A multiple alignment tool in 3D

Matthias Kruspe

Department of Computer Science, Bioinformatics Group
University of Leipzig

TBI Winterseminar
Bled, Slovenia
February 2005

Outline

Outline

Outline

Outline

Outline

Multiple Alignments

- direct alignment of more then three sequences via dynamic programming not practicable
- using of heuristics to reduce complexity
- using approximate methods: progressive, iterative, statistical

Typical framework of progressive alignment algorithms

1. determine pairwise distances of all sequences
2. calculate phylogenetic tree from the pairwise distances
3. calculate sequence weights according to their relationship
4. pairwise align sequences sequentially guided by tree

Problems with progressive alignments

Problems

- not guaranteed to find optimal alignment
- ultimate alignment depends on calculated phylogenetic tree
- ultimate alignment depends on early alignment steps
- introduced gaps remain fixed during whole progressive alignment process
- loss of information when building up alignment

		agca
a–ga	ag–a	ag–a
agga	agga	agga

Goal

Goal

- increase information transfer from sequence to alignment
- improve quality of introduced gaps
- find a more accurate description of underlying phylogenetic history

NNAlign

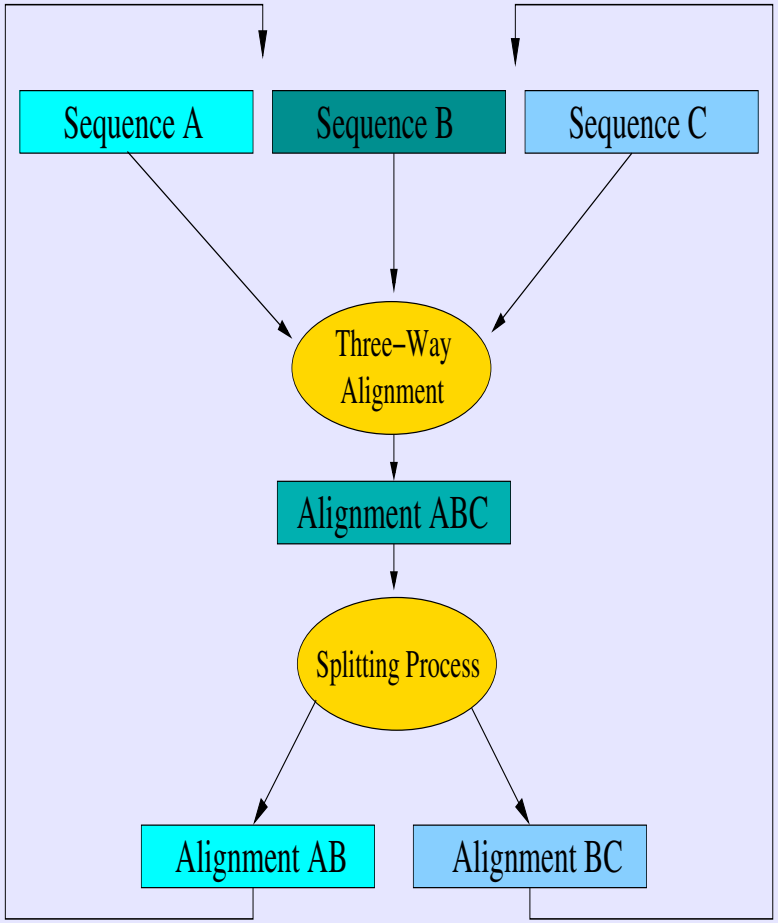
- progressive alignment method similar to *ClustalW*
- aligns both nucleic acid and amino acid sequences
- instead of aligning two profiles during progressive steps, *NNAlign* aligns three profiles simultaneously
- underlying phylogeny is not a tree but a network

Outline

The NNAlign method

Overview

1. determine sequence distances by pair-wise alignment
2. build a phylogenetic network using *Neighbor-Net*
3. align sequences sequentially according to phylogenetic network
   - align three sequences in each alignment step
   - while not the final alignment, split up into two alignments



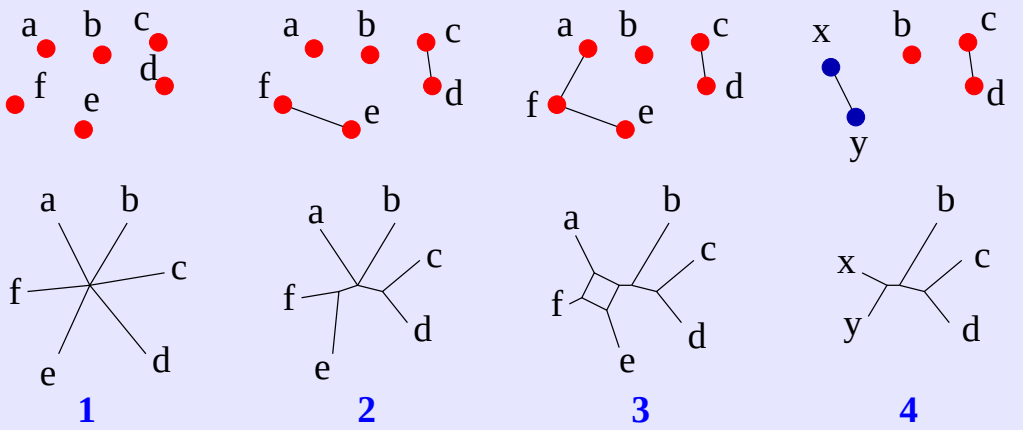
Reconstructing phylogenetic networks with *Neighbor-Net*[1]

Neighbor-Net

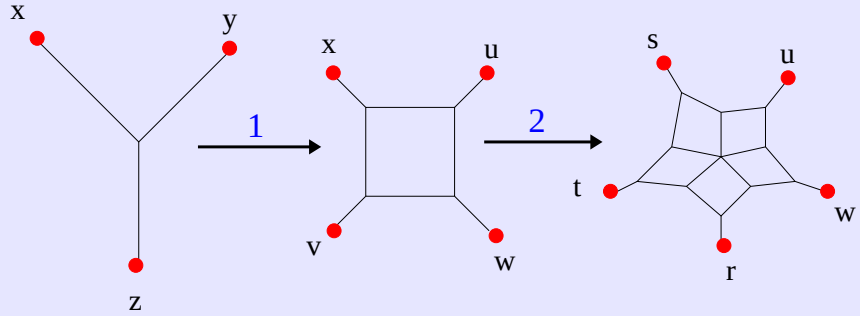
- introduced by *D. Bryant and V. Moulton* to model reticulate evolution
- distance based, agglomerative method similar to *Neighbor Joining*
- pairs nodes not immediately but waits until a node has been paired up a second time
- after all nodes are agglomerated they were expanded
- result after expansion is a planar splits graph

[1] D. Bryant, V. Moulton: Neighbor-Net: An Agglomerative Method for the Construction of Phylogenetic Networks, *Mol. Biol. Evol.*, 21(2), 255-265, 2004

Neighbor-Net: Agglomeration and Expansion



Agglomeration

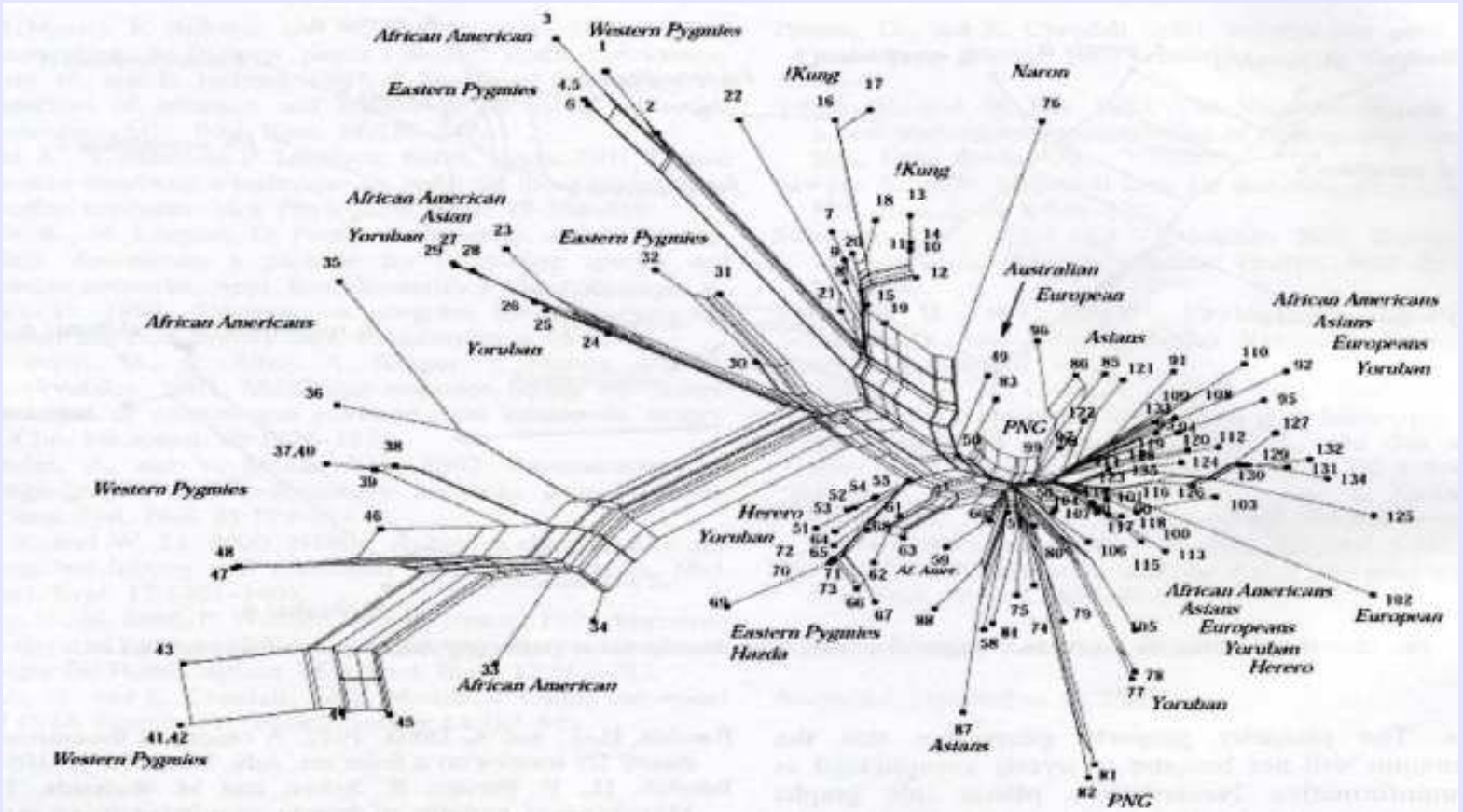


Expansion

1. begin with one node for each taxon
2. identify c,d as neighbors as well as e,f
3. identify f as neighbor of a as well as e
4. fusing a,e,f to new nodes x,y

1. expanding nodes y,z to u,w,v
2. expanding nodes v,x to r,s,t

Neighbor-Net example

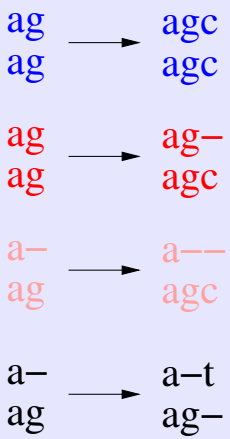


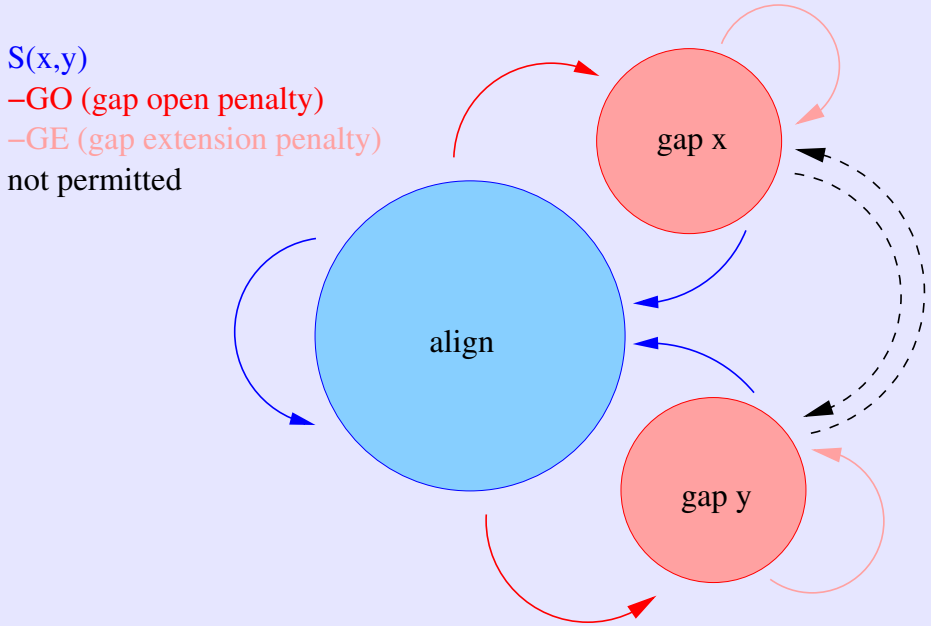
Alignment order

Getting alignment order out of phylogenetic network

- nodes in *Neighbor-Net* algorithm correspond to sequences
- every node fusion corresponds to a three-way alignment
- order of node fusion gives order of sequential alignments
- to keep framework consistent, alignment must be splitted up into two alignments (*NeighborNet* fuses three nodes to two nodes)

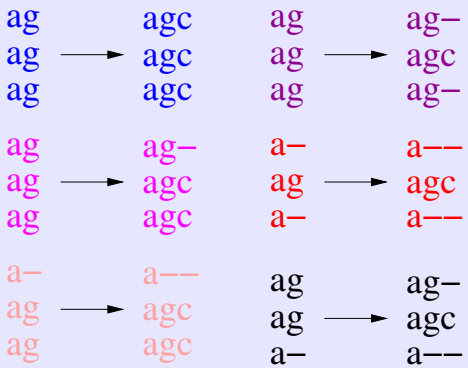
Affine gap penalties for pairwise sequence alignments



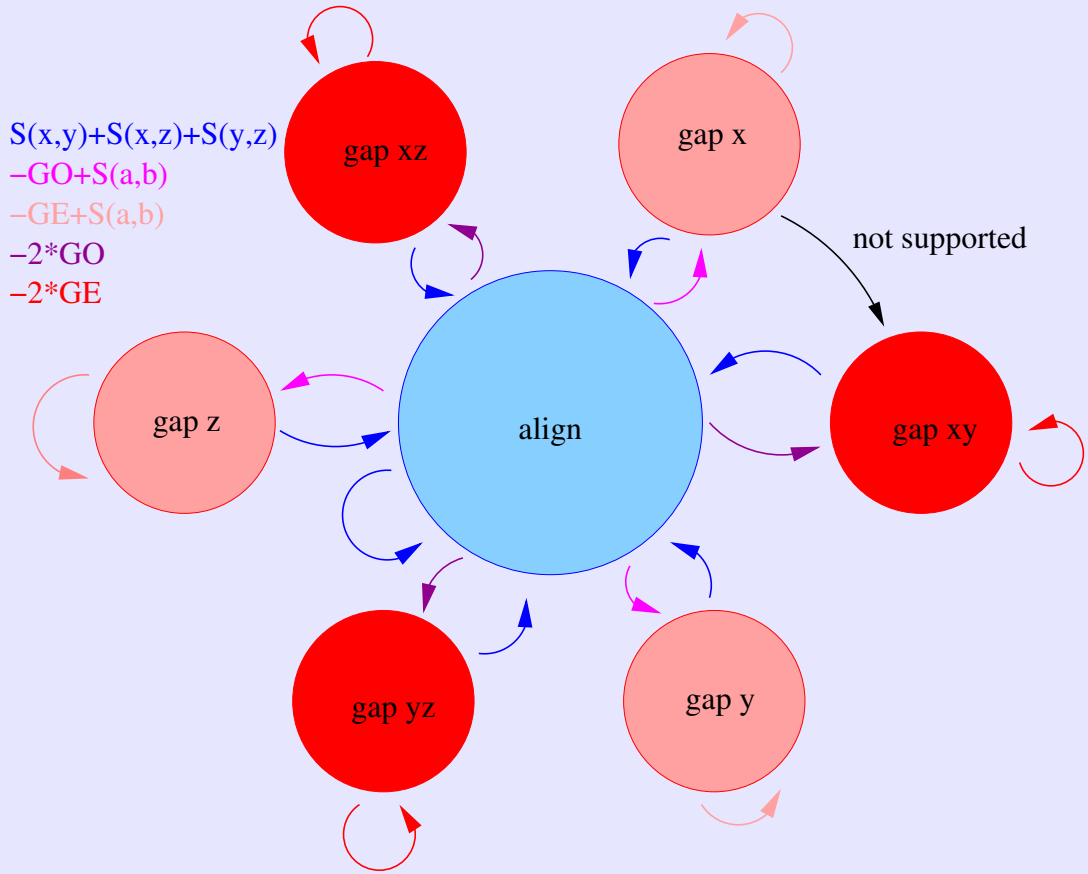


- consider possibilities for gap opening or extending as finite state machine with three states: align, gap in first sequence, gap in second sequence
- state transitions are shown in diagram

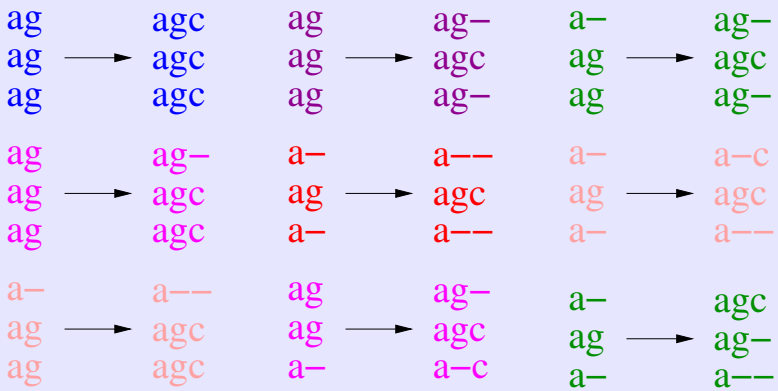
Quasi-natural gap costs for three sequences



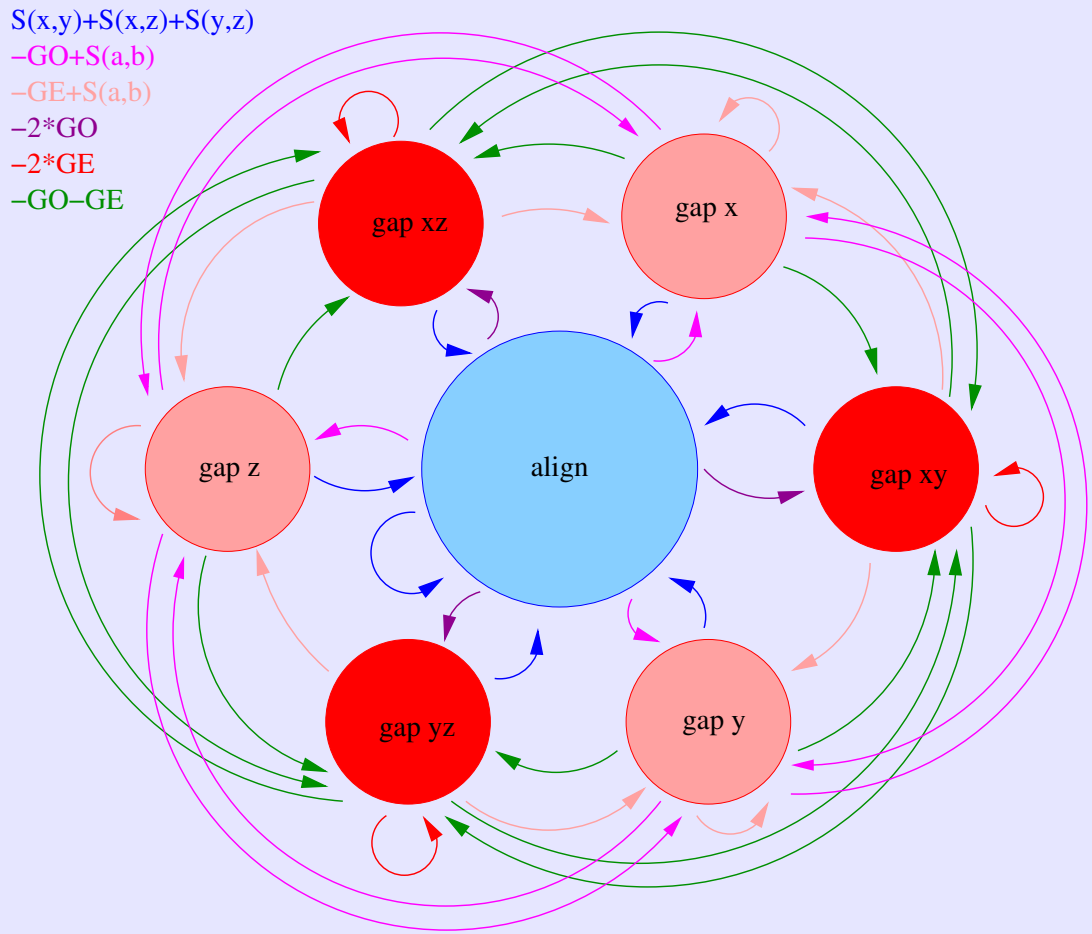
- not all reasonable state transitions possible
- fast approach, suitable results



Natural gap costs for three sequences



- all reasonable state transitions are possible
- higher computational effort necessary



Adjusting gap penalties

Global impacts

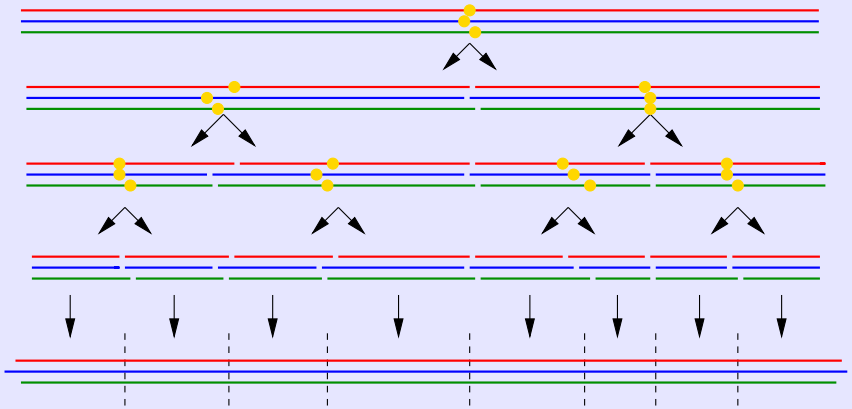
- length of three (groups of) sequences to be aligned
- length divergence of (groups of) sequences
- pairwise sequence identities
- scoring matrix (average mismatch score)

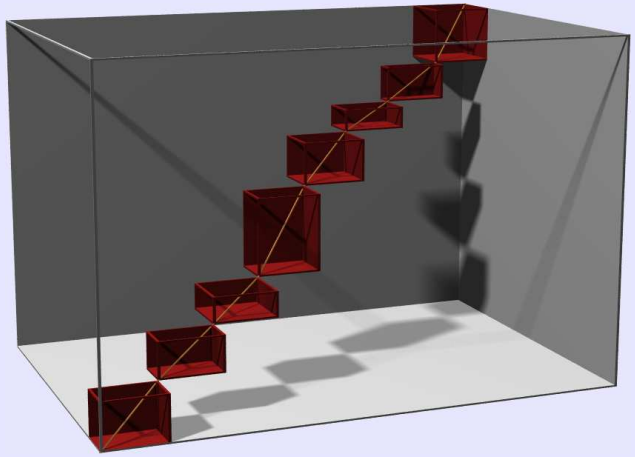
Position specific impacts

- number of sequences already containing gaps at this position
- distance from already introduced gaps
- presence of hydrophilic stretches in protein sequences
- residue specific gap penalties in protein sequences

Sequence weighting not implemented so far!

Divide & Conquer[1]





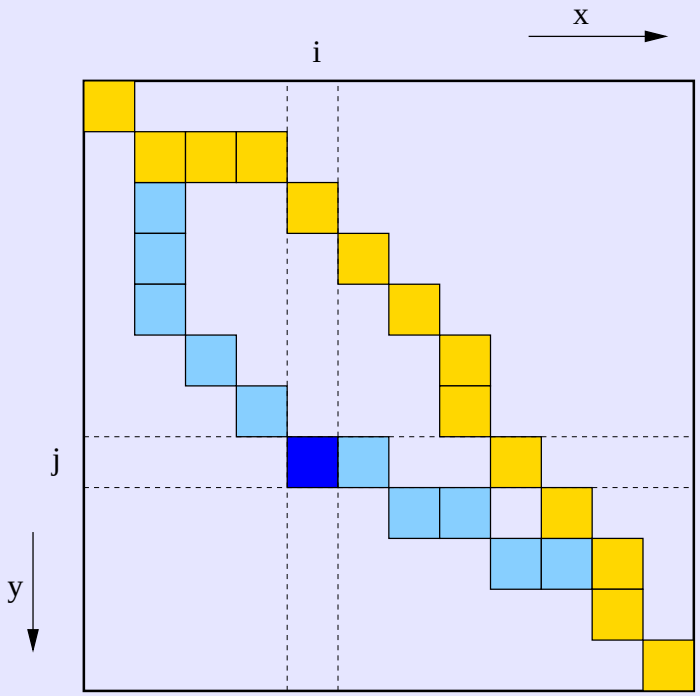
- because of cubic complexity in sequence length, very large sequences cannot be aligned *en bloc*
- use a divide-and-conquer-recurrence if sequence length exceeds a given limit
- choice of slicing positions has strong impact on alignment quality

[1] J. Stoye: Multiple Sequence Alignment with the Divide-and-Conquer Method, *Gene* 211(2), GC45-GC56, 1998. (Gene-COMBIS)

Calculating slicing positions

Choice of optimal slicing positions c_x, c_y and c_z?

Guess: Slicing positions c_x, c_y and c_z should lie on traceback path in dynamic programing algorithm.



yellow: optimal path

blue: optimal path running through (i, j)

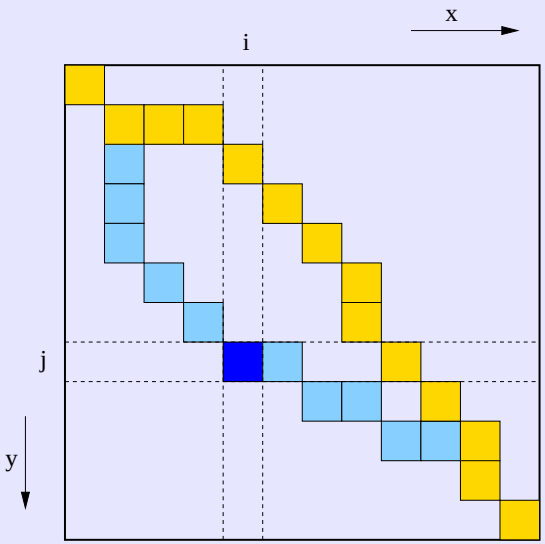
$\longrightarrow$ move (i, j) to optimal path

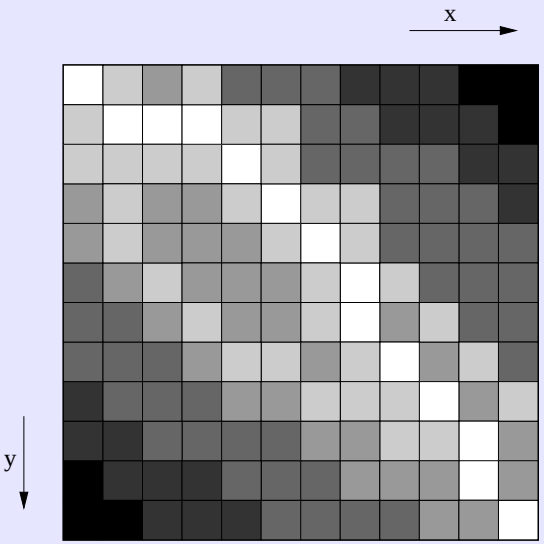
Problem: optimal path not known

Additional pairwise cost matrix

Definition

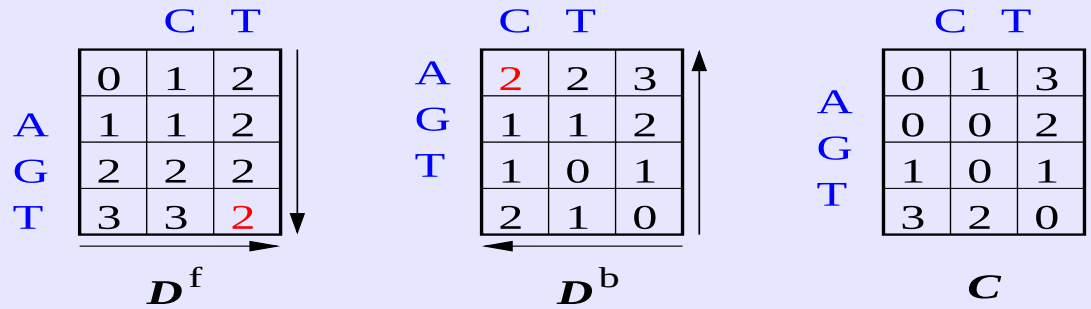
For each pair of possible slicing positions ($0 \leq i \leq |S_a|$, $0 \leq j \leq |S_b|$) the additional cost for slicing sequence S_a at position i and sequence S_b at position j is given by the additional pairwise cost matrix $C_{ab}(i, j)$.





- darker regions mean higher additional costs
- optimal traceback path has lowest additional cost

Calculating slicing positions

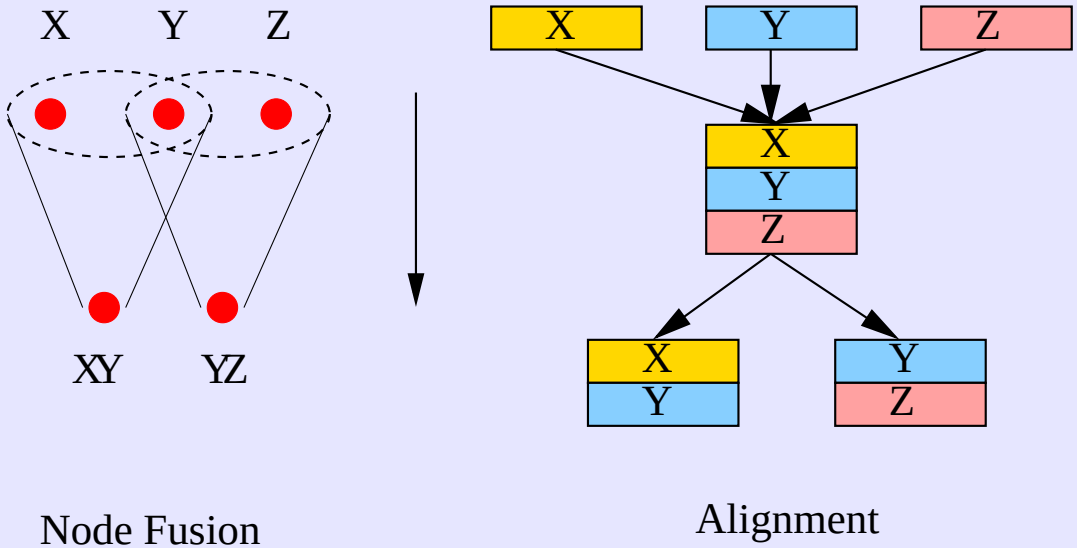


Algorithm

Given an additional score function ω:

1. Calculate for every pair a, b of sequences S_x, S_y, S_z additional cost matrix C_{ab} with: $C_{ab}(i,j) = D^f_{ab}(i,j) + D^b_{ab}(i,j) - \omega^{opt}_{ab}$ $(0 \leq i \leq |S_a|,\ 0 \leq j \leq |S_b|)$
2. Set $\hat{i} = \lceil |S_x|/2 \rceil$ fixed (assume that sequence x is longest)
3. Find j, k for which $C_{xy}(\hat{i},j) + C_{xz}(\hat{i},k) + C_{yz}(j,k)$ becomes minimal $(0 \leq j \leq |S_y|,\ 0 \leq k \leq |S_z|)$

Alignment splitting



Node Fusion

Alignment

- XYZ result of three-way alignment of X, Y and Z
- split XYZ up into XY and YZ
- Y contained in both alignments XY and $YZ \rightarrow$ iteratively delete sequences of Y either in XY or in YZ
- sequentially delete sequences from alignment that gains higher score after deletion

Complexity considerations

Once per program run

Algorithm step	**Time**	**Space**
Calculating distances	$\mathcal{O}(l^2 \cdot n^2)$	$\mathcal{O}(l^2 + n^2)$
Determing alignment order	$\mathcal{O}(n^3)$	$\mathcal{O}(n)$

For every 3-way alignment step

Calculate slicing positions	$\mathcal{O}(n^2 \cdot l^2)$	$\mathcal{O}(l^2)$
D&C-Alignment (D&C)	$\mathcal{O}(n^2 \cdot l \cdot L^2)$	$\mathcal{O}(L^3)$
Eliminating duplicates	$\mathcal{O}(n^2)$	$\mathcal{O}(n)$

l: average sequence length

n: number of sequences

L: divide & conquer length limit

Outline

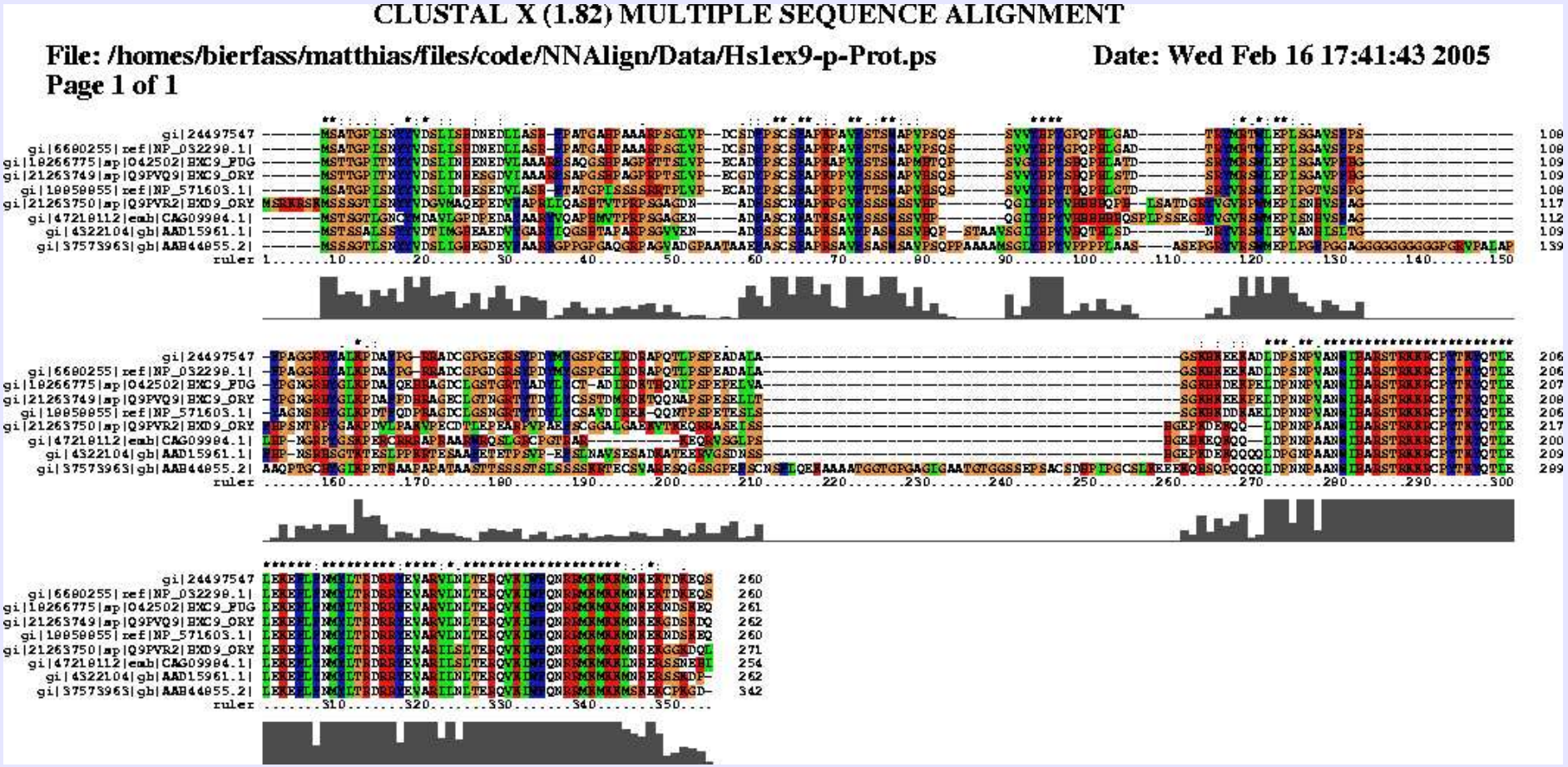
Results

Compare *NNAlign* with *ClustalW*

Parameter settings (default *ClustalW* settings):

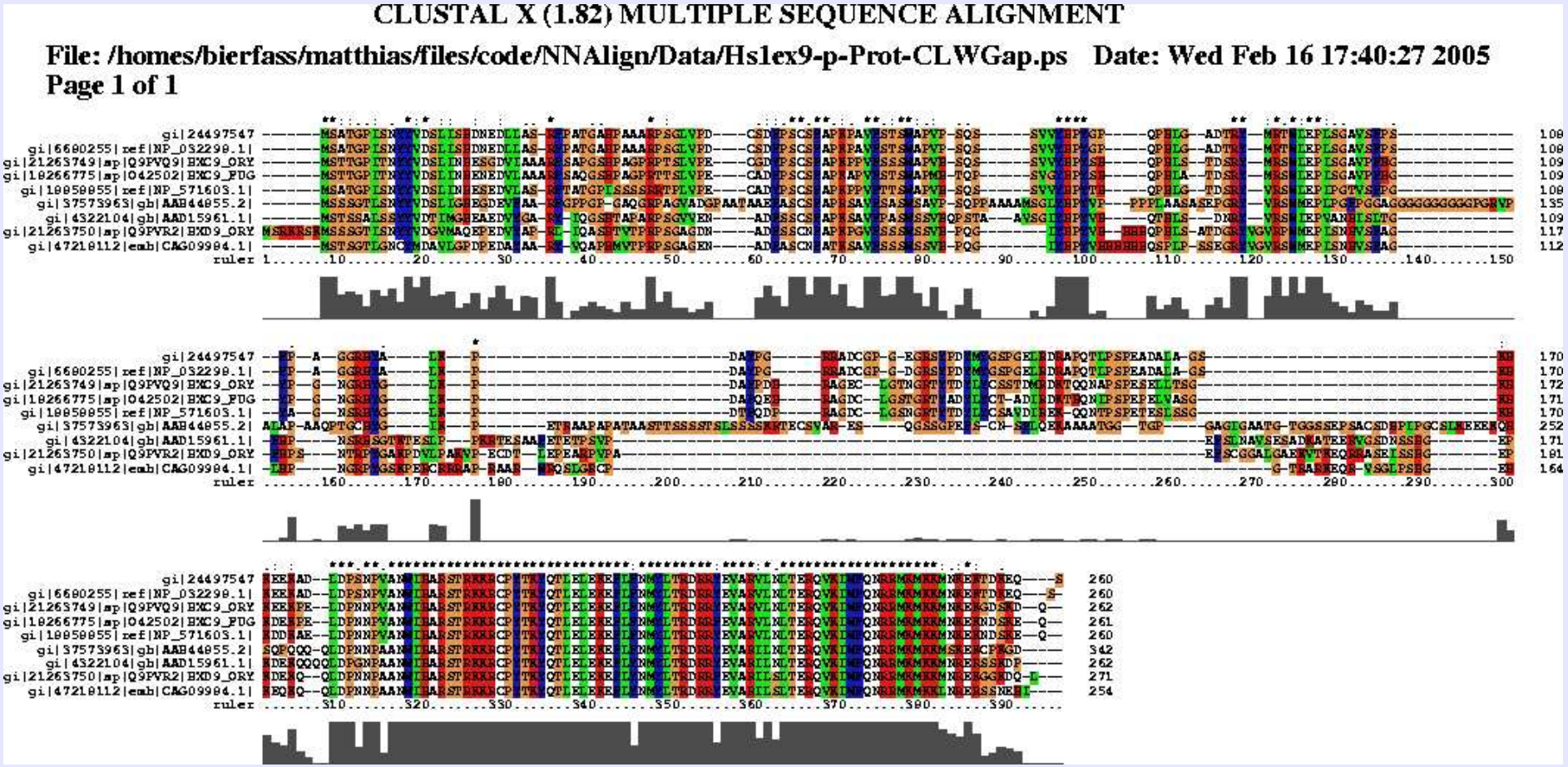
- gap-open penalty: 10.0
- gap-extension penalty: 0.2
- Protein scoring matrix: BLOSUM62
- terminal gaps are not weighted

Results: Exon 1 sequences of HOX C9/D9



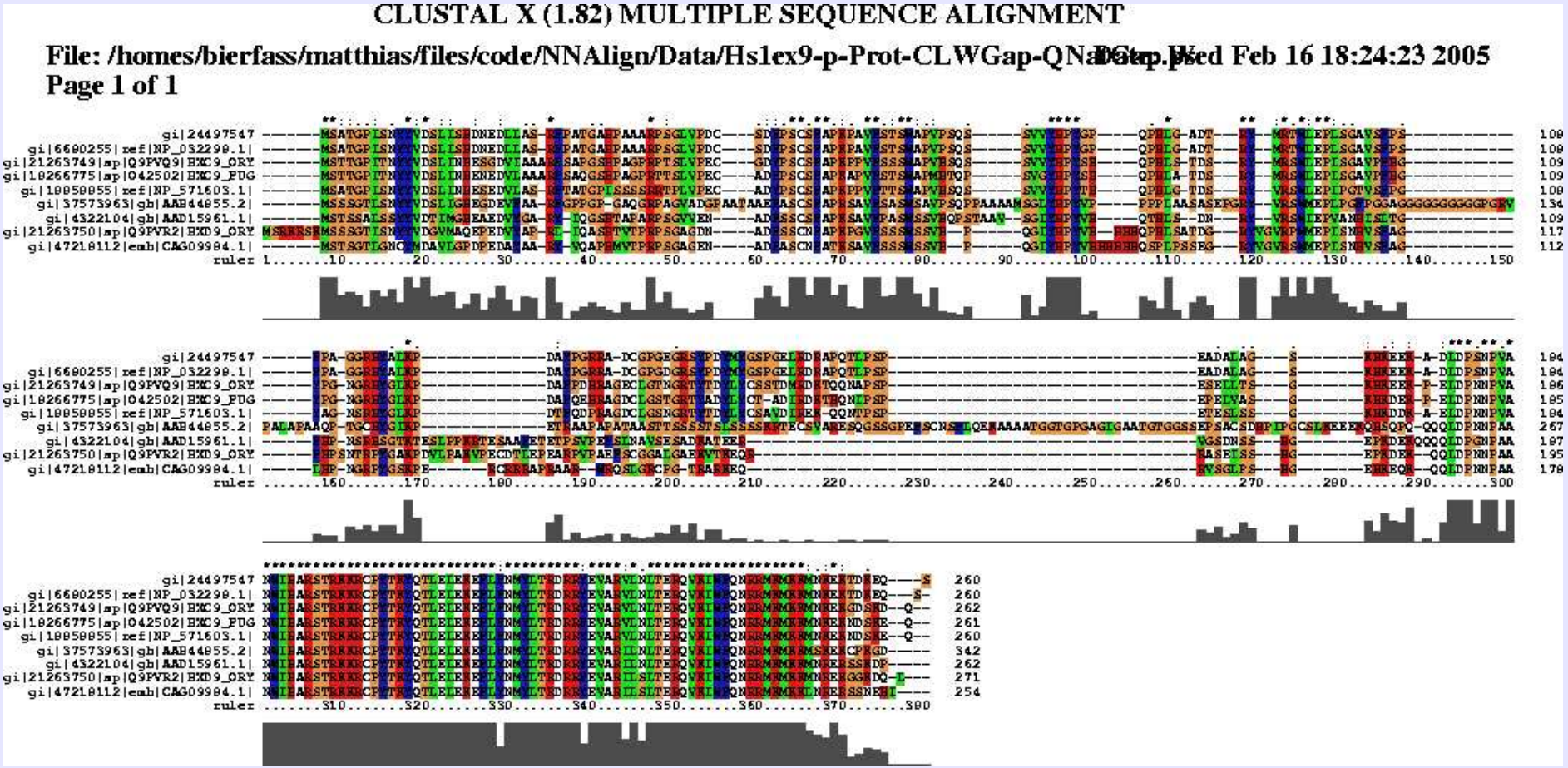
ClustalW: Score=741.57

Results: Exon 1 sequences of HOX C9/D9



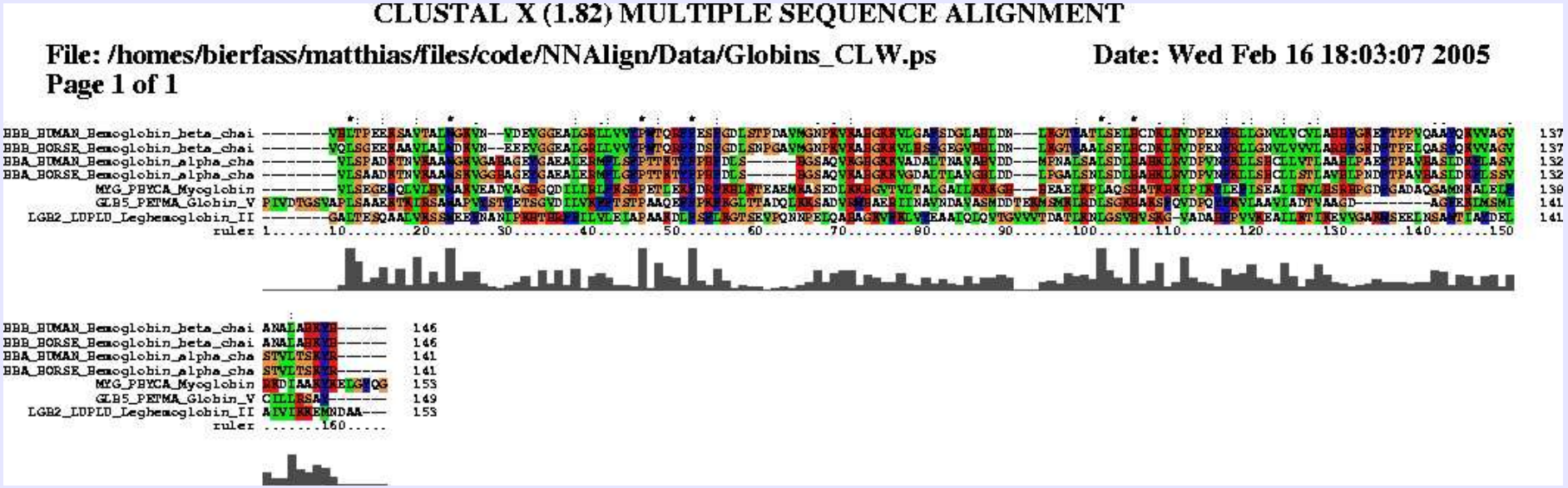
NNAlign: Score=776.85 (104.8% of ClustalW)

Results: Exon 1 sequences of HOX C9/D9



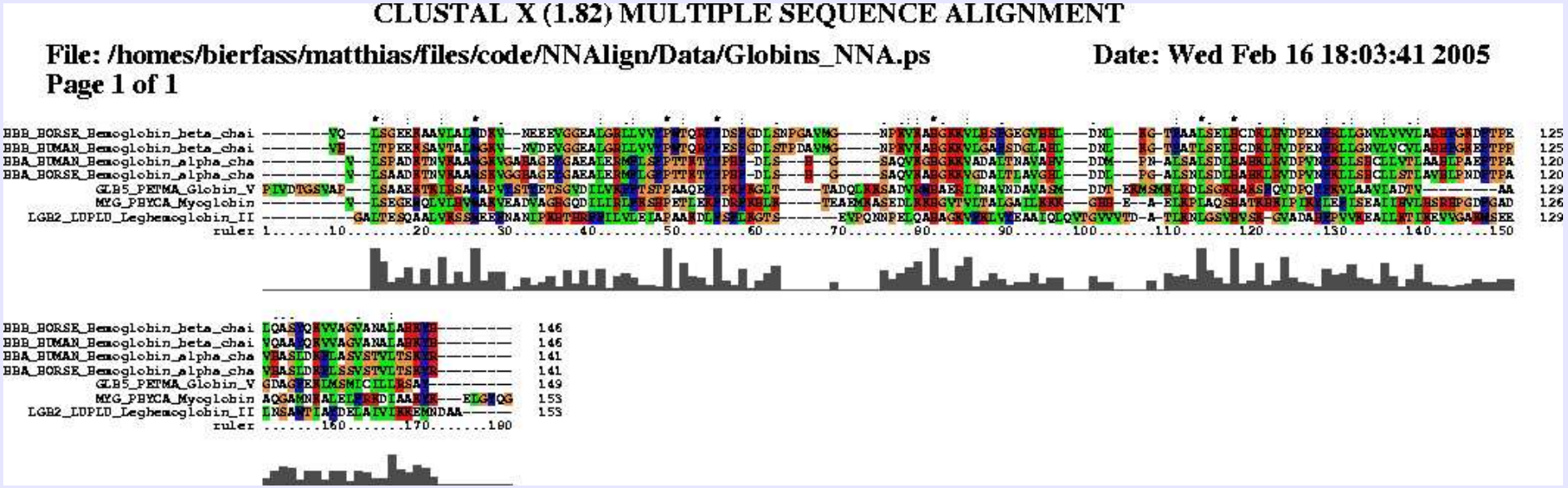
NNAlign (quasi-natural gap costs): Score=754.19 (natural gap costs: Score=776.85)

Results: Globin Domain



ClustalW: Score=151.89

Results



NNAlign: Score=158.38 (104.3% of *ClustalW*)

Outline

Summary and Outlook

Summary

- *NNAlign* alignments in many cases slightly better score than *ClustalW* alignment
- simultaneous alignment of three sequences has benefit
- natural gap costs increase alignment score compared to quasi-natural gap costs

Outlook

- implement ability for sequence weighting
- optimize splitting process
- optimize running time and memory consumption
- debug...

Thank you for your attention!