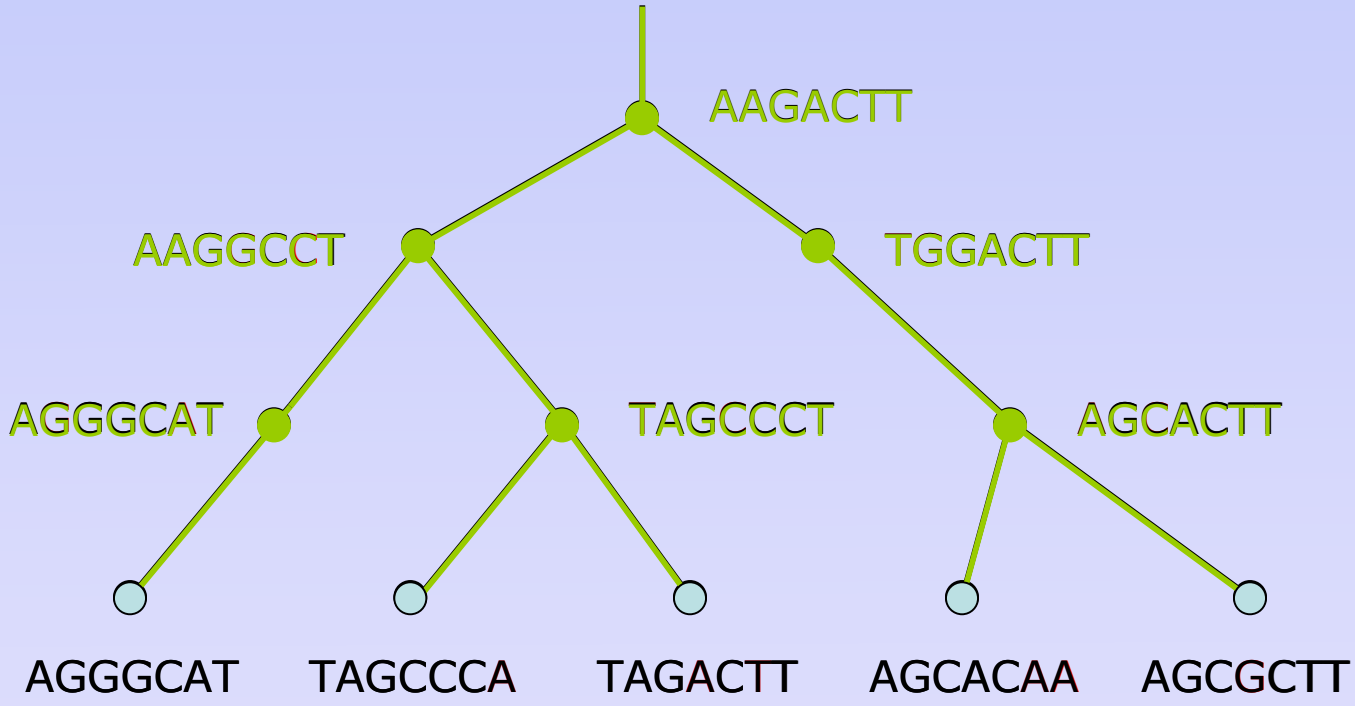
Sequence alignment

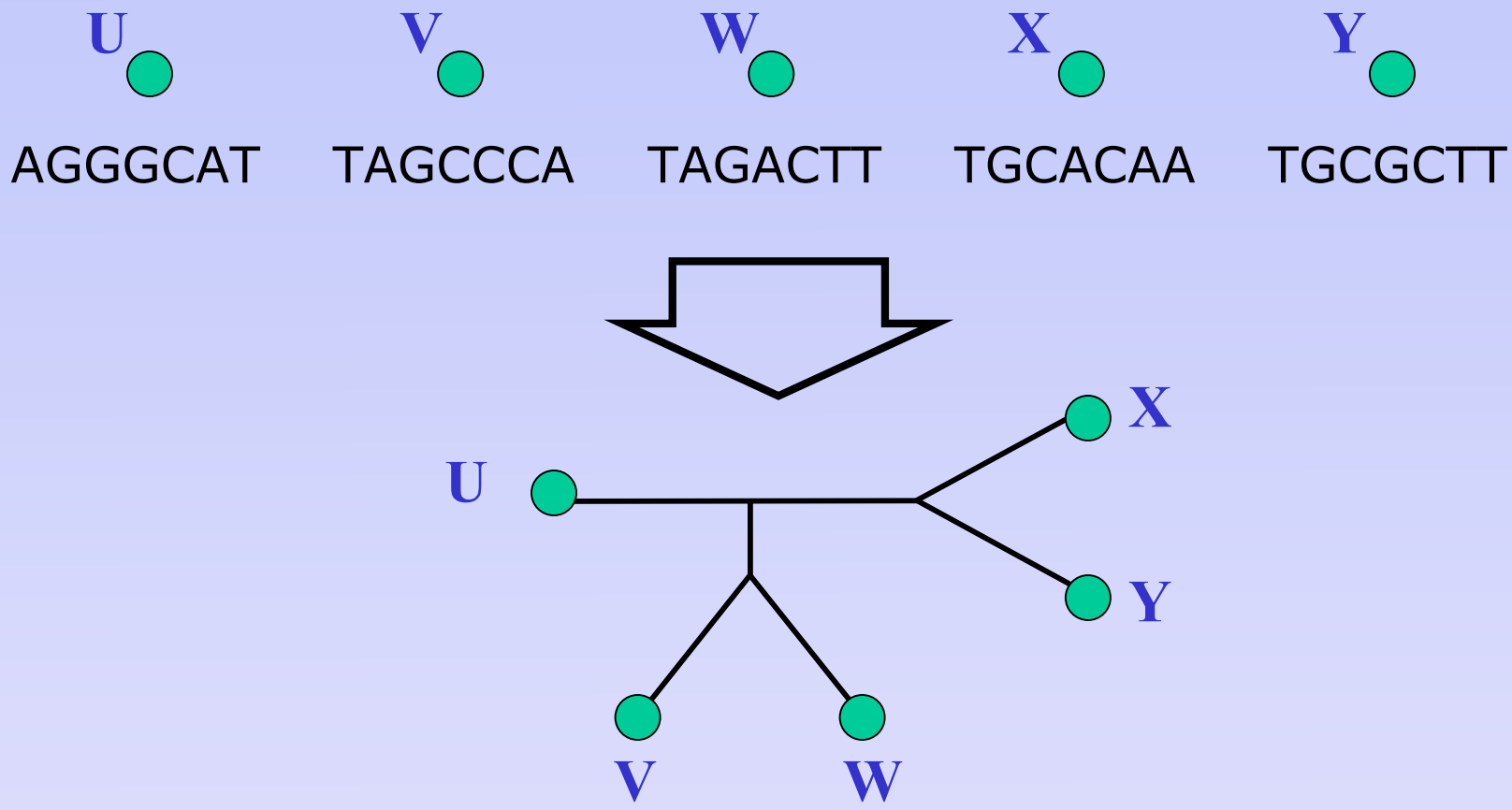
CS 394C

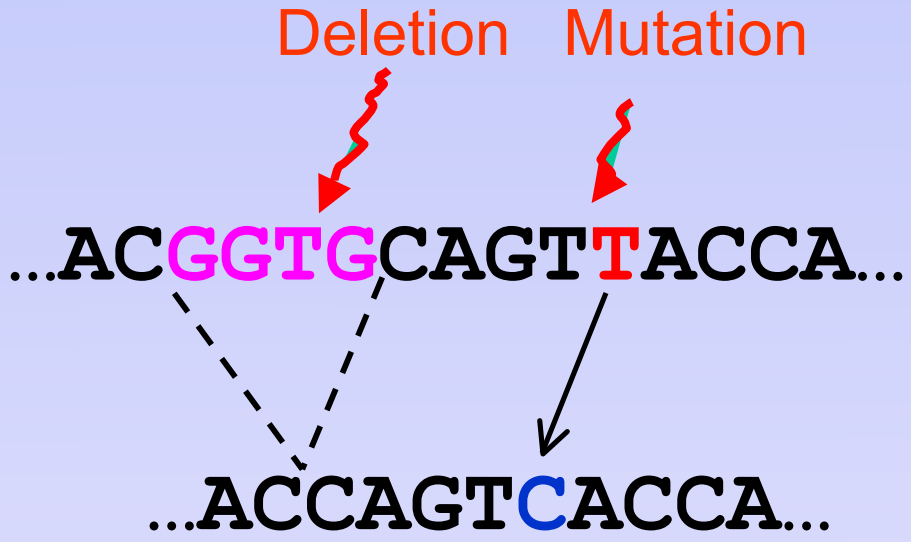
Tandy Warnow

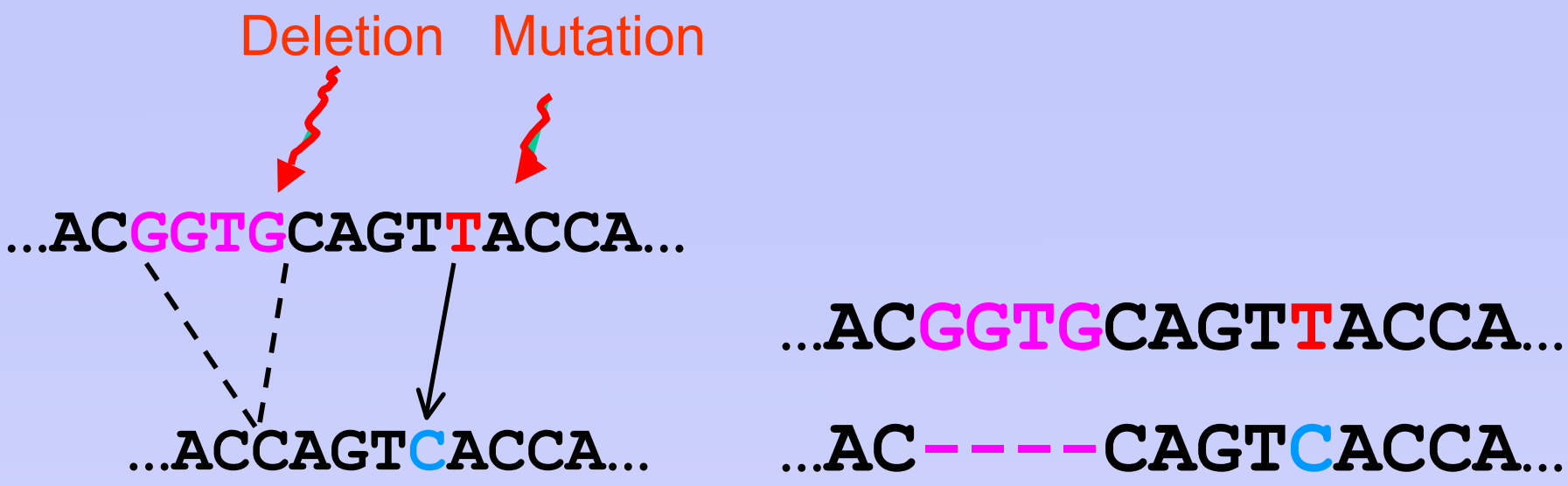
Feb 15, 2012

DNA Sequence Evolution




U
V
W
X
Y
AGGGCAT
TAGCCCA
TAGACTT
TGCACAA
TGCGCTT
X
U
Y
V
W


Deletion
Mutation
...ACGGTGCAGTTACCA...
...ACCAGTCACCA...



The true multiple alignment

– Reflects historical substitution, insertion, and deletion events in the true phylogeny

Input: unaligned sequences

```
S1 = AGGCTATCACCTGACCTCCA
S2 = TAGCTATCACGACCGC
S3 = TAGCTGACCGC
S4 = TCACGACCGACA
```

Phase 1: Multiple Sequence Alignment

```
S1 = AGGCTATCACCTGACCTCCA
S2 = TAGCTATCACGACCGC
S3 = TAGCTGACCGC
S4 = TCACGACCGACA
```



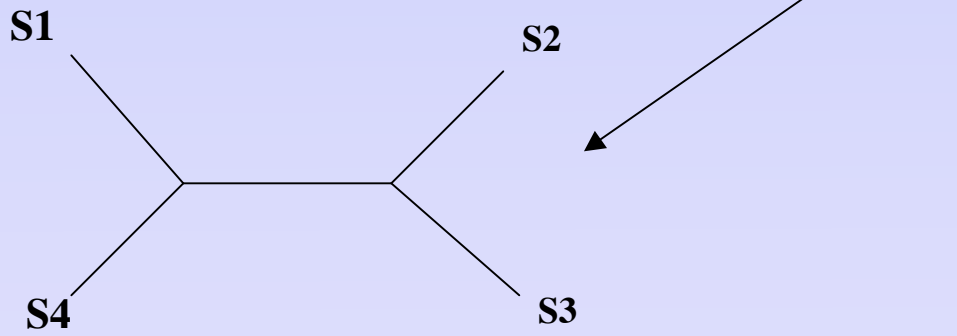
```
S1 = -AGGCTATCACCTGACCTCCA
S2 = TAG-CTATCAC--GACCGC--
S3 = TAG-CT-------GACCGC--
S4 = -------TCAC--GACCGACA
```

Phase 2: Construct tree

```
S1 = AGGCTATCACCTGACCTCCA
S2 = TAGCTATCACGACCGC
S3 = TAGCTGACCGC
S4 = TCACGACCGACA
```

→

```
S1 = -AGGCTATCACCTGACCTCCA
S2 = TAG-CTATCAC--GACCGC--
S3 = TAG-CT-------GACCGC--
S4 = -------TCAC--GACCGACA
```



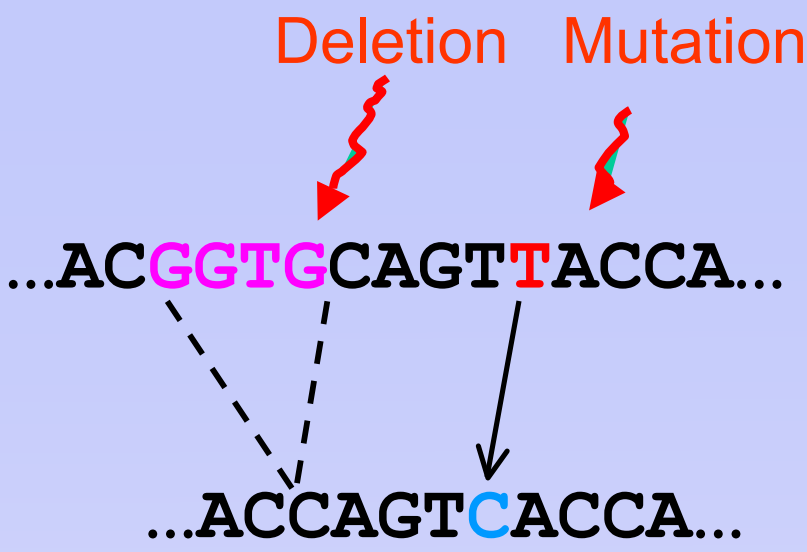
Many methods

Alignment methods

- Clustal
- POY (and POY*)
- Probcons (and Probtree)
- MAFFT
- Prank
- Muscle
- Di-align
- T-Coffee
- Opal
- Etc.

Phylogeny methods

- Bayesian MCMC
- Maximum parsimony
- Maximum likelihood
- Neighbor joining
- FastME
- UPGMA
- Quartet puzzling
- Etc.



...ACGGTGCAGTTACCA...

...AC----CAGTCACCA...

The true multiple alignment

– Reflects historical substitution, insertion, and deletion events in the true phylogeny

But how do we try to estimate this?

Pairwise alignments and edit transformations

- Each pairwise alignment implies one or more edit transformations
- Each edit transformation implies one or more pairwise alignments
- So calculating the edit distance (and hence minimum cost edit transformation) *is the same* as calculating the optimal pairwise alignment

Edit distances

- Substitution costs may depend upon which nucleotides are involved (e.g, transition/transversion differences)

- Gap costs
  - Linear (aka “simple”): gapcost(L) = cL
  - Affine: gapcost(L) = c+c’(L)
  - Other: gapcost(L) = c+c’log(L)

Computing optimal pairwise alignments

- The cost of a pairwise alignment (*under a simple gap model*) is just the sum of the costs of the columns

- Under affine gap models, it's a bit more complicated (but not much)

Computing edit distance

- Given two sequences and the edit distance function F(.,.), how do we compute the edit distance between two sequences?

- Simple algorithm for standard gap cost functions (e.g., affine) based upon dynamic programming

DP alg for simple gap costs

- Given two sequences A[1…n] and B[1…m], and an edit distance function F(.,.) with unit substitution costs and gap cost C,
- Let
  - $A = A_1, A_2, \ldots, A_n$
  - $B = B_1, B_2, \ldots, B_m$
- Let M(i,j)=F(A[1…i],B[1…j]) (i.e., the edit distance between these two prefixes)

Dynamic programming algorithm

Let M(i,j)=F(A[1…i],B[1…j])

- M(0,0)=0
- M(n,m) stores our answer
- How do we compute M(i,j) from other entries of the matrix?

Calculating M(i,j)

- Examine final column in some optimal pairwise alignment of A[1…i] to B[1…j]
- Possibilities:
  - Nucleotide over nucleotide: previous columns align A[1…i-1] to B[1…j-1]:
    $M(i,j)=M(i-1,j-1)+subcost(A_i,B_j)$
  - Indel (-) over nucleotide: previous columns align A[1…i] to B[1…j-1]:
    $M(i,j)=M(i,j-1)+indelcost$
  - Nucleotide over indel: previous columns align A[1…i-1] to B[1…j]:
    $M(i,j)=M(i-1,j)+indelcost$

Calculating M(i,j)

- Examine final column in some optimal pairwise alignment of A[1…i] to B[1…j]
- Possibilities:
  - Nucleotide over nucleotide: previous columns align A[1…i-1] to B[1…j-1]:
    $M(i,j)=M(i-1,j-1)+subcost(A_i,B_j)$
  - Indel (-) over nucleotide: previous columns align A[1…i] to B[1…j-1]:
    $M(i,j)=M(i,j-1)+indelcost$
  - Nucleotide over indel: previous columns align A[1…i-1] to B[1…j]:
    $M(i,j)=M(i-1,j)+indelcost$

Calculating M(i,j)

- M(i,j) = min {
  M(i-1,j-1)+subcost(A_i,B_j),
  M(i,j-1)+indelcost,
  M(i-1,j)+indelcost }

O(nm) DP algorithm for pairwise alignment using simple gap costs

- Initialize M(0,j) = M(j,0) = j∗indelcost

- For i=1…n
  - For j = 1…m
    - M(i,j) = min {
      M(i-1,j-1)+subcost(A_i,B_j),
      M(i,j-1)+indelcost,
      M(i-1,j)+indelcost
      }

- Return M(n,m)
- Add arrows for backtracking (to construct an optimal alignment and edit transformation rather than just the cost)

Modification for other gap cost functions is straightforward but leads to an increase in running time

Sum-of-pairs optimal multiple alignment

- Given set S of sequences and edit cost function F(.,.),
- Find multiple alignment that minimizes the sum of the implied pairwise alignments (Sum-of-Pairs criterion)

- NP-hard, but can be approximated
- Is this useful?

Other approaches to MSA

- Many of the methods used in practice do not try to optimize the sum-of-pairs
- Instead they use probabilistic models (HMMs)
- Often they do a progressive alignment on an estimated tree (aligning alignments)
- Performance of these methods can be assessed using real and simulated data

Many methods

Alignment methods

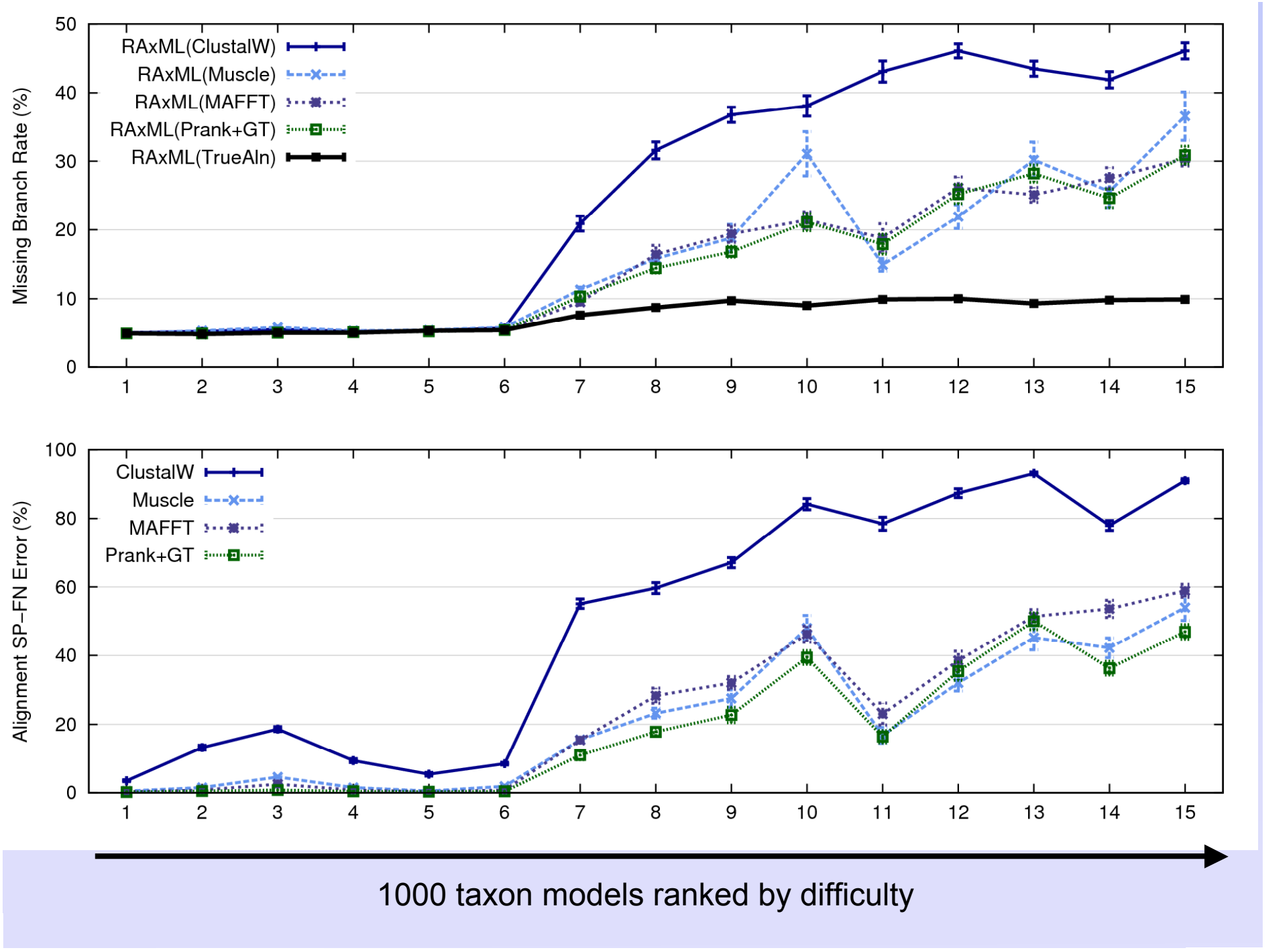
- Clustal
- POY (and POY*)
- Probcons (and Probtree)
- MAFFT
- Prank
- Muscle
- Di-align
- T-Coffee
- Opal
- Etc.

Phylogeny methods

- Bayesian MCMC
- Maximum parsimony
- Maximum likelihood
- Neighbor joining
- FastME
- UPGMA
- Quartet puzzling
- Etc.

Simulation study

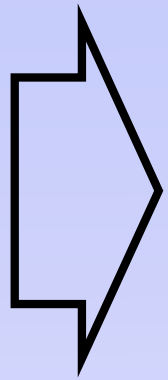
- ROSE simulation:
  - 1000, 500, and 100 sequences
  - Evolution with substitutions and indels
  - Varied gap lengths, rates of evolution
- Computed alignments
- Used RAxML to compute trees
- Recorded tree error (missing branch rate)
- Recorded alignment error (SP-FN)

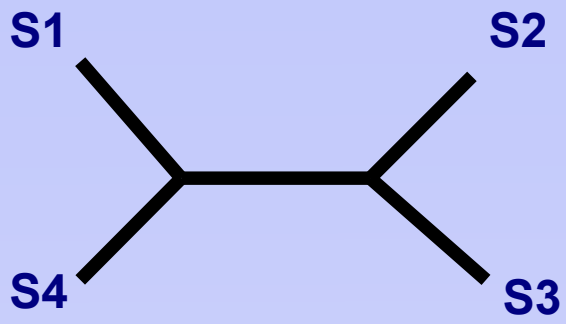

Missing Branch Rate (%)
RAxML(ClustalW)
RAxML(Muscle)
RAxML(MAFFT)
RAxML(Prank+GT)
RAxML(TrueAln)
Alignment SP–FN Error (%)
ClustalW
Muscle
MAFFT
Prank+GT
1000 taxon models ranked by difficulty

Problems with the two phase approach

- Manual alignment can have a high level of subjectivity (and can take a long time).
- Current alignment methods fail to return reasonable alignments on markers that evolve with high rates of indels and substitutions, especially if these are large datasets.
- We discard potentially useful markers if they are difficult to align.

```
S1 = AGGCTATCACCTGACCTCCA
S2 = TAGCTATCACGACCGC
S3 = TAGCTGACCGC
S4 = TCACGACCGACA
```





and

```
S1 = -AGGCTATCACCTGACCTCCA
S2 = TAG-CTATCAC--GACCGC--
S3 = TAG-CT-------GACCGC--
S4 = -------TCAC--GACCGACA
```

Simultaneous estimation of trees and alignments

Simultaneous Estimation Methods

- Likelihood-based (under model of evolution including insertion/deletion events)
  - ALIFRITZ, BAli-Phy, BEAST, StatAlign, others
  - Computationally intensive
  - Most are limited to small datasets (< 30 sequences)

Treelength-based

- Input: Set S of unaligned sequences over an alphabet $\sum$, and an edit distance function F(.,.) (must account for gaps and substitutions)

- Output: Tree T with sequences S at the leaves and other sequences at the internal nodes so as to minimize

$$\sum_e F(s_v, s_w),$$

where the sum is taken over all edges $e=(s_v,s_w)$ in the tree

Minimizing treelength

- Given set S of sequences and edit distance function F(.,.),
- Find tree T with S at the leaves and sequences at the internal nodes so as to minimize the treelength (sum of edit distances)
- NP-hard but can be approximated
- NP-hard even if the tree is known!

Minimizing treelength

- The problem of finding sequences at the internal nodes of a fixed tree was introduced by Sankoff.
- Several algorithmic results related to this problem, with pretty theory
- Most popular software is POY, which tries to optimize tree length.
- The accuracy of any tree or alignment depends upon the edit distance function F(.,.)

More

- SATé: our method for simultaneous estimation and tree alignment
- POY, POY*, and BeeTLe: results of how changing the gap penalty from simple to affine impacts the alignment and tree
- Impact of guide tree on MSA
- Statistical co-estimation using models that include indel events (Statalign, Alifritz, BAliPhy)
- Getting “inside” some of the best MSAs