# Formal Analysis of Phylogenetic Trees Under Uncertainty

An essay submitted in partial fulfillment of

the requirements for graduation from the

**Honors College at the College of Charleston**

with a Bachelor of Science in Data Science

Caroline Goodman

Spring 2023

Advisor: Dr. Kris Ghosh

**Abstract**

Phylogenetic studies play a crucial role in understanding the evolutionary relationships among species, individuals, and genes. However, constructing phylogenetic trees under imprecise data remains a challenging task. In this study, we propose a formalism that leverages stochastic models such as continuous-time Markov chains to address data imprecision. We use temporal logic to pose queries and specify properties that enable the evaluation of the formalism's computational feasibility. Probabilistic model checking using PRISM model checker is employed to verify the logic formulae, enabling researchers to focus on the model's outcomes rather than the accuracy and validity of the steps taken to reach the conclusion. Our study makes significant contributions to the field of phylogenetic studies by providing a framework that can handle imprecise data, ensuring more accurate evolutionary inferences.

# 1 Introduction

Phylogenetic studies have become an essential tool for understanding the evolutionary relationships between various species, individuals, and genes. These studies use phylogenetic trees to visualize and analyze the evolutionary history of organisms, with the ultimate goal of gaining insight into the patterns and mechanisms of evolution. However, the accuracy of these trees depends on the quality and precision of the data used to construct them, and dealing with imprecision and uncertainty in the data is a challenging problem. In recent years, researchers have turned to stochastic and probabilistic models, such as continuous-time Markov chains (CTMCs), to model the evolution of biological sequences and improve the accuracy of phylogenetic trees. This paper presents a formalism for constructing phylogenetic trees under imprecision of data using CTMCs and stochastics. Uncertainty is associated with phylogenetic trees because these trees are based on hypotheses about the evolutionary relationships between different organisms or sequences, and these hypotheses are subject to error and revision as new data and methods become available. In other words, phylogenetic trees are based on probabilistic models of evolution, which means that they are inherently uncertain and subject to error. One source of uncertainty in phylogenetic trees is the limited amount and quality of available data. Phylogenetic trees are typically constructed using molecular or morphological data, which can be subject to various sources of error such as incomplete sampling, sequencing errors, or homoplasy (convergence or reversal of characters due to independent evolution). The use of different methods and models of inference can also introduce uncertainty, as different approaches may yield different results or make different assumptions about the evolutionary process. Another source of uncertainty is the inherent complexity of evolutionary history. The true evolutionary relationships between different organisms or sequences may be more complex than can be represented by a single phylogenetic tree, and different parts of the tree may have different levels of support or confidence. In some cases, multiple competing hypotheses may be equally well-supported by the available data, making it difficult to determine the true evolutionary relationships. Ultimately, uncertainty is an inherent part of phylogenetic inference, and researchers must carefully consider and evaluate the sources and extent of this uncertainty when interpreting and using phylogenetic trees for evolutionary and biological studies. This formalism incorporates temporal logic specifications to evaluate the computational feasibility of the models, and PRISM model checking is used to assess the logic formulae. The use of these modeling techniques allows researchers to focus on the outcomes of their models, rather than the accuracy and validity of the steps taken to reach those conclusions. Overall, this work demonstrates the potential of stochastic and probabilistic models in improving the accuracy and reliability of phylogenetic studies.

# 2 Preliminaries

## 2.1 Phylogenetic Trees

Phylogenetic trees are diagrams that represent the evolutionary relationships among different organisms, genes, or other biological entities. They are constructed based on the shared characteristics or traits among these entities and are used to infer evolutionary histories, patterns of speciation, and genetic diversity. The construction of these trees is based on the principle of common descent. Common descent suggests that all living organisms share a common ancestor. This implies that the closer the relationship between two organisms, the more recent their common ancestor, and the more similar their characteristics or traits. Phylogenetic trees are widely used in many fields of biology, including ecology, evolutionary biology, systematics, genetics, and biogeography, among others. They provide a powerful tool for investigating the evolutionary history of life on Earth, understanding the relationships between organisms, and predicting future patterns of evolution.

## 2.2 Continuous-time Markov Chains

Continuous-time Markov chains (CTMCs) are stochastic models that are used to describe the evolution of a system over time in a probabilistic manner. CTMCs are widely used in various fields, including physics, chemistry, engineering, economics, and biology, among others, to model complex systems that exhibit random and unpredictable behavior. In CTMCs, a system is represented by a set of states, and transitions between these states occur randomly in continuous time, following the exponential distribution [2]. The probability of transitioning from one state to another depends only on the current state of the system and the transition rates between states. The transition rates are defined by a transition rate matrix, which specifies the rate of transitioning from one state to another. CTMCs have numerous applications, including modeling the spread of infectious diseases, the dynamics of biochemical networks, the behavior of financial markets, and the evolution of biological sequences, among others. CTMCs are also used in the field of phylogenetics to model the evolution of DNA and protein sequences, where they are used to estimate the rates of nucleotide or amino acid substitutions, which can be used to infer evolutionary relationships among species. Overall, CTMCs provide a flexible and powerful tool for modeling complex systems that exhibit stochastic behavior, making them a valuable tool for a wide range of applications in various fields of science and engineering.

## 2.3 Stochastic Processes

Stochastic processes are mathematical models that describe the behavior of random or unpredictable events. These events can occur in various fields, including physics, finance, engineering, biology, and computer science, among others. Stochastic processes provide a framework for modeling and analyzing systems that exhibit random behavior, allowing us to make probabilistic predictions about the future of the system. Stochastic models can be used to simulate the behavior of complex systems, such as the stock market, the spread of infectious diseases, the behavior of ecological communities, and the dynamics of biochemical networks, among others. Stochastic processes can also be used in machine learning, where they are used to model uncertainty in data and make predictions based on probabilistic models. Stochastic processes are often described by probability distributions, which specify the likelihood of different outcomes occurring. Overall, stochastic processes are a powerful tool for modeling complex systems that exhibit random or unpredictable behavior [3]. They provide a framework for making probabilistic predictions about the future of a system, allowing us to better understand and analyze the behavior of complex systems.

## 2.4 PRISM Model Checker

PRISM (`http://www.prismmodelchecker.org`) is a software tool for modeling, analyzing, and verifying probabilistic systems [5]. It is widely used in various fields, including computer science, engineering, and biology, among others, to model and analyze complex systems that exhibit random or unpredictable behavior. PRISM provides a flexible and powerful framework for modeling systems using different types of probabilistic models, including Markov decision processes (MDPs), continuous-time Markov chains (CTMCs), and probabilistic automata. One of the key features of PRISM is its model checking capabilities. Model checking is a formal verification technique used to verify that a given model satisfies a set of desired properties or specifications. PRISM uses various model checking algorithms to automatically analyze probabilistic models and check whether they satisfy certain desired properties. These properties can include safety properties, liveness properties, reachability properties, and temporal logic specifications, which is used in our formalism [3]. PRISM provides a range of analysis tools, including simulation, verification, and synthesis tools, to help users understand the behavior of their models and analyze them systematically. PRISM is a powerful tool for modeling and analyzing complex systems that exhibit random or unpredictable behavior. Its model checking capabilities make it particularly useful for verifying the computational feasibility of probabilistic models and ensuring that they satisfy desired properties and specifications.

## 2.5 Phylogenetic Model Checking Components

A phylogenetic tree is a rooted, labeled tree whose vertices represent the population of individuals depicted by their DNA. The vertices are further arranged along the tree in a fashion that reflects an evolutionary process. Kripke structures are also important to the process of model checking phylogenetic trees. A Kripke structure is a finite transition system defined on an atomic proposition (a statement or assertion that must be true or false) represented by a tuple: $M = (S, S_0, R, L)$ where

1. $S$ represents a finite set of states.

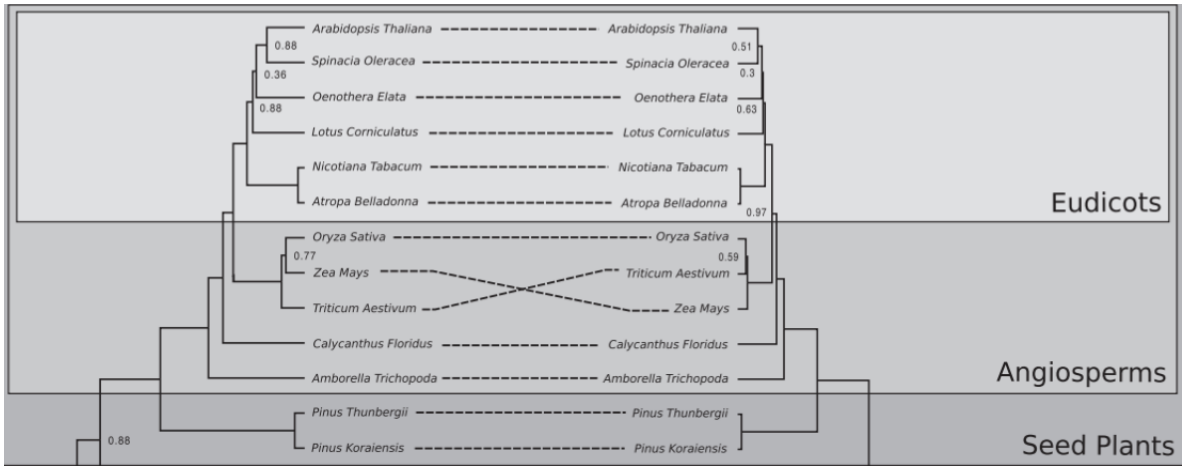



Figure 1: Phylogenetic tree representing the speciation of eudicots, angiosperms, and seed plants

2. $S_0$ is a subset of S that represents the set of initial states.
3. $R$ represents a total transition relation between states (for every element $s \in S$, there exists element $t$ in $S$ such that $(s, t)$ is in $R$.
4. $L$ is the labeling function that associates each state with the subset of atomic propositions that are true of it.

Branching-time is the distance from each node to the tips. In our work, we perform model checking on continuous-time Markov Chain using continuous stochastic logic (CSL) [2]. Model checking of a phylogenetic model was described by Blanco et al [3]. Sampling based models in phylogenetic analysis have been published [6]. There has been research that has focused on statistics measure of uncertainty of construction of phylogenetic trees using molecular sequences [7]. There has been ongoing research in construction of phylogentic trees under uncertainty [4].

# 3 Model

The data used as the phylogenetic tree model in this project was extracted from research conducted by Baele et al. [1], as shown in Figure 1.

Using the graphviz Python package, graphics of the model were created to visualize the PRISM code and simplify the visualization of the original phylogenetic tree. In Figure 2, each state represents either a branching point ($nt_ab$) or the formation of a species ($nt$). A rate is associated with each transition between states. The notion of uncertainty is implemented in these rates of transition. With regards to the naming conventions in the model, we used abbreviations of the various species based on the first letters of each organism.

For example, at in the model represents the organism *ArabidopsisThaliana*. Further, zm represents Zea mays. This simplified the creation of the model, and further improved the readability of the PRISM code. In Figure 3, you can find the translation of this tree into PRISM language.

Figure 3 represents a smaller subset of the entirety of the tree, six species in total. This code is a model of a Continuous Time Markov Chain (CTMC) in the probabilistic model checking language of PRISM. The variables 'at', 'so', 'oe', 'lc', 'nt', and 'ab' represent the states of various species of plants ('Arabidopsis Thaliana', 'Spinacia Oleracea', 'Oenothera Elata', 'Lotus Corniculatus', 'Nicotiana Tabacum', and 'Atropa Belladonna'). These variables are initialized to 'false' at the initial state. The boolean variables '*at_so_oe_lc_nt_ab*', '*at_so_oe_lc*', '*nt_ab*', '*at_oe*', and '*at_so*' represent different combinations of plant species that are present or absent.

These variables are initialized to 'true', 'false', 'false', 'false', and 'false', respectively, at the initial state. The '$tr1$' to '$tr10$' labels denote transitions in the CTMC model, each of which is associated with a rate ('$r1$' to '$r10$') defined in the module (not shown in figure).

The transitions update the values of the boolean variables and change the state of the system.

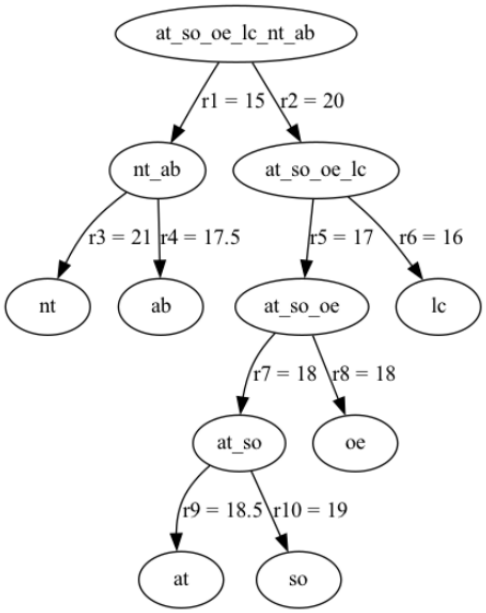



Figure 2: Visualization of Phylogenetic Tree

```
[tr1] (at_so_oe_lc_nt_ab = true) -> r1: (at_so_oe_lc_nt_ab' = false) & (nt_ab' = true);

[tr2] (at_so_oe_lc_nt_ab = true) -> r2: (at_so_oe_lc_nt_ab' = false) & (at_so_oe_lc' = true);

[tr3] (nt_ab = true) -> r3: (nt_ab' = false) & (nt' = true);

[tr4] (nt_ab = true) -> r4: (nt_ab' = false) & (ab' = true);

[tr5] (at_so_oe_lc = true) -> r5: (at_so_oe_lc' = false) & (at_so_oe' = true);

[tr6] (at_so_oe_lc = true) -> r6: (at_so_oe_lc' = false) & (lc' = true);

[tr7] (at_so_oe = true) -> r7: (at_so_oe' = false) & (at_so' = true);

[tr8] (at_so_oe = true) -> r8: (at_so_oe' = false) & (oe' = true);

[tr9] (at_so = true) -> r9: (at_so' = false) & (at' = true);

[tr10] (at_so = true) -> r10: (at_so' = false) & (so' = true);
```

Figure 3: Prism code Representation for Phylogenetic Tree in Figure 1

For example, '$tr1$' denotes a transition in which the state '$at\_so\_oe\_lc\_nt\_ab$' changes from 'true' to 'false' and the state '$nt\_ab$' changes from 'false' to 'true', with a rate of '$r1$'. The next model contains 10 species, and the last model created represented all 13 of the species included in Figure 1. These slight variations in the model will give us more data when verifying properties against the model and, ultimately, comparing PRISM's computation times. In these three models ($N = 6, N = 10, N = 13$), the rates associated with the transitions between states remain constant, but are used in later models to create further variability. Most of the varied models were derived from the $N = 6$ model, using the static rates defined already. As you can see in the tree above, the static rates from the

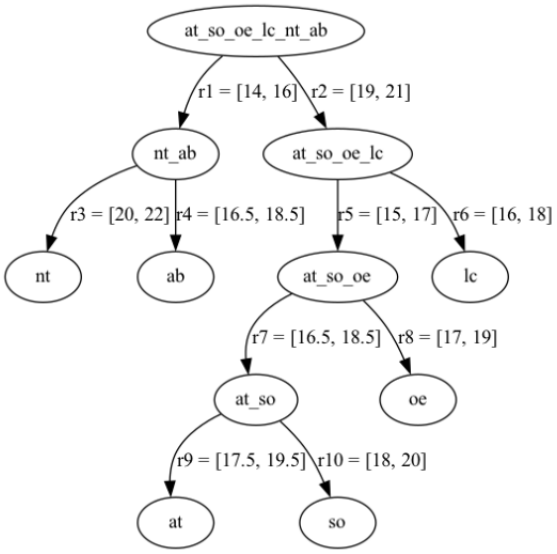



Figure 4: Range 1 of Phylogenetic Tree

previous tree were used to form a variation of the model. In this variation, the static rate was taken and turned into a range representing +/-1 from the original range. A random number is generated between this range and assigned to the respective rate. In the figure above, you can see that each of the rates specified is a float generated

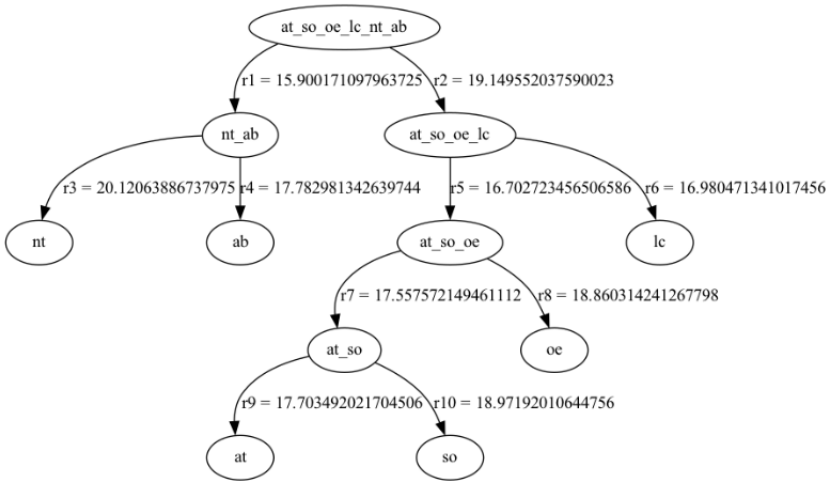
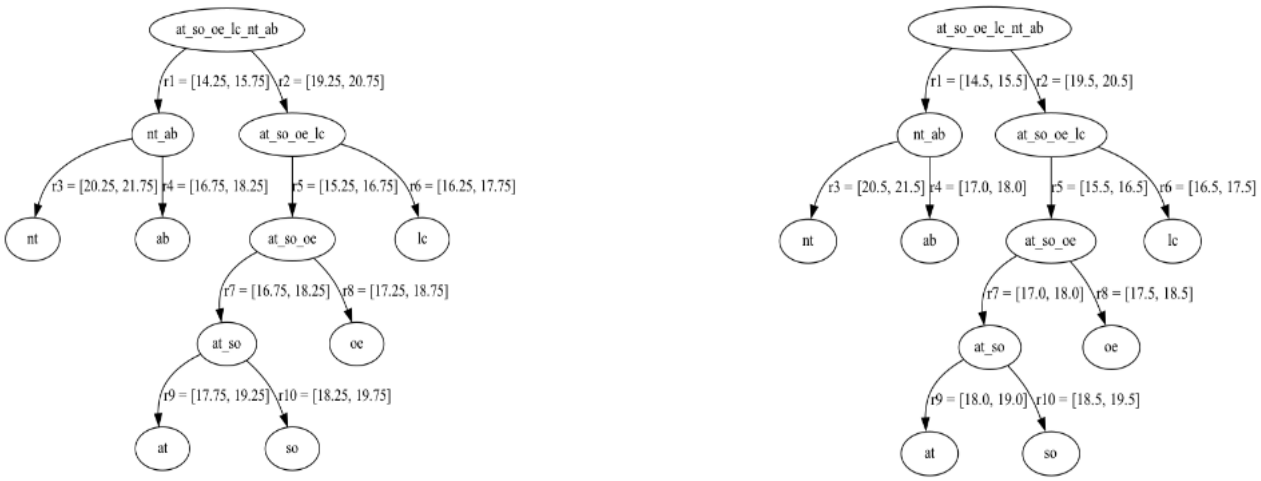



Figure 5: Random Rates of Phylogenetic Tree

within the specified range in the original tree. This model (in the +/-1 range) is generated ten times using an automated python script. Next, ten more models are created in the +/- 2 range, +/- 0.5 range, and +/- 0.75 range (as shown below in order as previously listed).

The creation of these varying models forms the foundation of this research, as it is where we incorporate the uncertainty into our model. These stochastic models give valuable insight, as phylogenetic trees are often uncertain




(a) (b)

Figure 6: Different ranges for Rates

and require a certain level of confidence to draw meaningful conclusions. After the generation of these models, the values are saved into a dataframe to perform analyses on the variation among each tree. In the end, there are over 60 model variations.

In the Figure 7, we have visualizations of the variation amongst each model variation, and further, the discrepancies among the ten randomized models of each variation. This data is based on the randomly generated rates based on the ranges specified. The figure is provided for the sole purpose of conveying the variations among a select few models involved in the total calculations.

# 4 Evaluation of Temporal Logic Properties

Specifying properties is the foundation of the PRISM model checking process. Property specification is important because it allows you to formally verify the correctness of your model. By specifying properties, you are essentially asking PRISM to check whether certain conditions hold true in your model. For example, you may want to verify that a particular state is reachable or that a certain event will eventually occur. By verifying these properties, you can gain confidence that your model is correct and behaves as expected. This is particularly important in safety-critical systems, where incorrect behavior could result in serious consequences. In addition, verifying properties using PRISM can also help you identify and fix errors in your model. If a property fails to hold true, you can use the diagnostic information provided by PRISM to identify the cause of the failure and make appropriate changes to your model. In this model, we are focused on reachability properties and temporal logic properties. The specified properties are as follows:

**Property 1:** $P =?[(ab\&nt\&oe)U(ab'\&nt'\&oe')\ ]$ This property asks whether eventually all three species ($sp1, sp2$, and $sp3$) become present in the model, which is determined by the reachability of the state where all three species are present. Note: This property's result will always evaluate to 0.0 because our model checks to see if a single species formed, but not multiple species.

**Property 2:** $P =?(nt = true)$ This is a reachability property asking if it is possible to reach a state where species ab is present in the model.

**Property 3:** $P =?(ab = true)$ This is a reachability property asking if it is possible to reach a state where species ab is present in the model.

**Property 4:** $P =?(at = true)$ This is a reachability property asking if it is possible to reach a state where species at is present in the model.

**Property 5:** $P =?(so = true)$ This is a reachability property asking if it is possible to reach a state where species so is present in the model.

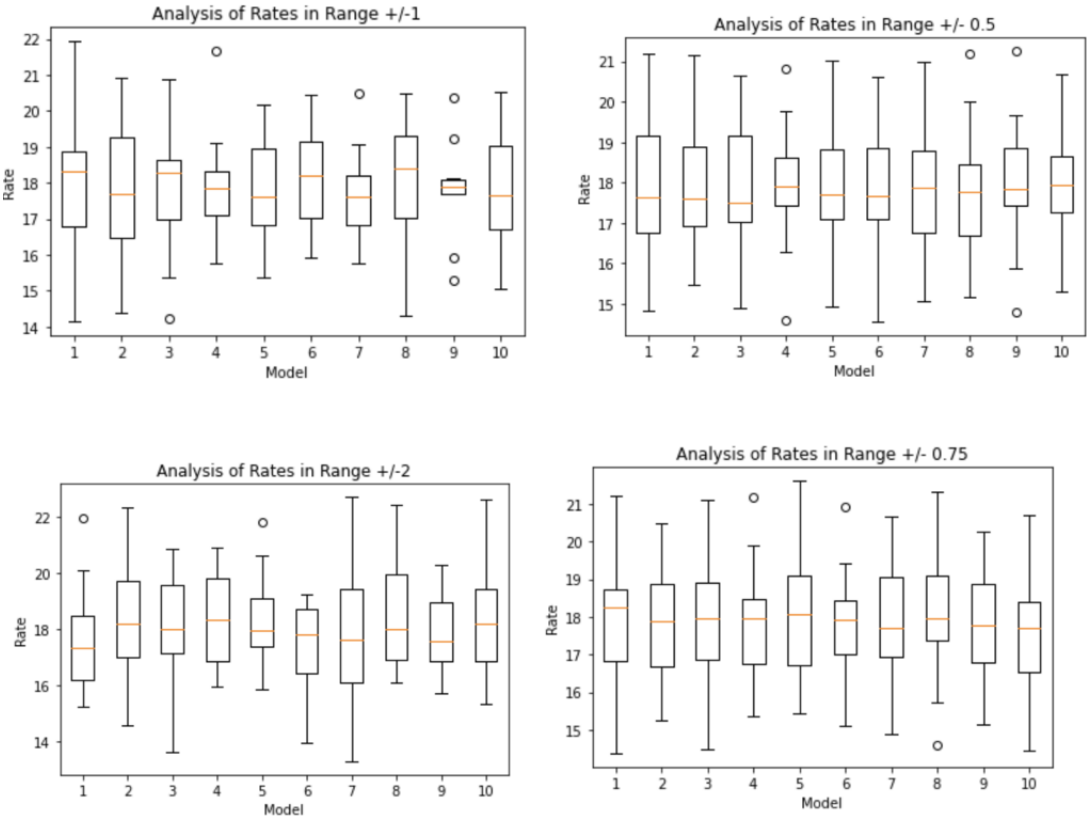



Figure 7: Random Rates of Phylogenetic Tree

**Property 6:** $P =?(oe = true)$ This is a reachability property asking if it is possible to reach a state where species oe is present in the model.

**Property 7:** $P =?(lc = true)$ This is a reachability property asking if it is possible to reach a state where species lc is present in the model.

These models and properties are imported into PRISM Model Checking software. The properties are then verified against the model, and the time taken to model check and the result are recorded.

The data presented in the Table 1 consists of the results obtained from checking a model for different variations of input size and properties. The model checking was performed using a probability metric, the value of which is presented in the "Result" column. Here, I will analyze this data and provide an in-depth analysis. The data consists of results obtained for seven different input sizes: n=6, n=10, and n=13, with varying properties. There are over 40 models, which refer to the varied rates specified earlier. Each variation has been checked for the same set of properties. For n=6, the results are further broken down into seven different variations of the input. For each input size and property combination, the "Time Taken for Model Checking" and "Result" are presented. The properties checked are numbered from 1 to 7. The time taken for model checking is recorded in seconds, and the result indicates whether the model satisfies the property checked. For example, for n=6 and property 2, the time taken for model checking was 0.009 seconds, and the result was 0.212459016, indicating that the model satisfied the property. The "Time Taken for Model Checking" column represents the time taken to perform the model checking for the given input size and property combination. This time can be used to compare the efficiency of the model checking process for different input sizes and properties. As expected, the time taken increases with the increase in the input size. For example, the time taken for model checking n=6 is significantly less than the time taken for n=13. However, there are some variations in time taken for the same input size, indicating that the properties of the input can also have an impact on the time taken for model checking. The "Result" column represents the metric used to evaluate the model checking process. The specific metric used is not defined, but it is clear that lower values of the metric are better. For all variations of input, the "Result" is less than 0.5, which indicates that the model checking process is effective. However, there are variations in the values of the "Result" for the same

| Model | Property | Time | Result |
|---|---|---|---|
| n=6 | 1 | 0.088 | 0 |
| | 2 | 0.009 | 0.21245902 |
| | 3 | 0.001 | 0.14754098 |
| | 4 | 0.001 | 0.071091 |
| | 5 | 0.001 | 0.07498668 |
| | 6 | 0 | 0.15454523 |
| | 7 | 0.001 | 0.33937615 |
| n=10 | 1 | 0.018 | 0 |
| | 2 | 0.202 | 0.31032538 |
| | 3 | 0.202 | 0.22084265 |
| | 4 | 0.245 | 0.13076218 |
| | 5 | 0.23 | 0.124134 |
| | 6 | 0.227 | 0.23570826 |
| | 7 | 0.195 | 0.41232858 |
| n=13 | 1 | 0.453 | 0 |
| | 2 | 41.65 | 0.31032538 |
| | 3 | 40.76 | 0.22084265 |
| | 4 | 53.945 | 0.13076218 |
| | 5 | 52.851 | 0.124134 |
| | 6 | 45.922 | 0.23570826 |
| | 7 | 40.515 | 0.41232858 |

Table 1: Execution of Time of Model Checking Queries

input size and property combination, indicating that the properties of the input can also have an impact on the effectiveness of the model checking process.

## 4.1 Analysis of Properties

Table 2 shows some properties are satisfied for all values of n, while others are not. Property 1 seems to be satisfied for all values of n, as indicated by the result of 0 for all entries. Property 7 is not satisfied for any value of n, as indicated by the result being 0 for all entries. This is due to the nature of our model. The model is not interested in whether multiple species formed, but whether or not a single organism is formed. Therefore, because Property 7 becomes false after the second transition, the result will always result in 0, which follows our predictions. To calculate the average result for each individual property, the data was grouped based on the property and the mean of the result column for each group was taken. Here is the table with the average result for each individual property (averages are rounded to four decimal places): The average result of each property reveals that some properties are

| Property | Average Result |
|---|---|
| 1 | 0.0 |
| 2 | 0.227723931 |
| 3 | 0.160131871 |
| 4 | 0.077751529 |
| 5 | 0.078001573 |
| 6 | 0.160862186 |
| 7 | 0.349791542 |

Table 2: Probabilities of Properties

more likely to be present in some models than others. Property 7 has the highest average probability of presence, with a 34.9% chance of eventually being present at some point in the model checking process. This could be due to various reasons, including but not limited to, the particular organism specified in the property and the values of the rates of transition associated with the property.

*Time Analysis for Model Checking:* Based on the data in Table 1 provided, it can be concluded that the time taken for model checking is significantly influenced by both the size of the model and the complexity of the property being checked. As the size of the model and complexity of the property increase, the time taken for model checking also increases. It is worth noting that the increase in time taken for model checking is dramatic. For example, the model checking for n=6 took only a few milliseconds, while the model checking for n=13 took over 40 seconds. Furthermore, the effect of model size on the time taken for model checking is more pronounced compared to the effect of property complexity. These findings suggest that when designing models and properties for model checking, consideration should be given to minimizing the size of the model and keeping the properties as simple as possible, without compromising their expressiveness.

*Analysis Results of Temporal Logic Queries:* Table 3 shows the comparison of a property across all the models. Property 2 is defined as follows: P=? [F nt = true]. This property checks whether the state 'nt' is eventually reached during the execution of the system. The result of the property verification for different models of varying sizes are given in the table below: For the model with n=6, the model checker took 0.009 seconds to verify the

| Model | Property | Time for Execution | Result |
|---|---|---|---|
| n =6 | 2 | .009 | 0.212459 |
| n =10 | 2 | 0.202 | 0.310325 |
| n =13 | 2 | 40.76 | 0.220842 |
| n =6 (variation 1) | 2 | .001 | 0.187863 |

Table 3: Comparison of time of execution across different models

property and returned a result of 0.212459. This means that there is a 21.25% chance that the system will eventually reach the state 'nt' during execution. For the model with n=10, the model checker took 0.202 seconds to verify the property and returned a result of 0.310325. This means that there is a 31.03% chance that the system will eventually reach the state 'nt' during execution. For the model with n=13, the model checker took 40.76 seconds to verify the property and returned a result of 0.220842649. This means that there is a 22.08% chance that the system will eventually reach the state 'nt' during execution. For the model with n=6 (variation 1), the model checker took 0.001 seconds to verify the property and returned a result of 0.18786327. This means that there is a 18.79% chance that the system will eventually reach the state 'nt' during execution. These results remain similar for each property, as the input size and model variables are altered. Based on these results, we can conclude that the larger the size of the model, the longer it takes for the model checker to verify the property. Additionally, we can see that the probability of reaching the 'nt' state during system execution is higher in the model with 'n=10' compared to the other models. This information can be useful in determining the feasibility of the system and making design decisions to optimize the system performance.

# 5 Discussion

The analysis of this data suggests that the efficiency and effectiveness of the model checking process can be affected by the size and properties of the input. Specifically, larger input sizes require more time for model checking, and the properties of the input can affect both the time taken for model checking and the effectiveness of the process. These observations indicate that careful consideration of the input size and properties is necessary when performing model checking. In particular, when dealing with larger input sizes, more time is required for the model checking process to complete. This is because larger inputs involve more states and transitions, which means that the state space is much larger and more complex. As a result, the model checker has to explore a larger number of possible paths through the state space, which can increase the overall time taken for model checking. Furthermore, the properties of the input can also impact the time taken for model checking and the effectiveness of the process. For example, inputs that have highly interconnected components or complex dependencies may require more time to check due to the need to explore multiple possible paths through the state space. Additionally, inputs with certain properties may be more difficult to verify, meaning that the model checking process may be less effective in identifying errors or verifying properties. Taken together, these observations suggest that careful consideration of the input size and properties is necessary when performing model checking. Model checkers need to be able to handle large inputs efficiently, and they need to be able to cope with inputs that have complex properties or dependencies. By taking these factors into account, researchers and practitioners can ensure that they use model checking in a way that is both efficient and effective, leading to better quality and more reliable software.

# 6 Conclusion

Ultimately, the formalism presented in this paper provides a powerful framework for constructing phylogenetic trees under imprecision of data, using stochastic and probabilistic models. The incorporation of temporal logic specifications and PRISM model checking enables researchers to assess the computational feasibility of the models and evaluate the accuracy of the results. These modeling techniques have the potential to improve the accuracy and reliability of phylogenetic studies, allowing researchers to gain deeper insights into the evolutionary relationships between species, individuals, and genes. As data continues to accumulate and become increasingly complex, the use of stochastic and probabilistic models will likely become even more important in phylogenetics and other areas of biological research. By combining mathematical and computational approaches with biological insights, we can better understand the processes of evolution and the diversity of life on our planet.

# References


[1] Guy Baele, Mandev S Gill, Paul Bastide, Philippe Lemey, and Marc A Suchard. Markov-modulated continuous-time markov chains to identify site-and branch-specific evolutionary variation in beast. *Systematic biology*, 70(1):181–189, 2021.

[2] Christel Baier, Boudewijn Haverkort, Holger Hermanns, and Joost-Pieter Katoen. Model checking continuous-time markov chains by transient analysis. In *Computer Aided Verification: 12th International Conference, CAV 2000, Chicago, IL, USA, July 15-19, 2000. Proceedings 12*, pages 358–372. Springer, 2000.

[3] Roberto Blanco, Gregorio de Miguel Casado, José Ignacio Requeno, and José Manuel Colom. Temporal logics for phylogenetic analysis via model checking. In *2010 IEEE International Conference on Bioinformatics and Biomedicine Workshops (BIBMW)*, pages 152–157. IEEE, 2010.

[4] John P Huelsenbeck, Bruce Rannala, and John P Masly. Accommodating phylogenetic uncertainty in evolutionary studies. *Science*, 288(5475):2349–2350, 2000.

[5] Marta Kwiatkowska, Gethin Norman, and David Parker. Prism 4.0: Verification of probabilistic real-time systems. In *Computer Aided Verification: 23rd International Conference, CAV 2011, Snowbird, UT, USA, July 14-20, 2011. Proceedings 23*, pages 585–591. Springer, 2011.

[6] Patricio Maturana Russel, Brendon J Brewer, Steffen Klaere, and Remco R Bouckaert. Model selection and parameter inference in phylogenetics using nested sampling. *Systematic biology*, 68(2):219–233, 2019.

[7] Borys Wróbel. Statistical measures of uncertainty for branches in phylogenetic trees inferred from molecular sequences by using model-based methods. *Journal of applied genetics*, 49:49–67, 2008.