Chapter 10

Simulating the evolution of genomes

Contents

I have been involved in a work that is related to an algorithm, PhyChro, which has been developed in the lab by Guénola Drillon. The algorithm is part of a bigger package that also includes the SynChro algorithm that has already been published (Drillon et al., 2014). The purpose of PhyChro is to reconstruct phylogenetic trees from synteny blocks. My work here was to create a program that simulates the evolution of genomes, in order to benchmark the PhyChro algorithm. This benchmark will be included in the revised version of the PhyChro manuscript[1], as running simulations was requested by the reviewers.

[1]Drillon, G., Champeimont, R., Oteri, F., Fischer, G., and Carbone, A. Phylogenetic reconstruction based on chromosomal rearrangements

10.1 Phylogenetic tree inference with PhyChro

The purpose of these two programs is to reconstruct phylogenetic trees using synteny blocks. SynChro computes the synteny blocks from the species genomes and their annotations, while PhyChro uses the synteny blocks computed by SynChro to reconstruct the phylogenetic tree.

Let G and G' be two genomes. We call a synteny block a succession of genes, for example 3 genes A, B, C, such that, if we consider their respective homologous genes A', B', C', they appear successively and in the same relative order in G'. This means that somewhere in G', there is the sequence $A'B'C'$ or $C'B'A'$. In fact, in SynChro, the $A'B'C'$ sequence is allowed to contain some other inserted genes that are homologous to genes not in the synteny block (for example $A'B'D'C'$), provided they are less than Δ, which is a parameter of the algorithm.

The first task SynChro has to perform is homologous gene matching, i.e. it has to find the corresponding A', B' and C' genes in the G' genome of our example. In fact, a gene may have several homologous genes in the other genome. This case is handled by SynChro by considering reciprocal best hits (RBH), which are pairs of genes (A, A'), with A in G and A' in G', such that A' is the closest gene to A in G' and A is the closest gene to A' in G. Both conditions may not hold, in which case SynChro marks it as a non-RBH homologous pair of genes.

Using both the information from RBH and non-RBH homologous genes, SynChro generates a list of synteny blocks. For a more detailed explanation of the algorithm, see Drillon et al. (2014).

In order to reconstruct the phylogenetic tree of a group of species, one has to run SynChro on all pairs of species, in order to find synteny blocks between all pairs of genomes. Then, PhyChro takes this information and is able to reconstruct the phylogenetic tree of these species.

10.2 Motivation for simulations

The PhyChro program has been applied to a set of 13 vertebrate species and 21 yeast species, and was able to reconstruct the phylogenetic tree correctly. A biological validation is a very good way to assess the algorithm performance, because it is biological applications that matter in the end, and nothing can be more realistic that biological data itself. However, there are still motivations for using simulations to further benchmark the algorithm:

- A simulation allows a large-scale testing, that can show the robustness of the algorithm, even if biological data is scarce. We can then be sure it did not find the correct answer "by chance" but works reliably.

- Since PhyChro provides confidence scores, a simulation can tell us whether these scores are useful information, in which case a high score should be correlated to a correct reconstruction.

- A simulation allows us to test the limits of the algorithm, and to know how much data and which data quality is required for the algorithm to work.

However, we should avoid the pitfalls related to simulations:

- Making the model so close to an experiment that it does not bring new information compared to the benchmark on biological data.
- Making the model so complex that we cannot know reliably the parameters to set.
- Making the model too simplistic so that it is not representative at all of what biological data would be like.
- Setting parameters that are good for the reconstruction program instead of being realistic.

We first tried to use the program MagSimus 2 developed (but not published yet) by Hugues Roest Crollius at the *Institut de Biologie de l'Ecole Normale Supérieure*. However, this program requires many parameters that are difficult to estimate, like the probability of each kind of event on each branch (problem 2 above). It is possible to fit it with biological data, but we thought it would make the simulation too close to the biological data (problem 1 above). We think the right level of imitation of the biological data is to take a yeast-like and a vertebrate-like ancestor, but not to re-create the same species that were studied, i.e. we do not want species-specific parameters. As a result, we decided to create our own simulation tool.

10.3 The model

I worked in collaboration with Gilles Fischer (head of the Biology of Genomes team in the lab) to create the model. I designed the mathematical formulation of the model and wrote the program, while he provided the biological knowledge of what events we should simulate, how many, with which frequency, etc. It is especially important that the model is realistic so we need to set the parameters to values which are the closest possible to biological truth.

10.3.1 Making the tree

We decided to create two different simulations, one that would mimic the evolution of vertebrates, and one that would mimic the evolution of yeasts. We want to get 13 vertebrates species and 21 yeast species, which have the same number of species as in the analysis reported in the PhyChro article.

We define time as going from 0, at the root, to 1, when we get the final species (leaves of the tree). We put the first branching at the root, as we are not interested in simulating events that are common to all species. Then, to get the right number of species, we choose randomly speciation times (branching) uniformly over $[0, 1]$. Then, for each speciation time, we have to split a leaf in two. We select that leaf by starting from the root of the tree, then, at each branching, we choose one of the two directions with a $1/2$ probability. When we reach a leaf, we split it in two at the speciation time previously defined. We repeat this operation until we have the right number of species.

Note that this procedure implies that there is not an identical probability to split every leaf, since a leaf that can be reached in less internal nodes is likelier to be chosen at every step. This has the consequence to make the tree more equilibrated than if leaves were chosen uniformly.

We do not claim that this tree shape is necessarily a perfect simulation of actual speciation in nature. Creating realistic speciation models is hard, and would not in fact be really relevant, because PhyChro is designed to be used on actually sequenced and annotated genomes, which are not randomly selected in the set of all existing species. There is a bias towards some species that are easier to find or study, more interesting than others, either for biological properties, industrial or medical applications. Some species are more ubiquitous than others and therefore discovered earlier, etc. The result is that we should not expect the set of studied species to be a uniform choice over all clades. This is especially obvious when looking at the vertebrates tree, which contains as many apes as fishes, while the number of species in these two groups have in fact a different order of magnitude.

An example of a tree generated with this method can be seen on Figure 10.1.

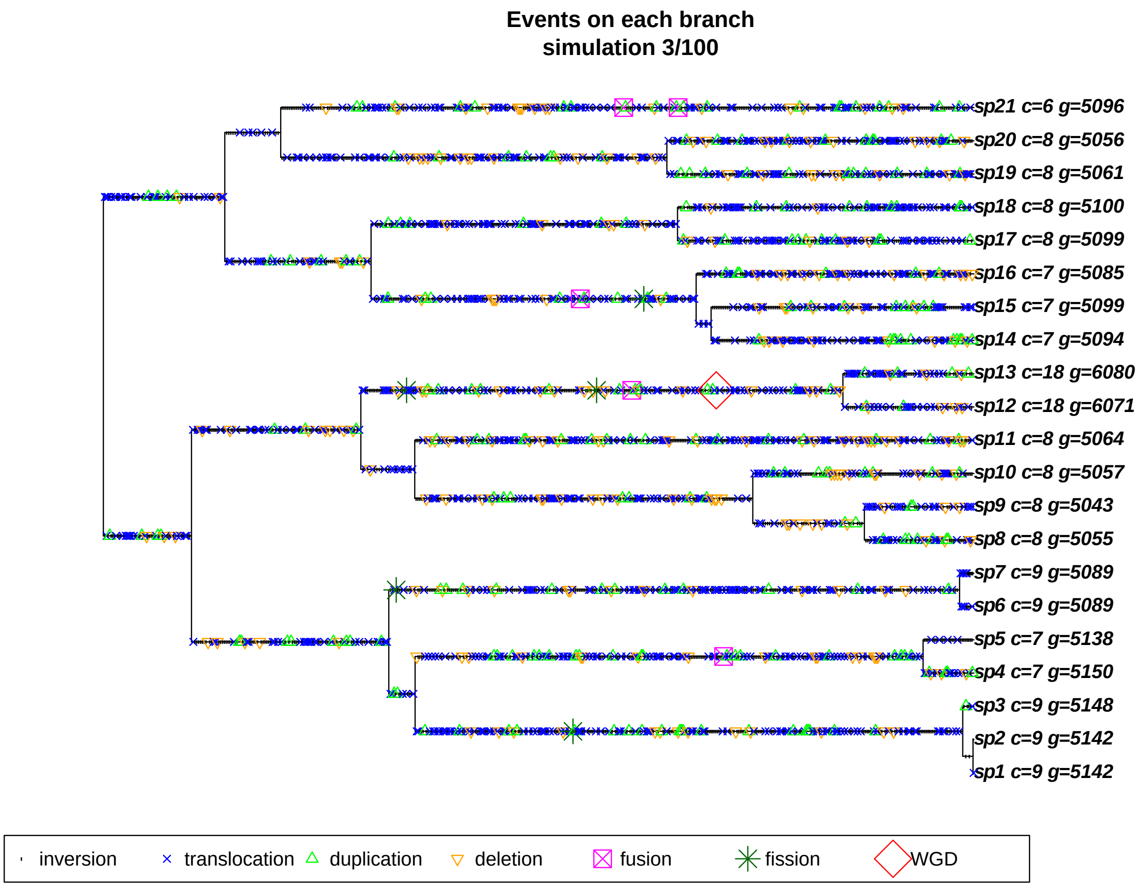

Figure 10.1: **A simulated tree with rearrangements events.**
The ancestor was given 5000 genes and 8 chromosomes, which are thought to be similar to the actual yeast ancestor (Gordon et al., 2009). The number of final simulated genes and chromosomes in each artificial species is given (resp. *g* and *c*). We have chosen to show this specific simulated tree (simulation # 3 in a set of 100) because it is the first to contain a WGD.

10.3.2 Genome model

We start with an ancestor with a number of genes and chromosomes which correspond to what can be inferred about the vertebrate and yeast ancestors. For the vertebrate ancestor, we have chosen 23 chromosomes and 18000 genes, while for the yeast ancestor, we have chosen 8 chromosomes and 5000 genes (Gordon et al., 2009). These genomes are implemented as lists of signed integers. Each integer represents a gene. A special case is the 0 which is used to mark the centromere. If two integers have an equal absolute value, it means that the genes are homologous: orthologs when between different species, paralogs within the same species. A different sign means the gene is on a different strand. In the ancestor, all genes are different (i.e. all integers appear exactly once).

10.3.3 Simulating the number of rearrangements

One of the parameters we define is the average number of events E from the root to any species. We know approximately what this number is if we take the two most remote species, estimate the number of rearrangements between them, and divide by two. This has led us to choose $E = 500$ events for yeasts and $E = 1000$ events for vertebrates (Drillon and Fischer, 2011).

To choose the number of events on each specific branch (internal or leading to a leaf), we use a Poisson distribution and set its parameter to $\lambda = E \times L$ where L is the branch length (in the time unit defined in the previous section). Choosing a Poisson distribution means that we make the hypothesis that the probability for a rearrangement to occur is the same at any point in the tree. Note that this Poisson distribution has the consequence that the expected number of events from the root to any species is E, which is consistent with the definition of this parameter.

There is only one exception to this rule, if a specific branch has less than m events (when chosen by the Poisson distribution), we still put m events on it. We choose $m = 1$, which means we forbid branches with no events at all. This makes sense because in the absence of any event, no algorithm can detect the split associated to the branch. We have also tested $m = 10$ which we discuss later.

10.3.4 Simulating each type of event

Once the number of events on each branch has been computed, we need to decide the type of each event. To do this, we select randomly an event type in this list with the following probabilities:

- Inversion (60%): A random sequence of genes is selected uniformly over the genome. This sequence is reversed. For example if we have $(1, 2, 3, 4, 5)$ and reverse the 3 genes in the middle, we get $(1, -4, -3, -2, 5)$. This phenomenon is the most frequent type of rearrangement in a genome. The number of genes involved in the inversion is chosen according to a Poisson distribution with an average of 5 genes. The number of 5 genes is chosen to be a realistic estimate of an average inversion size. The Poisson distribution was chosen for the sake of simplicity, since it requires no other parameter than the average, which is the only one we can reliably estimate. If the randomly chosen integer is 0, we try again until we get a non-zero value.

- Reciprocal translocation (29.79%): Two positions are chosen uniformly over the genome, but on different chromosomes. We then consider the part of the two chromosomes that start at the position chosen and reach the telomere, without crossing the centromere (therefore there is only one choice in doing so). These two parts are then exchanged between the chromosomes, such that the end that is the telomere remains the telomere in the other chromosome. The probability was chosen to be close to 30% but was decreased a little so that the probabilities sum up to 100% at the end.

- Duplication (5%): A randomly selected sequence of genes is duplicated. The number of genes involved is chosen with a Poisson distribution with an average of 5 genes for the same reasons as for inversions (see above). If the centromere is duplicated, randomly select one of the two centromeres and delete it.

- Deletion (5%): A randomly selected sequence of genes is deleted. The number of genes involved is chosen with a Poisson distribution with an average of 1 gene. Note that the value 0 is forbidden (since it would produce no event). If the sequence includes the centromere, we take care not to delete it.

- Fusion (0.1%): Two positions are randomly selected on the genome, on different chromosomes (like for reciprocal translocations). The two chromosomes are merged at random ends. The new centromere is randomly chosen between the two previous centromeres.

- Fission (0.1%): A random position is chosen uniformly over the genome. The chromosome to which it belongs is split at this position to form two chromosomes. One of the two chromosomes then lacks a centromere. We add one at a random position in the chromosome.

- Whole Genome Duplication (WGD) (0.01%): All chromosomes are duplicated. Each gene is then found in two copies. For each pair, we have an 80% probability to delete one of them (chosen randomly) and a 20% probability to keep both. The probability of WGD (0.01%) was chosen so that the average number of WGD in the yeast tree is 0.5 (most trees have 0 or 1 WGD, sometimes a few more).

In the special case in which we have only one chromosome in the species, translocation and fusion cannot be chosen, so the probabilities are redistributed between other events (according to their respective relative probabilities).

The whole process is run 100 times to create 100 trees with the associated species genomes. We do this simulation with both the vertebrate parameters and yeast parameters, resulting in two sets of 100 trees. The only differences between the two sets is the number of species per tree, the general frequency of events and the number of chromosomes and genes in the ancestor.

An example of simulation can be seen on Figure 10.1. In this tree, simulated WGD species sp12 has 18 chromosomes and 6071 genes, which is comparable to real WGD species *S. cerevisiae*, which has 16 chromosomes and 6275 genes. At the opposite, non-WGD species sp1 has 9 chromosomes and 5142 genes, similar to real non-WGD species *Lachancea kluyveri*, which has 8 chromosomes and 5321 genes. This means that our simulation simulates realistically the evolution of both WGD and non-WGD yeast species.

10.4 Benchmarking PhyChro

10.4.1 Preparing synteny blocks

After having created these two sets of 100 trees, we need to test PhyChro. However, a last step is needed because PhyChro takes a list of synteny blocks, not the genomes directly. On real genomes, we could use SynChro which would create the list of synteny blocks, but in our case, we do not have real genomes with DNA sequences. Instead, we have directly the homology relations (since integers with the same absolute value represent homologous genes). The step we need to add is the computation of synteny blocks. So I wrote an extra part in my simulation program which creates a list of synteny blocks. Here, I have chosen a strict definition of synteny block, that is a sequence of genes $x_1, x_2, \ldots, x_n$ (integers) such that the other genome contains either the sequence $x_1, x_2, \ldots, x_n$ or the sequence $-x_n, -x_{n-1}, \ldots, -x_1$. Contrary to SynChro, no insertion is allowed in the synteny block. This may make it harder for PhyChro to work, but this allows to test it as an independent program and to assess its performance even when it is not coupled with SynChro.

10.4.2 Results

After PhyChro has run on the 100 genomes and reconstructed a tree for each of them, we can compare these trees with the corresponding correct trees that were produced by the simulation program. The result for yeasts is shown in Figure 10.2. In the 100 trees, 61% have been perfectly inferred, i.e. have the same topology as the correct tree. With the vertebrate simulation, 79% of trees are perfectly reconstructed.

For the trees that are not perfectly reconstructed, we need to define a notion of error in order to count them. To do this, we use the notion of split. In a tree, each branch B defines a corresponding “split”, which is the bipartition of species formed by the two set of species corresponding to the two remaining subtrees if branch B is deleted. A tree of n species has $n-3$ splits. The correct tree and the PhyChro tree both define a set of splits. In order to count the number of errors in the PhyChro tree, we can count how many splits of the correct tree are missing in the PhyChro tree. This gives us a number of errors, which is reported in Figure 10.2 for the yeast simulation. As can be seen, only 9% of trees have more than 1 error.

Another way to measure performance is to consider the total ratio between the correct splits in all trees and the total number of splits. In this case, we find that PhyChro gets 97% of the splits correctly for both the yeasts and vertebrates simulations.

10.4.3 Further analysis

To further understand what makes it difficult for PhyChro to reconstruct the tree, we decided to measure several parameters of the branches which PhyChro fails to reconstruct properly. Figure 10.3 shows the distribution of branch lengths (in the correct tree) in different colors depending on whether PhyChro find the corresponding split. As can be seen, small branches are more difficult to reconstruct. This is expected, since it means PhyChro has less information to get the split right.

This observation has led us to test what the performance of PhyChro would be if we increase the minimal number of events on a branch (see section 10.3.3). When we increase it to 10 instead of 1, PhyChro is able to get 69% of the yeast trees correctly (instead of 61%), and 86% of the vertebrate trees correctly (instead of 79%). This confirms that this effect of small internal branches is significant.

Another test that is very interesting is to compare the confidence score given by PhyChro for correct and incorrect branches. The results are reported on Figure 10.4 for the yeast simulation. We can see that incorrect branches are, most of the times, given low confidence scores by PhyChro, which means PhyChro knows the branch is likely to be incorrect.

We made a last test which consisted in checking whether a certain type of event (inversion, duplication, etc.) was more associated with incorrectly inferred branches. However, none of the statistical tests were significant.

10.5 Conclusion

This simulation has enabled us to benchmark the PhyChro tree reconstruction algorithm. Overall, the algorithm performs remarkably well, deals properly with all types of rearrangements, and gives relevant confidence scores.

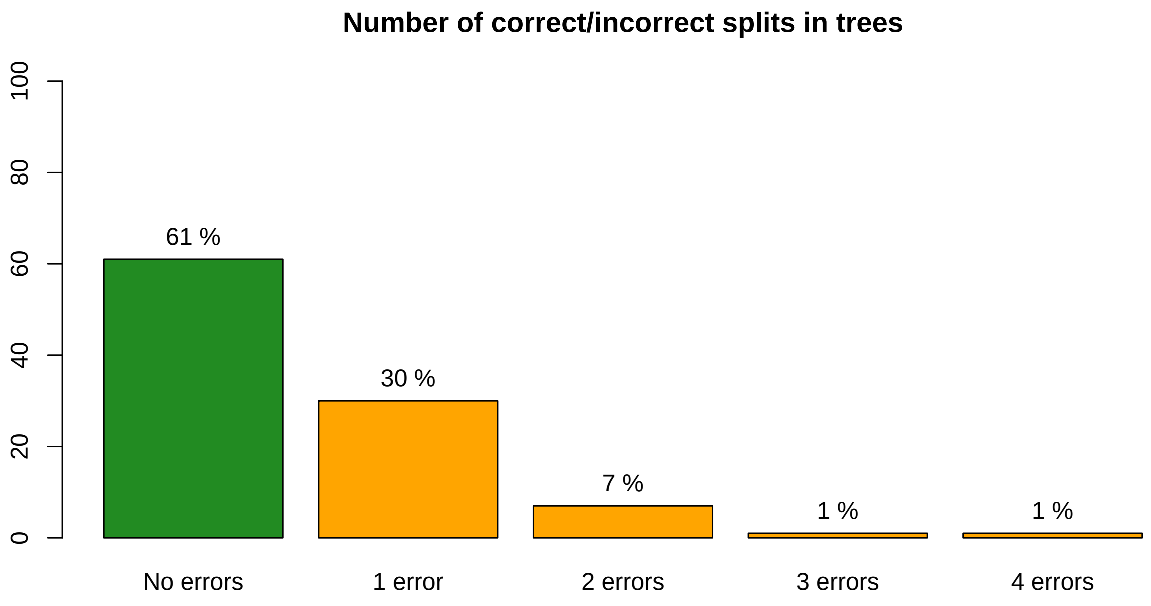

Figure 10.2: **PhyChro benchmark results for the yeast-like simulation.**
This histogram shows the number of incorrect splits in the 100 trees reconstructed by PhyChro.

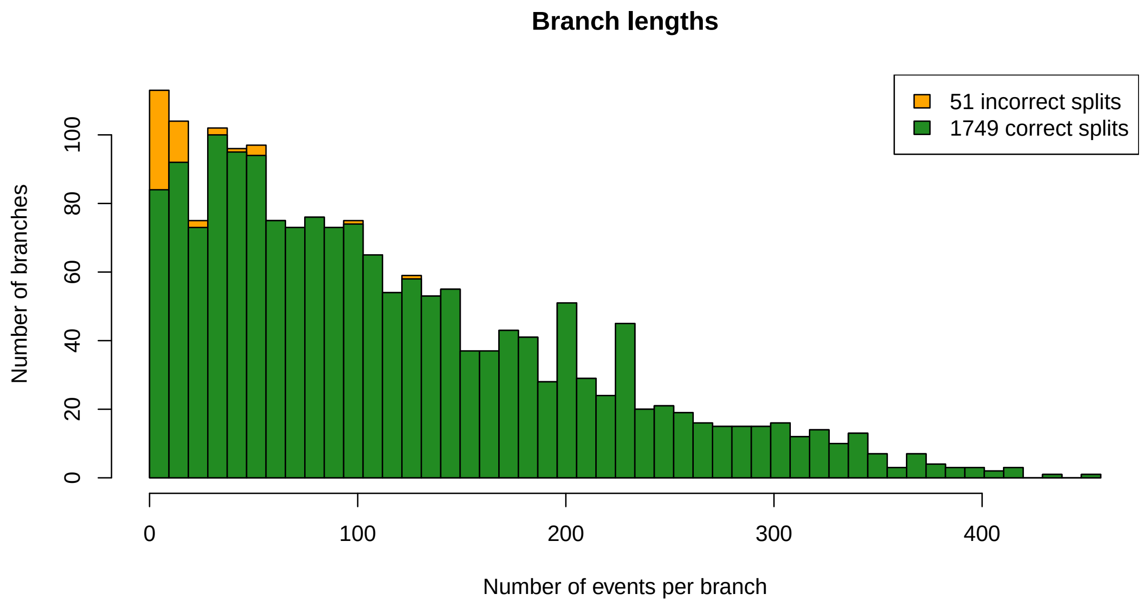

Figure 10.3: **Branch length distribution in simulated yeast trees.**
The histogram is colored differently depending on whether PhyChro has correctly inferred the split, i.e. whether the split is present in the tree it reconstructed.

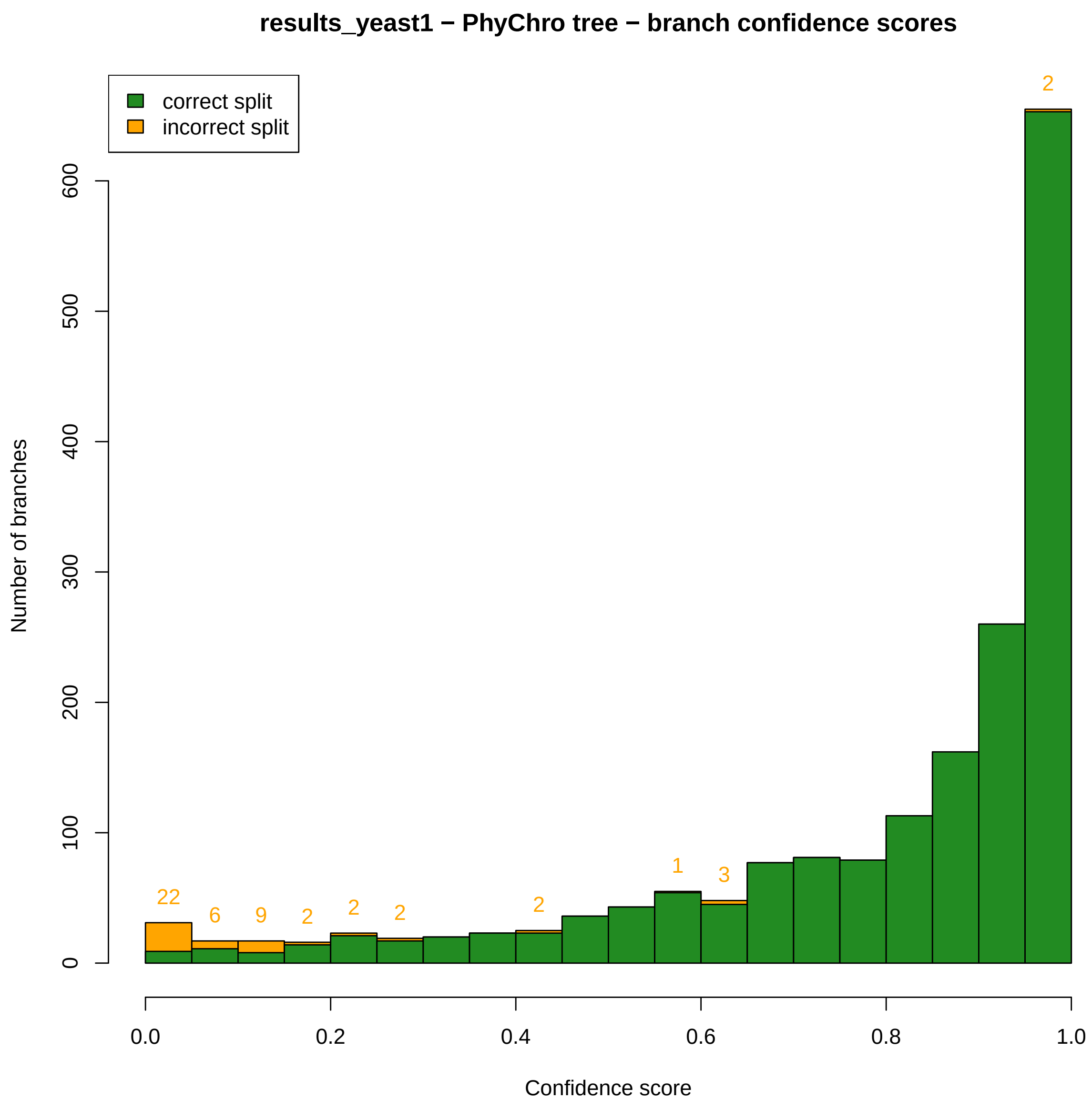

Figure 10.4: **Histogram of branch confidence scores given by PhyChro.**
The colors show whether the branch created by PhyChro corresponds to a correct split. The number of branches corresponding to the orange parts of the bars is written on top of them to make visualization easier.