

## Lecture 18

**Overview of Phylogeny**

*Lecturer: Prof. Serafim Batzoglou*  
(*serafim@cs.stanford.edu*)

*Scribe: Gaurav Garg*  
(*ggaurav@stanford.edu*)

We have been talking about phylogeny throughout the course but never really looked at it in depth, so in this last lecture we intend to tie everything together by covering in a little more depth phylogeny and evolution. We believe that all life forms are connected through an evolutionary tree. Also, evolution is something we want to understand on its own regard, so in this age of high throughput genomics we hope genomics could help us unravel the various mysteries of life through sequence comparison. The figure on last page shows a small part of the evolutionary tree, of mammals, which are a tiny fraction of all the organisms on the earth. For the mammals we have a pretty good idea of what the evolutionary tree should look like e.g. we know that we are closer to monkeys than to rodents, similarly cats and dogs are closer to one another than to cows or pigs. We also believe that we are closer to the rodents in terms of evolutionary time (not sequence) than to dogs, cats or cows. But this is still a somewhat unclear question, we know pretty well but not quite 100%. In fact when we look at organisms other than mammals, like viruses, bacteria we do not have a good idea of their evolutionary relationships at all. So this motivates us for the rest of the lecture.

**1. What are Phylogenies?**

Phylogenies are trees that show the history of life. A phylogeny tree shows the connection between various organisms and weight of the branches in the tree give an idea of time between evolutions of different organisms. A node with two edges out of it is the common ancestor through which speciation occurs, e.g. humans have a common ancestor with mice 80 million years ago. Beyond actual organisms particular sequence elements can also have their own evolution history within a genome. We have already seen selfish DNA. Selfish DNA is repeating transposable element, i.e. a selfish DNA makes a copy of it and inserts itself at a different position in the Genome. Now these elements have essentially fooled the genome into believing that it actually exists at two places within it. This can arise just like life, by chance. So if we have a DNA that transposes itself then more copies of its will transpose themselves more and in this way we get evolution of repeats. Hence within a Genome we can get a phylogeny of repeats. We can similarly have phylogeny of genes that copy themselves by duplication. Across organisms a particular gene family may have a whole history of how it evolved in the different branches of the tree. Hence we can conclude that phylogeny tries to answer the following questions. How do you establish relationships between (a) different organisms (b) between one family of elements within an organism or across organisms?

Before we move further let's define the *Orthologs* and *Paralogs*. Orthologs are two elements that have diverged because of speciation whereas Paralogs are two elements that have diverged because of duplication. So if we have a gene and this gene get duplicated and make two copies of it. And say we have two organisms; mouse and rat that evolved after the duplication then each one will have two copies of the gene. The two originals (between mouse and rat) are Orthologs, the two copies (between mouse and rat) are also Orthologs but an original and copy pair (between mouse and rat) is Paralogs. So both original versions of the gene in rat and mouse have come from the same gene, and both copies have come from the same copy version. But copy version in mouse and original in rat have not come from the same sequence, but they are very similar. They have actually come from a duplication of the original sequence before. Both orthology and paralogy are together called *homology*. Any two similar sequences are thus homologous, unless they are similar by some stroke of luck, which is very highly unlikely.

## 2. Inferring Phylogenies

Phylogeny trees can be inferred in many different ways e.g. by looking at the morphology of the organisms. E.g. According to morphology organisms with 4 legs could be grouped together but this is not true actually, the crocodile has 4 legs, dog has 4 legs and bat has 2 legs, but dogs are closer to bats than to crocodiles because they are mammals. So morphology cannot in general tell you a lot about the phylogeny of animals as compared to molecular methods that do a sequence comparison. Take the following example of species for which we do not know their phylogeny

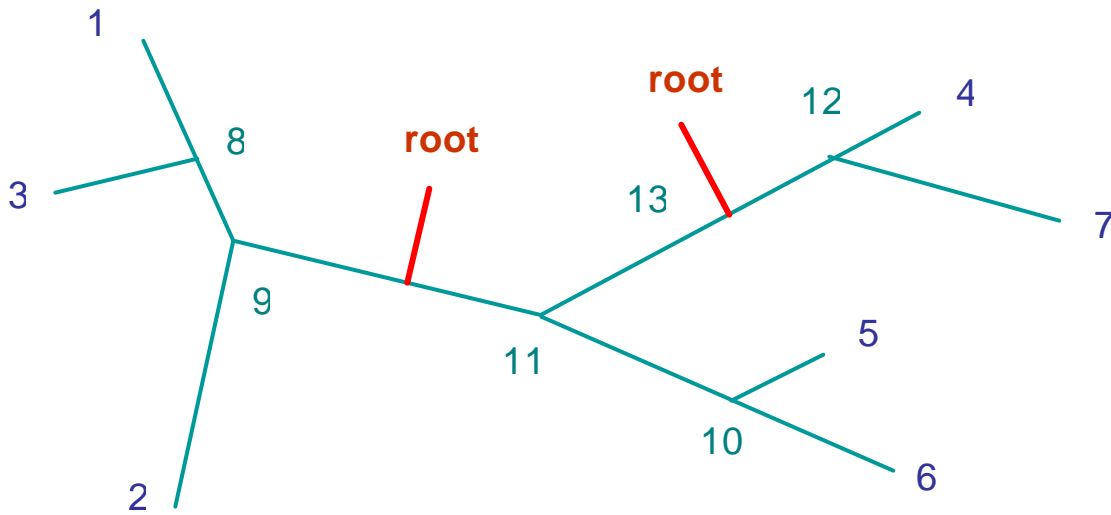


The different colors show the alignment. It is more common for dwarfs, humans and hobbits to have a similar nucleotide than for humans and orcs or humans and elves. But unlike this example it is not always easy to make this tree by the eye, so before we proceed further let's make a little background of the trees.

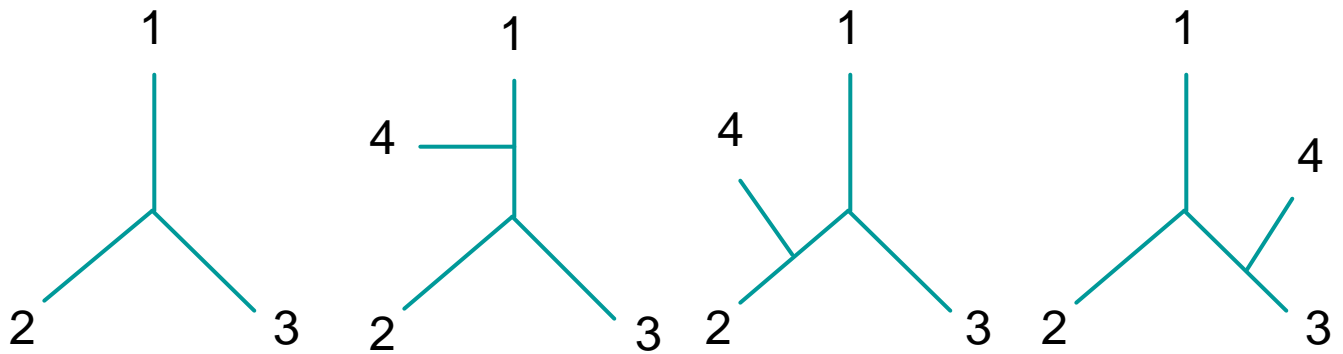
## 3. Background on Trees

Each edge of the tree can have length that specifies the time of evolution, or better the effective time of evolution because some species evolve much faster than the other

species, e.g. rodents are believed to evolve much faster than other organisms. The tree can be rooted or unrooted in general. If the tree is unrooted than we don't really know where the common ancestor should be placed e.g. as shown in the figure we could place the common ancestor (the root) at number of places. But there is no good way to figure out where to place this root.



A tree with  $N$  leafs has  $2N - 2$  nodes if it is unrooted and has  $2N - 1$  nodes if it rooted. This is because our trees are all binary. We assume all our trees to be binary because it is very highly unlikely that at exactly the same point three different species evolved. Also, even if they did we can always convert any tree to binary tree by adding more nodes. We also label the internal nodes of the tree and a convenient way to do this is to label in such a way that of the two leaf nodes, larger of the two is the smallest for all such possible leaf pairs. The parent then gets the next available label number.



If we look at the space of all possible trees, then there is one unrooted tree of three leaves but there can be three unrooted trees of four leaves as shown in above figure. In fact it turns out that there can be  $3 \times 5 \times 7 \times \dots \times (2N-5) = (2N-5)!!$  unrooted trees with  $N$  leaves and  $(2N-3)!!$  rooted trees with  $N$  leaves. So, we can see that the problem of constructing the best tree is pretty hard because of so many combinatorial choices for the tree.

## 4. Phylogeny and Sequence Comparison

All phylogeny and sequence comparison methods work on certain principles. The basic principles are:

- 1) Degree of sequence difference is proportional to length of independent sequence evolution.
- 2) Only use positions where alignment is pretty certain – avoid areas with (too many) gaps. Or in other words look at only the most conserved genes between organisms and use them to derive the rates of evolution.

Based on these principles, our problem is that given a number of sequences we want to build a phylogeny tree from them. The first thing to do is to define a distance measure between the sequences. Given two sequences  $x_i, x_j$ , define  $d_{ij}$  to be distance between the two sequences. Then one possible obvious definition is,

$d_{ij} =$  fraction  $f$  of sites where  $x_i[u] \neq x_j[u]$ .

The reader should bear in mind that we are talking about ungapped alignments here and that we have clipped out the bad regions of the alignment and are only looking at the best parts so as to do a reliable job. But the problem with this definition is that it maxes out to roughly 0.75. If we have four nucleotides and we align two random sequences comprising of them, then the alignment would differ in 75% of their locations. This makes it not a very good distance measure because as time of evolution goes to infinity the distance between sequences starts saturating at 0.75. Instead, a better model, which is due to Jukes-Cantor exists. As per this model,

$d_{ij} = -3/4 \log(1 - 4f/3)$

Clearly as per this model as  $f$  goes to 0.75  $d_{ij}$  goes to infinity, which is what we want. We will look at this model in more detail at the end of lecture. Now, having developed a metric for distance between two sequences we present a very simple clustering method for building a phylogeny tree.

### UPGMA (Unweighted Pair Group Method using Arithmetic Averages)

We are looking at how to cluster a bunch of sequences. If we think about it, a tree can also be thought of as a clustering where at any level you have a bunch of clusters. So, given two disjoint clusters  $C_i, C_j$  of sequences, define the distance between two clusters as sum of all pair wise distances between two clusters divided by total number of such possible pairs.

$$d_{ij} = \frac{1}{|C_i| + |C_j|} \sum_{\{p \in C_i, q \in C_j\}} d_{pq}$$

Note that if  $C_k = C_i \cup C_j$ , then distance of  $C_k$  to another cluster  $C_l$  is:

$$d_{kl} = \frac{d_{il}|C_i| + d_{jl}|C_j|}{|C_i| + |C_j|}$$

Thus, with above definitions we present our algorithm. The figure on the right shows an example of one such clustering and the resulting tree.

**Initialization:**

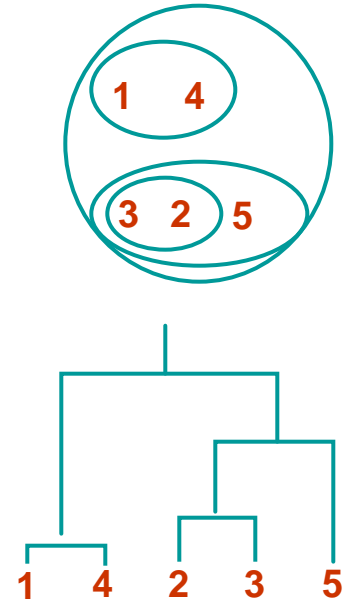
Assign each  $x_i$  into its own cluster  $C_i$   
 Define one leaf per sequence, height 0

**Iteration:**

Find two clusters  $C_i, C_j$  s.t.  $d_{ij}$  is min  
 Let  $C_k = C_i \cup C_j$   
 Define node connecting  $C_i, C_j$ , height  $d_{ij}/2$   
 Delete  $C_i, C_j$

**Termination:**

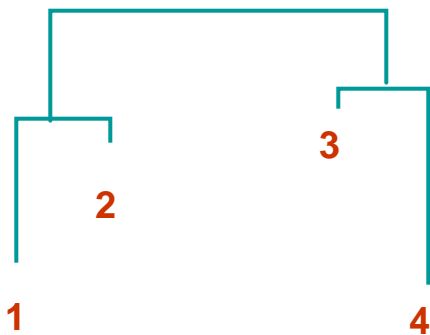
When all sequences belong to one cluster



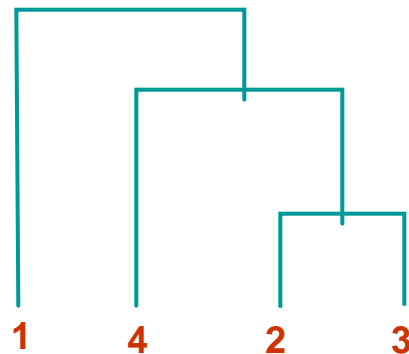
This algorithm has this property that from the root to the leaves all the heights are exactly the same.

A distance measure  $d_{ij}$  is called ultrametric if for any points  $i, j, k$  either all three distances  $d_{ij}, d_{jk}$  and  $d_{ki}$  are equal or two of them are equal and one is smaller. Our UPGMA trees follow this property. Thus, this method is guaranteed to produce the correct tree if distances between species actually follow this ultrametric property. But this actually turns out to be the weakness of UPGMA. E.g., from amongst 4 species, say species 1 and 4 started evolving much faster. But UPGMA will not find the correct tree because it assumes molecular clock where the implied time is constant for all species. The example where UPGMA messes up is:

Correct tree



UPGMA



The phylogeny trees can also be based on another property, namely additivity of distance. Given a tree, a distance measure is additive if the distance between any pair of leaves is the sum of lengths of edges connecting them. A maximum likelihood distance measure does obey this property if we are given a large amount of data. Now we will give another algorithm for building phylogeny tree, which is slightly more complicated than UPGMA, called Neighbor-joining.

### Neighbor-Joining

The neighbor-joining algorithm is guaranteed to produce the correct result if distance is additive, and it may even produce a good tree even when the distance is not additive. Step1 in this algorithm is to find the neighboring leaves of the tree. For this we define,

$$D_{ij} = d_{ij} - (r_i + r_j)$$

where

$$r_i = \frac{1}{|L| - 2} \sum_k d_{ik}$$

$r_i$  is the normalized sum of distance of leaf  $i$  to all other leaves  $k$ .

This method would not mess up like UPGMA, If we look at simple distances, in tree on the right then 1 and 3 look closer to one another than 1 and 4 but neighbor joining takes care of this by this normalization step. Thus, our complete algorithm looks like following:

#### Initialization:

Define  $T$  to be the set of leaf nodes, one per sequence  
Let  $L = T$

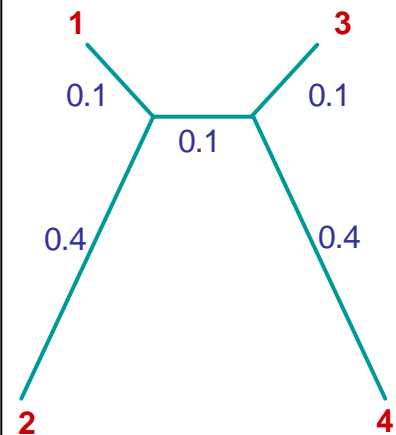
#### Iteration:

Pick  $i, j$  s.t.  $D_{ij}$  is minimal  
Define a new node  $k$ , and set  $d_{km} = \frac{1}{2}(d_{im} + d_{jm} - d_{ij})$   
for all  $m \in L$

Add  $k$  to  $T$ , with edges of lengths  $d_{ik} = \frac{1}{2}(d_{ij} + r_i - r_j)$   
Remove  $i, j$  from  $L$ ;  
Add  $k$  to  $L$

#### Termination:

When  $L$  consists of two nodes,  $i, j$ , and the edge between them of length  $d_{ij}$



The tree obtained from neighbor-joining method is though unrooted. In general it is very difficult as to how to root a tree. There is one way though, if you have what is called an outgroup. An *outgroup* is a species that we know is more distantly related to all remaining species, than they are to one another. E.g. Among human, mouse, rat, pig, dog, chicken and whale; chicken is the outgroup because it is not a mammal whereas rest are mammals. So, chicken can act as the root in the phylogeny tree. This is the best way currently available of rooting a tree. Now we will give probably the most popular method for building a phylogeny tree called Parsimony.

## Parsimony

The basic idea is that tree that explains the observed sequences with a minimal number of sequences. Parsimony can be sub-divided into two computational sub problems:

- 1) Given a tree find the parsimony cost of the tree (easy problem).
- 2) How to find the best tree with minimum parsimony cost (hard problem). This is hard because it requires a search through the space of all possible trees, which in general is not desirable. There is no good way to find the minimum cost parsimony tree in polynomial time.

We will now describe an algorithm, which is called *Traditional Parsimony Algorithm* to find parsimony score of a given tree.

Given a tree, given a column  $u$  of a multiple alignment:

### Initialization:

Set cost  $C = 0$ ;  $k = 2N - 1$

### Iteration:

If  $k$  is a leaf, set  $R_k = \{x^k [u]\}$

If  $k$  is not a leaf,

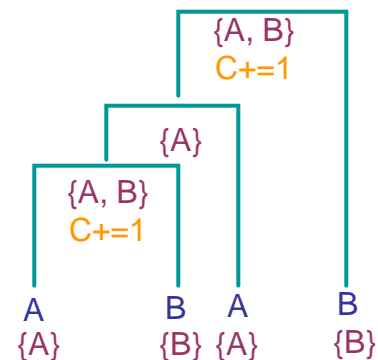
Let  $i, j$  be the daughter nodes;

Set  $R_k = R_i \cap R_j$ , if intersection is nonempty

Set  $R_k = R_i \cup R_j$ , and  $C += 1$ , if intersection is empty

### Termination:

Minimal cost of tree for column  $u = C$

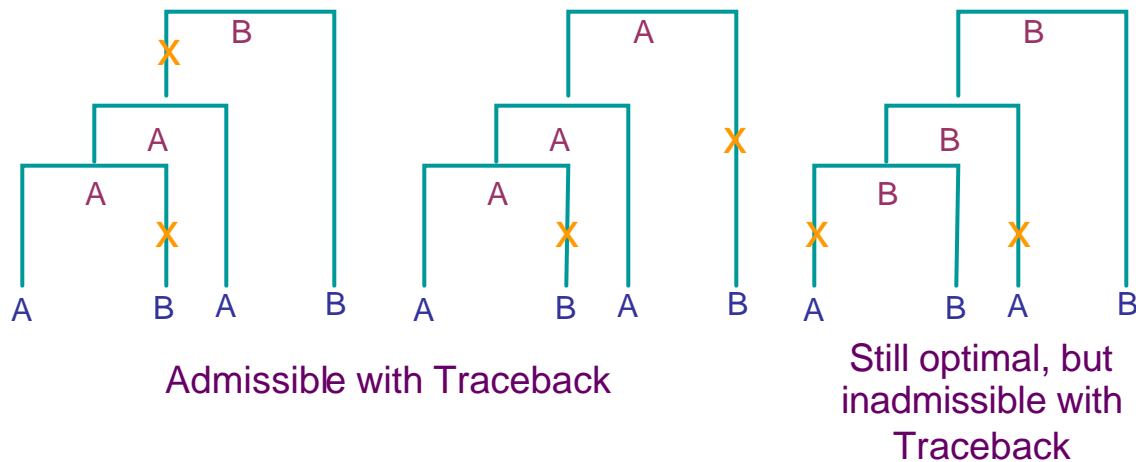


Thus, given a tree the above algorithm finds the traditional parsimony tree and traditional parsimony score for it. An example to illustrate this has been shown above. This

algorithm does not tell us what is the ancestral letter exactly, so we do a traceback to find the ancestral nucleotides. Traceback works as follows:

- 1) Choose an arbitrary nucleotide from  $R_{2N-1}$  for the root.
- 2) Having chosen nucleotide  $r$  for parent  $k$ , if  $r \in R_i$  then choose  $r$  for daughter, else choose arbitrary nucleotide from  $R_i$ .

This traceback method produces some assignment for cost  $C$ . The following figure explains this for our example of previous figure:



This traceback method gives us two possible trees for this scenario both of which are optimal. Though as we can clearly see that it misses out on some possibly optimal trees like the third one in above figure.

A variation of the traditional tree exists called *Weighted Parsimony*. The algorithm works as follows:

Let  $S_k(a) =$  minimal cost for the assignment of  $\underline{a}$  to node  $k$

**Initialization:**

Set  $k = 2N - 1$

**Iteration:**

If  $k$  is a leaf:

$S_k(a) = 0$  for  $a = x^k[u]$ ,  $S_k(a) = \infty$  otherwise

If  $k$  is not a leaf:

$S_k(a) = \min_b [S_i(b) + s(a,b)] + \min_c [S_j(c) + s(a,c)]$   
 where  $i$  and  $j$  are daughter nodes of  $k$

**Termination:**

Minimal cost of tree =  $\min_a S_{2N-1}(a)$

So both the Parsimony methods find the cost for a tree given the tree but the question still remains how to find the most optimal tree? Clearly searching through all the possible space of trees is exponential in complexity, so we cannot do that. But certain heuristics can be applied. Like by starting with an empty tree and adding edges to the tree one by one according to which edge is best. This is what UPGMA and neighbor-joining do. Here we present an exhaustive way to do this if tree size is manageable, called branch & bound.

## Branch and Bound

The strategy exploits the fact that adding an extra edge can only increase the number of substitutions in parsimony. The idea behind branch and bound is to begin systematically building trees with increasing number of leaves, but to abandon a particular avenue whenever the current incomplete tree has a cost exceeding the smallest cost so far for a complete tree. The algorithm then proceeds as follows:

- 1) Enumerate all unrooted trees with at most  $n$  leaves  $[i_3] [i_5] [i_7] \dots [i_{2N-5}]$ , where each  $i_k$  can take values from 0 (no edge) to  $k$ .
- 2) At each point keep  $C$  = smallest cost so far for a complete tree.
- 3) Start Branch & Bound with tree  $[1][0][0] \dots [0]$ .
- 4) Whenever cost of current  $T > C$ , then  $T$  is not optimal, and by the above fact any tree with more edges containing  $T$  is also not optimal.

The above algorithm works reasonably well for modest sized data.

## Bootstrapping to get best trees

The algorithm is based on the randomization theme. The main outline of this algorithm is following:

- 1) Select random columns from a multiple alignment – one column can then appear several times.
- 2) Build a phylogenetic tree based on the random sample from (1).
- 3) Repeat (1), (2) many (say, 1000) times.
- 4) Output the tree that is constructed most frequently.

## 5. Modeling Evolution

During an infinitesimal time  $\Delta t$ , there is not enough time for two substitutions to happen on the same nucleotide. So we can estimate  $P(x|y, \Delta t)$ , for  $x, y \in \{A, C, G, T\}$ . Let matrix of changes, the substitution model be defined as

$$S(\Delta t) = \begin{bmatrix} P(A|A, \Delta t) & \dots & P(A|T, \Delta t) \\ \vdots & \ddots & \vdots \\ P(T|A, \Delta t) & \dots & P(T|T, \Delta t) \end{bmatrix}$$

One reasonable assumption is that this matrix is multiplicative or in other words evolution is a stationary Markov process. Therefore,

$$S(t+t') = S(t)S(t')$$

That is,

$$P(x | y, \Delta t + \Delta t') = \sum_z P(x | z, \Delta t)P(z | y, \Delta t')$$

Simplest model of evolution is the *Jukes-Cantor* model of evolution. This model assumes a constant rate of evolution. Thus this model has only one parameter, alpha, which is the substitution rate of a nucleotide by a different nucleotide. Then, for a short time  $\epsilon$ ,

$$S(\epsilon) = \begin{bmatrix} 1-3\alpha\epsilon & \alpha\epsilon & \alpha\epsilon & \alpha\epsilon \\ \alpha\epsilon & 1-3\alpha\epsilon & \alpha\epsilon & \alpha\epsilon \\ \alpha\epsilon & \alpha\epsilon & 1-3\alpha\epsilon & \alpha\epsilon \\ \alpha\epsilon & \alpha\epsilon & \alpha\epsilon & 1-3\alpha\epsilon \end{bmatrix}$$

For longer times, it is easy to see that this reduces to

$$S(t) = \begin{bmatrix} r(t) & s(t) & s(t) & s(t) \\ s(t) & r(t) & s(t) & s(t) \\ s(t) & s(t) & r(t) & s(t) \\ s(t) & s(t) & s(t) & r(t) \end{bmatrix}$$

where,

$$r(t) = 1/4(1 + 3e^{-4\alpha t})$$

$$s(t) = 1/4(1 - e^{-4\alpha t})$$

This Jukes Cantor formula is used to define the distance measure earlier in the lecture.

Another model of evolution is the *Kimura* model of evolution.

This model takes into account the fact that rate of transitions (rate  $\alpha$ ) and transversions (rate  $\beta$ ) are different. Transitions (A/G, C/T) are much more likely than transversions (A/T, A/C, G/T, C/G). Thus,

$$S(t) = \begin{bmatrix} r(t) & s(t) & u(t) & s(t) \\ s(t) & r(t) & s(t) & u(t) \\ u(t) & s(t) & r(t) & s(t) \\ s(t) & u(t) & s(t) & r(t) \end{bmatrix}$$

where

$$s(t) = 1/4(1 - e^{-4bt})$$

$$u(t) = 1/4(1 + e^{-4bt} - e^{-2(a+b)t})$$

$$r(t) = 1 - 2s(t) - u(t)$$

There are also some models for modeling gaps. But in general, very little is known about evolution at this moment. There are still a lot of things that have not been modeled yet like, evolution at different parts of sequences, chromosome length, different elements, gaps, etc. These all form a part of the future research directions in Genomics.

