Sequence Analysis

Introduction to Bioinformatics
Dortmund, 16.-20.07.2007

Lectures:
Sven Rahmann

Exercises:
Udo Feldkamp, Michael Wurst
Overview

- Strings
- Pattern Matching
- Alignments
- Scoring Alignments (Cost, Score)
- Optimal (Global and Local) Alignments
- Modeling Optimal Local Alignment
- Multiple Alignment
- Modeling Optimal Global Multiple Alignment
Fundamentals

- Alphabet, e.g., \{A,C,G,T\}
- Strings = sequences over an alphabet
- Substring, AG of TGAGC (contiguous) number quadratic in string length
- Prefix, Suffix
- Subsequence, TAC of TGAGC (non-contiguous) number exponential in string length
Distances between Strings

- **q-gram Distance**
  - count and compare substrings of length $q$
  - Example: ACGT and GTAC with $q=2$:
    AC: (1,1), CG: (1,0), GT: (1,1), TA: (0,1)
    sum of differences: 2
    not a metric in the mathematical sense:
    can have distance=0 for two different strings

- **Hamming Distance** (if $s, t$ have same length)
  - Count number of differing positions
  d(ACGT, GTAC) = 4
Edit Distances

- Two defining principles:
  - set of operations to act on strings
  - maximum parsimony ("maximale Sparsamkeit")

- Perform operations on one string to create the other. Determine minimal number of operations required.

- Typical: Copy (=no op), Substitute, Delete, Insert

- Example: ACGT to AGTC: edit distance 2
Edit Distance Problem

• Given two strings s, t,
  – determine their edit distance
  – output a shortest “edit path”

• This is a problem specification formal enough for computer scientists to work on

• Applications:
  – Biological sequence comparison
  – Version control of text files
  – ...
Algorithms

• An algorithm is like a recipe (but more exact):
  – Input specification
  – Output specification
  – Series of well-defined steps

• Important properties:
  – Correctness (Incorrect algorithms are dangerous)
  – Termination after finite number of steps [?]
  – Efficiency (time, memory)
Problem, Algorithm, Program

- For one problem, there can be many algorithms.
- For one algorithm, there can be many implementations / programs
- Algorithm: often specified in pseudo-code
- Program: written in a formal language
Solving the Edit Distance Problem

• Consider every possible edit path
  – How many edit paths from s to t are there? Exponentially many
  – Would be a very slow algorithm

• Use structural properties of the edit distance
  – Optimal edit sequence for whole s, t contains optimal edit sequences for some prefixes of s, t
  – Solve the problem for all pairs of prefixes, starting with the small ones
  – Technique called “Dynamic Programming”
Edit Distance Algorithm

- Let $s = s_1...s_m$, $t = t_1...t_n$
- Let $D(i,j) := \text{edit distance of } s_1..s_i \text{ and } t_1...t_j$
- Clearly $D(0,j) = j$ for all $j$, $D(i,0)$ for all $i$
- In general, 
  $D(i,j) = \min \{ D(i-1,j)+1, D(i,j-1)+1, D(i-1,j-1) + 1[s_\neq t_j] \}$
- Three ways to edit $(i,j)$ $s_1..s_i$ into $t_1...t_j$ from shorter edit paths.
- Example: ACGT to AGTC
Recovering the Edit Path

- $D(m,n)$ is the edit distance of $s$ and $t$, but does not tell us the edit path!
- Remember which of the 3 possibilities for each $(i,j)$ lead to the minimum, i.e., keep “back-pointers” when filling in $D(i,j)$
- Trace back from $(m,n)$ to $(0,0)$
- Analysis:
  - Time: $O(mn)$, “quadratic”, not “exponential”!
  - Memory: $O(mn)$, can be reduced to $O(m+n)$
  - $O(x)$ means: not more than a constant times $x$
Pairwise Alignment

• Recall some edit paths ACGT to AGTC:
  A, C->G, G->T, T->C [suboptimal, 3]
  A, del C, G, T, ins C  [optimal, 2]

• Write this differently with “gap character”:
  ACGT   ACGT–
  AGTC   A–GTC
  :      :   :   :

• Each row without gaps shows original sequence.
• Each column shows one edit operation
• Matches highlighted with : or consensus letter
Scoring an Alignment

- Edit distance works with “distance” or “costs”, in particular “unit cost” (everything costs 1)
- Alignment often uses “scores”, can depend on type of indel or substitution. Example (purine/pyrimidine scoring):
  - score(A,A) = 1,
  - score(A,C) = score(A,T) = -1,
  - score(A,G) = 0,
  - score(A,-) = -3,
- Score of an alignment: Sum of column scores
Computing an Optimal Alignment

- Algorithm essentially unchanged (max, not min) “Needleman-Wunsch Algorithm”
- Simple because score is additive
- Extension (more tricky): **affine gap costs**
  Gap of length 2, 3, ... should be cheaper than 2, 3, ... times a gap of length 1.
- Affine gap costs not additive
Global vs Local Alignment

- Global Alignment aligns whole sequences. Makes sense when they are overall similar.
- Frequently, only the most similar substrings are of interest: “Local Alignment”
- Other variants:
  - Global with “free end gaps” (overhanging ends)
  - Approximate pattern matching (finding t in s)
- Examples (global, local, end gaps, pattern matching):
  
<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A----TC</td>
<td>TC</td>
<td>--ATC</td>
<td>ATC</td>
<td></td>
</tr>
<tr>
<td>AGTTC</td>
<td>TC</td>
<td>AGTTC</td>
<td>AGTTTC</td>
<td></td>
</tr>
</tbody>
</table>
Algorithms for Alignment

• All above problems can be solved with (simple) modifications of the global Needleman-Wunsch algorithm:
  – Local: Smith-Waterman algorithm
  – Approximate Pattern matching: Ukkonen's algorithm
Modeling Local Alignment

• Score of alignment: sum of column scores
• Local alignment: find highest-scoring substrings
• Problems:
  – shadow effect: long mediocre alignments mask short good alignments
  – mosaic effect: excellently aligning regions are interrupted by bad regions
• Reason: additivity of score
  – better to build long alignments to accumulate score
Length-Normalized Alignment

- Re-definition of score of an alignment:
  - \((\text{Sum of column scores}) \div (\text{Length})\)
  - Problem: Length 0? Short alignments?
  - Either use minimum length parameter,
  - or add pseudo-length \(L\):
    - \((\text{Sum of column scores}) \div (\text{Length} + L)\)

- Maximize this over all alignments of all substrings

- Algorithm known since 2002

- I don't know any www-based tools for it
Fast Local Alignment

• Exact standard local alignment runs in $O(mn)$ time ($m,n$: sequence lengths)
• Too slow for long sequences!
• Heuristics are algorithms that guarantee no optimal solution (but usually work well), but are often much faster
  – BLAST (Basic Local Alignment Search Tool), NCBI
  – FASTA
  – BLAT (Blast-Like Alignment Tool), UCSC
Where do Scores come from?

- Amino acids have physical and chemical properties
- Some amino acids are more similar than others.
- An expert could assign numerical “similarity scores”.
- Really?
- If score(I,L)=2 and score (V,W)=-3, what should score(P,A) be?
Log-Odds Scores

• During evolution, similar amino acids replace each other more frequently than dissimilar ones.

• Take this as definition!

• Amino acids are similar if they replace each other frequently, i.e., if we find them together in alignments more frequently than by chance.

• \( \text{score}(x,y; t) := \log \left( \frac{M(x,y; t)}{[f(x)\cdot f(y)]} \right) \)

• Parameters:
  - \( t \): a divergence time parameter
  - \( M(x,y; t) \): pair frequency at divergence time \( t \)
  - \( f(x) \): overall frequency of \( x \)
## BLOSUM62 Matrix

|   | A | R | N | D | C | Q | E | G | H | I | L | K | M | F | P | S | T | W | Y | V | X |
| R | -1 | 5 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| N | -2 | 0 | 6 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| D | -2 | -2 | 1 | 6 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| C | 0 | -3 | -3 | -3 | 9 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| Q | -1 | 1 | 0 | 0 | -3 | 5 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| E | -1 | 0 | 0 | 2 | -4 | 2 | 5 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| G | 0 | -2 | 0 | -1 | -3 | -2 | -2 | 6 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| H | -2 | 0 | 1 | -1 | -3 | 0 | 0 | -2 | 8 |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| I | -1 | -3 | -3 | -3 | -1 | -3 | -3 | -4 | -3 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |   |
| L | -1 | -2 | -3 | -4 | -1 | -2 | -3 | -4 | -3 | 2 | 4 |   |   |   |   |   |   |   |   |   |   |   |   |
| K | -1 | 2 | 0 | -1 | -3 | 1 | 1 | -2 | -1 | -3 | -2 | 5 |   |   |   |   |   |   |   |   |   |   |   |
| M | -1 | -1 | -2 | -3 | -1 | 0 | -2 | -3 | -2 | 1 | 2 | -1 | 5 |   |   |   |   |   |   |   |   |   |   |
| F | -2 | -3 | -3 | -2 | -3 | -3 | -3 | -1 | 0 | 0 | -3 | 0 | 6 |   |   |   |   |   |   |   |   |   |   |
| P | -1 | -2 | -2 | -1 | -3 | -1 | -1 | -2 | -2 | -3 | -3 | -1 | -2 | -4 | 7 |   |   |   |   |   |   |   |   |   |
| S | 1 | -1 | 1 | 0 | -1 | 0 | 0 | 0 | -1 | -2 | -2 | 0 | -1 | -2 | -1 | 4 |   |   |   |   |   |   |   |
| T | 0 | -1 | 0 | -1 | -1 | -1 | -1 | -2 | -2 | -1 | -1 | -1 | -1 | -2 | -1 | 1 | 1 | 5 |   |   |   |   |   |
| W | -3 | -3 | -4 | -4 | -2 | -2 | -3 | -2 | -3 | -2 | -3 | -1 | 1 | 1 | 4 | -3 | -2 | 11 |   |   |   |   |   |
| Y | -2 | -2 | -2 | -3 | -2 | -1 | -2 | -3 | 2 | -1 | -1 | -2 | -1 | 3 | -3 | -2 | -2 | 2 | 7 |   |   |   |   |
| V | 0 | -3 | -3 | -3 | -1 | -2 | -2 | -3 | 3 | 1 | -2 | 1 | -1 | -2 | -2 | 0 | -3 | -1 | 4 |   |   |   |   |
| X | 0 | -1 | -1 | -1 | -2 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -2 | 0 | 0 | -2 | -1 | -1 | -1 | -1 |   |
Multiple Alignment

• Alignment of $k \geq 3$ sequences
• Each row, without gaps, spells one sequence
• Scoring a multiple alignment
  – Sum-of-pairs score
    Sum up pairwise alignment scores of all pairs
  – Tree score
    Given a tree with sequences at inner nodes,
    sum up pairwise alignment scores along all edges
Pfam – Protein domains

- **URL:** http://www.sanger.ac.uk/Software/Pfam/

Pfam is a large collection of multiple sequence alignments and hidden Markov models covering many common protein domains and families. For each family in Pfam you can:

- Look at multiple alignments
- View protein domain architectures
- Examine species distribution
- Follow links to other databases
- View known protein structures

For more information on Pfam, on using this site, or on the changes between Pfam releases 21.0 and 22.0, click here.

Pfam can be used to view the domain organisation of proteins. A typical example is shown below. Notice that a single protein can belong to several Pfam families.

74% of protein sequences have at least one match to Pfam. This number is called the sequence coverage and is shown in the pie chart on the right.
Example: Serpin in Pfam

Pfam entry Serpin

Accession number: PF00079

Previous identifiers: serpin;

Serpin (serine protease inhibitor)

Structure is a multi-domain fold containing a bundle of helices and a beta sandwich.

INTERPRO description (entry IPR000215)

Peptide proteinase inhibitors can be found as single domain proteins or as single or multiple domains within proteins; these are referred to as either simple or compound inhibitors, respectively. In many cases they are synthesised as part of a larger precursor protein, either as a prepropeptide or as an N-terminal domain associated with an inactive peptidease or zymogen. Removal of the N-terminal inhibitor domain either by interaction with a second peptidease or by autocatalytic cleavage activates the zymogen.

Serpins (SERine Proteinase INHibitors) PUBMED:14705360, PUBMED:2630952, PUBMED:8417965 belong to MEROPS inhibitor family M4, clan ID. Serpins are proteins that are primarily known as irreversible serine protease inhibitors active against S1 (, S8 ( and C14 () peptidases. There are both extra- and intra-cellular serpins, which are found in all groups of organisms with the notable exception of fungi PUBMED:11116082, PUBMED:12411597.

Serpins and their homologues are a group of high molecular weight (40 to 50 kDa) structurally related proteins involved in a number of fundamental biological processes such as blood coagulation, complement activation, fibrinolysis, angiogenesis, inflammation, tumour suppression and hormone transport. All known serpins have been classified into 16 clades and 10 orphan sequences, the vertebrate serpins can be conveniently classified into six sub-groups PUBMED:11116092. In human plasma they represent approximately 2% of the total protein, of which 70% is alpha-1-antitrypsin.

In contrast to "rigid" proteinase inhibitors, such as those of the Kunitz or Kazal families, the serpins are metastable proteins (active-state proteins) which interact with their substrate and irreversibly trap the acyl intermediate as a result of a major conformational change PUBMED:11116079; they are best described as suicide substrate inhibitors. The common structure of these proteins is a multi-domain fold containing a bundle of 8 or 9 alpha helices and a beta sandwich formed by 3 beta sheets. The reactive centre loop (RCL) is found in the C-terminal part of these proteins. On the basis of strong sequence similarities, a number of proteins with no known inhibitory activity are said to belong to this family, these include: angiotensinogen, corticosteroid-binding globulin and thyroxin-binding globulin PUBMED:12824063.
Continued...

For additional annotation, see the PROSITE document PDOC00356.
Continued...

<table>
<thead>
<tr>
<th>Database References</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PDB</strong></td>
</tr>
<tr>
<td>You can find out how to set up Rasmol <a href="#">here</a>.</td>
</tr>
<tr>
<td><strong>PFAMB</strong></td>
</tr>
<tr>
<td><strong>PRODOM only:</strong></td>
</tr>
<tr>
<td>PB172042 PB172407 PB179135</td>
</tr>
<tr>
<td><strong>PROSITE</strong></td>
</tr>
<tr>
<td>PD0C0256</td>
</tr>
<tr>
<td><strong>HOMSTRAD</strong></td>
</tr>
<tr>
<td>serpin</td>
</tr>
<tr>
<td><strong>SYSTERS</strong></td>
</tr>
<tr>
<td>Serpin</td>
</tr>
<tr>
<td><strong>PANDIT</strong></td>
</tr>
<tr>
<td>Serpin</td>
</tr>
<tr>
<td><strong>FUNSHIFT</strong></td>
</tr>
<tr>
<td>Serpin</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Literature References</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Pfam specific information</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author of entry</strong></td>
</tr>
<tr>
<td><strong>Type definition</strong></td>
</tr>
<tr>
<td><strong>Source of seed members</strong></td>
</tr>
<tr>
<td><strong>Average Length</strong></td>
</tr>
<tr>
<td><strong>Average %id</strong></td>
</tr>
<tr>
<td><strong>Average Coverage</strong></td>
</tr>
</tbody>
</table>
Multiple Alignment

| THBG_RAT/38-415 | QNATLYKPSINADFAKRYLK.I5V .. ENPDLNIFSPVSISAALAMISFGCSSSTQTQ1LEIVLGGNITIDTPVKE .. |
| THBG_HUMAN/35-412 | PNNATLYKSSINADFAFRLYK.RTV .. ETDPKKNNIPSPVSIAALVMISFGCSSSTOT1LEIVLGGNITIDTPVKE .. |
| A1AT_RAT/37-409 | QSPYTRIKSSNLADFAFSLYK.LVH .. QSNTSNFSPSSITTAFAHLSGKDQDTRQLEIGFNTIQPEAD .. |
| A1AT2_MOUSE/37-410 | QSPASHETANLNGDAFALSYYL.RLV .. QSSNTSNIFSPSIAATAFAMLSGKGDQDHTTIILEGIFNITQSEAD .. |
| A1AT_BOVIN/41-413 | QEAACKHMNLNADFALSYYL.LAH .. QSSNTSNIFSPSIAATAFAMLSGKGDQDHTTIILEGIFNITSEAD .. |
| A1AT_HUMAN/43-415 | DEPTFKKLTNPMLAFALSYYL.RQ .. QSSNTSNIFSPSIAATAFAMLSGKGDQDHTTIILEGIFNITSEAD .. |
| A1AT_RABIT/38-410 | DHCPHRIAPSALFALSYYL.VAH .. QSSNTSNIFSPSIAATAFAMLSGKGDQDHTTIILEGIFNITSEAD .. |
| A1AT_CAVPO/29-400 | AQGPSQISIPSALFAHLSYYL.YLT .. QSSNTSNIFSPSIAATAFAMLSGKGDQDHTTIILEGIFNITSEAD .. |
| A1AT_DIDMA/36-407 | EYSSTRSIKSPYMTDFSLYR .. LVQ .. KSNTSNIFSPSIAATAFAMLSGKGDQDHTTIILEGIFNITSEAD .. |
| A1ATR_HUMAN/46-417 | EDLACQKSYVNDLAFDLKSYWVI .. HNQQHVLSPTLSAKAMISGLTIDRTTIILEGIFNITQSEAD .. |
| AACT_HUMAN/45-420 | VD .. LGQSNSLANTSYVNDLAFDLKSYWVI .. KAPKDNVISVTALESIFLSGKGDQDHTTIILEGIFNITQSEAD .. |
| CPI6_RAT/42-417 | LDS .. ITIATISNTDAFALYK.LAL .. RNPDKNVWSPSLISALAVLSGAQGKENDLJEIGFNTPETE .. |
| SPA3C_MOUSE/42-414 | LDS .. ITIATISNTDAFALYK.LAL .. RNPDKNVWSPSLISALAVLSGAQGKENDLJEIGFNTPETE .. |
| SPA3K_MOUSE/43-417 | DDS .. ITIATISNTDAFALYK.LAL .. RNPDKNVWSPSLISALAVLSGAQGKENDLJEIGFNTPETE .. |
| CPF1_RAT/40-415 | ILS .. ITIATWISNTDAFALYK.LAL .. RNPDKNVWSPSLISALAVLSGAQGKENDLJEIGFNTPETE .. |
| IPS5_HUMAN/34-406 | LTVSNITPAVPSSRDEFTLR .. ALAS .. AAPSQNONIPSPWSMSHLSAMISGKSKMNQIEGLNOSKSE .. |
| CBG_MOUSE/27-396 | DSSSRQILNPSSPARRQYNADA .. LV .. LSADKNTIISLSSALAMHLSAMISGKSKMNQIEGLNOSKSE .. |
| CBG_RAT/27-395 | SSNSHRGLAPTNYDFALYQR .. LVA .. LSADKNTIISLSSALAMHLSAMISGKSKMNQIEGLNOSKSE .. |
| CBG_HUMAN/32-404 | MNHHRGALASYDFFALYQR .. LVA .. LSADKNTIISLSSALAMHLSAMISGKSKMNQIEGLNOSKSE .. |
| CBG_RABIT/10-382 | TRSPPRCLAPAVDFALYQR .. LVS .. SAPDRNISCSWPSWALAMHLSAMISGKSKMNQIEGLNOSKSE .. |
| EP45_XENLA/61-432 | LTKEEITIISENSDFVWLNLSTESKRSRPKNIFSPSSPAFYMLGKASETSQIIKIGSIFNKKQESQ .. |
| HEP2_HUMAN/119-496 | GSRIQRQILNFAFNLRYR .. IKDO .. VNTFDINPAPVGSSTAMCHLSWIGKGETEIQHSHIIHFDFVNASKSKYET .. |
| OVALY_CHICK/1-388 | MDS .. VSINAVAFKCPFVENE .. HK .. HNQHVLSPTLSAKAMISGLTIDRTTIILEGIFNITQSEAD .. |
| OVALCHICK/2-386 | GS .. IGAASMECFDEVEE .. LV .. EAMNITPAATLAMALVIGKGDQDRTQ1NEKVRREDFKLGFSGIEAQ .. |
| SPBS_HUMAN/1-376 | MDV .. IAANSTFCFALLKLT .. LG .. KDNSKNFVPSFSSPLSAALYNVHGAQGTANNAQAANLIEKSSGGDD .. |
| IILEI_HORSE/1-379 | MEQ .. IASTNKFAVFLR .. LT .. SDPTQNIIPSPWSLAMIFLTPNGTANNAQQENKSSGGDD .. |
| SPBS_HUMAN/1-375 | MDA .. IGLANLSAVFLKV .. LG .. KEPLKNVWSPLSTLSLAVQKGDQDRTQ1NEKVRREDFKLGFSGIEAQ .. |
| ANIT_HUMAN/76-461 | TNRVWSVKNSEAFATTTQY .. IADS .. KDNNIIPSPWSLAMIFLTPNGTANNAQQENKSSGGDD .. |
| SERE_CHICK/23-396 | LSDKATTIADRSTTLAHLYHA .. MAX .. DKNMEN11LIPWSWASSISTLYGSLGKATTASQAQKAVISADKNNDDY .. |
| FRIZ_HORHU/6-395 | ATDVRLTIAH .. TEPFAHLS .. TCSNPERAASGSAFSPSLSVHLSLTTAG .. AATRDQWAIIGCGGDAKELNA .. |
If you use HMM-Logos in your publication, please cite
The paper is "open access": http://www.biomedcentral.com/1471-2105/5/7
Why Multiple Alignment?

- Multiple Alignment contains much more information than the pairwise alignments.
- Detect weak, but characteristic, signals for a family of sequences.
- Mainly global (maybe free end gaps): Only align related sequences.
- Local multiple alignment is rather “motif finding” (different problem).
Scoring a Multiple Alignment

• **Sum-of-pairs score**
  Sum up pairwise alignment scores of all pairs
  --: does not consider evolutionary relationships

• **Tree score**
  Given a tree with sequences at inner nodes,
  sum up pairwise alignment scores along all edges
  +: evolutionary basis
  –: complicated to compute (need a tree first)

• **Weighted sum-of-pairs score**
  “dirty hack” to make SP score look more like tree score, easier to compute
Finding the Best Multiple Alignment

• Highest Score ≠ Biologically correct (!)
  No one knows the ideal scoring function

• Computational problems:
  – (W)SP Problem: Given sequences, find multiple alignment that maximizes (W)SP score
  – Tree Alignment Problem: Given sequences + tree, find sequences at inner nodes + multiple alignment maximizing tree score
  – Generalized Tree Alignment Problem: Given sequences, find tree + sequences at inner nodes + multiple alignment maximizing tree score
Optimal Multiple Alignment is NP-Hard

• Problem complexity classes:
  – P: Problems for which there exists an algorithm that solves them in polynomial time
  – NP: Problem for which there exists a nondeterministic algorithm that solves them in polynomial time
  – Clearly NP contains P. Unknown whether P = NP (but seems unlikely)

• NP-hard problem: “hardest” problems in NP
  – If we find a polynomial algorithm for an NP hard problem, we can find one for any problem in NP.
Methods

- Exact methods (very slow, time exponential in the number of sequences)
  - multidimensional dynamic programming, similar to pairwise alignment, but over k dimensions for k sequences
  - runtime heuristics: safely cut away some parts of the search space.
  Idea of Carillo-Lipman: Optimal multiple alignment cannot contain too bad pairwise alignments
Methods

• Quality heuristics: do not guarantee optimal solution, but faster
  – Center-Star method
  – Divide & Conquer Alignment (DCA)
  – Progressive alignment (“clustering”)
  – many more …
Progressive Alignment

- CLUSTALW / CLUSTALX widely used
- Basic idea: Reduce multiple alignment to a series of pairwise alignments
- Order determined by a guide tree:
  - Compute distances between sequences
  - Create tree from distances
  - Process tree bottom-up
- Intuition: Align most similar sequences first