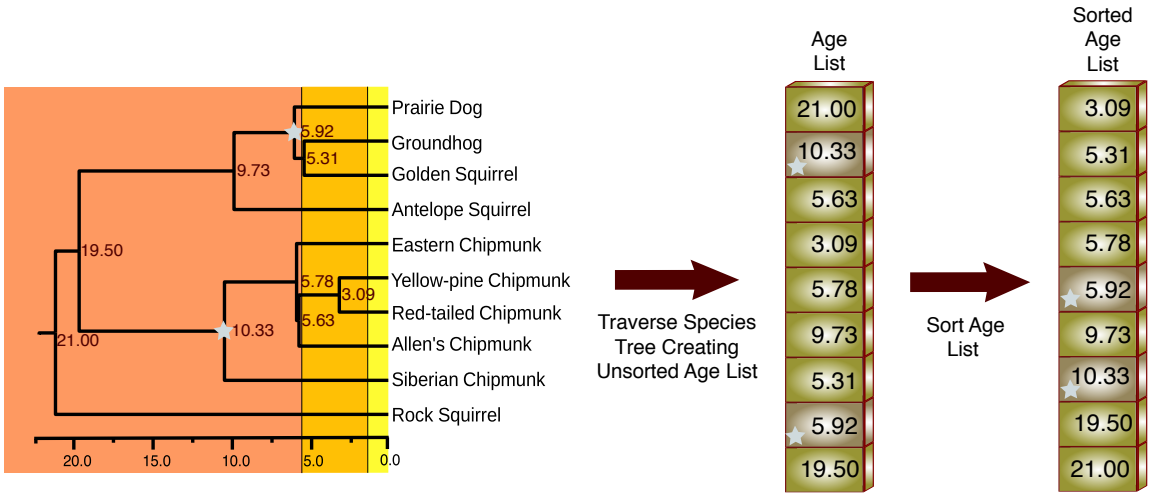


Figure 5.1: *Building the Sorted Age List.* During a depth-first traversal of the species tree, the ages of the nodes are added to a list. Entries in the list that are associated with calibration nodes are marked as such and the list is sorted.

In practice, this is computed using the logs of the values

$$\ln f\left(\mathbf{t}_{\bar{c}}|\mathbf{t}_c\right) = \sum_{i=1,i\notin c}^{s-2} \ln g(t_i) + \sum_{i=0}^{c} \ln h(i) - \sum_{i=0}^{c} \ln G'(i) \tag{5.18}$$

5.3 Algorithm Description

In MCMCTree the prior of ages is computed through a depth first traversal of the species tree each time a new age proposed. During the traversal nodes are added to the list of ages as shown in Figure 5.1.

If a calibration node is encountered during the traversal, the contribution of the node to the calibration node density (section 5.2.1) is computed. As shown in Figure 5.2 the PDF of the current age of the calibration node is computed from the distribution specified. The product of these probabilities is then computed as the sum of the natural logarithms of the values.

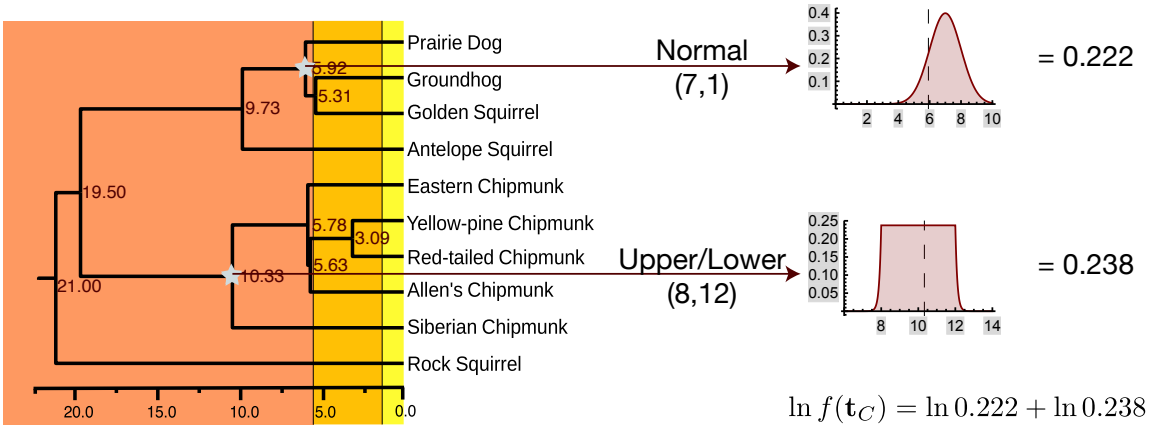


Figure 5.2: *Computing the Density of the Calibration Nodes.* This figure shows an example of the computation of the density of a pair of calibration nodes. During a traversal of the species tree, the current ages of nodes with fossil calibrations are used to determine the PDF associated with the fossil distribution. The logs of these PDF values are summed to arrive at the contribution of the calibration nodes to the prior.

If a non-calibration node is encountered, the BDS PDF value for the age is determined and it's log summed to the non-calibration density (the first term in Equation (5.18).

Once the traversal is complete, the list of ages is sorted in ascending order allowing the ranks of the calibrations nodes to be determined. The area under the PDF curve is then divided into segments. The dividing points for the segments are the ages of the calibration nodes as shown in Figure 5.3. The density used in the computation of the conditional probability is computed as the ratio between the factorial of number of non-calibration nodes in the segment (the difference between the ranks of the calibration nodes bounding the segment) and the area under the PDF curve for the segment raised to the power associated with the number of non-calibration nodes in the segment.

The root is excluded from this calculation even if it has a calibration but it's

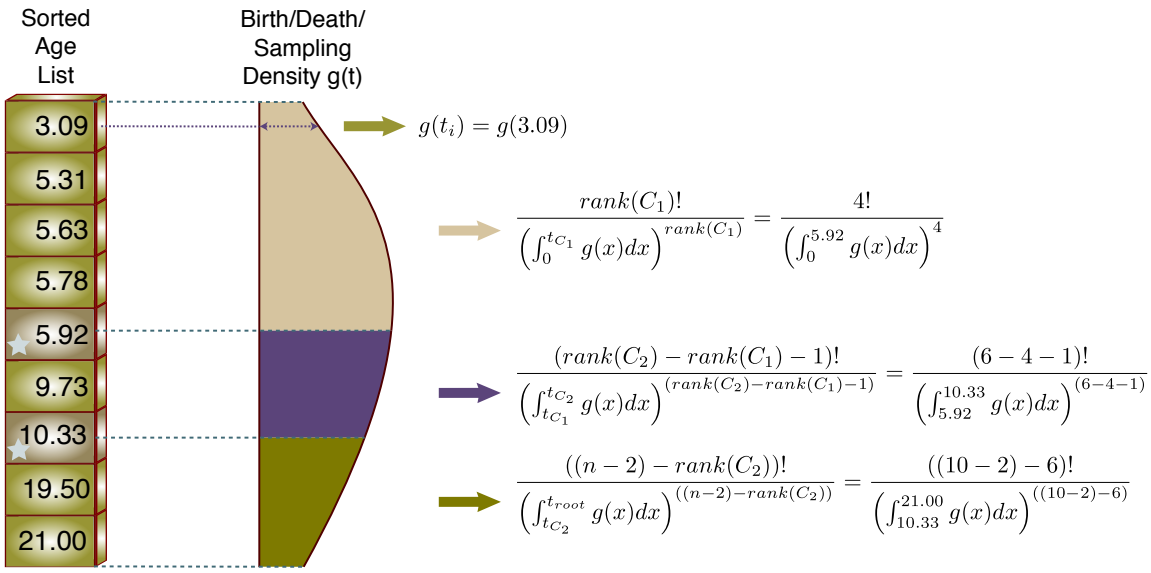


Figure 5.3: *Computing the Conditional Probability of the Non-Calibration Nodes Given the Calibration Nodes.* The $g(t)$ function is computed for each non-calibration node. The marginal density of the calibration nodes is computed by dividing the area under the PDF curve of the BDS distribution into segments with the dividing points defined by the ages of the calibration nodes. The relative weight given to the segment is defined by the number of non-calibration nodes in the segment.

age is used as the upper limit of the BDS distribution. The number of segments is therefore $c+1$. For the first segment, zero is used as the lower bound on the segment.

Computationally, factorials are computed by taking advantage of the fact that the gamma function value for a positive integral value is the factorial. The log gamma function provides an efficient computation of the log of the factorial while reducing the chance of numeric underflows.

A closed form solution to the integral of the $g(t)$ function is available as

$$G(t) = \frac{\rho\lambda}{v_{t_R}} \times \frac{1 - \mathrm{e}^{(\mu-\lambda)t}}{\rho\lambda + [\lambda(1-\rho) - \mu]\,\mathrm{e}^{(\mu-\lambda)t}} \tag{5.19}$$

which simplifies to

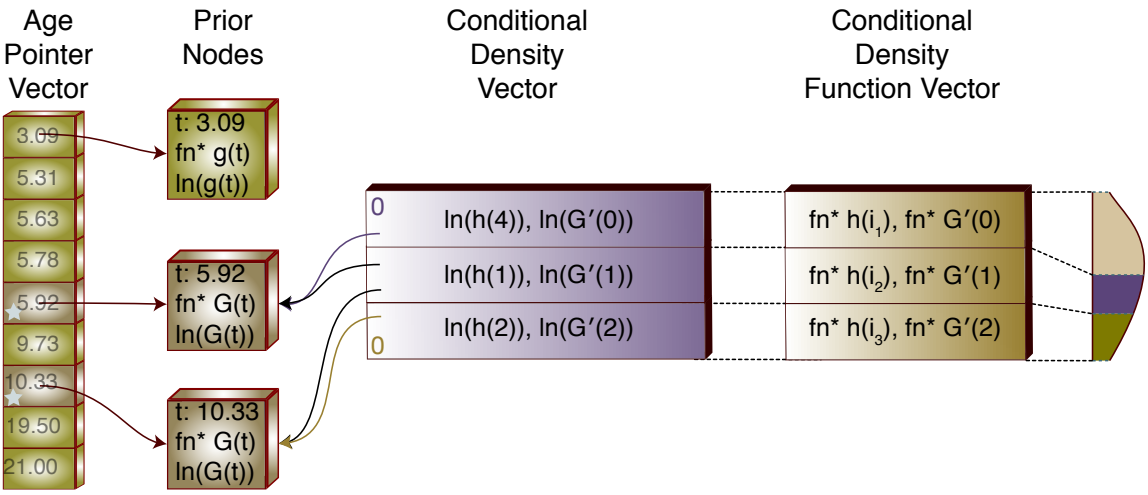


Figure 5.4: *Our Prior of Ages Data Structures.* A prior node (PN) is associated with each species tree node. The age pointer vector (APV) is built during replication. Each segment of the PDF of the BDS distribution has entries in the conditional density vector (CDV) and conditional density function vector (CDFV).

$$G(t) = \frac{(1 - \rho\lambda t_R)t}{t_R(1 + \rho\lambda t)} \tag{5.20}$$

in the case $\lambda = \mu$.

5.3.1 Our Prior of Ages Algorithm

The key to out new prior algorithm is a set of data structures that allow intermediate values to be retained across computations.

These structures are shown in Figure 5.4. All structures are built during the replication process (Section A and Figure A.2). The structures are populated during the initial value process and will persist throughout the execution of the model.

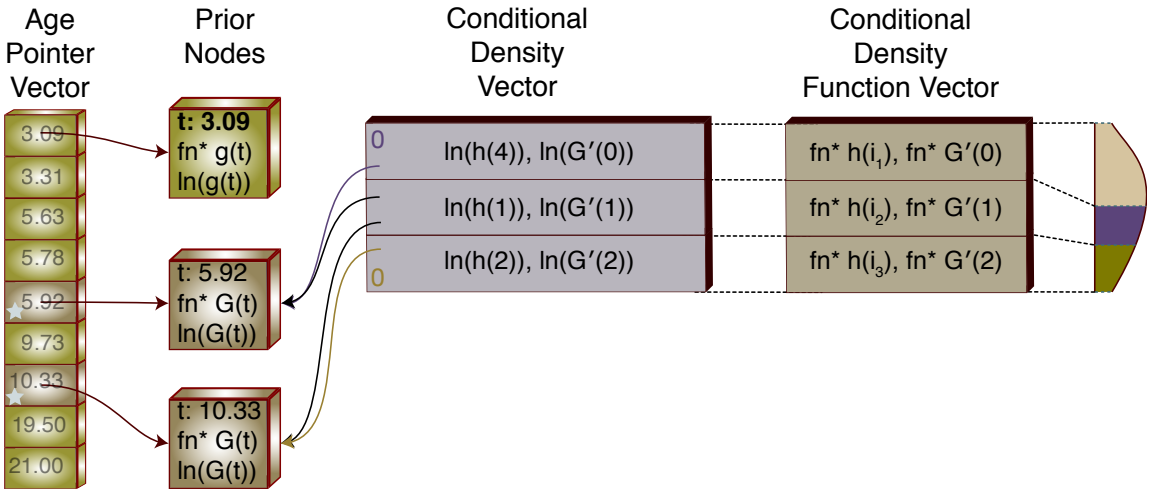
Each non-leaf node in the species tree will have an associated prior node (PN). A common prior node class is further extended into non-calibration and calibration node classes depending on whether the species tree node has an associated fossil calibration. Instead of a list of ages a vector of pointers (APV) to the PNs is main-

tained. By maintaining this list as a vector it is possible to compute the rank value by subtracting indicies into the vector without the requirement to traverse a list.

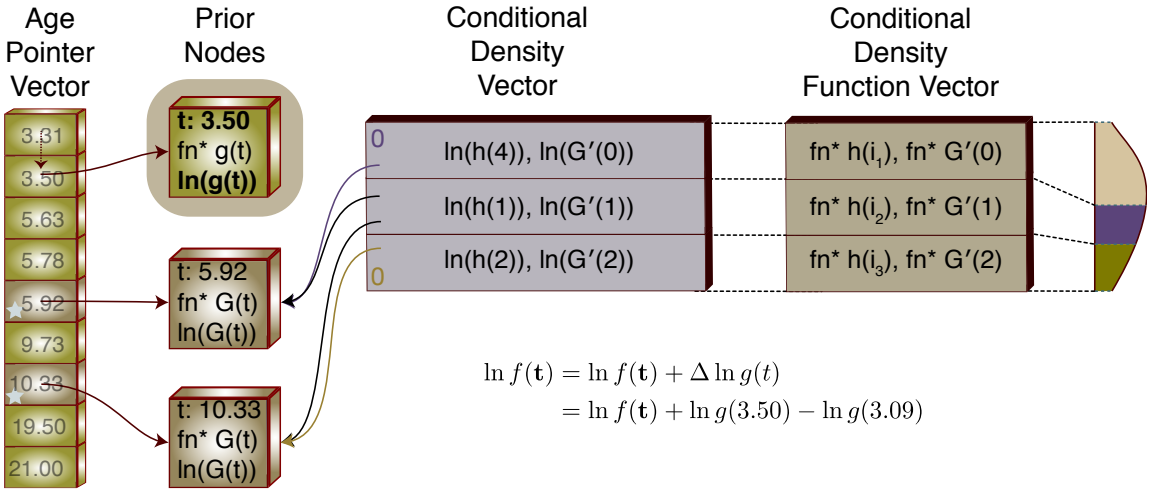
A reference to the parameter containing the current age of the node along with the index of the node in the APV is held in each PN instance. The non-calibration subclass extends this information with the addition of a pointer to the function responsible for computing the $g(t)$ value along with the log of the current $g(t)$ function value.

A pair of vectors is associated with the segments of the birth-death-sampling PDF. Each segment will have an entry in the conditional density vector (CDV) as well as an entry in the conditional density function vector (CDFV). Each CDV entry will contain the current values of the $h(i)$ and $G'(i)$ functions (the second and third terms in Equation (5.18) associated with the segment. The CDV entry will also hold pointers to the starting and ending prior node instances. A CDFV entry is associated with but independent of a CDV entry as it's information is static throughout the execution of the model while the CDV entry contents are volatile. The CDFV entry contains pointers to the functions used to compute the $h(i)$ and $G'(i)$ values for the segment. In reality these functions are functional objects (functors) that are initialized depending on the position of the CDFV entry in the CDV list. For example, the first $h(i)$ CDFV entry will always compute it's function value with the knowledge that it's the first segment. This is particularly important for the first and last CDFV entries as their computations differ from the computations for "middle" nodes (see Equations (5.15) and (5.16)).

Computation of the prior is handled as transactions against the data structures with the goal being minimization of the computation required for any individual transaction. A new proposed age for a node in the species tree triggers the transactions. Transaction are categorized depending on whether the node with the age



(a) Initial State of the Data Structures



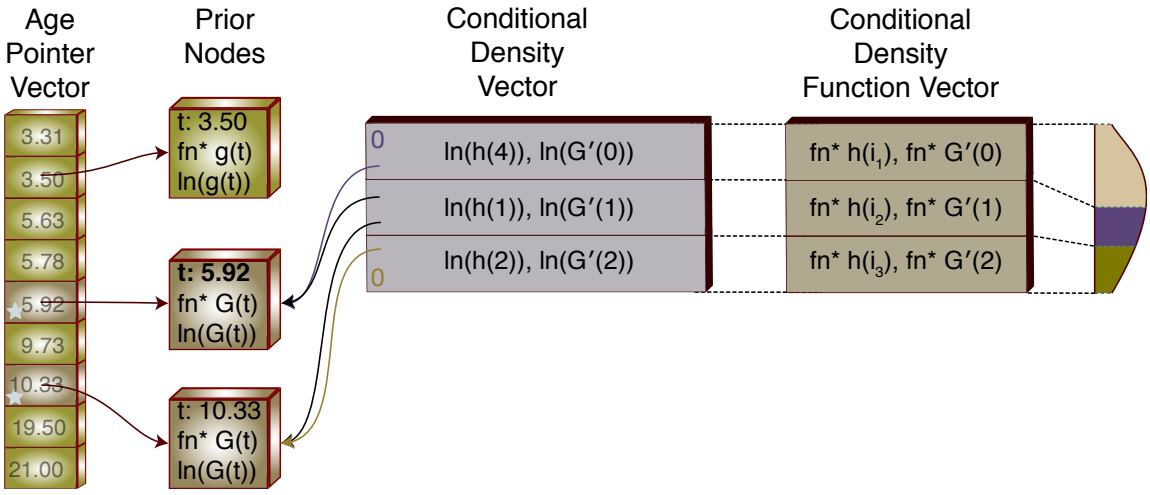
(b) Changed State of the Data Structures

Figure 5.5: *Change in the Age of a Non-Calibration Node.* In the figure the first PN in the APV has a new age proposed. This causes a shift in the position of the node in the APV. The only computation required is to calculate the change in the $g(t)$ value for the node.

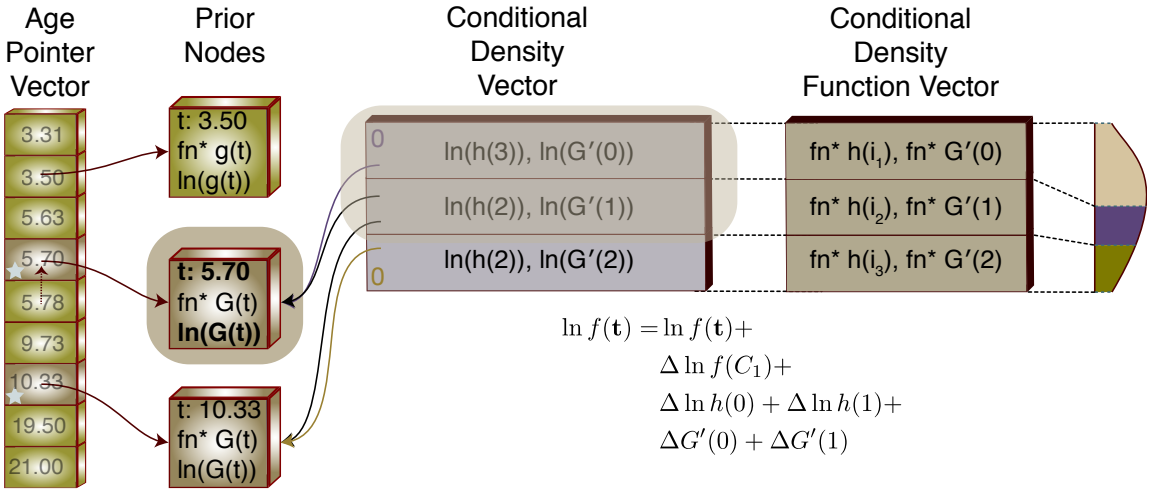
proposal holds a fossil calibration.

- A change in the date of a non-calibration node that does not affect the ordering of the APV.

  In this case, none of the rankings of the calibration nodes change and therefore



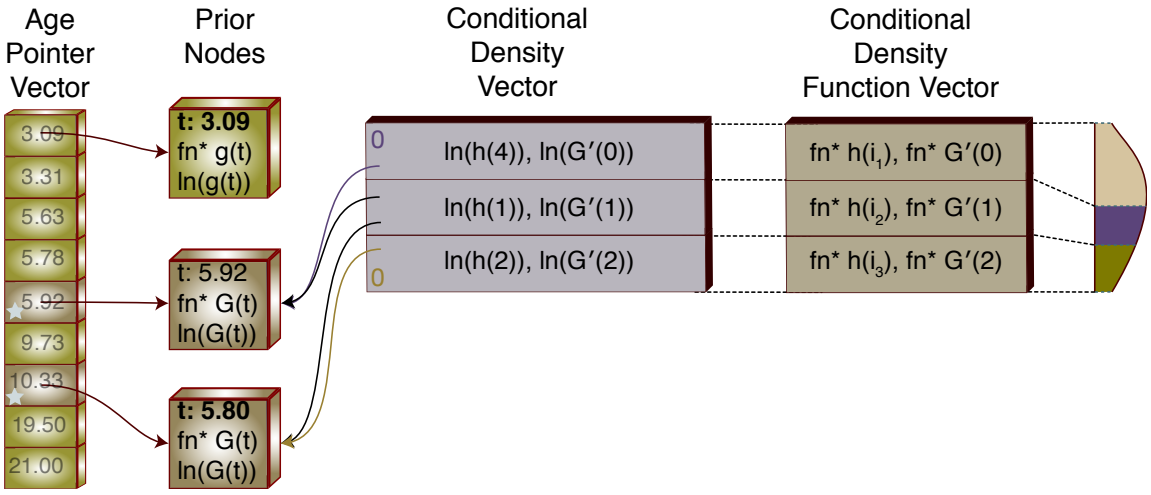
(a) Initial State of the Data Structures



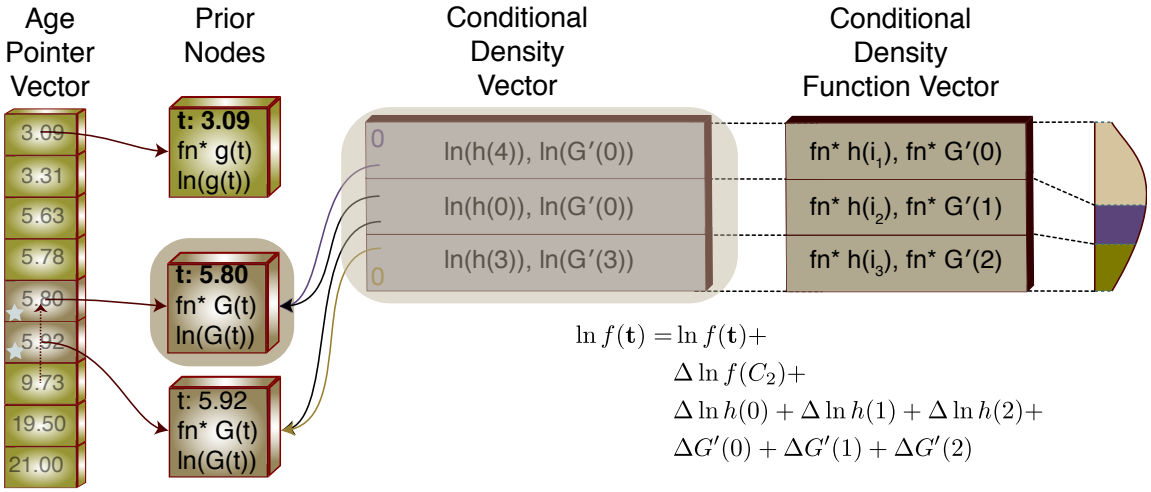
(b) Changed State of the Data Structures

Figure 5.6: *Change in the Age of a Calibration Node.* In this example the age of the first calibration node is changed. The position of the PN in the APV changes requiring calculation of the CDV entries that border the calibration's PN.

there is no change to any of the values in the CDV. The new value for the prior can be computed as the old value updated with the change in the $g(t)$ value associated with the single non-calibration node.



(a) Initial State of the Data Structures



(b) Changed State of the Data Structures

Figure 5.7: *Change in the Age and Position of a Calibration Node.* In this example the age of the first calibration is changed such that the ordering of the calibration PNs is changed. In this case all CDV entries that reference either of the calibration PNs require recomputation.

$$\ln f(t) = \ln f(t) + \Delta lng(t) \tag{5.21}$$

- A change in the date of a non-calibration node that changes the ordering of

the APV.

In this case, the new node age is either younger or older than one or more nodes in the APV. If the movement of the node does not alter the ranking of the calibration nodes the order of the entries in the APV is changed. The remainder of the transaction is handled the same as for the previously discussed change that did not affect the ordering. In other words if the new node age did not change the position of any calibration nodes in the APV. This case is shown in Figure 5.5 for the first non-calibration node.

If the new node age does cause a change in the position of one or more calibration nodes the contents of both CDV entries that border on the changed calibration node(s) need to be recomputed.

$$\begin{aligned} \ln f(t) = & \ln f(t) + \\ & \Delta \ln g(t) + \\ & \Delta \ln h(i-1) + \Delta \ln h(i) + \\ & \Delta G'(i-1) + \Delta G'(i) \end{aligned} \tag{5.22}$$

- A change in the date of a calibration node that does not change the ordering of the APV.

  The PDF value for the new calibration date is computed whenever a new date is proposed for a calibration node. In the case where the ordering of the APV is not changed the new value for the calibration nodes $G(t)$ function is computed and the values for the $G'(i)$ values for the two CDV entries that refer to the calibration node are recalculated. Note that the $h(i)$ functions do not require

recalculation since the ranking of the nodes has not changed.

$$\begin{aligned}\ln f(t) = &\ln f(t) + \\ &\Delta \ln f(t_{PN}) + \\ &\Delta G'(i-1) + \Delta G'(i)\end{aligned} \tag{5.23}$$

The $G'(i)$ value as shown in Equation (5.16) is computed using the difference between the CDF values for the bordering CDV segments. This value is raised to the power associated with the number of non-calibration nodes associated with the segment. Since, in this case, only one of the CDF values has changed, the change in the log of the $G'(i)$ can be computed as

$$\Delta G'(i) = (rank_{i-1} - rank_i)(\ln G'_{new}(i) - \ln G'_{old}(i)) \tag{5.24}$$

requiring only one computation of the CDF.

- A change in the date of a calibration node that changes the ordering of the APV.

  As with any change to a calibration node, the PDF value for the new date is computed. A change to the position of a calibration node in the APV will by definition change the ranking for at least that node. This will require recalculation of the two CDV nodes that border the node. In this case the rankings of the nodes have changed and both the $h(i)$ and $G'(i)$ values will need to be recomputed. This case is shown in Figure 5.6 where the first calibration node is now younger that one of the non-calibration nodes.

$$\begin{aligned}\ln f(t) = &\ln f(t) + \\ &\Delta \ln f(t_{PN}) + \\ &\Delta \ln h(i-1) + \Delta \ln h(i) + \\ &\Delta G'(i-1) + \Delta G'(i)\end{aligned} \tag{5.25}$$

If the change in the ordering of the APV is such that the node moves past one or more calibration nodes, the set of CDV entries requiring recomputation will increase to encompass any entries whose borders have changed. This case is shown in Figure 5.7

5.4 Analysis

In the MCMCTree algorithm, the time complexity for the computation in a single MCMC step is composed of the following components, repeated for each of the $n-1$ inner nodes:

- The traversal of the species tree building the age list: $\mathcal{O}(n)$.
- A traversal of the species tree computing the density of the calibration nodes: $\mathcal{O}(n)$.
- Sorting the age list: $\mathcal{O}(n \log n)$.
- Traversing the sorted age list computing the density of the non-calibration nodes: $\mathcal{O}(n)$.

giving an overall complexity of $(n-1)(n \log n) \sim \mathcal{O}(n^2 \log n)$.

As previously discussed the difference between this computation and the likelihood computation is that the likelihood computation is a fairly simple set of multiplications and additions repeated a very large number of times. This computation is repeated a much smaller $n-1$ number of times for each MCMC step. But while the computational complexity of a single age parameter proposal computation $\mathcal{O}(n \log n)$ is itself reasonable, there is a large constant multiplier with this calculation associated with the computation of PDF of each calibration node time and the $g(t)$, $h(i)$, $G(t)$ and $G'(i)$ function values.

The space complexity is only that associated with the age list (both sorted and unsorted): $\mathcal{O}(n)$.

For our algorithm, the complexity is related to the distance up or down the list a node moves as the result of a new age proposal. Since the data structures are only adjusted during each MCMC step, there is no cost associated with the list generation or sort during MCMC processing.

In the worst case, it is theoretically possible that an age proposal could cause a calibration node to move from one end of the APV to the other. If, as well, all inner nodes had calibrations associated with them the result would be the recalculation of $n+1$ total CDV entries for a worst case complexity of $\mathcal{O}(n)$ for one age parameter proposal and $\mathcal{O}(n^2)$ for an MCMC step.

In practice this is extremely unlikely for two reasons. First, the step size used for age proposals is small. If a step size were used that caused nodes to move large distances within the APV the MCMC process itself would be unstable and probably never reach stationarity. Second, the number of nodes with calibrations tends to be a small percentage of the overall nodes (4% in the case of the primates dataset) so that the length of the CDV is small relative to the total number of inner nodes $(n-1)$.

The best case complexity for a single age parameter proposal is simply $\mathcal{O}(1)$ and for an MCMC step $\mathcal{O}(n)$. If there is no movement in the APV, only a single calculation of $g(t)$ or $G(t)$ and 2 associated CDV entries would be required for a single age parameter calculation.

Experimental analysis of 110,000 MCMC steps over the primates dataset showed that in 73% of the new age proposals no positional changes occurred in the APV and of the remaining 27% of the proposals only 4 proposals (out of a total of 3,480,000 age proposals) moved a node (calibration or non-calibration) more than one slot in the APV. This shows that, in reality, the performance of our new algorithm is much closer to $\mathcal{O}(n)$ than $\mathcal{O}(n^2)$.

In terms of space complexity there is some additional memory required but no change in the actual complexity. There is a prior node and an entry in the APV for each inner node (including the root) in the species tree to give a space complexity of $\mathcal{O}(n)$ for these structures. For the CDV and CDFV, the number of entries is equal to one more than the number of calibrations; $\mathcal{O}(c+1)$. Since the number of calibration nodes cannot exceed the number of non-leaf nodes $c \leq n$ the worst space space complexity for these structures will also be $\mathcal{O}(n)$ giving a total complexity of $\mathcal{O}(n)$ for all the structures.

Doing an experimental analysis of the performance of the prior using the same criteria discussed in section 4.4, it is estimated that the MCMCTree prior computation for one new age proposal required:

- 205 FLOPS for each node with a calibration. This does not include the cost of computing the PDF of the calibration node which will vary depending on the distribution and associated parameters selected for the calibration.
- 144 FLOPS for each node without a calibration.

On the primates dataset, there are a total of 349 taxa $- 1 = 348$ inner nodes of which 14 have calibrations attached. Therefore, the average FLOPS required for a computation of the prior is

$$(394 - 14) * 144 + (14 * 205) = 50,703 \tag{5.26}$$

repeated for each of the 348 non-leaf nodes for which the time is being inferred

$$50,703 * 348 = 17,644,644 \tag{5.27}$$

Repeating this calculation with AncestralAge using experimentally determined value of 1.35 as the number of calculations required for a calibration or non-calibration node age proposal:

$$1.35 * 144 + 1.35 * 205 = 471 \text{ FLOPS per age proposal} \tag{5.28}$$

which when repeated for each of the 348 non-leaf nodes yields

$$471 * 348 = 163,908 \tag{5.29}$$

total FLOPS required for an MCMC step, a 99.1% improvement

5.5 Summary

By retaining the various intermediate values associated with prior computation, it has become possible to reduce the computation of the prior to an effective per-MCMC step complexity of $\mathcal{O}(n)$ from an original complexity of $\mathcal{O}(n^2 \log n)$. This improvement comes at the expense of some additional memory required per n but no overall change the the space complexity ($\mathcal{O}(n)$).

It has been experimentally demonstrated that, using the primates dataset, a 99.1% improvement in the computation of the prior was achieved.

6 SUPPORT MEASURE FOR AGES

The AncestralAge platform has the capability to generate multiple jackknife replicates each with a subset of the fossil calibrations.

Bootstrapping and jackknifing are similar statistical mechanisms for resampling data. In the case of bootstrapping the sampling is with replacement. For example in phylogenetic bootstrapping, columns from the multiple sequence alignment (MSA) are sampled into a bootstrap replicate but since the sampling is with replacement some columns will appear more than once in the replicate and some will not appear at all. In jackknifing the sampling is without replacement, either a sample appears once or not at all.

While it would be possible to generate bootstrap replicates for divergence time by sampling the set of fossil calibrations with replacement, there is no current significance to having a calibration appear more than once on a node (AncestralAge does allow multiple calibrations on a node but their ages are averaged). Therefore we decided to use jackknifing where replicates were generated with subsets of the calibrations randomly sampled without replacement from the full set of calibrations.

By generating sets of replicates and dating them independently it is possible to provide a new support measure to assess the divergence time process.

6.1 Motivation

Fossil calibrations are typically obtained through the use of radiometric dating. These methods all carry their own limitations[3]. Mixtures of fossil dates using different methods may exhibit different bias and variance. As methods have improved previously dated fossils may not fit well with more recently dated specimens. Additionally, in many taxonomic areas the fossil record is sparse and other methods may

be used (e.g. geological strata) to estimate specimen age with potentially larger bias and/or variance.

In phylogenetics the bootstrap is used to provide a measure of support for the topology. Sets of bootstrap replicates are generated and tree inferred. The support for a particular subtree is then determined as the percentage of replicates in which the subtree appears.

Until now there has been no comparable measure for divergence time. We propose a new support measure for divergence time based on multiple jackknife replicates. At a particular node, n, the ages for the replicates t_r are compared with the age of the node in the baseline tree t_b (with all calibrations). While a number of comparisons are possible we choose to compare the t_r replicate ages with the 95% credibility interval (CI) for t_b computed during MCMC processing. The support for the baseline age t_b is then computed as the percentage of replicate ages t_r that are within the 95% CI.

We believe that if the set of calibrations is varied but the age of a node is stable (within the 95% CI) then there is strong support for the age reported. Conversely, if the ages of the node vary outside of the 95% CI when the calibration set is varied, we believe this indicates instability in the model, either in the set of fossil calibrations or in the other model parameters (e.g. MSA, evolutionary model).

It has always been possible to manually generate multiple divergence time runs with different sets of fossil calibrations. But, the performance of the divergence time inference process and the practical difficulty of specifying multiple calibration sets has limited the ability of researchers to experiment with different sets of calibrations. With the new AncestralAge algorithms, it is now possible to date multiple trees efficiently making jackknifing computationally practical.

6.2 Algorithm

During the replication phase of the process (see Appendix A) the directed acyclic graph (DAG) data structure containing the trees is replicated in it's entirety. Each of these replicates is allowed to run on a separate, parallel, execution thread.

To support calibration jackknifing, the user specifies, in addition to the calibration information itself:

- The number of jackknife replicates desired.
- The percentage of the fossil calibrations to include in each replicate (p_r).

The first replicate always contains the full set of of calibrations and is referred to as the baseline. Subsequent replicates are generated by randomly selecting calibrations from the full set. Each calibration has probability p_r of being selected.

Dating then proceeds normally with the jackknife replicates executing on separate execution threads.

6.3 Experimental Analysis

To investigate the support for the ages in the primates dataset we generated a set of 20 jackknife replicates each containing a randomly selected 50% of the calibrations and dated both the base tree and the replicates. We then computed the support measure for each age in the baseline as the percentage of replicates whose ages fell within the 95% CI for the baseline age. Support values are only reported if the support value is $> 90\%$. Calibrations are also shown on the appropriate tree nodes to allow their placement to be analyzed.

The full tree is reported in Appendix E. In Figure 6.1, a section of the tree is shown for the Lorisoidae, a superfamily of nocturnal, largely arboreal primates. There is a single calibration specified for the MRCA of the superfamily. While not

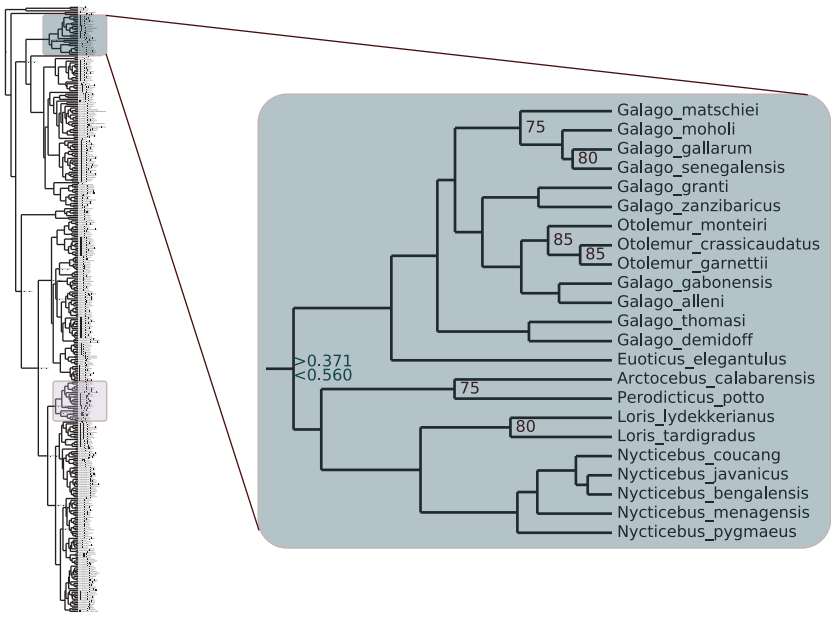


Figure 6.1: *Support for the Lorisoidae.* The superfamily Lorisoidae evidences a number of nodes with $< 90\%$ support.

shown in Figure 6.1 there is an additional calibration between the MRCA and the root applied to next older ancestor of the superfamily. There are other subtrees of similar size in the overall tree that are well supported, this subtree had a large number of nodes with $< 90\%$ support.

Focusing in on one of the values, the 75% value for the ancestor of the Galago matschiei, we see in Figure 6.2 that all the replicates with the exception of one have dates older than the baseline.

If we look at which calibrations are included in which replicate it is apparent that these two calibrations supply a stabilizing influence on the ages. In cases where both calibrations appear in the replicate, the ages for the node most closely approximate the baseline age. There were two replicates where neither of the calibrations appeared in the replicated. These two replicates showed the greatest deviation from the baseline.

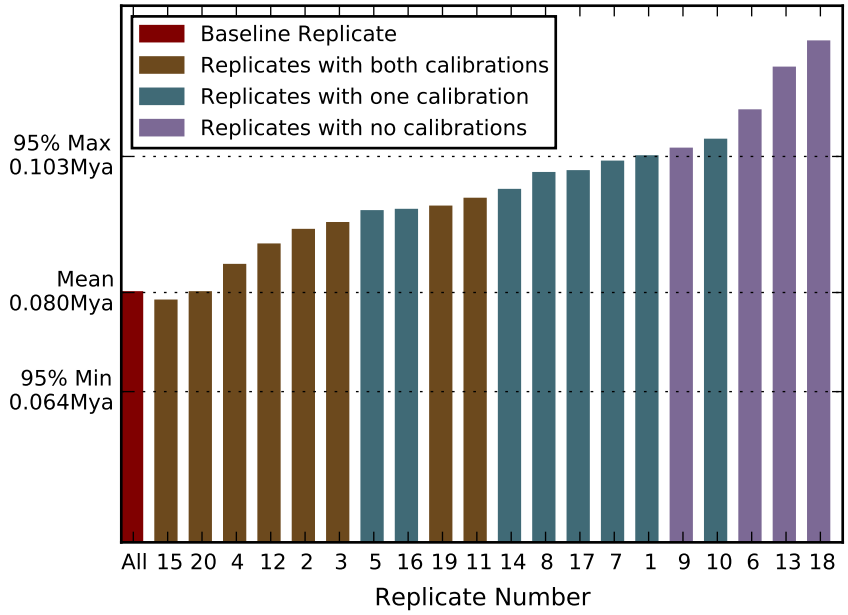


Figure 6.2: *Jackknife Values for the Galago 75% Support Node.* The individual replicate ages associated with the immediate ancestor of Galago matschiei are shown ordered by age. The baseline age is shown at the far right and the 95% CI is marked on the Y axis.

Even considering the inclusion or exclusion of the calibrations, it was apparent that there was a more variation in ages in this subtree than in others in the overall tree. It would be interesting to compare the age support values here with phylogenetic bootstrap support values to determine if there is any correlation between the bootstrap and jackknife support values.

For comparison Figure 6.3 shows the clade that contains the Lemurs, family Lemuridae. In this case, there is only a single calibration between the taxa and the root of the tree. But even with only a single calibration the ages were remarkably stable.

We hypothesize that the presence of only a single calibration might, in some cases, be preferable to multiple calibrations in the path to the root. In cases where

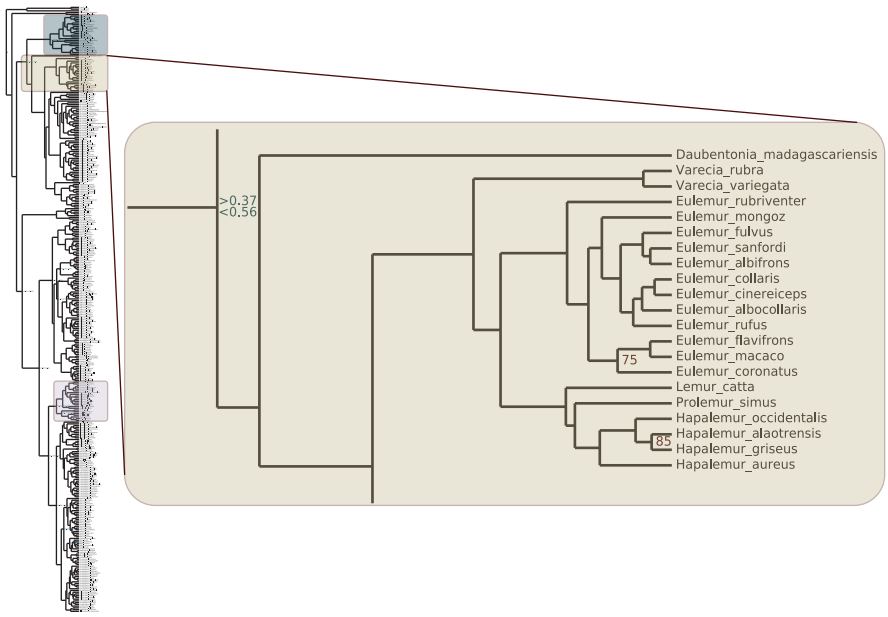


Figure 6.3: *Support for the Lemuridae.* The family Lemuridae is shown.Even though this family only had a single calibration, it's node ages were largely stable.

there are multiple calibrations along a path to the root there is a greater chance that the calibrations would, if not conflict, at least tend to pull the ages in opposing directions. Consider again the case of the Loris superfamily where there were two calibrations between the taxa and the root. Both calibrations were specified with the same age range. It is possible that one of the calibration ages was pulled to the maximum age and the other to the minimum. If one or the other of the calibrations (or both) were removed, the node would be free to move more than it might have otherwise and demonstrated increased variance in the dates of the descendant nodes. In the case of the Lemurs, with a single calibration there would be no potential for this antagonism between calibrations.

As a counterpoint to this analysis consider the situation with the Hominids as shown in Figure 6.4. In this case, there are calibrations on three of the four nodes between Homo sapiens and the MRCA of the family and, not shown, an additional 3

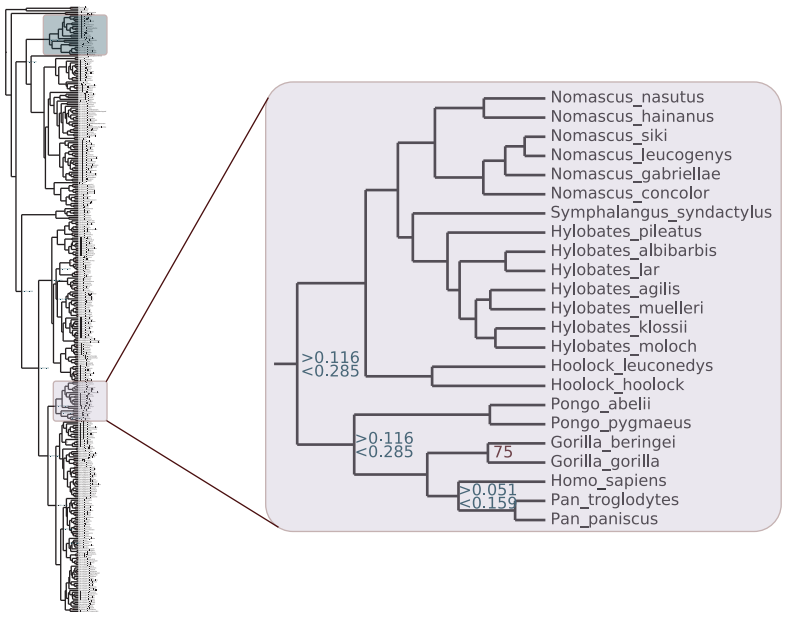


Figure 6.4: *Support for the Hominidae.* The hominids show high degrees of support. The presence of three calibrations is believed to provide stability to the ages inferred.

between the MRCA and the root. The family showed considerable stability across the replicates. In this case, with 50% sampling, there were always at least 3 calibrations between H. sapiens and the root so that the impact of removing one, two or even three calibrations in the path was minimized.

6.4 Summary

Calibration jackknifing provides a powerful tool for understanding the impact of fossil calibrations on divergence time inference.

With the addition of this capability, is has become possible to create a whole new class of support measures on divergence time. We demonstrated one such support measure based on credibility intervals around the ages in the original tree with the full set of calibrations.

Using this support measure, we investigated the stability of the ages on the

primates dataset. Several key points and opportunities for further research were apparent:

- Additional calibrations can help stabilize the ages provided the specifications do not conflict. Calibration node ages that are pushed to one limit or another should be examined.

- If properly specified, even a small number of calibrations can generate stable ages. We found considerable stability in cases where there was only a single calibration on the path to the root.

- We wonder if there is there any correlation between phylogenetic bootstrap support and age jackknife support value? Do branches with poor bootstrap support also tend to have poor age support.

At present, AncestralAge produces unique output trees for each jackknife replicate. For this work, the computation of the jackknife support was a separate post-process of this data. In the future, it would be desirable to either integrate this capability directly into AncestralAge or provide a post-analysis tool to process the results of AncestralAge experiments.

7 ALGORITHMS FOR DATING MULTIPLE PHYLOGENETIC TREES*

We hypothesized that by dating multiple trees prior to the consensus process it would be possible to produce a more statistically supportable dated consensus tree. In order to facilitate research in this are we devised algorithms for efficiently dating multiple trees using AncestralAge. In this chapter we will first discuss the QuickQuartet (QQ) algorithm used to select related sets of trees. Using structures based on those in QQ we will then discuss the algorithms and issues with multiple tree dating in AncestralAge.

7.1 Motivation

With previous methods, the cost of dating single trees was high enough to make dating multiple trees, even through the use of high performance clusters, time prohibitive. Now with our high performance single tree methods it becomes, at least theoretically, possible to date multiple trees efficiently and study the differences between performing the consensus before and after dating the tree(s). Our focus in this research has been the development of efficient methods for dating which will facilitate the analysis of these new methods as well as the dating of larger studies.

The first aspect of the multiple tree dating problem is the selection of the set of trees which will be dated. Large modern Bayesian phylogenetic studies have generated 10^5 or more trees from a single run[72][58]. While it might be computationally possible with our new algorithms to date large numbers of trees, it becomes very difficult to properly assign fossil calibrations across large sets of topologically diverse

*Portions of this section reprinted with permission from *"A Fast Algorithm for Computing the Quartet Distance for Large Sets of Evolutionary Trees."* by R. Crosby & T. Williams, (January 2012), In Lecture Notes in Computer Science, Bioinformatics Research and Applications, (pp 60-71) ©Springer Berlin Heidelberg

trees. In practice, we have found in difficult to create valid calibration sets across as few as 50 input trees. Therefore, the selection of the set of trees to date becomes of significant interest.

The second aspect of the multiple tree dating problem is the development of algorithms that are more efficient than the brute force independent processing of each of t trees. In our work on quartet distance algorithms we found that, in a typical phylogenetic dataset, there was a considerable amount of common structure in the trees. There will be portions of the phylogeny that are well supported by the molecular data and will appear across some or all of the trees. Given this commonality, our research focused on leveraging common topological features of the trees.

7.2 Tree Distance Measures

The first step in the new workflow is the selection of a set of trees for dating. It would be tempting to date all available trees and in fact there may be cases where this is an appropriate approach. But, there are a couple of factors to consider:

1. A sampled set of trees may produce equally accurate results at a much lower computational cost. Sampled methods are know to produce results as good as 100% testing for many processes. Will this hold true for divergence time? What level of sampling is appropriate?

2. The dating process is expensive. Even with new, high efficiency, algorithms it may not be practical to date the large numbers of trees produced by Bayesian inference. For example sets of biological data our lab has worked with have 33,306 and 20,000 trees respectively.

There are several possible approaches to tree selection. Of particular interest are the following three:

1. Random selection of a subset of the trees. This is by far the simplest approach and would provide a representative sample of the input. It should be possible to prove analytically that the results of a sampled approach will, within some margin of error ϵ, approximate the results from dating the entire set of trees.

2. Choose a set of trees "nearest" to the consensus tree.

   This approach would bias the selected trees toward the consensus tree but still allow for the inclusion of topological variance around the existing consensus.

   To determine the trees nearest the consensus likelihood scores could be used or tree distance measures could be employed to find a subset of trees that most closely resemble the existing consensus tree.

3. Select trees from closely related clusters. Use clustering methods to find the set of logical clusters in the data then, using distance methods, select sets of trees around those clusters. This approach would allow for the removal of outliers in the data and still be sensitive to the structure of the data. For example, in the previously mentioned 20,000 tree dataset there are two distinct clusters resulting from two runs of Bayesian analysis. Just using the trees most closely related to the existing consensus tree would ignore this inherent structure in the data.

For either of the last two approaches, a distance measure could be employed. Distance measures can be used to quantify the relationship between trees. Smaller "distances" imply more similarity between trees than larger distance values. While

there are a number of distance measures available, the quartet distance is of particular interest in this case.

7.2.1 The Quartet Distance

The quartet distance has several properties that make it interesting:

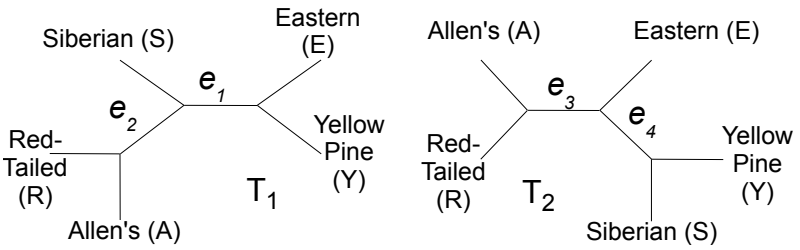
1. It is efficient to compute. Practical algorithms exist that allow for computation of the quartet distance in $\mathcal{O}(n^2)$ where n is the number of taxa. Algorithms exist with even better computational complexities but are not, in practice, more efficient for phylogenetic data.

2. It is relatively insensitive to minor changes in topology. Some other measures such as the Robinson-Foulds distance can show maximal distances for minor perturbations of the topology.

3. The range of distance values for a pair of trees is from zero to $\binom{n}{4}$. This allows for a finer set of distinctions than is possible for measures based on other aspects of tree topology like bipartitions which only allow $n-2$ possible values.

A binary tree is uniquely defined by it's set of $\binom{n}{4}$ quartets. The quartet distance is the number of quartets that differ between two trees. Consider Figure 7.1. Tree T_1 contains two internal edges e_1 and e_2. Removing edge e_1 from the tree generates the quartets $SR|EY$, $AS|EY$ and $AR|EY$. Edge e_2 generates additional quartets $AR|SY$ and $AR|SE$. Let $Q(T_1)$ and $Q(T_2)$ represent the set of quartets in trees T_1 and T_2, respectively. Then, the quartet distance between the two trees is $\frac{|Q(T_1)-Q(T_2)|+|Q(T_2)-Q(T_1)|}{2}$, where $Q(T_1)-Q(T_2)$ is the set of quartets in T_1 that are not in T_2. In Figure 7.1, the quartet distance between T_1 and T_2 is $\frac{2+2}{2}=2$.

The original definition of the quartet distance between trees containing n leaves was done by Estabrook [30]. An $O(n^2)$ time algorithm, upon which my recent work



$Q(T_1)$	SR|EY	(e_1)
	AS|EY	(e_1)
	AR|SY	(e_2)
	AR|SE	(e_2)
	AR|EY	(e_1, e_2)

$Q(T_2)$	**AR|EY**	(e_3)
	AR|SY	(e_3, e_4)
	AR|SE	(e_3)
	RE|SY	(e_4)
	AE|SY	(e_4)

Figure 7.1: *Two Evolutionary Trees and their Set of Quartets.* Quartets in bold are shared by both trees.

was based, was developed by Bryant *et al.*[12]. Subsequently additional algorithms with time complexities of $O(n \log^2 n)$ and $O(n \log n)$ were developed by Brodal *et al.*[9] [10]. The $O(n^2)$ and $O(n \log^2 n)$ algorithms were implemented in the QDist program [60] by Mailund *et al.*. All of these algorithms are based on an insight related to ordered quartets; Brodal et al. [9] observed that the number of quartets for a tree can be viewed as twice the number of ordered quartets (where the direction of the center, connecting, edge is significant). Let A represent the set of taxa (leaves) based on the edge pointing upward. B and C represent the sets of taxa from the other two downward pointing edges. Equation 7.1 defines the total number of ordered quartets can be evaluated as the sum of the ordered quartets present at each of the inner nodes of a tree.

$$\binom{|A|}{2}|B||C| + \binom{|B|}{2}|A||B| + \binom{|C|}{2}|A||B| \tag{7.1}$$

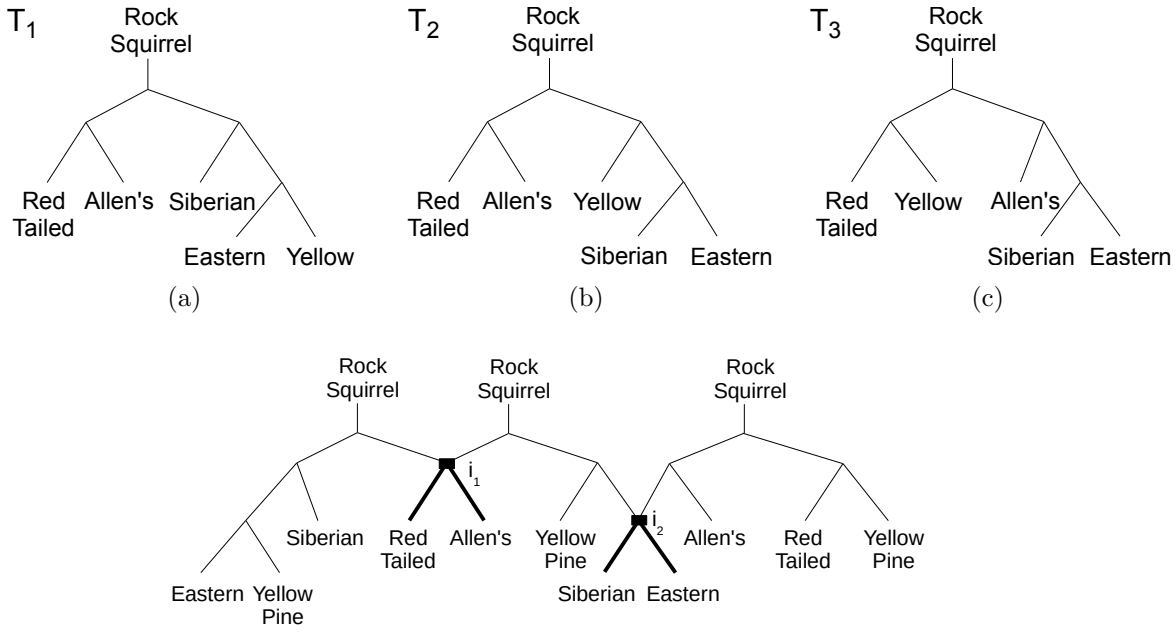


Figure 7.2: *Generation of the DAG Structure.* The first taxa encountered in the input is used as the common root (e.g. Rock Squirrel). Each of the three trees shown in (a), (b) and (c) have their own copies of the root node. Common subtrees (i_1 and i_2) are detected and stored only once in the DAG as shown in (d).

Stissing *et al.* further extended this work with an implementation of the QDist program that computed the all pairs distance for sets of trees [78] once again using the $O(n^2)$ and $O(n \log^2 n)$ algorithms to give complexities of $O(t^2 n^2)$ and $O(t^2 n \log^2 n)$ for all pairs distances. To efficiently compute the quartet distance, Stissing loaded all the trees under analysis into a common directed acyclic graph (DAG) data structure (Figure 7.2). This structure provides a unique root for each tree but subtrees shared across multiple trees are also shared in the DAG as shown in Figure (d). During processing of the trees the taxa (leaves) of the trees are colored with respect to the first, source, tree T_1 and the number of taxa with each color is computed with respect to the second, target, tree T_2.

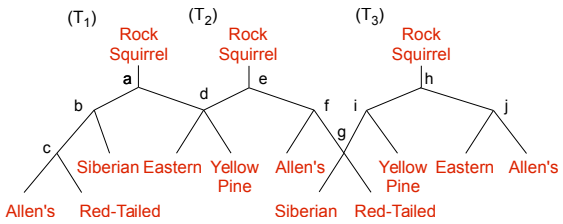
When comparing two trees, the set of shared ordered quartets represented by a node in the first tree and a node in the second tree can be determined for a particular pair of edges [78]. By summing the values for all pairs of edges ($3 \times 3 = 9$) incident to the two nodes, the count of directed quartets at the pair of nodes can be determined. This value, when divided by 2, gives the number of shared, unordered quartets exposed by the pair of nodes.

7.2.2 The QuickQuartet Algorithm

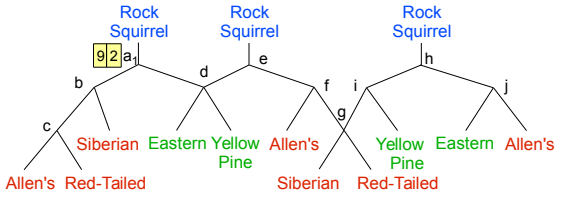
Our QuickQuartet algorithm[19] is based on QDist's implementation of the Stissing et al. algorithm for single tree pair comparison but with a new all-to-all pairs algorithm. When computing the quartet distance between two trees, QuickQuartet identifies additional common bipartitions in the DAG with the use of color strings. These shared bipartitions are not recognized by the Stissing *et al.* approach and results in QuickQuartet's performance improvement by up to two orders of magnitude over QDist.

For each internal edge, there is a single bipartition and many quartets. That is, a bipartition exposes a set of quartets. Given a bipartition B, the number of quartets contained in it is $\binom{p}{2} \times \binom{q}{2}$, where p and q are the number of taxa in the sets X and Y, respectively. For the bipartition $EJL|ST$, the number of embedded quartets is $\binom{3}{2} \times \binom{2}{2}$ or 3. In this example, if another tree had bipartition $EJL|ST$, then we could automatically say that they had 3 quartets in common based on exploring a single bipartition. This is a significant source of cost savings leveraged by QuickQuartet with the use of color strings.

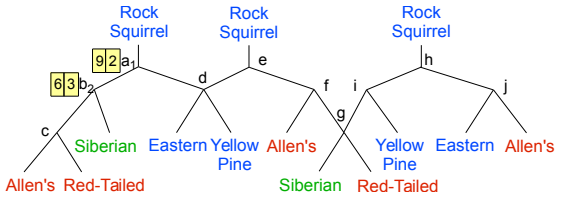
Let U be the set of unique taxa names in any arbitrary order. $|U| = n$, which is the number of taxa. More specifically, $U = \{u_1, u_2, \ldots, u_n\}$, where a_1 is the first taxa name, a_2 is the second taxa name, etc. We can represent a coloring of a node i



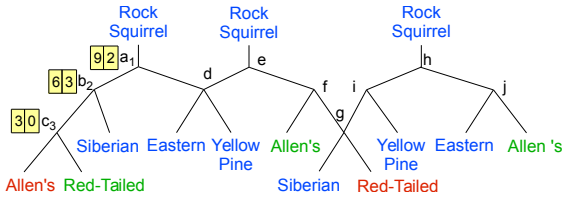
(a) Initial coloring of the DAG.



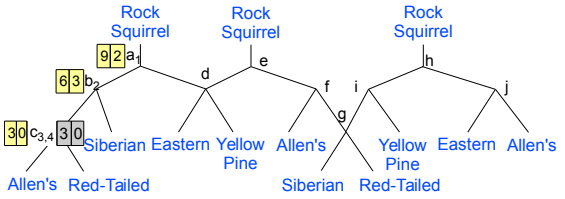
(b) Step 1, compute down vector for node a.



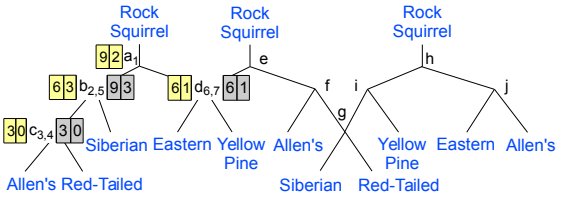
(c) Step 2, compute down vector for node b.



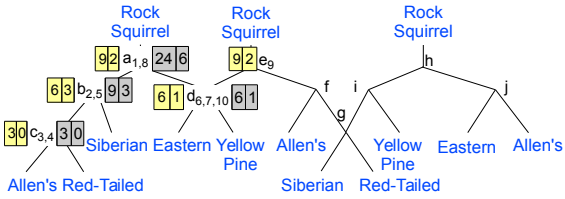
(d) Step 3, compute down vector for node c.



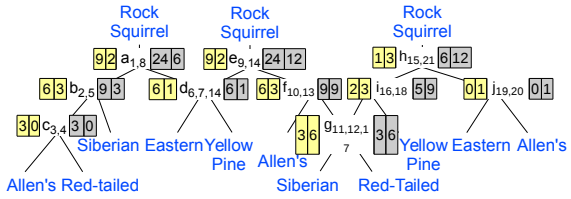
(e) Step 4, compute up vector for node c.



(f) Step 7, compute up vector for node d.



(g) Step 10, reuse down vector for node d.



(h) QDist algorithm finished.

Figure 7.3: *The QDist Algorithm.* The yellow and gray boxes at a source tree's inner node represent the down and up vectors computed during the downward upward tree traversals, respectively. The up vector at node k in the source tree is the sum of the its down vector and its children's down vectors. Each box contains an entry for each target tree.

compactly by using a color string S, where $|S| = |U|$. That is $S = \{s_1, s_2, s_3, .., s_n\}$, where each s_i is the color assigned to taxa a_i. The possible colors of a node are

represented by $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$. Consider the trees in Figure 7.3. Let U= {Rock Squirrel, Red-Tailed Squirrel, Allen's Squirrel, Siberian Squirrel, Eastern Squirrel, Yellow Squirrel}. The coloring for node a in Figure 7.3b is $S = \{\mathcal{AACBBA}\}$ for tree T_1. Node e for tree T_2 has the same color scheme even though the trees T_1 and T_2 have different topologies and are represented as distinct nodes in the DAG. Finding a color string at a previously computed source tree inner node eliminates traversing the remaining target trees resulting in significant running time improvements.

The QDist algorithm creates cost savings by identifying nodes in the trees that have the exact same subtrees (e.g., node d in Figure 7.3b). However, in Figure 7.3b, QDist would redo the calculation for nodes a and e even though both nodes represent the same bipartition and induce the exact same coloring. QuickQuartet, on the other hand, identifies those nodes as being identical bipartitions and retrieves a previously calculated and saved down vector. This eliminates the need to perform the traversal of any of the target trees at that specific source tree inner node. Thus, by saving the down vectors associated with bipartitions (color strings) a t tree traversals can be eliminated every time a common bipartition is found.

A map (implemented as a hash table) keyed by the color string is used to store the down vector associated with the coloring. The hash key is generated by a standard universal hashing algorithm operating on the color string. Prior to performing the traversal of a target tree, the hash table is queried using the current color string. If the down vector has already been computed for the color string, it is retrieved from the hash table and the set of target tree traversals is not performed. If the color string is not in the hash table, the down vector is computed and inserted in the hash table.

While the identification of shared bipartitions using color strings is responsible for the time performance improvement in QuickQuartet, there was an additional

improvement that to reduce memory utilization. It was noted that the life of the down vectors is known. The life of a down vector is bounded by the first and last trees in which the vector's associated inner node appears. Since inner nodes are not duplicated in the DAG, by keeping track of the last tree that uses an inner node, we can free down vectors after the last tree containing the node is processed as the source tree. As a result, the vector storage increases to an upper limit during the middle of the quartet distance computation but then decreases as vectors were released thus reducing the algorithm's overall memory requirements.

Although QuickQuartet outperforms QDist in practice, the overall best and worst case time complexities of the algorithms remain the same. In the worst case, there would be no bipartitions in common and there would be no common DAG nodes. A common DAG node is a sufficient but not a necessary condition for the existence of a common bipartition. Without any common DAG nodes, each source tree inner node will force the traversal of each other target tree with the resulting worse case time complexity of $O(t^2n^2)$ for t trees over n taxa. In the best case, all bipartitions would be in common (i.e., all t trees are identical) and discovered during the first and only traversal of a source tree. During this one source tree traversal, only the first target tree traversal set would produce unique results. No subsequent target traversals would be required as the counts for all DAG nodes would have already been computed. This yields the same best case time complexity as QDist of $O(t^2 + n^2)$.

QuickQuartet performance is significantly better than that of the original QDist program on biological data. Figure 7.4 shows the results of sets of tests on a biological dataset from the lab of Dr. Doug Soltis[72] comparing QuickQuartet with the original QDist algorithm. As can be seen QuickQuartet is 125 times faster than QDist which was unable to process more than 10,000 of the 33,306 trees in the dataset due to memory limitations on a 32GB supercomputer node.

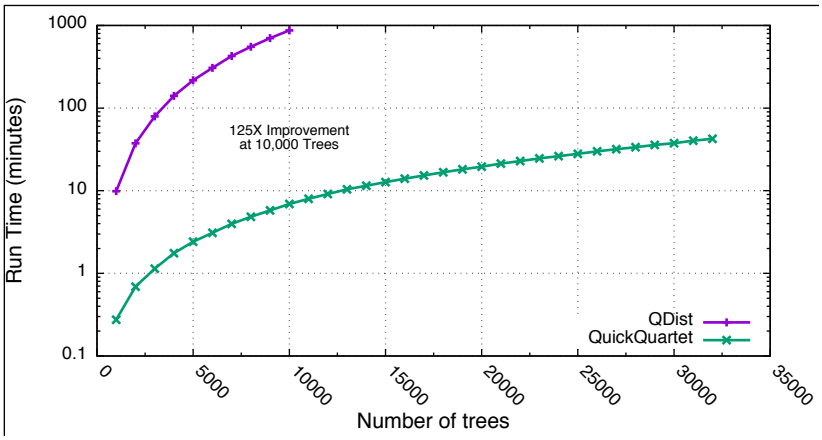
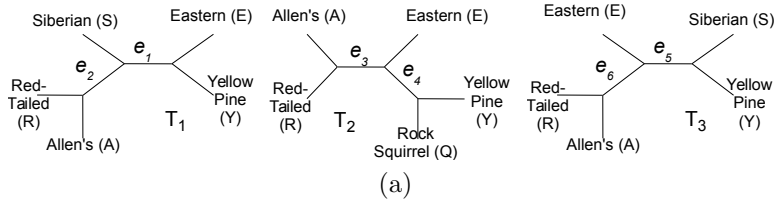


Figure 7.4: *QQ Algorithm Performance.* The performance of the *QQ* algorithm is compared to the old QDist algorithm. On a biological dataset of 33,306 trees[72] over 567 taxa QDist was only able to process 10,000 trees before running out of memory (32GB). *QQ* was able to process the entire 33,306 tree dataset. At the 10,000 tree level *QQ* performed 125 times faster than QDist.



(a)

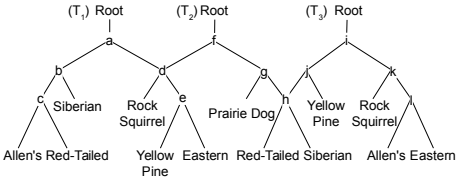
	T_1	T_3	T_3
T_1	0	4	2
T_2	4	0	4
T_3	2	4	0

(b)

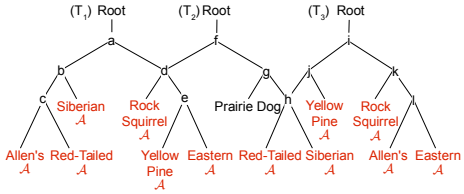
Figure 7.5: *Quartet Distance Between Three Dissimilar Trees.* (a) shows three dissimilar trees. (b) is the quartet distance matrix generated for the trees.

7.2.3 The Heterogeneous QuickQuartet Algorithm

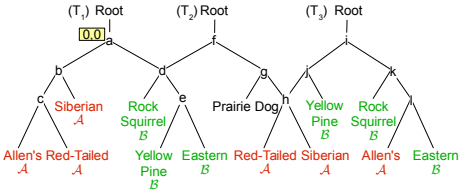
The algorithm of Stissing *et al.*[78] assumes that a coloring of the taxa relative to T_1 will be indicative of the coloring relative to the second tree, T_2. This is not the case when T_2 contains taxa not appearing in T_1 (heterogeneous trees). Figure 7.5 illustrates the computation of the quartet distance across three trees containing different sets of taxa.



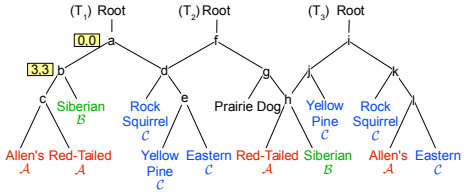
(a) All leaves set to the $\mathcal{NC}$ color



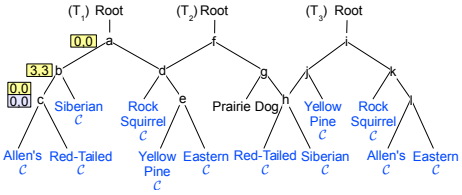
(b) All leaves in the source tree (T_1) have been set to $\mathcal{A}$.



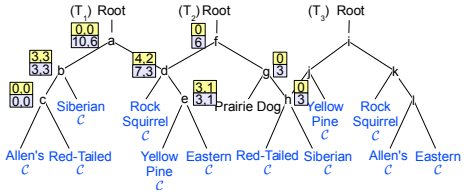
(c) The larger subtree of a has been colored $\mathcal{B}$ and taxa reachable from the upward edge of node a colored $\mathcal{C}$ (of which there are none). A traversal of each of the other trees is then performed and the counts at node a computed.



(d) The process was repeated for node b.



(e) The counts for the subtree rooted at node c are computed as sum of the counts at node c and its subtrees.



(f) All vectors after completion of the algorithm.

Figure 7.6: *QQ_{Het} Algorithm on Trees from Figure 7.2.*

In order to handle heterogeneous trees the impact of any taxa appearing in T_2 but not in T_1 needs to be reconciled. This is accomplished by the addition of an additional "no color" color or $\mathcal{NC}$. Additionally, since the algorithm of Stissing *et al.*[78] arbitrarily rooted the DAG at the first leaf encountered, there is no guarantee that this leaf appears in the full set of trees. To handle this condition, a dummy root leaf is inserted into each tree (see Figure 7.6). This dummy leaf is always set to

color $\mathcal{NC}$.

For the computation of the directed shared quartet count at a particular node the number of taxa with each of the three colors at each of the three edges is required. Within QuickQuartet the values for the downward edges were recursively computed. The color counts associated with the upward facing edge were computed by subtracting the counts for the downward edges from the total counts for each color.

But this count would be incorrect in the case where T_1 and T_2 do not contain the same set of leaves. As shown in Figure (a) to properly compute the counts for the upward subtree, all leaves in all trees are initially set to color $\mathcal{NC}$. Then, a depth-first traversal of T_1 excluding the added root is performed setting all leaves to the initial color $\mathcal{A}$ (see Figure (b)). Since the total number of leaves with each of the colors will not be known in this case it is necessary to get the number of taxa with each of the colors respective to T_2 prior to each traversal of T_2.

This requires a traversal of T_2 getting counts for each color prior to the main traversal of T_2 to compute the directed quartets in common. When the all pairs distances are being computed for a set of t trees, for each inner node in T_1 each of the other $t-1$ trees is considered T_2 in turn and a traversal of T_2 performed. The algorithm is illustrated in Figure 7.6.

As expected the performance of the QQ_{Het} algorithm was slightly worst within a constant factor than that of the QQ algorithms as shown in the Figure 7.7. The figure shows the ratio of the two algorithm's performance on subsets of biological datasets of 20,000 trees over 150 taxa and 33,306 trees over 567 taxa.

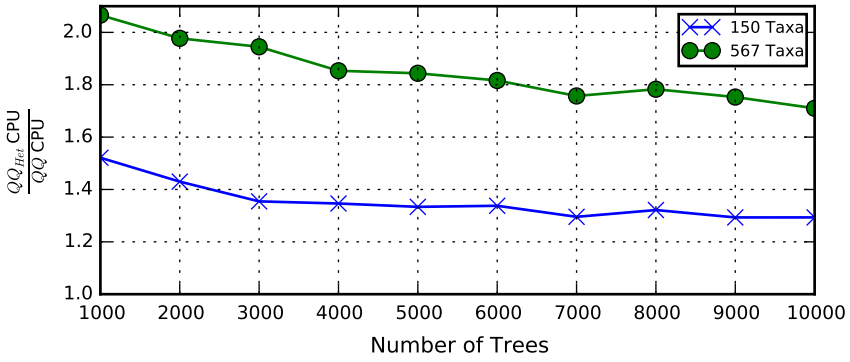


Figure 7.7: *QQh Algorithm Performance.* The ratio of the performance of the QQ and QQ_{Het} algorithms is shown for varying numbers of trees with two different biological datasets. As the number of trees increases the performance of the heterogeneous algorithm approaches that of the homogeneous algorithm.

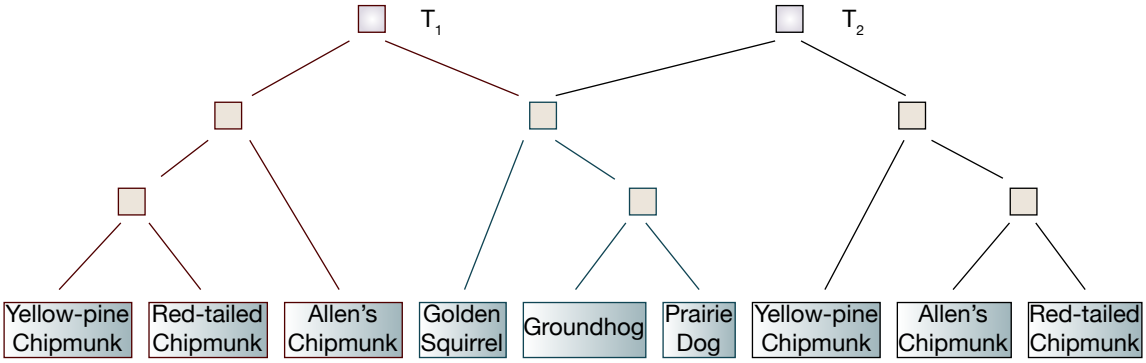


Figure 7.8: *The Directed Acyclic Graph Data Structure.* Two trees, T_1 and T_2 share a common subtrees that is represented by the MRCA of the Golden Squirrel, Groundhog and the Prairie Dog. As with the QQ DAG this common subtree is connected to both T_1 and T_2.

7.3 Data Structures

We based the data structures in AncestralAge on the DAG structure used in QQ. But, for divergence time both the species tree and the associated gene trees both need to be considered in the structure of the DAG. The current AncestralAge implementation also requires that each species tree include the same set of taxa

(homogeneous trees). This structure is illustrated in Figure 7.8.

7.3.1 Species Tree Representation

The construction of the DAG in *QQ* was based on the original algorithm of Mailund[78] which processed each tree through a set of steps:

1. The tree was parsed and internally an unrooted tree was built.

2. The unrooted tree was converted into a rooted tree using a common root.

3. The rooted tree was traversed and for each internal node encountered, the subtree associated with the root was used to search the current DAG. If the set of leaves in the subtree were found in a subtree of the DAG, the structures was compared. If the structures matched, the new tree was attached to the appropriate point in the DAG.

There were two issues with this approach; first, divergence time required rooted trees. Second, the search algorithm of Mailund could not be proven to find all common subtrees as the ordering of the leaves in the input could effect the search.

To address these issues a new algorithm was developed to generate the species tree DAG in AncestralAge. This algorithm processes each tree as follows:

1. The input is assumed to be a rooted binary tree.

   Methods are provided in AncestralAge to allow specification of an outgroup.

2. A depth-first traversal of the tree is performed.

   During traversal the set of leaves associated with each inner node of the tree is accumulated. When the inner node is seen during the backtracking phase of the traversal, a string is build to represent the subtree. The string is similar

to a newick string in that it is a parenthetical representation of the structure of a tree. To build the string the subtrees of a given node are ordered based on the collating sequence of the leaves in each of the subtrees. For example a subtree of ((d,b),(c,a)) would generate subtree keys of (b,d) and (a,c) for the subtrees that would ordered into a key for the node of (a,c),(b,d). The ordering is required to guarantee that common subtrees are identified regardless of any isomorphism. For example, within a newick string a specification of (a,c) and (c,a) define the same subtree with only a difference in ordering.

The string generated for each inner node is used as the key to a hash table. The value in the hash table is a pointer to an existing node in the DAG that matches the subtree. If a string is found in the hash table, the new tree is connected to the existing subtree in the DAG. Otherwise the string is added to the hash table as a new key.

3. Once all the trees have been loaded into the DAG the hash table is not longer required and is freed.

While the overall computational complexity of the DAG construction process is the same as that for QQ, $\mathcal{O}(n)$, the AncestralAge algorithm requires only a single traversal of the input as opposed to two traversals required in the QQ algorithm.

But, even if there are topological similarities between trees, there will potentially be differences in the edge lengths between the trees. In order to support these differences, each node in the DAG contains a vector of entries (tree nodes) each of which contains information associated with a single tree. For example, if an inner node appears in common between four species trees, there will be four entries in the tree specific node vector at the inner node. Within the tree node the following information is found:

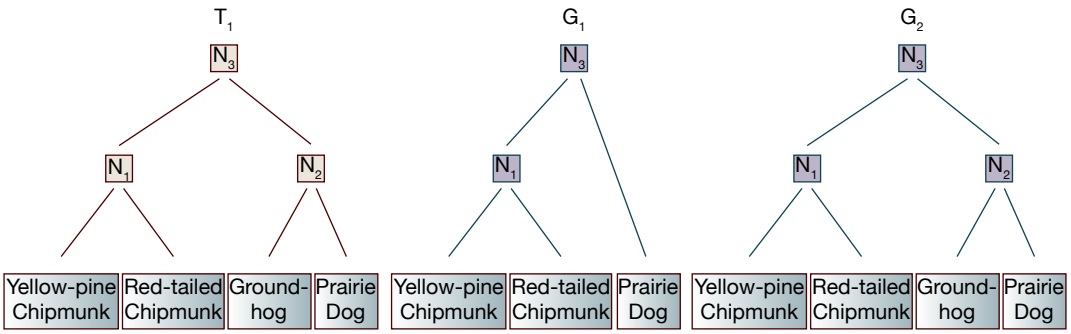


Figure 7.9: *Species and Gene Trees.* The tree on the left, T_1, contains 4 taxa with internal nodes. The first gene tree, G_1, only has DNA sequences for three of the taxa while the second gene tree, G_2, has DNA sequences for all four taxa.

- The age parameter for the node in the tree.
- A pointer to the prior of ages entry for the node (Sections 5.3.1).
- Gene tree pointers.

7.3.2 Gene Tree Representation

Each gene tree will be composed of a subset of the leaves in each species tree. The gene trees are represented as a set of overlay's on the species tree. The topology of a given gene tree is not directly specified in the input. What is specified is the multiple sequence alignment (MSA) associated with the gene tree. The species tree then provides the topological backbone for the gene tree. But since the gene tree MSA might (in practice frequently) not include all taxa in the input, the tree may be be missing various nodes corresponding to leaves and connecting nodes. The species tree and gene tree relationship is shown in Figure 7.9. In this figure, a subset of the Marmots of four species is represented in the species tree. But, DNA is only available for the first gene for three of the species. DNA is available for all four species in the second gene.

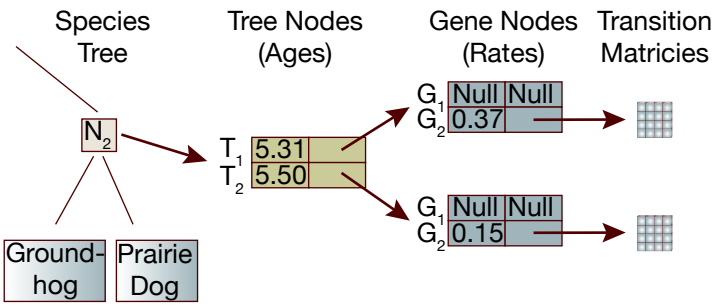


Figure 7.10: *Age and Rate Structures.* The relationship between the species tree nodes, the tree nodes and the gene nodes is shown. In this example, the subtree rooted at N_2 appears in both T_1 and T_2 so there are two entries in it's tree node vector with separate ages for each tree. But, N_2 does not appear in G_1 since there is no DNA sequence for the Groundhog in G_1. Therefore, the gene node vector for each tree has a null entry in the slot for G_1 while the entries for G_2 are both populated with the rates and TPM pointers associated with each tree.

To represent both the gene trees and allow independence between multiple species tree nodes that share common subtrees, the structures shown in Figure 7.10 are used. A node in the species tree (N_2 in this case) points to the previously discussed tree node vector. In this case, N_2 is common between two trees, T_1 and T_2. In addition to the age of the node in the associated tree, the tree node contains a pointer to a vector of gene tree nodes. There is an entry in this gene tree node vector for each gene tree. Since there is no DNA sequence data for the Groundhog in G_1 the entry is marked as missing (null). Since N_2 does exist in G_2, each gene node contains a rate value and transition probability matrix (TPM) for the ancestral edge associated with the gene tree node.

7.4 Dating Models

While the DAG structure discussed in the preceding sections allows for the representation of each species tree as an effectively independent entity with separate ages and gene trees, it also allows for models that leverage the common structures.

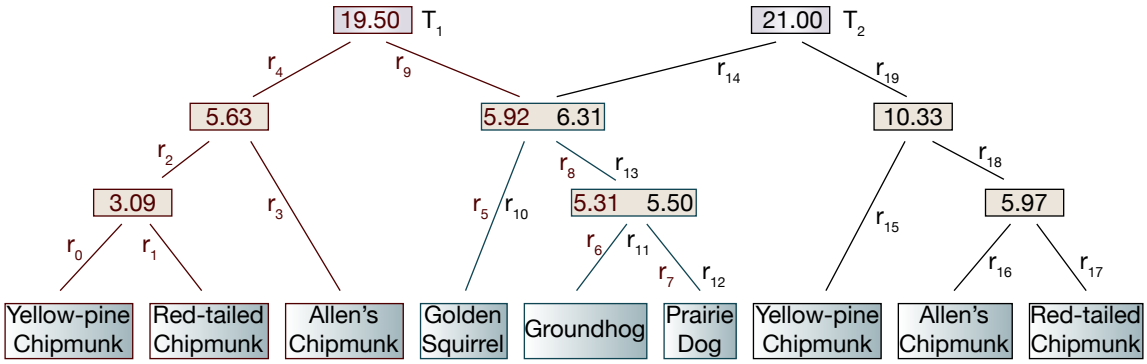


Figure 7.11: *The Independent Ages, Independent Rates Model.* In this model each tree has it's own set of edge rates and node ages. For the common subtree, each of the edges has two rates, one for each of the trees that include the subtree. The same is true of the node ages, each node in the common subtree has a pair of ages, one for each of T_1 and T_2.

In the following discussion, the number of parameters associated with each model is discussed. An MCMC step is an iterative process through the model parameters proposing new values for each parameter in turn. The computational complexity for an MCMC step is therefore function of the number of parameters as well as the complexities of the computation associated with each individual parameter proposal. Our approach to improving the performance of the multiple tree dating process has been to reduce the total number of parameters.

7.4.1 Independent Ages, Independent Rates

This is the base model wherein each species tree is effectively independent. The structures discussed in the preceding sections are fully populated as shown in Figure 7.11. Each of the t trees composed of n taxa has a unique age parameter for each non-leaf node in the model giving $\mathcal{O}(tn)$ age parameters.

Furthermore, each of the g gene trees is replicated for each of the t trees. For clock models other than the molecular clock, each node in a gene tree with the exception

of the root has a rate parameter that gives the mean rate across the edge leading to the parent of the node. For the molecular clock model, there is only a single rate parameter for each gene tree. Therefore, there are $\mathcal{O}(tgn)$ rate parameters exposed in this model.

There is no computational advantage to this model over a set of completely independent runs with one tree per execution.

To gain an understanding of the impact of the multiple tree models experiments were performed using a dataset of 567 angiosperms[13]. In this study a total of 10,339 DNA base pairs in 5 genes were used to produce 33,306 sampled trees.

For our experimental analysis QQ was used to generate a distance matrix of the 33,306 sampled trees including the published tree. The sampled trees were then ranked by their quartet distance from the published tree.

Sets of trees of sizes $t \in \{1, 5, 10...100\}$ were generated from this ranked list such that the first tree in every set was the published tree and the remaining trees were the $t-1$ trees "nearest" the published tree. Since divergence time inference had not been performed as part of the original study, a set of calibrations was synthetically estimated to allow dating.

As shown in Figure 7.12 , the time per iteration increases, as expected, linearly with the number of trees. Iteration times varied 21.89 seconds for the single tree up to 16 minutes 12.92 seconds for the 45 tree set. Another way of looking at this data is to consider the time required to perform an MCMC step on one tree from the set. The time required to take an MCMC step on a single tree was 21.89 seconds for the single tree set. This value decreased to 21.62 in the 45 tree set. This small decrease was not statistically significant.

Figure 7.12: *Iteration Times for the Fully Independent Model.* On the 567 taxa angiosperm dataset sets of trees varying in size from a single tree to 100 trees were run using the fully independent model. Times shown are those required for a single MCMC step. The chart illustrates the linear scaling expected from this model.

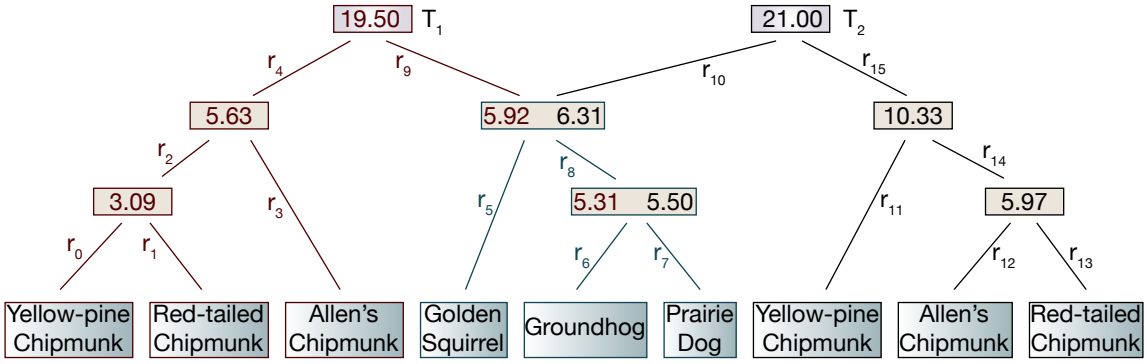


Figure 7.13: *The Independent Ages, Common Rates Model.* In this model each edge in the common subtree has only a single rate (r_5, r_6, r_7 and r_8). Each inner node in the common subtree still has two independent ages corresponding to trees T_1 and T_2.

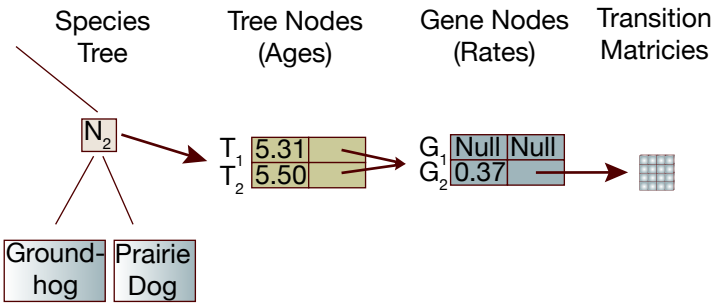


Figure 7.14: *Age and Rate Structures for the Common Rates Model.* To support the common rates model the existing structures are modified such that only one vector of gene nodes is used. Each of the tree nodes points to this single, common node.

7.4.2 Independent Ages, Common Rates

In this model, the ages of each node in each species tree are allowed to be independent. But, the rates associated with the edges in the gene trees for common subtrees are considered to be the same across trees. This case is illustrated in Figure 7.13. The hypothesis in this case is that even if structural differences in the overall species tree will impact the ages of the nodes, within a common gene subtree, an edge is truly common and would therefore only have a single rate regardless of the species tree.

To support this model the structures from Figure 7.10 are modified as shown in Figure 7.14. For common subtrees a single vector of gene tree nodes is instantiated at each common species tree node. Each of the tree nodes is then pointed to this common vector.

As with the fully independent model, each of the t trees composed of n taxa has a unique age parameter for each non-leaf node in the model giving $\mathcal{O}(tn)$ age parameters.

But, the number of rate parameters is reduced by the number of common subtrees found. In the worst case where the are no common subtrees, the number of rate

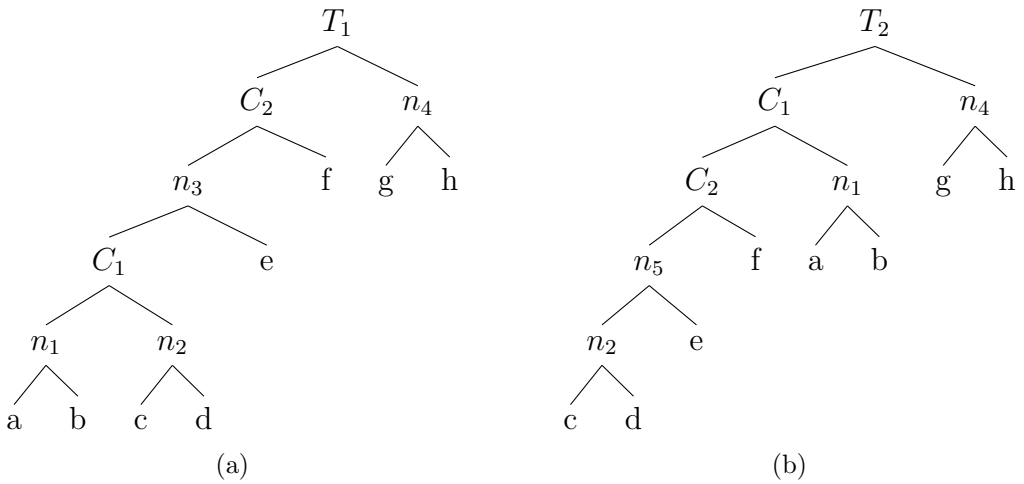


Figure 7.15: *Trees with Conflicting Calibrations.* Two calibrations are specified; C_1 is believed to be the ancestor of existing taxa a and c while C_2 is the ancestor of existing taxa e and f. In tree T_1 this places the calibration node C_2 closer to the root than C_1 implying that C_2 must be older than C_1. But in T_2 the movement of the subtree containing taxa a and b causes C_1 to appear closer to the root than C_2 creating a conflict in the specification with T_1.

parameters remains the same as in the fully independent model $\mathcal{O}(tgn)$. In the best case, where the trees share a 100% of the structure, the number of rate parameters reduces to $\mathcal{O}(gn)$ since each rate for each gene tree node will be shared across all species trees.

As with the fully independent model, QQ was used to generate a distance matrix of the 33,306 sampled trees and the published tree from the study. The sampled trees were then ranked by their quartet distance from the published tree.

The same sets of trees as were used in the fully independent model were once again tested, this time using the independent ages, common rates model. It was discovered that specification of calibrations across multi-tree sets caused various conflicts in the calibration dates that restricted the number of trees that could be dated, even with small numbers of calibrations (< 10).

Consider the trees in Figure 7.15. There are two calibrations specified. C_1 is believed to be the extinct ancestor of taxa a and c. C_2 is believed to be the extinct ancestor of e and f. In T_1 the topology is such that C_1 will be farther from the root than C_2 so the calibration specification indicates $C_1 <= C_2$. But, given the specification of $C_1 <= C_2$ there is a conflict in T_2. In this case C_1 will, due to it's morphological specification, appear closer to the root than C_2. Now the specification of $C_1 <= C_2$ will be in conflict with the topology of the tree. While this is an obvious conflict, it is our belief that there are a number of other combinations that will also generate conflicts. The full development of the conflict set and development of methods to handle these conflicts are outside of the scope of this work.

For the purpose of this work the set of conflicts was managed such that we were able to date up to 65 trees using the independent ages, common rates model.

Figure 7.16 shows the results using set sizes of $\{1, 5, 10...65\}$. Using the independent ages, common rates model, the impact of common subtrees can be seen. The second set of points in Figure 7.16 shows the same set of trees processed using the independent ages/common rates model. In this case the iteration times also appear to increase linearly, but at a much slower rates reflecting the commonality among the trees. It should be remembered that these trees were selected to all be "close" to published tree, so it is expected that a large set of common subtrees would be found. Iteration times for this model with a single tree are the same 21.89 seconds as for the full independent model. But this value only increases to 9 minutes 13.60 seconds at 65 trees, an improvement of 61.0%. Looking at the time required to perform an MCMC step on one tree from the set, the initial 21.89 second time for the fully independent model decreased to 8.52 seconds at 65 trees.

It had been expected that the percentage of improvement would increase with the number of trees. While not apparent from Figure 7.16, there was a continuous

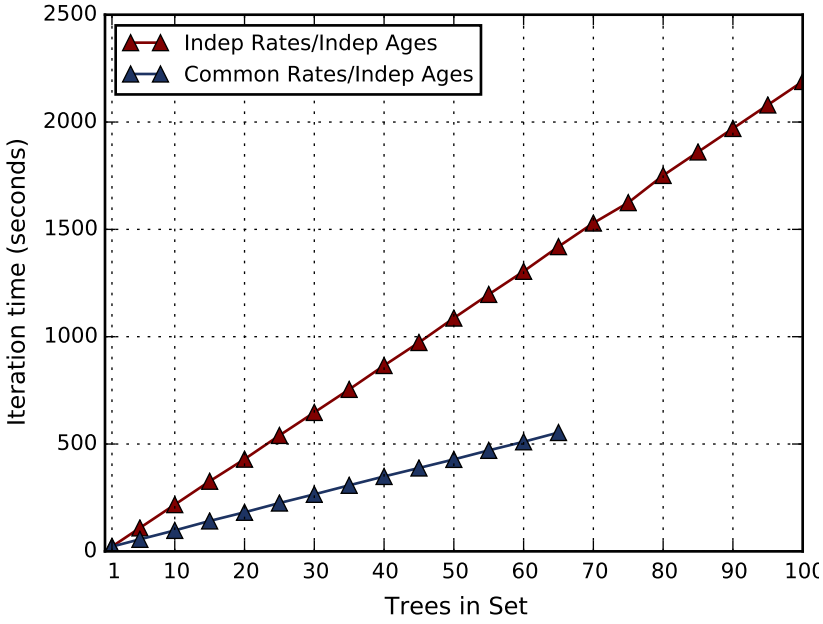


Figure 7.16: *Iteration Times for the Independent Ages, Common Rates Model.* On the 567 taxa angiosperm dataset, sets of trees varying in size from a single tree to 100 trees were run using both the fully independent and independent ages/common rates models. Times shown are those required for a single MCMC step. The chart illustrates the increasing advantage seen from the reduction in the number of rate parameters.

increase in the percentage of improvement. With the set of 5 trees, the improvement was 49.1%. As the number of trees was increased, this improvement also increased at every step to the aforementioned 61.0% improvement for the 65 tree dataset.

We hypothesize that since the trees selected were all "close" to each other the common portions of the tree were nearly the same. We believe that were the set of trees to be chosen by another method (e.g. random selection) a different curve would be seen.

7.4.3 Common Ages, Common Rates

This model constrains the ages as well as the rates for common subtrees. The idea here is that a common subtree could be considered as phylogenetically isolated.

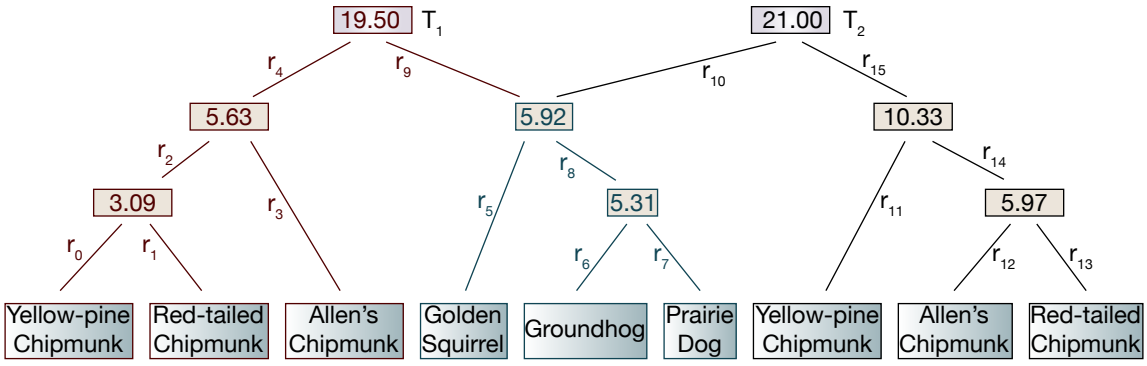


Figure 7.17: *The Common Ages, Common Rates Model.* In this model there are is only a single age for each of the inner nodes in the common subtree.

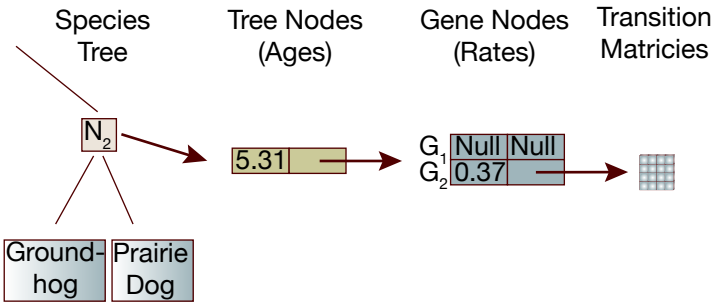


Figure 7.18: *Age and Rate Structures for the Common Ages and Rates Model.* To support the common ages and rates model the existing structures are modified such that only one tree node is instantiated. Each of the species nodes points to this single, common tree node.

The common ancestor of the taxa in the subtree is considered to root a tree that diverged at one specific time only, not at different times for different species trees. This model is illustrated in Figure 7.17.

The structures from the common rates model are further simplified as shown in figure 7.18. One tree node is associated with each node in the species tree. This single tree node in turn points to a single gene node vector containing the evolutionary rates.

In this model the number of rate parameters is the same as for the independent

ages, common rates model with best and worst case parameter counts of $\mathcal{O}(gn)$ and $\mathcal{O}(tgn)$ respectively.

Once again, for the age parameters, if there are no common subtrees the number of age parameters will be the same as the number for the fully independent model, $\mathcal{O}(tn)$. In the best case where the trees fully share the same structure, the number of parameters would reduce to just the number of non-leaf nodes $\mathcal{O}(n)$.

This model suffered from the same set of issues with calibrations as the independent ages, common rates model. While the model did successfully compute the ages, the ages inferred were, in many cases unsupportable. Using the fully independent model results as the baseline, over 30% of the node ages at the 65 tree level for the common ages, common rates model fell outside of the 95% CI on the baseline results. This variation did not occur with the independent ages, common rates model where less than 8% of the ages in the 65 tree set were outside of the 95% baseline CIs.

We hypothesize that the algorithm for the prior of ages may not properly handle this model. In the case of a common subtree that, for example, contains a calibration, how should the age prior vector and conditional density vector structures be organized? Should these structures remain separate across the trees?

Figure 7.19 shows the results using set sizes of $\{1, 5, 10...65\}$. The third set of points in Figure 7.19 shows the same set of trees processed using the common ages, common rates model. In this case, the iteration times also appear to increase linearly but at an even slower rates reflecting the reduce set of model parameters. Iteration times for this model with a single tree are the same 21.89 seconds as for the full independent model. But this values only increases to 2 minutes 35.06 seconds at 65 trees, an improvement of 89.1% over the fully independent model. Looking at the time required to perform an MCMC step on one tree from the set, the initial 21.89 second time for the fully independent model decreased to 2.39 seconds at 65 trees.

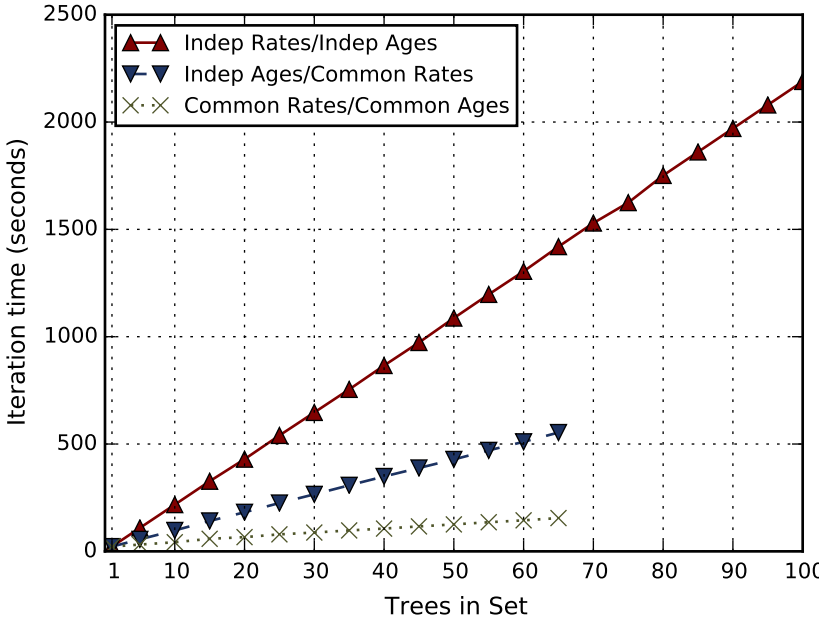


Figure 7.19: *Iteration Times for the Common Ages, Common Rates Model.* On the 567 taxa angiosperm dataset, sets of trees varying in size from a single tree to 100 trees were run using all three models. Times shown are those required for a single MCMC step. The chart illustrates the increasing advantage seen from the reduction in the number of age parameters.

7.5 Summary

In order to efficiently data multiple species trees in a single run, three models have been presented.

1. The independent ages, independent rates model provides the maximum independence between the individual trees but with no performance advantage over separate dating of the trees.

2. The independent ages, common rates model posits that rates for a edge in a gene tree are the same regardless of the final species tree topology or ages. This model provided statistically value results with a significant improvement in performance.

3. The common ages, common rates model assumed that the ages of nodes in common subtrees would always be the same. This is in addition to the assumption that the rates for an edge in a gene tree in a common subtree would always be the same.

In all models it was found that the specification of the calibrations was challenging over large sets of trees. Calibrations in AncestralAge are created on the MRCA of sets of taxa specified by the user. Specification of this set, in cases there the tree topology varied around the taxa specified, frequently caused situations where ages either could not be estimated or would have been illogical.

Using the 567 taxa angiosperm dataset of Soltis *et al.* we demonstrated a 89.1% improvement in CPU time for a set of 65 trees using the common ages, common rates model. While this is certainly significant, it is important to note that the time required for an MCMC iteration is still approximately 2.5 minutes making an MCMC run of 100,000 steps require just under seven months to complete.

To gain understanding of the impact of all three models on the MCMC process, we determined the number of MCMC parameters for each of the models varying the number of trees. Since MCMC step processing iteratively proposes new values for each of the MCMC parameters, we expected to see the relationship between the number of parameters and the time for each MCMC step. The MCMC parameter counts for each of the models with varying numbers of trees is shown in Figure 7.20.

As expected, the number of parameters for the fully independent model increased linearly with the number of tree. But, of particular interest, parameter counts for the common rates/independent ages model increase much more slowly that would have been expected from Figure 7.16. In fact at 65 trees, there were 80.0% fewer parameters but only a 61.0% improvement in step time. We are of the opinion that

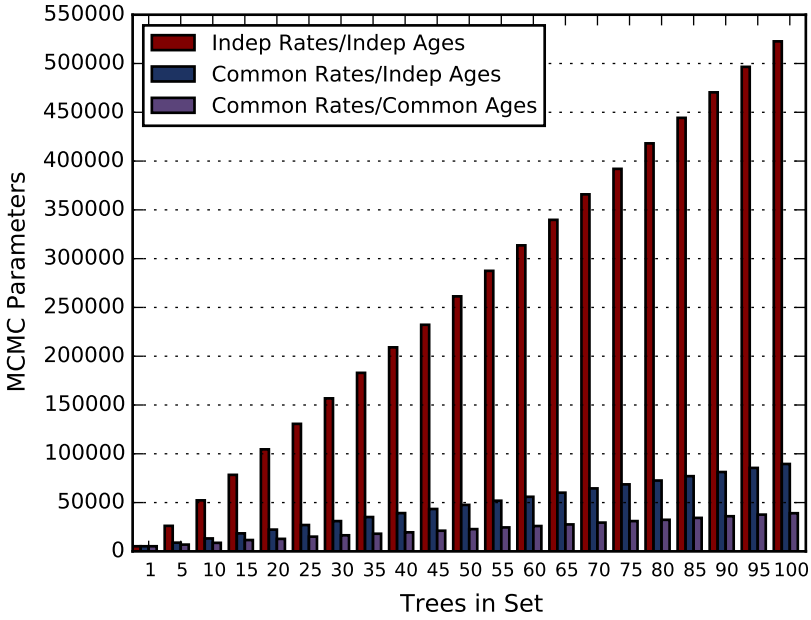


Figure 7.20: *MCMC Parameter Counts for the Multiple Tree Models.* The number of MCMC parameters for each of the three models is shown for varying numbers of trees drawn from the 567 taxa angiosperm dataset.

this reflects the cost of the set of rate parameter proposals (and the computation of the prior of rates) relative to the overall cost of an MCMC step. The rate parameter proposals comprise the largest group of parameters. But, age and nuisance parameters remain and will continue to scale with the number of trees even as the portion of the MCMC step time associated with rate parameter proposals decreases. With this decrease in the time spend computing rate parameter proposals, the portion of the CPU time spent computing age parameter proposals increases.

In the case of the common ages, common rates model, the improvement in CPU time did more closely correlate to the number of parameters. CPU time decreased by 90.8% at the 65 tree level and parameter counts decreased by 89.1% compared to the fully independent model. We believe this indicates that computation of rate and age parameter proposals consume far more of the total CPU than the remaining

nuisance parameters.

8 ANALYSIS

In this chapter we provide an analysis of the AncestralAge from both the perspective of the dates inferred and the performance of the algorithms.

8.1 Model Validation

The objective of this research was not to develop new statistical models for divergence time. The objective was to develop new algorithms using the existing statistical model of MCMCTree. We considered it critical that the output of AncestralAge was both consistent with MCMCTree and reproducable[8]. To this end we provide a comparison of the results from comparable executions of AncestralAge and MCMCTree.

Three datasets were used, two that are provided as examples with the MCMCTree program and our Family Scuridae dataset. Each of the datasets was run with both AncestralAge and MCMCTree using the same parameters (evolutionary model, MSA, input tree and prior hyperparameters). ForMCMCTree the datasets were processed using both the approximated and exact likelihood algorithms.

Conventional comparisons of phylogenetic trees rely on the comparison of likelihood values between the trees. In order for the likelihood values to be comparable they must be generated using identical methods. Typically, likelihood values vary significantly between phylogenetic programs even if the programs produce identical trees. These differences are a consequence of the numerical methods used for the computation. This situation was found to also be the case between AncestralAge and MCMCTree. While the actual dates inferred were very close in all cases, the likelihood values varied significantly.

So, for purposes of this validation process, the results for each of the datasets

were compared based on an analysis of the dates returned by the two programs. Differences in dates between the two programs were normalized by the mean of the two dates giving an indication of the relative difference.

$$d = 2(d_1 - d_2)/(d_1 + d_2) \tag{8.1}$$

It was expected that the differences would be within a small tolerance, ϵ. Additionally, it was hypothesized that the dates returned by AncestralAge would fall within the 95% credibility interval as returned by MCMCTree and the dates from MCMCTree would fall within the 95% credibility intervals returned by AncestralAge.

- Dataset 1 - DatingSoftBounds

  This dataset was provided as an example in the PAML distribution and was extensively analyzed by Yang, *et al.*[94]. There are 7 taxa in the tree with 3331 DNA sites in one gene.

  The dates returned by MCMCTree for both the approximated and exact likelihood algorithms were compared with the results returned by AncestralAge. In all cases the results from AncestralAge were within $\epsilon = 0.01$ ($.99 < d < 1.01$) of the results returned by the MCMCTree exact likelihood algorithm and within $\epsilon = 0.015$ ($.985 < d < 1.015$) of the results returned by the MCMCTree approximated likelihood algorithms.

  In all cases the dates returned by AncestralAge were within the 95% credibility interval returned by MCMCTree and the dates returned by MCMCTree fell within the 95% credibility interval returned by AncestralAge.

- Dataset 2 - TipDate.FluH1

  This dataset was provided as an example in the PAML distribution and was

also extensively analyzed by Yang, *et al.*[74]. There are 289 viral taxa in the tree with 1710 DNA sites in one gene. In the original analysis, tip dating was used. But, for our purposes, conventional, branch, dating was applied.

The dates returned by MCMCTree for both the approximated and exact likelihood algorithms were compared with the results returned by AncestralAge. In all cases the results from AncestralAge were within $\epsilon = 0.005$ ($.995 < d < 1.005$) of the results returned by the MCMCTree exact likelihood algorithm and within $\epsilon = 0.015$ ($.985 < d < 1.015$) of the results returned by the MCMCTree approximated likelihood algorithms.

In all cases the dates returned by AncestralAge were within the 95% credibility interval returned by MCMCTree and the dates returned by MCMCTree fell within the 95% credibility interval returned by AncestralAge.

- Dataset 3 - Scuridae

  This dataset of 80 taxa with 7618 total DNA sites in 5 genes was analyzed by the author as part of the Texas A&M Quantitative Phylogenetics class.

  The dates returned by MCMCTree for both the approximated and exact likelihood algorithms were compared with the results returned by AncestralAge. In all cases the results from AncestralAge were within $\epsilon = 0.01$ ($.99 < d < 1.01$) of the results returned by the MCMCTree exact likelihood algorithm and within $\epsilon = 0.03$ ($.97 < d < 1.03$) of the results returned by the MCMCTree approximated likelihood algorithms.

  In all cases the dates returned by AncestralAge were within the 95% credibility interval returned by MCMCTree and the dates returned by MCMCTree fell within the 95% credibility interval returned by AncestralAge.

8.2 Performance Analysis

To understand the performance on AncestralAge a set of synthetic data was generated. This approach allowed the parameters associated with the theoretical computation complexity to be varied and their impact verified.

8.2.1 Methods

A tree generator, *TGen*, capable of constructing sets of trees with varying numbers of taxa, was developed to facilitate testing. The generator allowed specification of the similarity between the trees in terms of the percentage of quartets that would appear in common across the trees (the quartet consensus). The generator also allowed the user to specify how balanced (or unbalanced) the resulting trees should be.

The trees produced by *TGen* were used as input to the seq-gen program[66] to produce DNA sequences varying the lengths of the sequences and the numbers of partitions (genes) of the data. All sequences were generated using the HKY[81] evolutionary model with a transition/transversion ratio, κ, of 2 and gamma variation among sites using 4 discrete categories.

All sets were processed by AncestralAge. Attempts to use MCMCTree were unsuccessful as even the smaller tests produced extremely long runs times (days to months).

Each test was allowed to run for 100 MCMC steps. Performance information was obtained through API calls to the Linux kernel as well as the through the PAPI performance library[23]. We hypothesized that 100 MCMC steps would be sufficient to overcome the impact of any initialization and termination. In all cases the CPU time required for initialization and termination outside of the MCMC process itself was less than 1% of the total CPU.

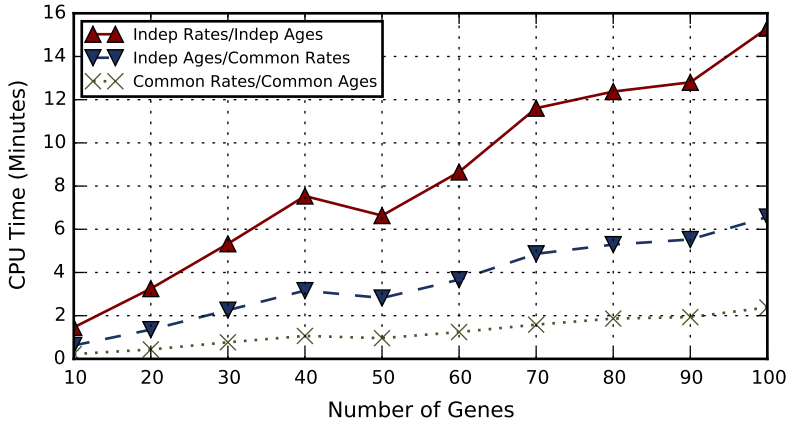


Figure 8.1: *CPU Times for Varying Numbers of Genes.* The number of genes, $l \in \{10, 20...100\}$, was varied for a constant sequence length of 2000 sites per gene. Each set contained 10 trees of 100 taxa each.

Values reported are the average times required for a single MCMC step. All tests were run on the Texas A&M University Brazos high performance cluster (http://brazos.tamu.edu). Each node on the cluster consists of dual 2.5GHz Intel quad core processors and 32GB of memory.

8.2.2 Varying Numbers of Genes

Sets of trees with $l \in \{10, 20...100\}$ genes were generated. Each gene had independent data but a constant sequence length of 2000 sites. Each set contained 10 trees with 100 taxa per tree.

Based on the computational complexities of the components of the MCMC step it was believed that the performance would be linear with respect to the number of genes. In Figure 8.1 the results of the running each of the sets on each of the three multi-tree models is shown. The overall curve for each model does show a generally linear increase for each of the models. The performance of the common rates and common ages models tracked closely with the fully independent model. Since the number of trees in each set was constant (although the trees themselves were not the

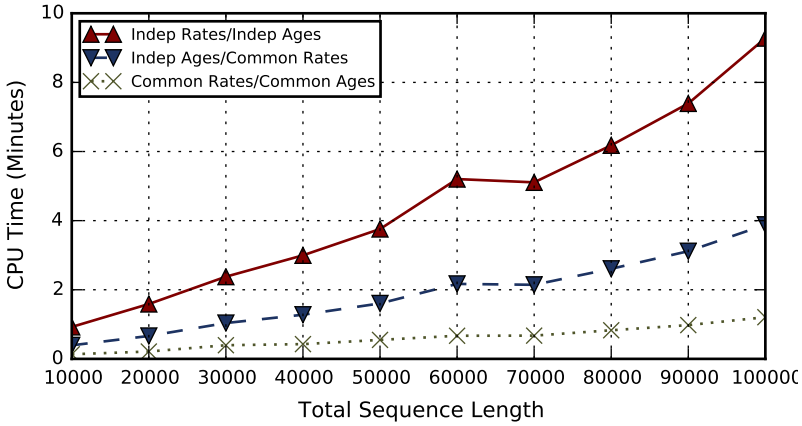


Figure 8.2: *CPU Times for Varying DNA Sequence Lengths.* The total number of sites across 10 genes was varied from 10000 to 100000 in increments of 1000. Each set contained 10 trees with 100 taxa. The times displayed are the average for MCMC steps.

same) the level of sharing would be relatively constant.

The curves for each of the models show unexpected behavior from 40 to 60 genes with the value for 40 genes higher than expected and the values for 50 and 60 genes lower than expected. There are several possible explanations for this behavior. We believe the discontinuity is most likely an artifact of the synthesis process although it is also possible that some other issue occurred (e.g. memory constraints in AncestralAge). Interestingly the impact of this discontinuity decreased with the common rates and common ages models. Further experimentation will be required to understand this discontinuity.

8.2.3 Varying Sequence Lengths

Sets of trees with varying DNA sequence lengths, $s \in \{1000, 2000...10000\}$, sites per gene were generated. Each set contained 10 trees with 100 taxa and 10 genes per tree.

As with the number of genes, it was anticipated that the performance would be

linear with respect to the number of DNA sites. In Figure 8.2 the results for the three multi-tree models are shown. While all three curves appear linear at the start, as the number of sites increases above 6000 the values appear to increasing at a greater than linear rate (possibly quadratic).

To understand this behavior we did an analysis of the site data. We found that for shorter sequences (particularly $s < 4000$) the lengths of the subtree site compressed vectors was generally less than the theoretical limits ($\min(5^n, s)$) but that as the length of the sequences increased this shifted such that by the time we had reached 10,000 sites per gene the majority of the vectors were of the maximum lengths. We therefore are of the opinion that if the number of sites was increased further, all the sites would be at their limits and the performance increase would be linear from that point on.

As with the test varying the number of genes, there was a discontinuity found in the middle of the range. In this case the discontinuity only appeared to impact one set of points (the 6,000 sites/gene set) and we are of the opinion that this is an artifact of the generation process.

8.2.4 Varying Numbers of Taxa

Sets of trees with varying numbers of taxa, $n \in \{20, 40...200\}$, were generated. Each set contained 10 trees with 10 genes of 200 sites each.

In this case, the computational complexity indicated that the performance should be quadratic in the number of taxa. For this experiment, the results as shown in Figure 8.3 could support either linear or quadratic models. With the fully independent model, the data could support a quadratic model but with a modest constant multiplier. For the common rates and common ages models, the curves appear linear but we would argue that with further increases in the size of the data the quadratic

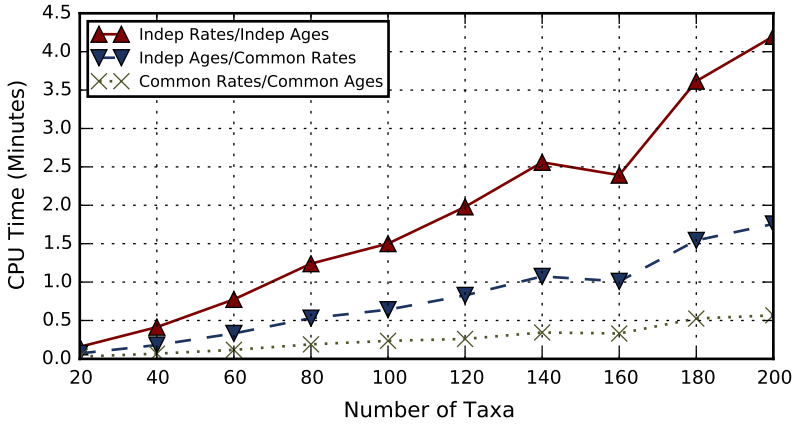


Figure 8.3: *Step CPU Times for Varying Numbers of Taxa.* The number of taxa n, was varied from 20 to 200 in increments of 20. Each set contained 10 trees with 10 genes of 200 sites each. The times displayed are the average for MCMC steps.

nature of the complexity would become evident.

8.2.5 Varying Numbers of Trees

Sets with varying numbers of trees, $t \in \{10, 20...100\}$, were generated. Each tree had 100 taxa and 10 genes of 200 sites.

As shown in Figure 8.4 the performance of all three models was, as expected, linear in the number of trees. It was anticipated that the performance benefit from the common ages and rates models would increase. What was of particular interest was the performance increase with the common ages model over the other models as the number of tree increased. As previously discussed, with the primates data the performance increase with the common ages model was approximately 2x over the common rates model. But with the synthetic data, at 200 taxa, the increase was closer to 6x. We hypothesize that there was a higher degree of subtree sharing in the synthetic data sets. It would be interesting to generate additional sets of synthetic data varying the level of quartet consensus and observing the change in performance at the different levels.

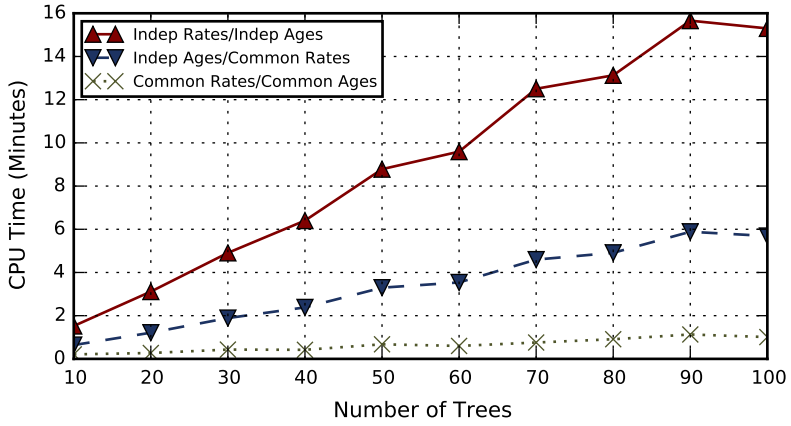


Figure 8.4: *CPU Times for Varying Numbers of Trees.* The number of trees in a set was varied from 10 to 100 in increments of 10. Each tree had 100 taxa and 10 genes of 200 sites each. The times displayed are the average for MCMC steps.

8.2.6 Summary

We have shown experimental validation for the theoretical computational complexity of AncestralAge. Varying the key components of the complexity; the numbers of trees, genes, taxa and sites we showed that the performance was within expected values.

When we varied the number of sites, we observed what appeared to be a greater than linear increase in CPU time. We believe this is a result of the nature of the subtree site compression algorithms and that linear scaling would be demonstrated with larger sample sizes.

When we varied the number of taxa, we observed a much closed to linear scaling than would have been expected. We believe that the complexity is in actuality quadratic but that the constant multiplier is small and it would take significantly larger sample sizes to demonstrate quadratic behavior.

There were several discontinues noted in the data. Further investigation will be required to understand if these are artifacts of the synthetic data generation process

or if there are aspects of the AncestralAge algorithms yet to be uncovered.

9 CONCLUSIONS AND FUTURE WORK

Phylogenetic divergence time has proven to be a rich area for computational research. When we started our work with the MCMCTree program it became apparent that this program had been developed as a test bed for statistical theory relating to divergence time. Issues of computational performance were secondary (or worse). The other significant application, Beast, took a different approach to the problem hypothesizing that combining phylogenetic inference with divergence time inference would produce a better model. In some, small, cases this has been proven [41] [36] but this combination also limits the ability of the biologist to independently estimate the phylogenetic tree and the dates for the nodes of the tree.

We found that neither of these programs scaled to the hundreds of trees and thousands of base pairs of DNA that are being used in modern studies. We were able to improve the performance of MCMCTree through a new approximated likelihood algorithm. But, this did not help in cases where the molecular clock was violated and the approximation model failed.

In order to provide a framework for further research into the process of phylogenetic divergence time, we developed the AncestralAge framework. This framework has facilitated research into new algorithms in a number of areas relating to divergence time including likelihood computation and Bayesian prior computation.

In order to improve the efficiency and performance of the dating process we first focused on developing new algorithms for dating individual trees. Our subtree site compression algorithm for the computation of phylogenetic likelihood has demonstrated a 90.1% improvement over existing methods. Our incremental prior of ages algorithm has shown as better than 100x improvement over the comparable compu-

tation in MCMCTree.

The modular design of the AncestralAge framework has also allowed research into alternative parallel methods for likelihood computation. Our research into a highly parallel GPU algorithm demonstrated that our high optimized subtree site compression algorithm did not translate well as the level of parallelism possible had been reduced as the overall amount of work required had also been reduced.

One of the significant questions we investigated was the sensitivity of the model to changes in the set of calibrations. With the development of calibration jackknifing we were able to develop a novel new support measure for the ages of the nodes. We demonstrated the use of this measure on the primates dataset. With previous methods (MCMCTree) it required 14 days to date a single tree with the approximated likelihood algorithm. We were able to date the base tree and 20 replicates using out exact likelihood algorithm in just under 10 days on a single cluster node.

We hypothesized that dating multiple trees prior to the consensus would allow for reduced bias and variance in the final tree. We produced new, high performance, algorithms for the computation of the quartet distance across all pairs of a set of trees allowing selection of sets of trees of interest for dating. While the fundamental questions around multiple tree dating remain outstanding we have developed models for sharing structures across multiple trees that will, in the future, facilitate research into this question.

9.1 Future Work

Frequently research will generate new sets of questions for study. This work has been no exception. There are a number of areas that are worthy of additional study including:

- The common rate and common age multiple tree models.

Additional research will be required to best determine how to specify calibrations in the multiple tree models. How should conflicting calibrations be handled and what are the implications of that handling?

The prior of ages algorithm will require additional research to understand how to manage the computation of the prior when the sets of calibrations may vary between the trees.

- Development of more efficient parallel GPU algorithms.

  The initial GPU implementation was less efficient than the serial version of the same algorithm. We found that through the subtree site compression algorithm the width of the calculations sent to the GPU was not sufficient to counter the overhead of dispatching the work to the GPU. We are of the opinion that it will be possible to overcome these issues by computing the likelihood for multiple nodes in parallel. Of particular interest will be understanding the threshold where the GPU width is sufficient to overcome the cost of dispatching work to the GPU.

- Use of AncestralAge to study the impact of performing the consensus after dating.

  While it will be desirable to further improve the performance and capacity of the multiple tree dating capability, there is addition research that can be done with the numbers of trees that can currently be dated.

  It will be interesting to generate consensus trees from the set of dated trees generated and compare the results with the published consensus trees.

- Additional experimentation using of AncestralAge to gain further understand into the impact of the support measure for node ages.

We analyzed a single tree from one study. But, there is considerably more research that can now be done to understand the impact of the numbers, types and distributions associated with fossil calibrations on the divergence time process.

- Extension of the AncestralAge framework to support additional capabilities.

  There are a number of additional capabilities that would prove valuable including:

  - Support for dating using Amino Acid sequences.
  - Extension of the evolutionary models to include the percentage of invariant sites.
  - Support for "Tip" dating where the dates for leaves can be specified. This is of particular interest in virology.

9.2 Conclusion

We have demonstrated new algorithms for phylogenetic divergence time that push the boundaries of our knowledge in this area. Our hope is that this work will one one hand provide biologists a new set of tools for phylogenetics and, on the other hand, provide computer scientists with a new framework and algorithms for research into computational biology.

REFERENCES

[1] Wikimedia User Asiir. A black-tailed prairie dog. `http://commons.wikimedia.org/wiki/File:Prairie_Dog_Washington_DC_1.jpg`, 2007. Licensed under the Creative Commons Attribution-Share Alike 2.5 Generic license.

[2] Daniel L Ayres, Aaron Darling, Derrick J Zwickl, Peter Beerli, Mark T Holder, Paul O Lewis, John P Huelsenbeck, Fredrik Ronquist, David L Swofford, Michael P Cummings, Andrew Rambaut, and Marc A Suchard. Beagle: an application programming interface and high-performance computing library for statistical phylogenetics. *Systematic Biology*, 61(1):170 – 173, 2012.

[3] Halfdan Baadsgaard, GL Cumming, RE Folinsbee, and JD Godfrey. Limitations of radiometric dating. *Geochronology of Canada: Royal Soc. Canada Spec. Pub*, 8:20–38, 1964.

[4] Dennis A. Benson, Ilene Karsch-Mizrachi, David J. Lipman, James Ostell, and Eric W. Sayers. Genbank. *Nucleic Acids Research*, 37(suppl 1):D26–D31, 2009.

[5] F. Blagojevic, A. Stamatakis, C.D. Antonopoulos, and D.S. Nikolopoulos. Raxml-cell: Parallel phylogenetic tree inference on the cell broadband engine. In *Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International*, pages 1–10, 2007.

[6] George EP Box and Norman R Draper. *Empirical Model-building and Response Surfaces.* John Wiley & Sons, 1987.

[7] Grant Brammer. *Algorithms for Searching and Analyzing Sets of Evolutionary Trees.* PhD thesis, Texas A&M University, 2014.

[8] Grant R. Brammer, Ralph W. Crosby, Suzanne J. Matthews, and Tiffani L. Williams. Paper mâché: Creating dynamic reproducible science. *Procedia Computer Science*, 4(0):658 – 667, 2011. Proceedings of the International Conference on Computational Science, {ICCS} 2011.

[9] Gerth Brodal, Rolf Fagerberg, and Christian Pedersen. Computing the quartet distance between evolutionary trees in time $O(n \log^2 n)$. In Peter Eades and Tadao Takaoka, editors, *Algorithms and Computation*, volume 2223 of *Lecture Notes in Computer Science*, pages 731–742. Springer Berlin / Heidelberg, 2001. 10.1007/3-540-45678-3_62.

[10] GS Brodal, R Fagerberg, and CNS Pedersen. Computing the quartet distance between evolutionary trees in time $O(n \log n)$. *Algorithmica*, 38(2):377–395, 02 2004.

[11] David Bryant. A classification of consensus methods for phylogenetics. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 61:163–184, 2003.

[12] David Bryant, John Tsang, Paul Kearney, and Ming Li. Computing the quartet distance between evolutionary trees. In *SODA '00: Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 285–286, Philadelphia, PA, USA, 2000. Society for Industrial and Applied Mathematics.

[13] J Gordon Burleigh, Khidir W Hilu, and Douglas E Soltis. Inferring phylogenies with incomplete data sets: a 5-gene, 567-taxon analysis of angiosperms. *BMC Evolutionary Biology*, 9(1):61, 2009.

[14] Patrick Cardinal, Pierre Dumouchel, Gilles Boulianne, and Michel Comeau. GPU accelerated acoustic likelihood computations. In *INTERSPEECH*, pages 964–967, 2008.

[15] Wikimedia User Cephas. Groundhog on Laval University campus, Quebec, Canada. http://commons.wikimedia.org/wiki/File:Marmota_monax_UL_04.jpg, 2013. Licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license.

[16] T.E. Cerling, J.M. Harris, B.J. MacFadden, M.G. Leakey, J. Quade, V. Eisenmann, and J.R. Ehieringer. Global vegetation change through the Miocene/-Pliocene boundary. *Nature*, 389(6647):153–158, 1997. cited By (since 1996)747.

[17] A Cornish-Bowden. IUPAC-IUB symbols for nucleotide nomenclature. *Nucleic Acids Res*, 13:3021–3030, 1985.

[18] David Roxbee Cox and David Victor Hinkley. *Theoretical statistics*. CRC Press, 1979.

[19] Ralph W Crosby and Tiffani L Williams. A fast algorithm for computing the quartet distance for large sets of evolutionary trees. In *Bioinformatics Research and Applications*, pages 60–71. Springer, 2012.

[20] Ralph W. Crosby and Tiffani L. Williams. High performance phylogenetic divergence time inference. In *Biotechnology and Bioinformatics Symposium*, Brigham Young University, Proro, Utah, December 2014.

[21] D. Darriba, A. Aberer, T. Flouri, T.A. Heath, F. Izquierdo-Carrasco, and A. Stamatakis. Boosting the performance of Bayesian divergence time estimation with the Phylogenetic Likelihood Library. In *Parallel and Distributed Processing Symposium Workshops PhD Forum (IPDPSW), 2013 IEEE 27th International*, pages 539–548, May 2013.

[22] Michael DeGiorgio and James H. Degnan. Robustness to divergence time underestimation when inferring species trees from estimated gene trees. *Systematic*

Biology, 63(1):66–82, 2014.

[23] Jack Dongarra, Kevin London, Shirley Moore, Phil Mucci, and Dan Terpstra. Using PAPI for hardware performance monitoring on linux systems. In *In Conference on Linux Clusters: The HPC Revolution, Linux Clusters Institute*, 2001.

[24] Mario dos Reis, Jun Inoue, Masami Hasegawa, Robert J Asher, Philip CJ Donoghue, and Ziheng Yang. Phylogenomic datasets provide both precision and accuracy in estimating the timescale of placental mammal phylogeny. *Proceedings of the Royal Society B: Biological Sciences*, 279(1742):3491–3500, 2012.

[25] Mario dos Reis and Ziheng Yang. Approximate likelihood calculation on a phylogeny for Bayesian estimation of divergence times. *Molecular Biology and Evolution*, 28(7):2161–2172, 2011.

[26] Mario Dos Reis and Ziheng Yang. The unbearable uncertainty of Bayesian divergence time estimation. *Journal of Systematics and Evolution*, 51(1):30–43, 2013.

[27] Alexei Drummond and Andrew Rambaut. BEAST: Bayesian evolutionary analysis by sampling trees. *BMC Evolutionary Biology*, 7(1):214, 2007.

[28] Alexei J. Drummond, Marc A. Suchard, Dong Xie, and Andrew Rambaut. Bayesian phylogenetics with BEAUti and the BEAST 1.7. *Molecular Biology and Evolution*, 29(8):1969–1973, 2012.

[29] Wikimedia User Eborutta. Golden-mantled ground squirrel. `http://commons.wikimedia.org/wiki/File:Goldmantelziesel.jpg`, 2003. Licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license.

[30] George F. Estabrook, F. R. McMorris, and Christopher A. Meacham. Comparison of undirected phylogenetic trees based on subtrees of four evolutionary

units. *Systematic Zoology*, 34(2):193–200, 06 1985.

[31] J Felsenstein and G A Churchill. A Hidden Markov Model approach to variation among sites in rate of evolution. *Molecular Biology and Evolution*, 13(1):93–104, 1996.

[32] Joseph Felsenstein. Evolutionary trees from DNA sequences: a maximum likelihood approach. *Journal of Molecular Evolution*, 17(6):368–376, 1981.

[33] Joseph Felsenstein. *Inferring Phylogenies*. Sinauer Associates, 2003.

[34] J. Patrick Fischer. Siberian chipmunk (Tamias sibiricus) in Gyeongju, South Korea. `http://commons.wikimedia.org/wiki/File:Asiatisches_Streifenhoernchen.jpg`, 2010. Licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license.

[35] T. Flouri, F. Izquierdo-Carrasco, D. Darriba, A.J. Aberer, L.-T. Nguyen, B.Q. Minh, A. von Haeseler, and A. Stamatakis. The Phylogenetic Likelihood Library. *Systematic Biology*, 2014.

[36] A. Gavryushkina, D. Welch, T. Stadler, and A. Drummond. Bayesian inference of sampled ancestor trees for epidemiology and fossil calibration. *ArXiv E-prints*, June 2014.

[37] Stuart Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-6(6):721–741, Nov 1984.

[38] Gilles Gonthier. Tamias striatus. `http://commons.wikimedia.org/wiki/File:Tamias_striatus2.jpg`, 2006. Licensed under the Creative Commons Attribution 2.0 Generic license.

[39] Dan Graur and William Martin. Reading the entrails of chickens: molecular timescales of evolution and the illusion of precision. *TRENDS in Genetics*, 20(2):80–86, 2004.

[40] Terry Gray. Red-tailed chipmunk. `http://commons.wikimedia.org/wiki/File:Red-tailed_Chipmunk_(Tamias_ruficaudus).jpg`, 2009. Licensed under the Creative Commons Attribution 2.0 Generic license.

[41] Simon J. Greenhill, Alexei J. Drummond, and Russell D. Gray. How accurate and robust are the phylogenetic estimates of Austronesian language relationships? *PLOS ONE*, 5(3):e9573, 03 2010.

[42] Masami Hasegawa, Hirohisa Kishino, and Taka-aki Yano. Dating of the human-ape splitting by a molecular clock of mitochondrial DNA. *Journal of Molecular Evolution*, 22(2):160–174, 1985.

[43] W.K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[44] Tracy A Heath. A hierarchical Bayesian model for calibrating estimates of species divergence times. *Systematic Biology*, 61(5):793–809, 2012.

[45] Tracy A. Heath, Mark T. Holder, and John P. Huelsenbeck. A Dirichlet process prior for estimating lineage-specific substitution rates. *Molecular Biology and Evolution*, 29(3):939–955, 2012.

[46] Tracy A. Heath, John P. Huelsenbeck, and Tanja Stadler. The fossilized birth–death process for coherent calibration of divergence-time estimates. *Proceedings of the National Academy of Sciences*, 111(29):E2957–E2966, 2014.

[47] Joseph Heled and Alexei J. Drummond. Calibrated tree priors for relaxed phylogenetics and divergence time estimation. *Systematic Biology*, 61(1):138–149,

2012.

[48] David M. Hillis, Tracy A. Heath, and Katherine St. John. Analysis and visualization of tree space. *Systematic Biology*, 54(3):471–482, 2005.

[49] John P. Huelsenbeck, Fredrik Ronquist, Rasmus Nielsen, and Jonathon P. Bollback. Bayesian inference of phylogeny and its impact on evolutionary biology. *Science*, 294:2310–2314, 2001.

[50] Jun Inoue, Philip CJ Donoghue, and Ziheng Yang. The impact of the representation of fossil calibrations on Bayesian estimation of species divergence times. *Systematic Biology*, 59(1):74–89, 2010.

[51] Fernando Izquierdo-Carrasco, Stephen Smith, and Alexandros Stamatakis. Algorithms, data structures, and numerics for likelihood-based phylogenetic inference of huge trees. *BMC Bioinformatics*, 12(1):470, 2011.

[52] Ryan Johnson. Harris's antelope squirrel from Gold Canyon, Arizona, USA. , 2010. Licensed under the Creative Commons Attribution 2.0 Generic license.

[53] Thomas H Jukes and Charles R Cantor. Evolution of protein molecules. *Mammalian Protein Metabolism*, 3:21–132, 1969.

[54] David G Kendall. On the generalized" birth-and-death" process. *The Annals of Mathematical Statistics*, pages 1–15, 1948.

[55] Motoo Kimura. A simple method for estimating evolutionary rates of base substitutions through comparative studies of nucleotide sequences. *Journal of Molecular Evolution*, 16(2):111–120, 1980.

[56] Hirohisa Kishino, Jeffrey L Thorne, and William J Bruno. Performance of a divergence time estimation method under a probabilistic model of rate evolution. *Molecular Biology and Evolution*, 18(3):352–361, 2001.

[57] Damean Kuhn. Yellow-pine chipmunk at the Skyline trail, Mount Rainier National Park. `http://commons.wikimedia.org/wiki/File:Yellow_pine_chipmonk_2008.JPG`, 2008. Licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license.

[58] Louise A Lewis and Paul O Lewis. Unearthing the molecular phylodiversity of desert soil green algae (chlorophyta). *Systematic Biology*, 54(6):936–947, 2005.

[59] Wen-Hsiung Li, Masako Tanimura, and PaulM. Sharp. An evaluation of the molecular clock hypothesis using mammalian DNA sequences. *Journal of Molecular Evolution*, 25(4):330–342, 1987.

[60] T Mailund and CNS Pedersen. QDist–quartet distance between evolutionary trees. *Bioinformatics*, page bth097, 2004.

[61] Suzanne Matthews. *Efficient Algorithms for Comparing, Storing, and Sharing Large Collections of Phylogenetic Trees*. PhD thesis, Texas A&M University, 2012.

[62] Suzanne J Matthews, Seung-Jin Sul, and Tiffani L Williams. A novel approach for compressing phylogenetic trees. In *Bioinformatics Research and Applications*, pages 113–124. Springer, 2010.

[63] Camilo Mora, Derek P. Tittensor, Sina Adl, Alastair G. B. Simpson, and Boris Worm. How many species are there on earth and in the ocean? *PLOS Biology*, 9(8):e1001127, 08 2011.

[64] Thomas J. Near and Michael J. Sanderson. Assessing the quality of molecular divergence time estimates by fossil calibrations and fossil–based model selection. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 359(1450):1477–1483, 2004.

[65] Andrew Rambaut. Figtree v1. 3.1: Tree figure drawing tool. `http://tree.bio.ed.ac.uk/software/figtree`, 2009.

[66] Andrew Rambaut and Nicholas C. Grass. Seq-gen: an application for the monte carlo simulation of DNA sequence evolution along phylogenetic trees. *Computer Applications in the Biosciences : CABIOS*, 13(3):235–238, 1997.

[67] Bruce Rannala and Ziheng Yang. Probability distribution of molecular evolutionary trees: a new method of phylogenetic inference. *Journal of Molecular Evolution*, 43(3):304–311, 1996.

[68] Bruce Rannala and Ziheng Yang. Inferring speciation times under an episodic molecular clock. *Systematic Biology*, 56(3):453–466, 2007.

[69] David James Russell. *Multiple Sequence Alignment Methods*. Springer, 2014.

[70] Eric W. Sayers, Tanya Barrett, Dennis A. Benson, Stephen H. Bryant, Kathi Canese, Vyacheslav Chetvernin, Deanna M. Church, Michael DiCuccio, Ron Edgar, Scott Federhen, Michael Feolo, Lewis Y. Geer, Wolfgang Helmberg, Yuri Kapustin, David Landsman, David J. Lipman, Thomas L. Madden, Donna R. Maglott, Vadim Miller, Ilene Mizrachi, James Ostell, Kim D. Pruitt, Gregory D. Schuler, Edwin Sequeira, Stephen T. Sherry, Martin Shumway, Karl Sirotkin, Alexandre Souvorov, Grigory Starchenko, Tatiana A. Tatusova, Lukas Wagner, Eugene Yaschenko, and Jian Ye. Database resources of the National Center for Biotechnology Information. *Nucleic Acids Research*, 37(suppl 1):D5–D15, 2009.

[71] Greg Schechter. Neotamias senex, Allen's chipmunk. `http://commons.wikimedia.org/wiki/File:Neotamias_senex.jpg`, 2011. Licensed under the Creative Commons Attribution 2.0 Generic license.

[72] Douglas E. Soltis, Matthew A. Gitzendanner, and Pamela S. Soltis. A 567-taxon data set for angiosperms: The challenges posed by Bayesian analyses of large data sets. *Int. J. Plant Sci.*, 168(2):137–157, 2007.

[73] Mark S Springer, Robert W Meredith, John Gatesy, Christopher A Emerling, Jong Park, Daniel L Rabosky, Tanja Stadler, Cynthia Steiner, Oliver A Ryder, Jan E Janečka, et al. Macroevolutionary dynamics and historical biogeography of primate diversification inferred from a species supermatrix. *PLOS ONE*, 7(11):e49521, 2012.

[74] Tanja Stadler and Ziheng Yang. Dating phylogenies with sequentially sampled tips. *Systematic Biology*, 62(5):674–688, 2013.

[75] Alexandros Stamatakis and Michael Ott. Efficient computation of the phylogenetic likelihood function on multi-gene alignments and multi-core architectures. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 363(1512):3977–3984, 2008.

[76] A.P. Stamatakis, T. Ludwig, H. Meier, and M.J. Wolf. Accelerating parallel maximum likelihood-based phylogenetic tree calculations using subtree equality vectors. In *Supercomputing, ACM/IEEE 2002 Conference*, pages 40–40, Nov 2002.

[77] Mike Steel and Andy McKenzie. Properties of phylogenetic trees generated by yule-type speciation models. *Mathematical Biosciences*, 170(1):91–112, 2001.

[78] M Stissing, T Mailund, CNS Pedersen, GS Brodal, and R Fagerberg. Computing the all-pairs quartet distance on a set of evolutionary trees. *Journal of Bioinformatics & Computational Biology*, 6(1):37–50, 2008.

[79] D. L. Swofford. PAUP*: Phylogenetic analysis using parsimony (and other methods), 2002. Sinauer Associates, Underland, Massachusetts, Version 4.0.

[80] Wikimedia User Tadam. Spermophilus variegatus at Zion National Park. `http://commons.wikimedia.org/wiki/File:Spermophilus_variegatus.jpg`, 2008. Licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license.

[81] K Tamura. Estimation of the number of nucleotide substitutions when there are strong transition-transversion and g+c-content biases. *Molecular Biology and Evolution*, 9(4):678–687, 1992.

[82] K Tamura and M Nei. Estimation of the number of nucleotide substitutions in the control region of mitochondrial DNA in humans and chimpanzees. *Molecular Biology and Evolution*, 10(3):512–526, 1993.

[83] Koichiro Tamura, Fabia Ursula Battistuzzi, Paul Billing-Ross, Oscar Murillo, Alan Filipski, and Sudhir Kumar. Estimating divergence times in large molecular phylogenies. *Proceedings of the National Academy of Sciences*, 109(47):19333–19338, 2012.

[84] Koichiro Tamura, Glen Stecher, Daniel Peterson, Alan Filipski, and Sudhir Kumar. MEGA6: molecular evolutionary genetics analysis version 6.0. *Molecular Biology and Evolution*, 30(12):2725–2729, 2013.

[85] Simon Tavaré. Some probabilistic and statistical problems in the analysis of DNA sequences. *Lectures on Mathematics in the Life Sciences*, 17:57–86, 1986.

[86] Jeffrey L Thorne, Hirohisa Kishino, and Ian S Painter. Estimating the rate of evolution of the rate of molecular evolution. *Molecular Biology and Evolution*, 15(12):1647–1657, 1998.

[87] Tandy Warnow. Standard maximum likelihood analyses of alignments with gaps can be statistically inconsistent. *PLOS Currents*, 4, 2012.

[88] Eric W. Weisstein. Likelihood. From MathWorld—A Wolfram Web Resource. `http://mathworld.wolfram.com/Likelihood.html`. Last visited on 4/3/2015.

[89] Joel O. Wertheim and Michael J. Sanderson. Estimating diversification rates: How useful are divergence times? *Evolution*, 65(2):309–320, 2011.

[90] Z Yang and B Rannala. Bayesian phylogenetic inference using DNA sequences: a Markov Chain Monte Carlo method. *Molecular Biology and Evolution*, 14(7):717–724, 1997.

[91] Ziheng Yang. Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: approximate methods. *Journal of Molecular Evolution*, 39(3):306–314, 1994.

[92] Ziheng Yang. *Computational Molecular Evolution*, volume 21. Oxford University Press Oxford, 2006.

[93] Ziheng Yang. Paml 4: phylogenetic analysis by maximum likelihood. *Molecular Biology and Evolution*, 24(8):1586–1591, 2007.

[94] Ziheng Yang and Bruce Rannala. Bayesian estimation of species divergence times under a molecular clock using multiple fossil calibrations with soft bounds. *Molecular Biology and Evolution*, 23(1):212–226, 2006.

[95] Ziheng Yang and Anne D. Yoder. Comparison of likelihood and Bayesian methods for estimating divergence times using multiple gene loci and calibration points, with application to a radiation of cute-looking mouse lemur species. *Systematic Biology*, 52(5):705–716, 2003.

[96] Miriam L Zelditch, Jingchun Li, Lucy AP Tran, and Donald L Swiderski. Relationships of diversity, disparity and their evolutionary rates in squirrels (Sciuridae). *Evolution*, 2015.

[97] Yuchi Zheng, Rui Peng, Masaki Kuro-o, and Xiaomao Zeng. Exploring patterns and extent of bias in estimating divergence time from mitochondrial DNA sequence data in a particular lineage: A case study of salamanders (order Caudata). *Molecular Biology and Evolution*, 28(9):2521–2535, 2011.

[98] E. Zuckerkandl and L. Pauling. *Molecular Disease, Evolution and Genetic Heterogeneity*, pages 189–225. Academic Press, 1962.

APPENDIX A

THE ANCESTRAL AGE FRAMEWORK

A framework was required that would allow efficient implementation of the divergence time algorithms. This implementation would act as the test platform for the various algorithms as well as providing a working implementation for use by biologists.

During the design phase of the project, it was determined that the existing divergence time programs, Beast and MCMCTree, were not appropriate for the implementation of the new algorithms. In the case of Beast, the architecture is extensible, but, the focus of the platform is on phylogenetic inference with divergence time as a secondary feature. The fear was that the impact of the new divergence time algorithms would be hard to determine in the context of phylogenetic inference. In the case of MCMCTree the previously discussed software engineering issues would have significantly complicated the implementation of new algorithms. Additionally, our previous work on MCMCTree had shown that the monolithic structure of the program made implementation of new data structures extremely difficult.

It was therefore determined that the best approach to the implementation of the new algorithms would be in a new divergence time framework, AncestralAge. This framework was designed to meet the following goals:

- Allow for simplified implementation of algorithms for discrete aspects of the divergence time problem. Addition of a new version of an algorithm should not require significant modification to the framework.

- The framework would itself be high performance. Overhead for the framework

would be kept to a minimum.

- The framework would be operating system independent. It should run on any of the major systems (Unix, OS/X, Windows) in common use at this time.
- The framework would provide detailed performance data as part of the architecture.
- The framework would provide a set of common, extensible, services including task management and operating logging. These services would themselves be modular and extensible.

AncestralAge was created to meet these goals.

As part of this research a rigorous analysis was performed of the statistical model and it's computational implications. The goal was to minimize the work required for each and every parameter change. In the following discussion the implication to each of the model components for a change in a particular type of parameter is discussed. This analysis provided key insights into the structure of the computational model implemented by AncestralAge.

Age Parameter Proposals. When a new age is proposed for an ancestral node, only some components of the model require updating. In equation (3.7) it can be seen that the node ages ($\mathbf{t}$) are not included in the prior on nuisance parameters and therefore there is no need to compute that prior as part of the acceptance ratio for ages. For the other model components the following apply:

Likelihood. Since it is frequently the case that not all species appear in all gene trees, there will be nodes "missing" from the gene trees. When a new age is proposed for a parameter associated with node in the species tree, the likelihood will

change for each gene tree that contains the node but omitting gene trees where the node is missing.

But, it is not necessary to compute the likelihood for the entire gene tree. Contributions to the likelihood from siblings of the node with the new age will not have changed and do not require recomputation. Additionally, as the gene tree is walked from the changed node to the root, siblings of any of the ancestors of the nodes along the path to the root will not require recomputation either. But, as the transition probabilities for a branch are functions of the both the times and the rates at the branch, it is necessary to recompute the values for the immediate children of the changed node. Therefore, in summary, if intermediate values are preserved within the gene trees, it is only necessary to recompute from the children of the changed node to the root for each gene tree in which the changed node appears.

Prior on Ages. Due to the model used for the prior of ages, the scope of a change to the prior will vary depending on the magnitude of the change. To compute the prior the ages of each node in the species tree are sorted and this sorted list is used for the computation of the prior. If an age is changed, the impact of the change will depend on whether the ordering in the sorted list is changed. In the simplest case, the list ordering is not changed and the only computation required will be for the changed node. If the list ordering has changed, calculations will need to be performed for that portion of the list whose ordering has changed. For example, if the node A was previously older than node B, but, as a result of the proposed new age it is now younger than node B, the contribution of both nodes A and B to the prior will need to be recomputed.

Prior on Rates. The impact of an age parameter change will depend on the rate model used:

- For the molecular clock model, there is no relationship between the single rate for a gene tree and the node ages. Therefore there is no requirement to update the rate prior on an age change.

- For the independent clock model, the node ages do not figure into the calculation of the rates at the gene tree nodes. Therefore, as with the molecular clock, there is no need to update the prior on an age change.

- For the correlated clock model, the value of the prior is calculated using the edge length in the gene trees. This length is calculated as the product of the rate and the time duration of the edge. Therefore any change in node age will effect the prior computed from the immediate descendants and immediate ancestor of the changed node. Our early work on MCMCTree determined that a key component of the excessive time estimate for the primates study was a result of recomputing the prior on rates across the entire gene tree. This optimization allowed for a significant reduction in the time required for MCMCTree.

Rate Parameter Proposals. As with changes to age parameters, only some model components require updating on a new rate proposal. Referring once again to equation (3.7), it can be seen that the rate does not factor into the prior on ages or the prior on the nuisance parameters and therefore those model components would cancel out on a rate change.

It should be noted that in the molecular clock model there are no rate parameters on any of the branches. The only parameter related to rates is the overall rate for each gene tree.

Likelihood. As rates are associated with a single gene tree, a new rate parameter will only require updating of at most one gene tree. Assuming again that

intermediate values have been saved during computation of the overall gene tree, a rate change will only impact the edge with the changed rate and it's ancestors leading up to the root. Once again, siblings of the changed edge and the siblings of it's ancestors will not be effected and will not require recomputation.

Prior on Rates. When a rate is changed, the edge length computed as the product of the rate and the time duration for the edge changes for that branch only, therefore only the contribution of that edge to the overall prior needs to be recomputed.

Nuisance Parameter Proposals. This is not a single type of parameter although, in general, all the nuisance parameters operate at the gene tree level. For example, if an evolutionary model parameter is changed, all transition probability matrices for the gene tree will need to be updated and the likelihood for the entire gene tree recomputed.

In the case of these parameters, none of the changes will impact the ages of the nodes and therefore the prior on ages would cancel out.

Likelihood. As mentioned, the nuisance parameters have an effect across an entire gene tree and therefore a change in any of these parameters will require re-computation of the likelihood for the entire gene tree.

Prior on Rates. Changes to evolutionary model parameters will not effect the rates at any of the edges and therefore will allow the prior on rates to cancel out. But, changes to the mean or variance parameters at the root of the gene trees will effect the rates prior in the independent and correlated models. For the molecular clock model there are no rate parameters and therefore no prior on the rates. The only parameters in this model are the mean and variance parameters on the rate at the root of each gene tree..

Prior on Nuisance Parameters. Each of the nuisance parameters will make a contribution to the overall prior for the nuisance parameters but only requires computation of the prior for the single changed parameter. In general, this will only involve determination of the probability density value for the new parameter value in the distribution specified by the hyperparameters involved.

Capabilities of AncestralAge. While the long term goal will be to provide a robust set of divergence time methods the focus of the initial implementation has been on the subset of the capabilities provided by MCMCTree and Beast that would most meet the needs of biologists working on phylogenetic studies. These capabilities are:

1. Support for the DNA nucleotide model. At some future point, the Amino Acid or Protein model could be added.

2. Support for a subset of the IUPAC codes for nucleotides[17]. The codes for nucleotides T,C,A and G along with "any" character and missing are supported. Codes for the other possible combinations of nucleotides are not supported at this time.

3. Support for the most common evolutionary models:

    - Jukes-Cantor model (JC69)[53].
    - Kimura two parameter model (K80)[55].
    - Felsenstein 1981 model (F81)[32].
    - Felsenstein 1984 model (F84)[31].
    - Hasegawa, Kishino and Yano 1985 model (HKY85)[42].
    - Tamura 1992 model (T92)[81].

- Tamura and Nei 1993 model (TN93)[82].
- Generalized Time Reversible model (GTR)[85].

4. Support for common statistical distributions for fossil calibrations:
   - Fixed or soft lower bounds[94].
   - Fixed or soft upper bounds[94].
   - Fixed or soft upper and lower bounds[94].
   - Gamma distributed calibration ages.
   - Normally distributed calibration ages.
5. Support for discrete gamma variation between sites. At some future point support for the percentage of invariant sites could be added.
6. Support for three clock (evolutionary rate) models as defined by Yang[68]:
   (a) A single rate for all branches in a gene tree (the molecular clock).
   (b) Independent rates for branch drawn from a common gamma distribution.
   (c) Branch rates drawn from a common gamma distrbution with the mean of the distribution set based on the rate for the ancestral branch (the correlated model of Yang and Rannala).
7. Support for the prior of node ages using the birth-death-sampling process of Yang and Rannala[94]. Additional algorithms such as the Yule process or Dirichlet process prior[45] for node ages could be implemented at some future point.
8. Support for multiple input trees. Common structure will be leveraged to improve computational efficiency.

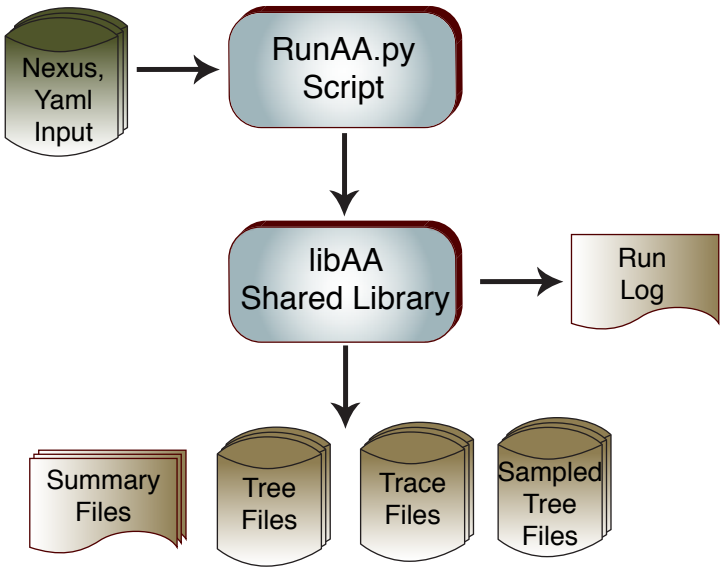


Figure A.1: *AA High Level Architecture.* This figure illustrates the overall structure of the AncestralAge framework. The RunAA.py command line interface is responsible for parsing program options and loading the input data. These options and data are loaded into the LibAA shared library for processing. The LibAA shared library is responsible for writing the various output files from the process.

9. Support for multiple MCMC chains or multiple jackknife replicates. Each tree is included in each of the chains or jackknife replicates. At this time either multiple MCMC chains or multiple jackknife replicates may be used but multiple MCMC chains across multiple jackknife replicates is not supported.

Architecture of AncestralAge. As shown in Figure A.1, the core of AncestralAge is a shared library, LibAA, providing the framework for the algorithms implemented. In programs such as MCMCTree a considerable amount of code is required to manage the basic housekeeping tasks of interpreting options and loading input data. But these tasks are handled easily and with minimal programming in scripting languages such as Python. By the use of a scripting language it becomes possible to quickly support multiple input formats and the large set of configura-

tion options required by divergence time methods. Therefore the setting of program options and acquisition of the input data is handled by a Python script: RunAA.py.

The LibAA shared library exposes it's capabilities though an API with "C", "C++" and Python language bindings available. All outputs are generated through the LibAA library. In addition to a single log file output, for each tree in each replicate (MCMC chain or jackknife) the following files are produced:

- The final tree generated as a Nexus file suitable as input to the FigTree[65] visualization program.
- A textual summary file is produced giving detailed information on all the model parameters.
- Trace records in CSV format that can be used as input to the Tracer program to provide visual and statistical validation of the MCMC process.
- The trees sampled during the MCMC process as a Nexus file suitable for input to the TreeHouse[7], TreeZip[62] or Phash[61] tree analysis tools.

LibAA: A Shared Library for Divergence Time Inference. The LibAA shared library is implemented in C++ conforming to the C++11 language standard. The components of the library are shown in Figure A.2.

A typical use of the library is in four phases:

1. The process parameters and data are loaded into the library using methods provided. The data includes the trees to be dated, the DNA sequences for the trees and set of genes (loci) and taxa to be included. Trees are loaded into a common directed acyclic graph data structure reducing duplication in cases where common subtrees exist.

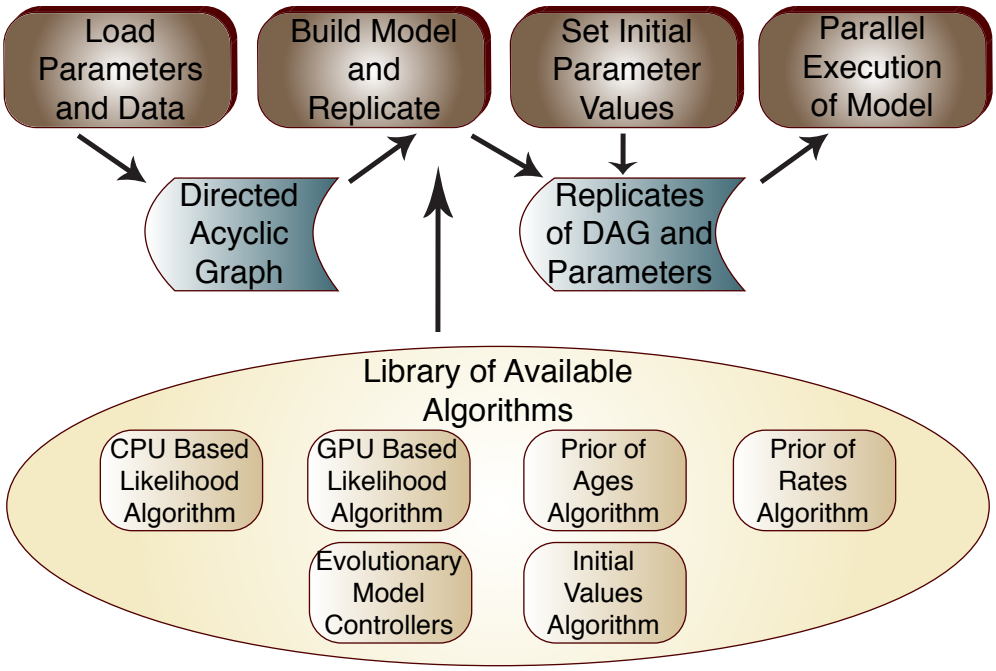


Figure A.2: *Components of the AA Shared Library.* The four phases of processing are shown at the top of the figure. Parameter and data loading produces the common directed acyclic graph data structure. This data structure is replicated during model construction using a library of available algorithms and initial values are assigned to all parameters. The replicates are then passed to a set of processing tasks to execute the MCMC steps.

2. A model is generated from the input parameters and data. Since, at this point, there is complete information on the set of operations to be performed in each MCMC step it is possible to generate a model that includes only those algorithms required (e.g. rate clock models) for the run in process. This allows the MCMC step execution path to be optimized prior to starting the process thereby improving processor efficiency.

   The model is assembled from a library of algorithms including:

   - CPU and GPU versions of the likelihood computation (see Chapter 4).
   - The initial value model in use (discussed below).

- A set of algorithms to build transition probability matrices depending on the evolutionary model chosen.
- Algorithms for the computation of the priors of node ages and node rates. Different versions of the algorithms are used depending on the clock model in use (see Section A).

The model is then replicated with a replicate being generated for each MCMC chain or jackknife replicate requested.

3. Initial values are determined for all model parameters. MCMCTree does not allow for the input of branch lengths as part of the tree under analysis, initial values for all parameters are determined through an iterative fitting process that might or might not produce results similar to the branch lengths returned from the phylogenetic inference.

   It is our hypothesis that by using branch lengths provided in the input trees, a better starting point for the process will be used thereby contributing to quicker MCMC convergence. We therefore use any branch length information provided in the input along with the fossil calibrations to set the initial ages of the inner nodes and roots as well as the rates associated with the edges of the gene trees.

4. Finally, the model is executed. Each replicate is allowed to run on a separate thread with the typical case being that the number of replicates would exceed the number of cores available on a given processor.

RunAA.py: An Extensible Command Line Interface to LibAA. The RunAA.py script is provided as a command line interface to the LibAA shared library. This script may be run directly from the command line on Unix and OS/X systems

and through the python interpreter on Windows systems. Library options (e.g. number of MCMC steps to run) may be provided either on the command line or as part of an input file. Two formats of input file are supported, the standard Nexus format as well as in a Python friendly, Yaml (Yet Another Markup Language), format.

The options available from the command line are shown in the script output when the "–help" option is provided. A sample help output is provided in Appendix B. A sample Nexus format file is provided in Appendix C and a sample Yaml format file is provided in Appendix D.

APPENDIX B

RunAA.py HELP OUTPUT

```
$ ./RunAA.py --help
usage: RunFDivT.py [-h] [--noconsole] [--dontrun]
                   [--stdinfmt {yaml,nexus,newick}] [--bdlambda FLOAT]
                   [--bdmu FLOAT] [--bdrho FLOAT] [--burnin INT]
                   [--clockmeanalpha FLOAT] [--clockmeanbeta FLOAT]
                   [--clockmeandcon FLOAT] [--clockmodel CMODEL]
                   [--clockvaralpha FLOAT] [--clockvarbeta FLOAT]
                   [--clockvardcon FLOAT] [--dumpreplicates]
                   [--no-dumpreplicates] [--dumptrees] [--no-dumptrees]
                   [--filename FILENAME] [--fileoverwrite]
                   [--no-fileoverwrite] [--gapchar CHAR] [--hashbits INT]
                   [--logdebug] [--no-logdebug] [--logfile FILENAME]
                   [--logperf] [--no-logperf] [--logstats] [--no-logstats]
                   [--missingchar CHAR] [--ngen INT] [--nreplicates INT]
                   [--nruns INT] [--nthreads INT] [--replicatepct FLOAT]
                   [--samplefreq INT] [--seed INT] [--seqchars STRING]
                   [--usegpu] [--no-usegpu]
                   files [files ...]

Perform phylogenetic divergence time analysis on an input file. Command line
options are applied after any files are loaded. This allows overriding of any
options in the files.

positional arguments:
  files                 Input File(s). At least one file must be included.
                        Multiple files may be specified and will be read in
                        the order specified. Since there are some constraints
                        on how parameters are loaded (e.g. taxa before trees)
                        this must be considered in the order of the files on
                        the command line. If '@' is specified as one of the
                        file names, input will next be read from stdin at that
                        point. Note other files can be read before and after
                        stdin if required. The type of file is assumed based
                        on the file extension. For nexus files extensions
                        '.nexus' and '.nex' are supported. For newick tree
                        files, '.tre' and '.newick' are supported. For yaml
                        input, '.yaml' is value.

optional arguments:
  -h, --help            show this help message and exit
  --noconsole           Do not produce console output. Normally, all output is
                        directed to the console along with any file specified
                        with the --logfile parameter. If this option is
                        specified, console output will be disabled and only
                        --logfile output produced. In this case if --logfile
                        isn't specified there isn't any output at all.
  --dontrun             Do not automatically run the MCMC process after
                        loading all input. Normally, all input files will be
                        read and loaded then the MCMC process run.
  --stdinfmt {yaml,nexus,newick}
                        Format to use for input read from stdin. The default
                        is the standard yaml format as described in the
                        documentation. If 'nexus' is specified, the stdin must
                        be a properly formatted nexus file. If 'newick' is
                        specified, the input must consist of one or more
```

```
                        newick format trees
--bdlambda FLOAT        Time prior - Birth death lambda. Value is a floating
                        point number
--bdmu FLOAT            Time prior - Birth death mu. Value is a floating point
                        number
--bdrho FLOAT           Time prior - Birth death rho. Value is a floating
                        point number
--burnin INT            Numer of burnin generations to exclude from output.
                        The value must be less than the number of generations
                        specified (ngen). Default is zero.
--clockmeanalpha FLOAT
                        Clock models - mean of rates gamma alpha
                        hyperparameter. Value is a floating point number.
                        Default is 1.0
--clockmeanbeta FLOAT
                        Clock models - mean of rates gamma beta hyperparameter
                        Value is a floating point number. Default is 1.0
--clockmeandcon FLOAT
                        Clock models - mean of rates dirichlet concentration
                        hyperparameter. Value is a floating point number.
                        Default is 1.0
--clockmodel CMODEL     Clock model (global, independend, correlated).
--clockvaralpha FLOAT
                        Clock models - variance of rates gamma alpha
                        hyperparameter
--clockvarbeta FLOAT    Clock models - variance of rates gamma beta
                        hyperparameter
--clockvardcon FLOAT    Clock models - variance of rates dirichlet
                        concentration hyperparameter
--dumpreplicates        Dump trees after replication
--no-dumpreplicates
--dumptrees             Dump trees after load prior to replication
--no-dumptrees
--filename FILENAME     File name root for all output files. This name is used
                        as the base of all the non-console output files (e.g.
                        trace, tree files). If the files already exist the run
                        will be aborted without overwriting the files. If the
                        value '%(date)' is included in the filename, the
                        current date of the form yy-mm-dd will be included in
                        the generated names. If the value '%(time)' is
                        included in the filename, the current time of the form
                        hh-mm-ss will be included in the generated names.
--fileoverwrite         Allow any existing output files to be overwritten.
                        Default action is to abort if existing output files
                        are found.
--no-fileoverwrite
--gapchar CHAR          Sequence alignment: gap character. Default is '-'.
--hashbits INT          Number of bits in the subtree hash table. Default is
                        24 which should be adequite for most analysis.
                        Performance of the subtree hash can be monitored by
                        enabling --logstats
--logdebug              Enable debugging output. The module must have been
                        compiled with debugging specified (DEBUG=1 on the
                        scons command line). If debugging wasn't compiled in
                        this option is ignored
--no-logdebug
--logfile FILENAME      Write console output to this file as well as stdout.
                        If the file already exists it will be overridden
--logperf               Enable performance data output (if compiled in)
--no-logperf
--logstats              Enable run statistics output
--no-logstats
--missingchar CHAR      Sequence alignment: missing data character. Default is
                        '?'.
```

--ngen INT	Number of MCMC generations. This includes any burnin generations requested. A value must be specified **for** this parameter, there is no default
--nreplicates INT	Number of jackknife replicates to generate. The first replicate always contains all the calibrations. Subsequent replicates might or might not contain all the calibrations depending on the replication percentage requested. This value is mutually exclusive with the number of MCMC chains (nruns).
--nruns INT	Number of MCMC chains. This value determines the number of MCMC chains (runs) to be **done** on each input tree. This value is mutually exclusive with jackknife replication (nreplicates).
--nthreads INT	Number of parallel CPU threads. Default is the number of cores available.
--replicatepct FLOAT	Percentage of calibrations to include **in** each jackknife replicates. The value must be > 0.0 and < 100.0. The default is to 100.0 which will include all calibrations **in** all replicates.
--samplefreq INT	Number of generations between samples of the chains. Both trees and all parameter values are sampled at the same time. Default is to take 1000 samples across the non-burnin generations **if** the number of generations will support it (non-burnin generations >= 10,000). If the number of non-burnin generations is less than 10,000 a value is choosen to produce at least 100 samples.
--seed INT	Random seed to use. If not specified a value will be generated.
--seqchars STRING	Sequence alignment: **set** of characters **in** the alignment. Default is 'TCAG'.
--usegpu	Enable GPU support. If GPU support is unavailable, this value will be ignored. Default will be to **enable** support.
--no-usegpu	

APPENDIX C

SAMPLE NEXUS INPUT FILE

```
#NEXUS
[
Sample NEXUS File
]
begin data;
        dimensions ntax=8 nchar=49;
        format datatype=dna missing=? gap=-;
matrix
    Taxa_1    AAAAAAAAAAGGGGGGGGGGCCCCCCCCCCTTTTTTTTTT----------
    Taxa_2    TATATATATATATATATATATATACGCGCGCGCGCGCGCGCGCGCGCGCG
    Taxa_3    A?G?A?GA?G?A?GA?G?A?GA?G?A?GA?G?A?GA?G?A?GA?G?A?GA
    Taxa_4    ACGACGACGACGACGACGACGTGATGATGATGATGATATATATATATATA
    Taxa_5    GGGGGGGGGGCCCCCCCCCCTTTTTTTTTT----------AAAAAAAAAAT
    Taxa_6    TATATATATATATATATATATATACGCGCGCGCGCGCGCGCGCGCGCGCG
    Taxa_7    GGGGGGGGGGCCCCCCCCCC----------??????????AAAAAAAAAA
    Taxa_8    CCCCCCCCCCTTTTTTTTTT----------AAAAAAAAAAGGGGGGGGGG
end;

begin sets;
      charset 'Locus_1' = 1-20;
      charset 'Locus_2' = 21-49;
end;

begin taxa;
    Dimensions ntax=8;
    Taxlabels Taxa_1 Taxa_2 Taxa_3 Taxa_4 Taxa_5 Taxa_6 Taxa_7 Taxa_8;
end;

begin trees;
    tree t1 = ((Taxa_1:1 ,Taxa_2:2):3, ((Taxa_3:4, (Taxa_4:5, Taxa_5:6):7):8,
              (Taxa_6:9, (Taxa_7:10, Taxa_8:11):12):13):14);
    tree t2 = ((Taxa_1:.1, Taxa_3:.3):11, ((Taxa_2:.2, (Taxa_7:.7, Taxa_8:.8)
              :12):13, (Taxa_6:.6, (Taxa_4:.4, Taxa_5:.5):14):15):16);
    tree t3 = ((Taxa_1:1.1, (Taxa_2:1.2, (Taxa_3:1.3, Taxa_4:1.4):21):22):23,
              (Taxa_5:1.5, (Taxa_6:1.6, (Taxa_7:1.7, Taxa_8:1.8):24):25):26);
end;

begin fdivt;
    options clockmodel=correlated ngen=100000 samplefreq=1000 burnin=1000
            nruns=3 fileoverwrite=true;
    emodel  model=khky85 parms=(1.0, 0.5) gammacats=4 gammaparms= ( 0.5, 1.0 );
    outgroup (Taxa_1);
end;

begin calibrations;
    cal_1 descendents=(Taxa_3, Taxa_4) type=lowerupper minage=1.0 maxage=4.5;
end;
```

APPENDIX D

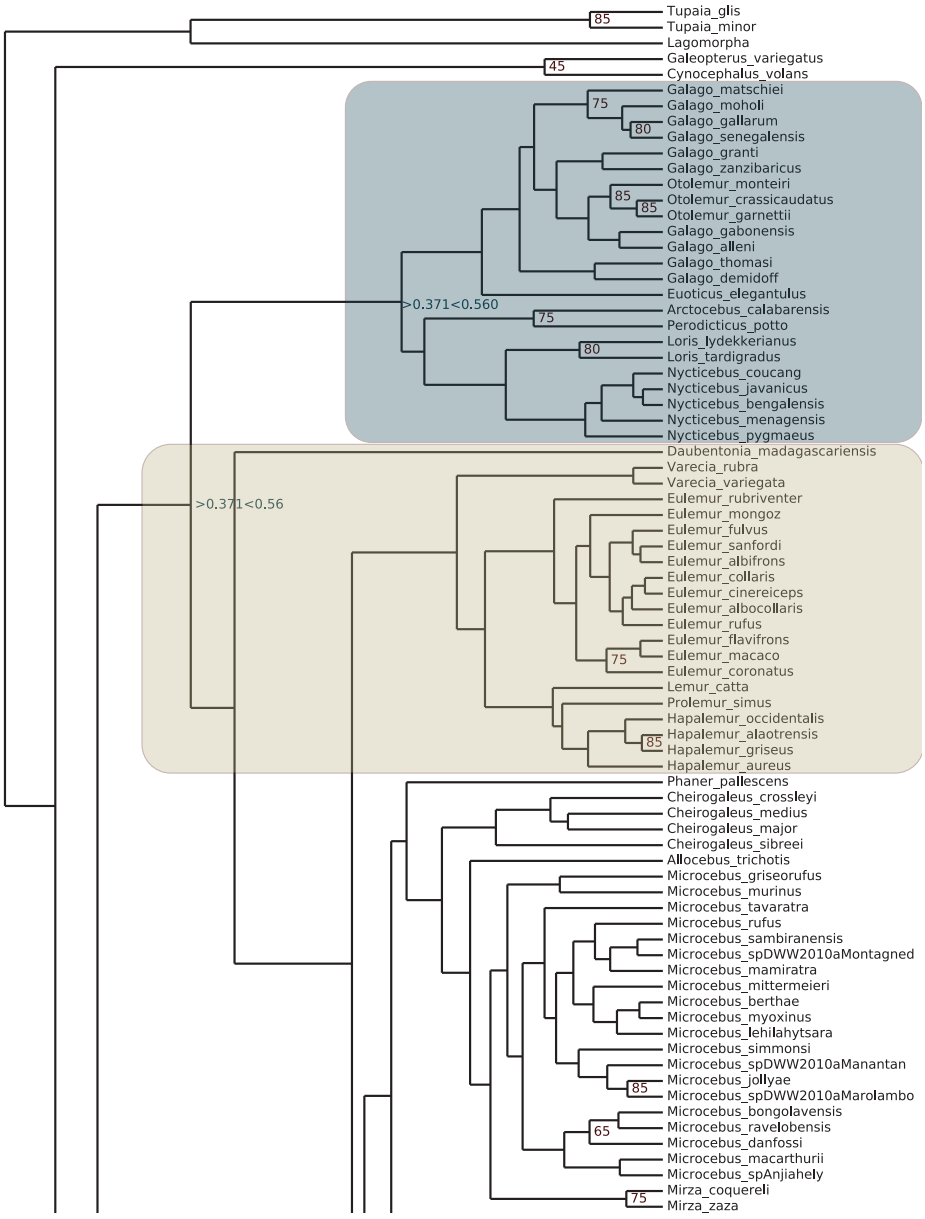
SAMPLE YAML INPUT FILE

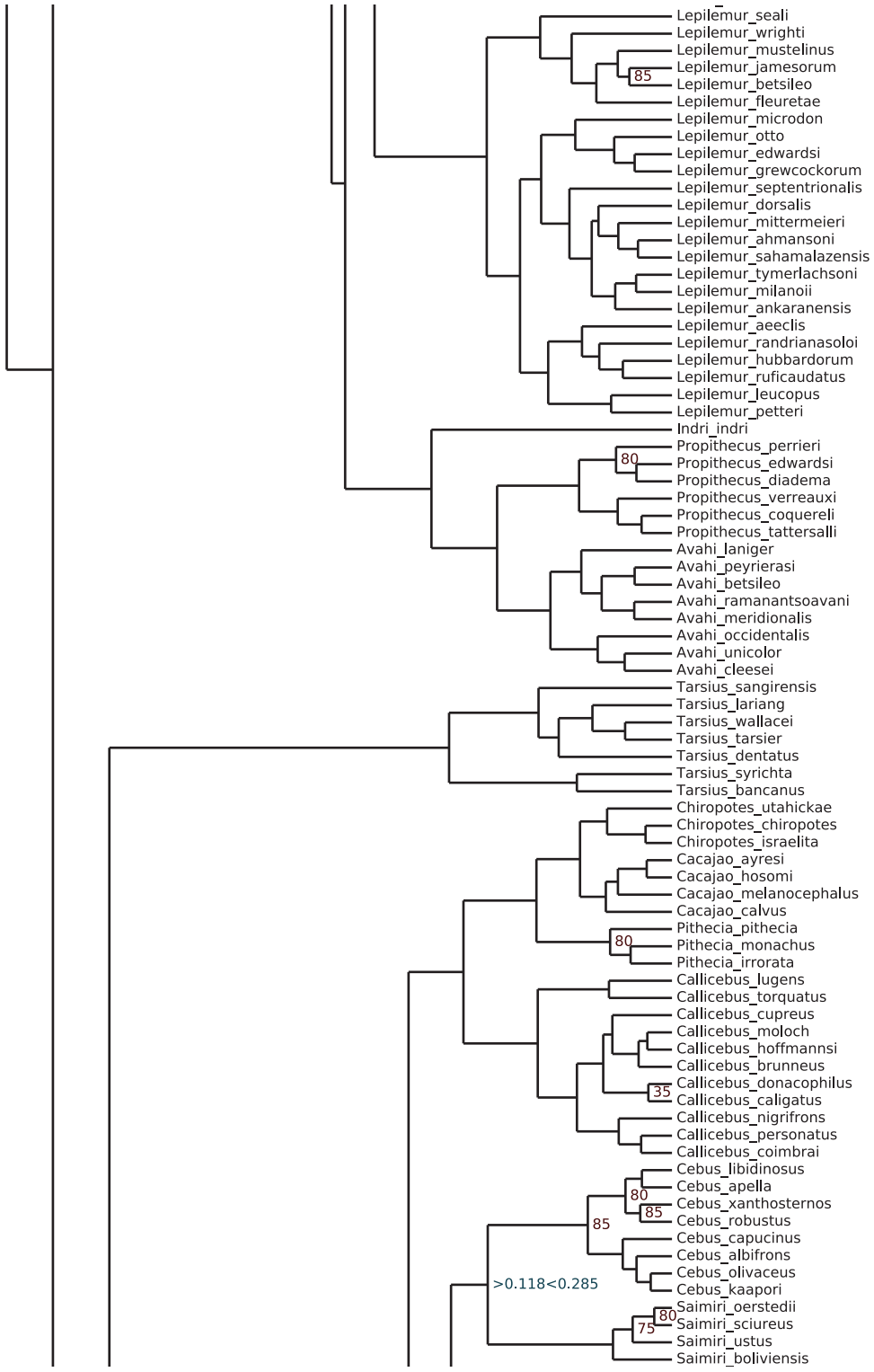
```
#
# Sample Yaml File
#

options:

  Seed            : 16361

  ClockModel      : Correlated

  NGen            : 100000
  SampleFreq      : 1000
  Burnin          : 1000
  NRuns           : 4
  FileOverwrite   : true

emodel:
  model       : HKY85
  parms       : [1.0, 0.5]
  gammacats   : 4
  gammaparms  : [ .5, 1]

taxa:
  - Taxa_1
  - Taxa_2
  - Taxa_3
  - Taxa_4
  - Taxa_5
  - Taxa_6
  - Taxa_7
  - Taxa_8

loci:
  - Loci_1
  - Loci_2
  - Loci_3
  - Loci_4
  - Loci_5
  - Loci_6

sequences:
  Taxa_1:
    Loci_1: AAAAAAAAAA?GGGGGGGGGCCCCC
    Loci_2: AAAAAAAAAAGGGGGGGGGGCCCCCCCCCCTTTTTTTTTT----------
    Loci_3: AAAAAAAAAAGGGGGGGGGGCCCCCCCCCCTTTTTTTTTT----------
    Loci_4: AAAAAAAAAAGGGGGGGGGGCCCCCCCCCCTTTTTTTTTT----------A
    Loci_5: AAAAAAAAAAGGGGGGGGGGCCCCCCCCCCTTTTTTTTTT----------AAAAAAAAAA
            GGGGGGGGGGCCCCCCCCCCTTTTTTTTTT----------
    Loci_6: AAAAAAAAAAGGGGGGGGGGCCCCCCCCCCTTTTTTTTTT----------AAAAAAAAAA
            GGGGGGGGGGCCCCCCCCCCTTTTTTTTTT----------AAAAAAAAAAGGGGGGGGGGCCCCC
  Taxa_2 :
    Loci_1: TTTTTTTTTT??????????AAAAA
    Loci_3: TATATATATATATATATATATACGCGCGCGCGCGCGCGCGCGCGCG
    Loci_5: A?G?A?GA?G?A?GA?G?A?GA?G?A?GA?G?A?GA?G?A?GA?G?A?GA?G?A?GA?
            G?A?GA?G?A?GA?G?A?GA?G?A?GA?G?A?GA?
```

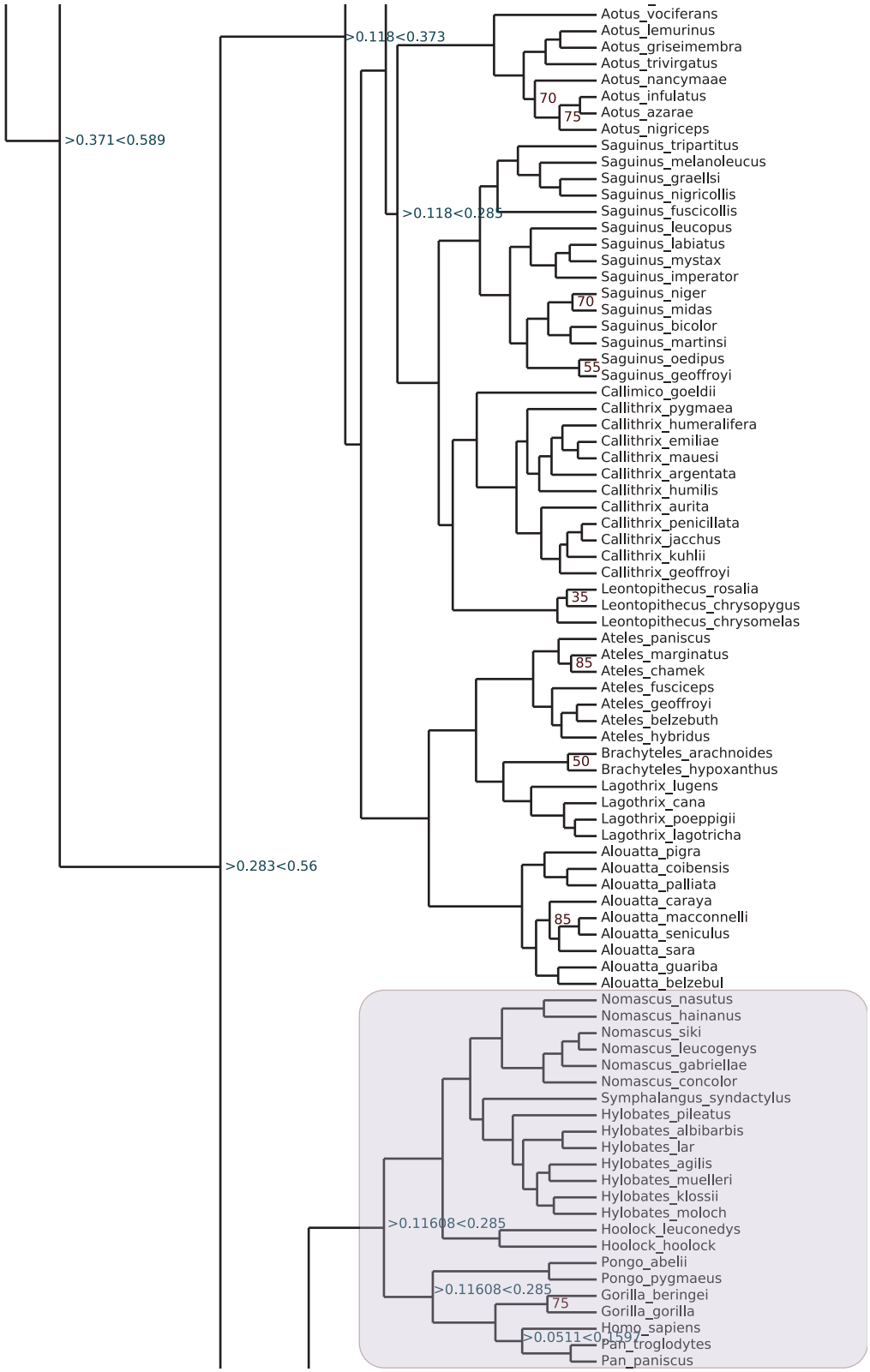
```
  Taxa_6 :
    Loci_2: GGGGGGGGGGCCCCCCCCCC----------?????????AAAAAAAAA
    Loci_4: ACGACGACGACGACGACGACGTGATGATGATGATGATATATATATATATAT
    Loci_6: T-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT
            -GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT-GAT

trees:
  t1: ((Taxa_1:1 ,Taxa_2:2):3, ((Taxa_3:4, (Taxa_4:5, Taxa_5:6):7):8, (Taxa_6:9,
      (Taxa_7:10, Taxa_8:11):12):13):14);
  t2: ((Taxa_1:1, Taxa_3:3):11, ((Taxa_2:2, (Taxa_7:7, Taxa_8:8):12):13,
      (Taxa_6:6, (Taxa_4:4, Taxa_5:5):14):15):16);
  t3: ((Taxa_1:1, (Taxa_2:2, (Taxa_3:3, Taxa_4:4):21):22):23, (Taxa_5:5, (Taxa_6:6,
      (Taxa_7:7, Taxa_8:8):24):25):26);

calibrations:
  - descendents:
    - Taxa_7
    - Taxa_8
    label   : Some fossil
    type    : Gammadist
    alpha   : 2.0
    beta    : 1.0
  - descendents:
    - Taxa_3
    - Taxa_4
    Type  : NormalDist
    Mu    : 10.001
    Sigma : 2.000
```

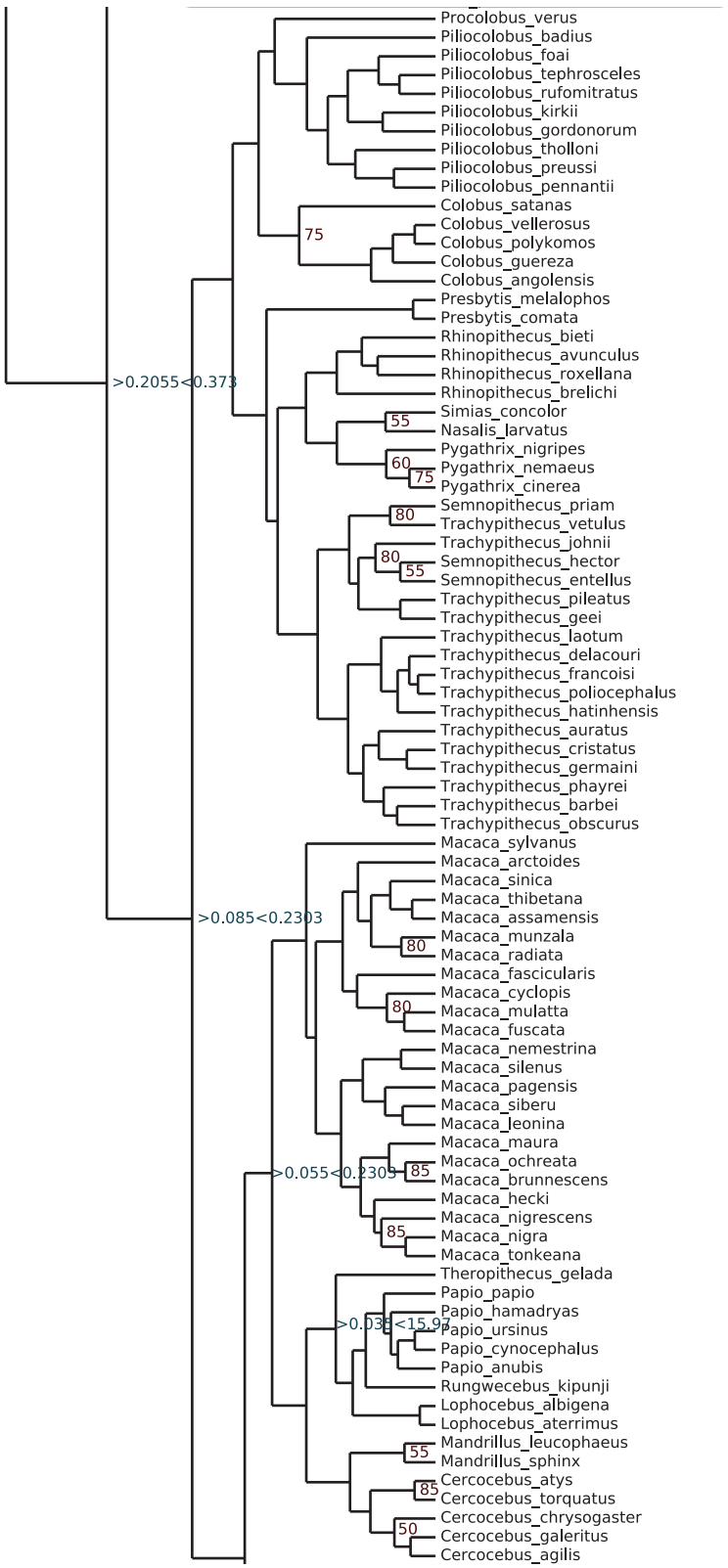
APPENDIX E

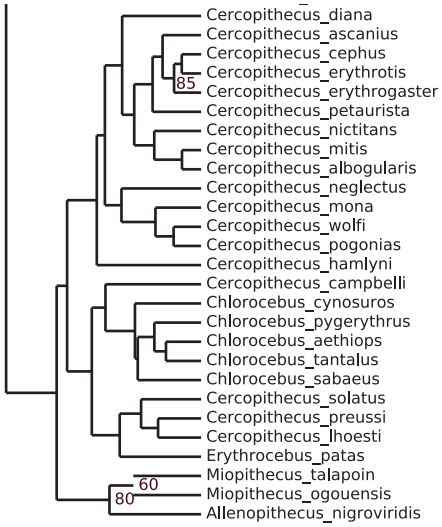
JACKKNIFE SUPPORT FOR THE PRIMATES




Lepilemur_seali
Lepilemur_wrighti
Lepilemur_mustelinus
Lepilemur_jamesorum
85
Lepilemur_betsileo
Lepilemur_fleuretae
Lepilemur_microdon
Lepilemur_otto
Lepilemur_edwardsi
Lepilemur_grewcockorum
Lepilemur_septentrionalis
Lepilemur_dorsalis
Lepilemur_mittermeieri
Lepilemur_ahmansoni
Lepilemur_sahamalazensis
Lepilemur_tymerlachsoni
Lepilemur_milanoii
Lepilemur_ankaranensis
Lepilemur_aeeclis
Lepilemur_randrianasoloi
Lepilemur_hubbardorum
Lepilemur_ruficaudatus
Lepilemur_leucopus
Lepilemur_petteri
Indri_indri
Propithecus_perrieri
80
Propithecus_edwardsi
Propithecus_diadema
Propithecus_verreauxi
Propithecus_coquereli
Propithecus_tattersalli
Avahi_laniger
Avahi_peyrierasi
Avahi_betsileo
Avahi_ramanantsoavani
Avahi_meridionalis
Avahi_occidentalis
Avahi_unicolor
Avahi_cleesei
Tarsius_sangirensis
Tarsius_lariang
Tarsius_wallacei
Tarsius_tarsier
Tarsius_dentatus
Tarsius_syrichta
Tarsius_bancanus
Chiropotes_utahickae
Chiropotes_chiropotes
Chiropotes_israelita
Cacajao_ayresi
Cacajao_hosomi
Cacajao_melanocephalus
Cacajao_calvus
Pithecia_pithecia
80
Pithecia_monachus
Pithecia_irrorata
Callicebus_lugens
Callicebus_torquatus
Callicebus_cupreus
Callicebus_moloch
Callicebus_hoffmannsi
Callicebus_brunneus
Callicebus_donacophilus
35
Callicebus_caligatus
Callicebus_nigrifrons
Callicebus_personatus
Callicebus_coimbrai
Cebus_libidinosus
Cebus_apella
80
Cebus_xanthosternos
85
85
Cebus_robustus
Cebus_capucinus
Cebus_albifrons
Cebus_olivaceus
>0.118<0.285
Cebus_kaapori
Saimiri_oerstedii
80
75
Saimiri_sciureus
Saimiri_ustus
Saimiri_boliviensis


>0.371<0.589
>0.118<0.373
>0.118<0.285
>0.283<0.56
>0.11608<0.285
>0.11608<0.285
>0.0511<0.1597
70
75
70
55
85
50
85
75
35
Aotus_vociferans
Aotus_lemurinus
Aotus_griseimembra
Aotus_trivirgatus
Aotus_nancymaae
Aotus_infulatus
Aotus_azarae
Aotus_nigriceps
Saguinus_tripartitus
Saguinus_melanoleucus
Saguinus_graellsi
Saguinus_nigricollis
Saguinus_fuscicollis
Saguinus_leucopus
Saguinus_labiatus
Saguinus_mystax
Saguinus_imperator
Saguinus_niger
Saguinus_midas
Saguinus_bicolor
Saguinus_martinsi
Saguinus_oedipus
Saguinus_geoffroyi
Callimico_goeldii
Callithrix_pygmaea
Callithrix_humeralifera
Callithrix_emiliae
Callithrix_mauesi
Callithrix_argentata
Callithrix_humilis
Callithrix_aurita
Callithrix_penicillata
Callithrix_jacchus
Callithrix_kuhlii
Callithrix_geoffroyi
Leontopithecus_rosalia
Leontopithecus_chrysopygus
Leontopithecus_chrysomelas
Ateles_paniscus
Ateles_marginatus
Ateles_chamek
Ateles_fusciceps
Ateles_geoffroyi
Ateles_belzebuth
Ateles_hybridus
Brachyteles_arachnoides
Brachyteles_hypoxanthus
Lagothrix_lugens
Lagothrix_cana
Lagothrix_poeppigii
Lagothrix_lagotricha
Alouatta_pigra
Alouatta_coibensis
Alouatta_palliata
Alouatta_caraya
Alouatta_macconnelli
Alouatta_seniculus
Alouatta_sara
Alouatta_guariba
Alouatta_belzebul
Nomascus_nasutus
Nomascus_hainanus
Nomascus_siki
Nomascus_leucogenys
Nomascus_gabriellae
Nomascus_concolor
Symphalangus_syndactylus
Hylobates_pileatus
Hylobates_albibarbis
Hylobates_lar
Hylobates_agilis
Hylobates_muelleri
Hylobates_klossii
Hylobates_moloch
Hoolock_leuconedys
Hoolock_hoolock
Pongo_abelii
Pongo_pygmaeus
Gorilla_beringei
Gorilla_gorilla
Homo_sapiens
Pan_troglodytes
Pan_paniscus


Procolobus_verus
Piliocolobus_badius
Piliocolobus_foai
Piliocolobus_tephrosceles
Piliocolobus_rufomitratus
Piliocolobus_kirkii
Piliocolobus_gordonorum
Piliocolobus_tholloni
Piliocolobus_preussi
Piliocolobus_pennantii
Colobus_satanas
Colobus_vellerosus
Colobus_polykomos
Colobus_guereza
Colobus_angolensis
Presbytis_melalophos
Presbytis_comata
Rhinopithecus_bieti
Rhinopithecus_avunculus
Rhinopithecus_roxellana
Rhinopithecus_brelichi
Simias_concolor
Nasalis_larvatus
Pygathrix_nigripes
Pygathrix_nemaeus
Pygathrix_cinerea
Semnopithecus_priam
Trachypithecus_vetulus
Trachypithecus_johnii
Semnopithecus_hector
Semnopithecus_entellus
Trachypithecus_pileatus
Trachypithecus_geei
Trachypithecus_laotum
Trachypithecus_delacouri
Trachypithecus_francoisi
Trachypithecus_poliocephalus
Trachypithecus_hatinhensis
Trachypithecus_auratus
Trachypithecus_cristatus
Trachypithecus_germaini
Trachypithecus_phayrei
Trachypithecus_barbei
Trachypithecus_obscurus
Macaca_sylvanus
Macaca_arctoides
Macaca_sinica
Macaca_thibetana
Macaca_assamensis
Macaca_munzala
Macaca_radiata
Macaca_fascicularis
Macaca_cyclopis
Macaca_mulatta
Macaca_fuscata
Macaca_nemestrina
Macaca_silenus
Macaca_pagensis
Macaca_siberu
Macaca_leonina
Macaca_maura
Macaca_ochreata
Macaca_brunnescens
Macaca_hecki
Macaca_nigrescens
Macaca_nigra
Macaca_tonkeana
Theropithecus_gelada
Papio_papio
Papio_hamadryas
Papio_ursinus
Papio_cynocephalus
Papio_anubis
Rungwecebus_kipunji
Lophocebus_albigena
Lophocebus_aterrimus
Mandrillus_leucophaeus
Mandrillus_sphinx
Cercocebus_atys
Cercocebus_torquatus
Cercocebus_chrysogaster
Cercocebus_galeritus
Cercocebus_agilis
>0.2055<0.373
>0.085<0.2303
>0.055<0.2305
>0.036<15.97
75
55
60
75
80
80
55
80
85
85
55
85
50


Cercopithecus_diana
Cercopithecus_ascanius
Cercopithecus_cephus
Cercopithecus_erythrotis
85
Cercopithecus_erythrogaster
Cercopithecus_petaurista
Cercopithecus_nictitans
Cercopithecus_mitis
Cercopithecus_albogularis
Cercopithecus_neglectus
Cercopithecus_mona
Cercopithecus_wolfi
Cercopithecus_pogonias
Cercopithecus_hamlyni
Cercopithecus_campbelli
Chlorocebus_cynosuros
Chlorocebus_pygerythrus
Chlorocebus_aethiops
Chlorocebus_tantalus
Chlorocebus_sabaeus
Cercopithecus_solatus
Cercopithecus_preussi
Cercopithecus_lhoesti
Erythrocebus_patas
Miopithecus_talapoin
60
80
Miopithecus_ogouensis
Allenopithecus_nigroviridis