Sequence Alignment Methods: Dynamic Programming and Heuristic Approaches.

Jaroslaw Meller

Biomedical Informatics, Children's Hospital Research Foundation, University of Cincinnati

Dept. of Informatics, Nicholas Copernicus University

Problems and methods:

Problem → Algorithms → Programs

Sequencing → Fragment assembly problem → The Shortest Superstring Problem → Phrap (Green, 1994)

Gene finding → Hidden Markov Models, pattern recognition methods → GenScan (Burge & Karlin, 1997)

Sequence comparison → pairwise and multiple sequence alignments → dynamic algorithm, heuristic methods → BLAST (Altschul et. al., 1990)

Trying out the routine BLAST searches:

- Let us BLAST some sequences …

  NCBI HomePage.htm

  Scoring matrix (BLOSUM62 etc.), PSSM and PsiBLAST, gap penalties, Smith-Waterman vs. heuristic alignment, repeats filtering, p-value, E-value, B-value …

- Why homology is so useful?

Sequence similarity is at the core of computational biology …

- DNA/RNA/Protein machinery: from sequence to sequence to structure to function to sequence,
- High sequence similarity implies usually significant functional and/or structural similarity,
- Profiles and multiple alignments methods such as PsiBLAST are significantly more sensitive (with very high specificity) than pairwise sequence alignments.

What do we need to measure similarity between sequences?

- A similarity measure: substitution (or scoring) matrices, such as PAM or BLOSUM matrices.
- An alignment algorithm: dynamic programming generates an optimal (approximate) string matching in quadratic time, still to slow? – use faster heuristics!
- A statistical significance measure to filter our well scoring matches by chance: extreme value distribution and various estimates of statistical confidence.

Deriving substitution matrices from known protein families:

- PAM matrices (Dayhoff et. al): extrapolating longer evolutionary times from data for very similar proteins with more than 85% sequence identity (short evolutionary time),

  $s(a,b \mid t) = \log P(b|a,t)/q_a$ e.g. $P(b|a,2)= \Sigma_c P(b|c,1)P(c|a,1)$

- BLOSUM matrices (Henikoff & Henikoff): multiple alignments of more distantly related proteins (e.g. BLOSUM50 with 50% sequence identity),

  $s(a,b) = \log p_{ab}/q_a q_b$ where $p_{ab} = F_{ab} / \Sigma_{cd} F_{cd}$

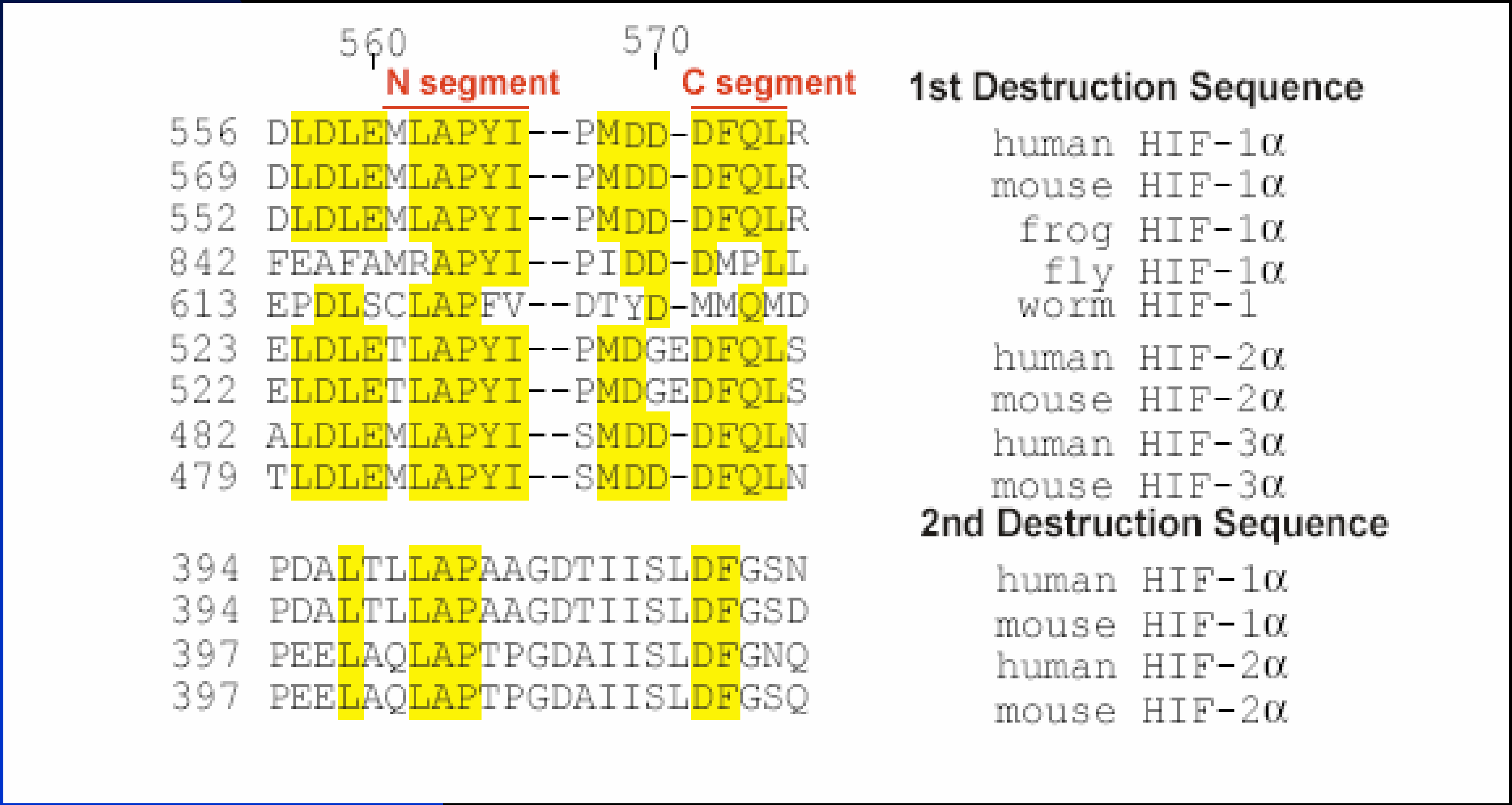
Expected score: $\Sigma_{ab}\, q_a q_b\, s(a,b) = - \Sigma_{ab}\, q_a q_b \log q_a q_b / p_{ab} = -H(q||p)$

Example (BLOSUM50):

	H	E	A	G	A	W	G	H	E
P	-2	-1	-1	-2	-1	-4	-2	-2	-1
A	-2	-1	5	0	5	-3	0	-2	-1
W	-3	-3	-3	-3	-3	15	-3	-3	-3
H	10	0	-2	-2	-2	-3	-2	10	0
E	0	6	-1	-3	-1	-3	-3	0	6

$d = 8$

Multiple alignment (family profiles) and Position Specific Scoring Matrices …



```
          560           570
           N segment     C segment     1st Destruction Sequence
556 DLDLEMLAPYI--PMDD-DFQLR            human HIF-1α
569 DLDLEMLAPYI--PMDD-DFQLR            mouse HIF-1α
552 DLDLEMLAPYI--PMDD-DFQLR             frog HIF-1α
842 FEAFAMRAPYI--PIDD-DMPLL              fly HIF-1α
613 EPDLSCLAPFV--DTYD-MMQMD             worm HIF-1
523 ELDLETLAPYI--PMDGEDFQLS            human HIF-2α
522 ELDLETLAPYI--PMDGEDFQLS            mouse HIF-2α
482 ALDLEMLAPYI--SMDD-DFQLN            human HIF-3α
479 TLDLEMLAPYI--SMDD-DFQLN            mouse HIF-3α
                                       2nd Destruction Sequence
394 PDALTLLAPAAGDTIISLDFGSN            human HIF-1α
394 PDALTLLAPAAGDTIISLDFGSD            mouse HIF-1α
397 PEELAQLAPTPGDAIISLDFGNQ            human HIF-2α
397 PEELAQLAPTPGDAIISLDFGSQ            mouse HIF-2α
```

Gap penalties – evolutionary and computational considerations:

- Linear gap penalties:

  $\gamma(g) = - g\, d$ for a gap of length g and constant d.

- Affine gap penalties:

  $\gamma(g) = - d - (g - 1)\, e$,

  where d is opening gap penalty and e an extension gap penalty.

Dynamic programming algorithm:

- Goal – find an optimal matching for two strings $S_1 = a_1a_2\ldots a_n$ and $S_2 = b_1b_2\ldots b_m$ over certain alphabet Σ, given a scoring matrix $s(a,b)$ for each a and b in Σ and (for simplicity) a linear gap penalty .
- Relation to minimal edit distance (number of insertions, deletions and substitutions) problem.
- Three stages: <u>recurrence relation, tabular computation and traceback</u>.

Recurrence relations:

- Global alignment (Needleman-Wunsch):

$F(0,0) = 0; F(k,0) = F(0,k) = - k d;$

$F(i,j) = \max \{ F(i-1,j-1)+s(a_i,b_j) ; F(i-1,j)-d ; F(i,j-1)-d \}$

- Local alignment (Smith-Waterman):

$F(0,0) = 0; F(k,0) = F(0,k) = 0;$

$F(i,j) = \max \{ 0 ; F(i-1,j-1)+s(a_i,b_j) ; F(i-1,j)-d ; F(i,j-1)-d \}$

Building DP table and finding the optimal alignment:

- Use the recurrence relations, starting from the left upper corner (convention).
- Find the highest score in the DP table (last, bottom right cell in the global alignment by definition)
- Trace back the alignment using the pointers in the DP graph that show how the best local steps led to the best overall alignment.

Examples (global alignment):

		H	E	A	G	A	W	G	H	E
	0	<-8	<-16	<-24	<-32	<-40	<-48	<-56	<-64	<-72
P	^-8	*-2	*-9	*-17	<-25	*-33	<-41	<-49	<-57	*-65
A	^-16	^-10	*-3	*-4	<-12	*-20	<-28	<-36	<-44	<-52
W	^-24	^-18	^-11	*-6	*-7	*-15	*-5	<-13	<-21	<-29
H	^-32	*-14	*-18	*-13	*-8	*-9	^-13	*-7	*-3	<-11
E	^-40	^-22	*-8	<-16	^-16	*-9	*-12	^-15	*-7	*3

HEAGAWGHE

--P-AW-HE

Examples (local alignment):

		H	E	A	G	A	W	G	H	E
	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0
A	0	0	0	*5	0	*5	0	0	0	0
W	0	0	0	0	*2	0	*20	<12	<4	0
H	0	*10	<2	0	0	0	^12	*18	*22	<14
E	0	^2	*16	<8	0	0	^4	^10	*18	*28

AWGHE

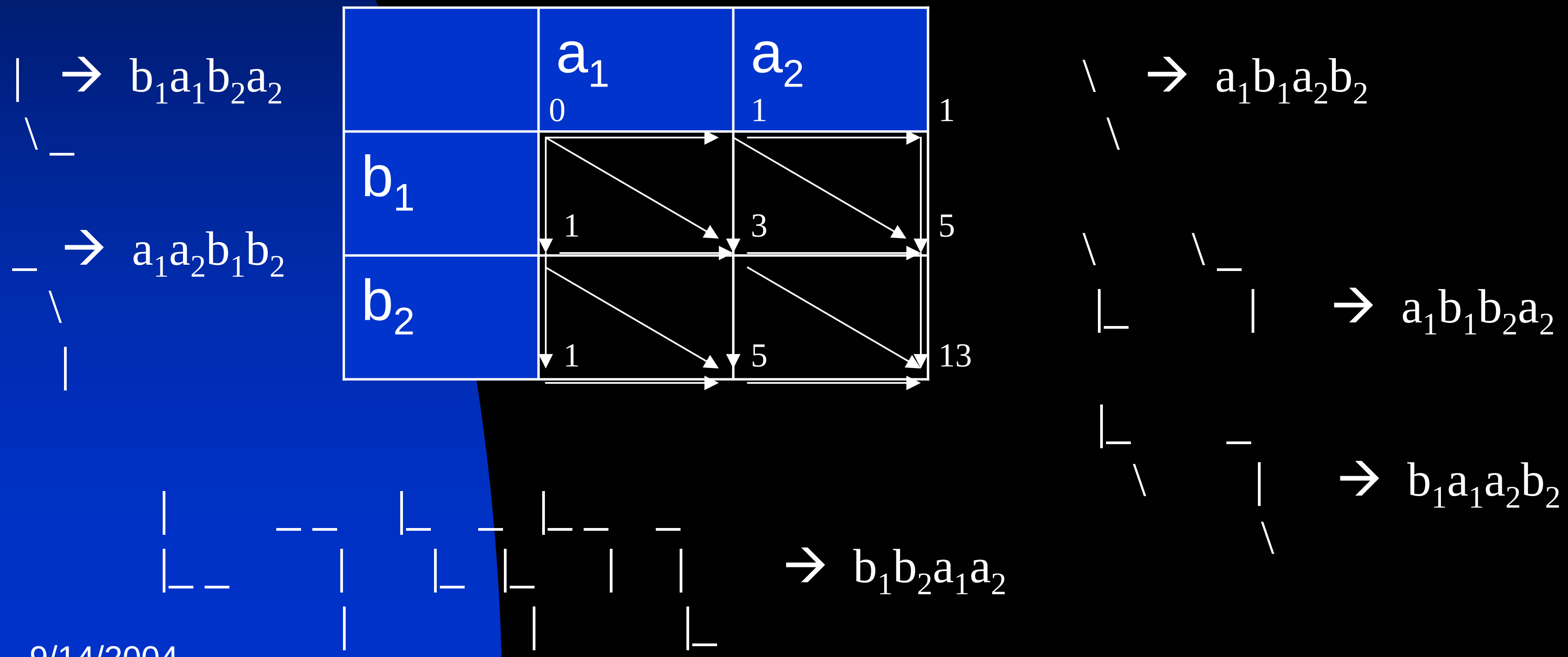
AW-HE

Why it works?

- All the possible alignments (with gaps) are represented in the DP table (graph).
- The score is a sum of independent piecewise scores, in particular, the score up to a certain point is the best score up to the point one step before plus the incremental score of the new step.
- Once the best score in the DP table is found the trace back procedure generates the alignment since only the best “past” leading to the present score is represented by the pointers between the cells.

Why it works?

- All the possible alignments (with gaps) are represented in the DP table (graph). Consider an example with two strings of length 2 and complete enumeration of all possible alignments:



Def.

A sequence of length n+m, obtained by intercalating two sequences $S_1 = a_1a_2\ldots a_n$ and $S_2 = b_1b_2\ldots b_m$, while preserving the order of the symbols in S_1 and S_2, is called an intercalated sequence (denoted by $S_{1/2}$).

Def.

Two alignments are called redundant if their score is identical. The relationship of "having the same score" may be used to define equivalence classes of non-redundant alignments. For example, the class $a_1b_1b_2a_2$:

```
\    \_
 |_    |
```

$\rightarrow a_1b_1b_2a_2 \rightarrow \begin{array}{l} a_1-a_2 \\ b_1b_2- \end{array} ; \begin{array}{l} a_1a_2- \\ b_1-b_2 \end{array}$

Lemma.

There is one-to-one correspondence (bijection) between the set of non-redundant gapped alignments of two sequences S_1 and S_2 and the set of intercalated sequences $\{S_{1/2}\}$.

Corollary.

The number of non-redundant gapped alignments of two sequences, of length n and m, respectively, is equal to (n+m)!/[m!n!].

Proof.

Since the order of each of the sequences is preserved when intercalating them, we have in fact n+m positions to put m elements of the second sequence (once this is done the position of each of the elements of the first sequence is fixed unambiguously). Hence, the total number of intercalated sequences $S_{1/2}$ is given by the number of m-element combinations of n+m elements and the corollary is a simple consequence of the one-to-one correspondence between alignments and intercalated sequences stated in the lemma. QED

Ex.

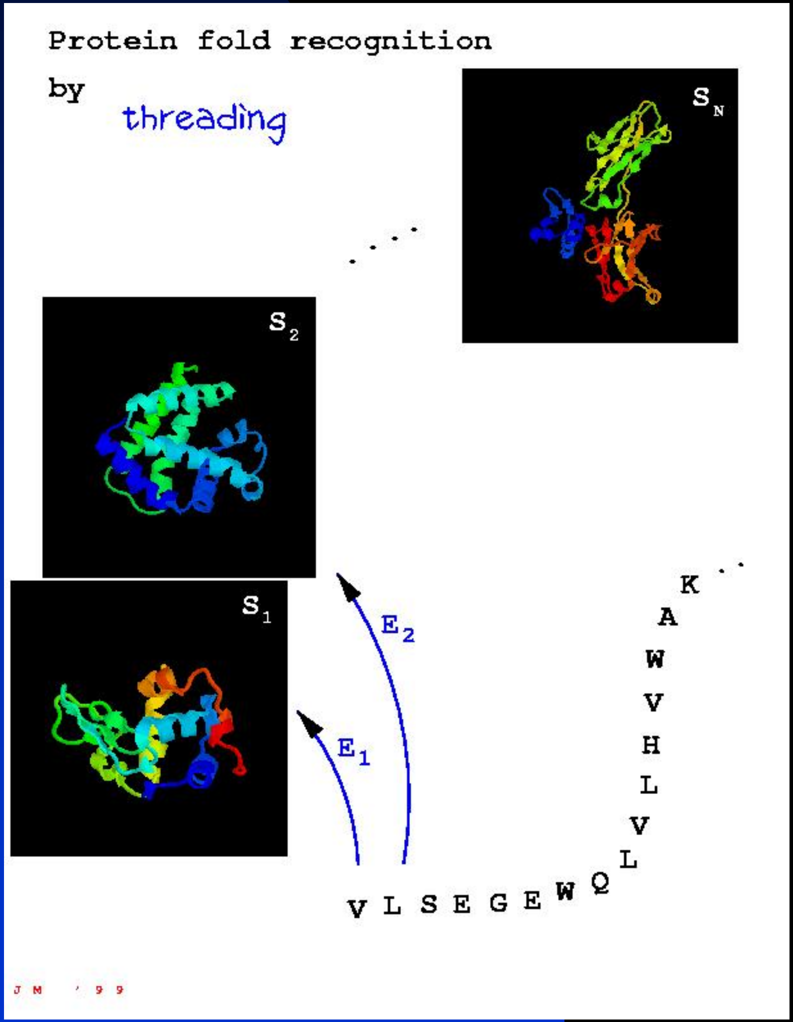
Consider for simplicity two sequences of the same length. Using the Stirling formula ($x! \sim (2\pi)^{1/2} x^{x+1/2} e^{-x}$) show that

$(n+n)!/[n!n!] \sim 2^{2n} / (2\pi n)^{1/2}$

Note that for a ridiculously small by biology standards length of 50 we already have about 10^{30} basic operations to perform an exhaustive search, making the naïve approach infeasible.

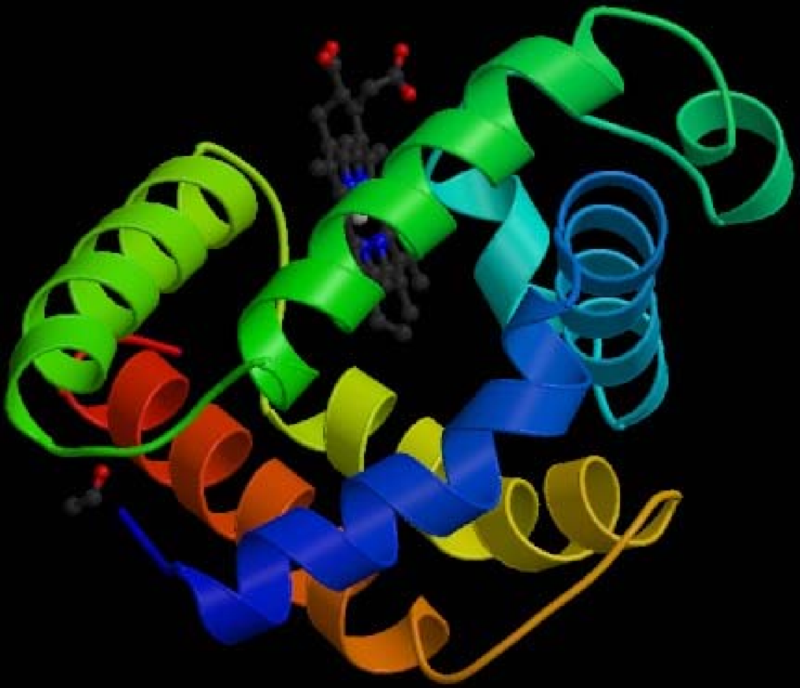
Dynamic programming provides in polynomial time an optimal solution for a class of potentially exponentially scaling problems. However, the traveling salesman (and other NP-complete problems) cannot be solved by dynamic programming because the cost of local decisions depends on the overall trajectory (or path on the DP graph). Pairwise contact potentials in sequence-to-structure matching result in the same problem.

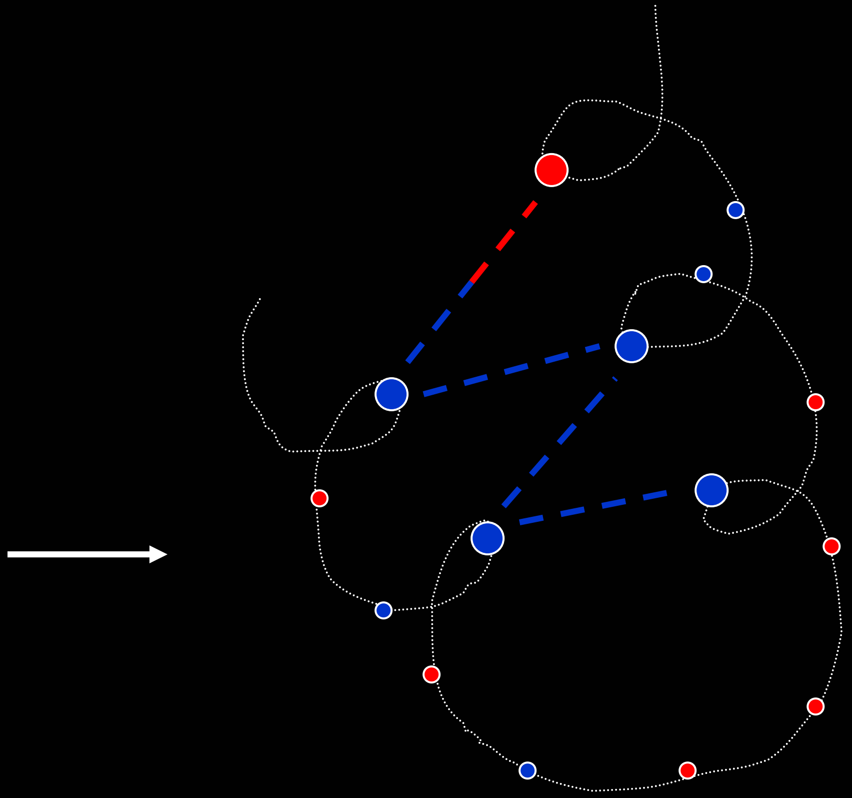
Assigning fold and function utilizing similarity to experimentally characterized proteins:



- Sequence similarity: BLAST and others
- Beyond sequence similarity: matching sequences and shapes (threading)

Reduced representation of protein structure





Each amino acid represented by a point in the 3D space; simple contact model – two amino acids in contact if their distance smaller than a cutoff.

- Because of the non-local character of scores due to a given structural site, finding an optimal alignment with gaps using pairwise models of the energy:

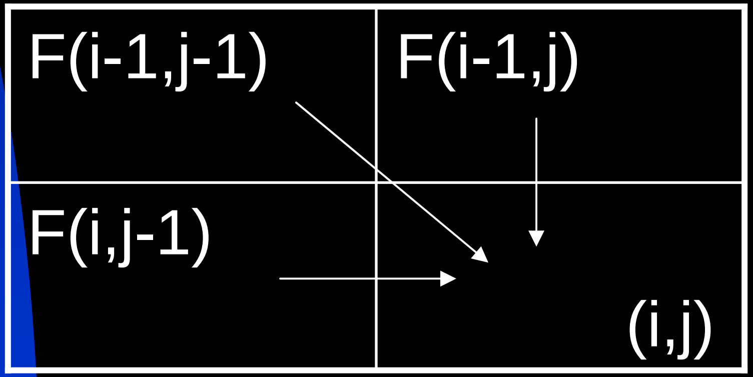
$$E = \sum_{k<l} \varepsilon_{kl} \, ,$$

  is NP-hard!

  R.H. Lathrop, Protein Eng. 7 (1994)

Why it works?

- The score is a sum of independent piecewise scores, in particular, the score up to a certain point is the best score up to the point one step before plus the incremental score of the new step.



a_j $-$ a_j

b_i ; b_i ; $-$

An argument instead of a proof.

We know already that all the possible alignments are included in the DP graph (each path defines one alignment, some of them redundant with respect to the score). What we need to consider now is if the path found using DP recurrence relations and tracback procedure is indeed optimal, i.e. it maximizes the score.

In case of global alignment each path starts at cell (0,0) and must end at cell (n,m). Consider the latter cell and the immediate past that led to it through one (most favorable together with the cost of the incremental step) of the 3 neighboring cells. Changing the last step (e.g. from initially chosen, optimal diagonal step) to an alternative one does not affect the scores at the preceding cells that represent the best trajectory up to this point. Clearly we get suboptimal solution if we assume that optimal solutions have been found in the previous steps. Hence, formalizing this argument we get proof by induction with reductio ad absurdum.

Examples (global alignment):

		H	E	A	G	A	W	G	H	E
	0	<-8	<-16	<-24	<-32	<-40	<-48	<-56	<-64	<-72
P	^-8	*-2	*-9	*-17	<-25	*-33	<-41	<-49	<-57	*-65
A	^-16	^-10	*-3	*-4	<-12	*-20	<-28	<-36	<-44	<-52
W	^-24	^-18	^-11	*-6	*-7	*-15	*-5	<-13	<-21	<-29
H	^-32	*-14	*-18	*-13	*-8	*-9	^-13	*-7	*-3	<-11
E	^-40	^-22	*-8	<-16	^-16	*-9	*-12	^-15	*-7	*3

```
HEAGAWGHE
--P-AW-HE
```

Examples (local alignment):

		H	E	A	G	A	W	G	H	E
	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0
A	0	0	0	*5	0	*5	0	0	0	0
W	0	0	0	0	*2	0	*20	<12	<4	0
H	0	*10	<2	0	0	0	^12	*18	*22	<14
E	0	^2	*16	<8	0	0	^4	^10	*18	*28

AWGHE

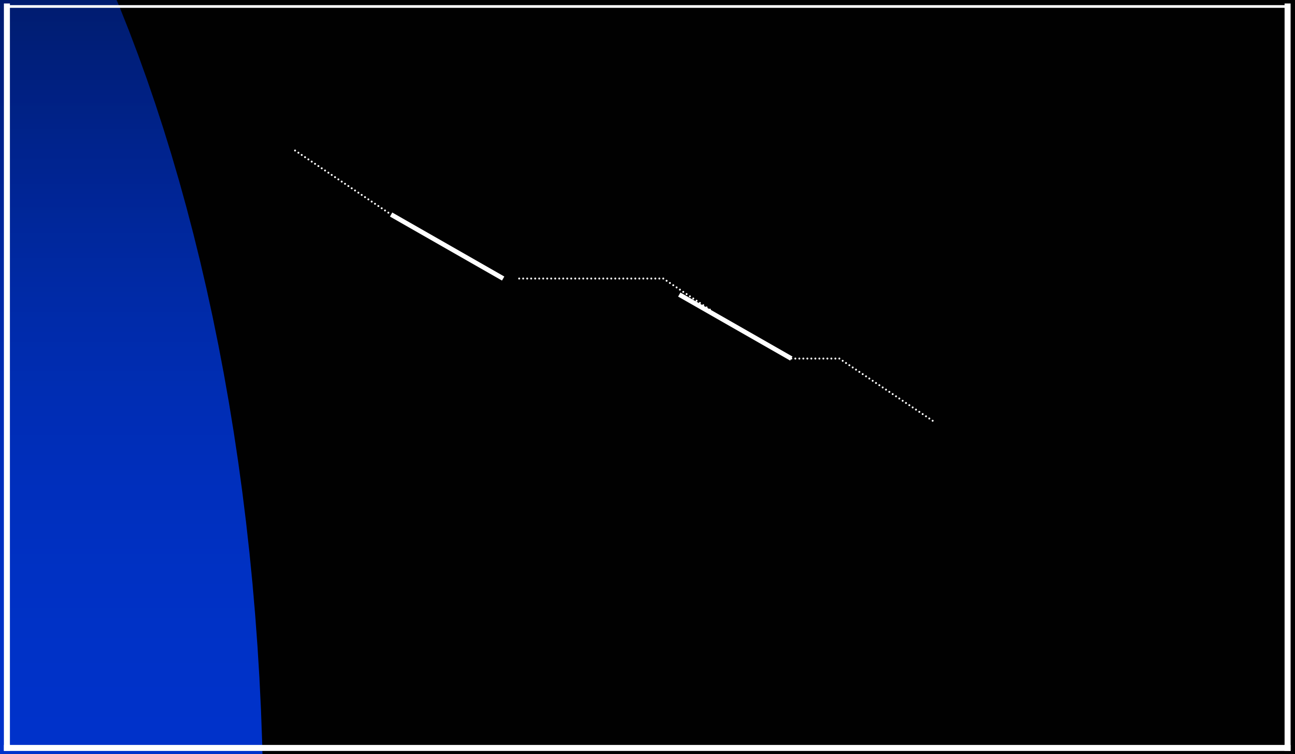
AW-HE

Reduction of complexity:

- Naïve search - ~ 2^{2n}
- DP search – $O(n^2)$
- Global optimization problem solved in polynomial time for specific (local or piecewise, pairwise scoring function).
- For large scale comparisons heuristic methods that find approximate solutions are used, e.g. BLAST.

Approximate, heuristic solutions may be nearly as good and much faster ...

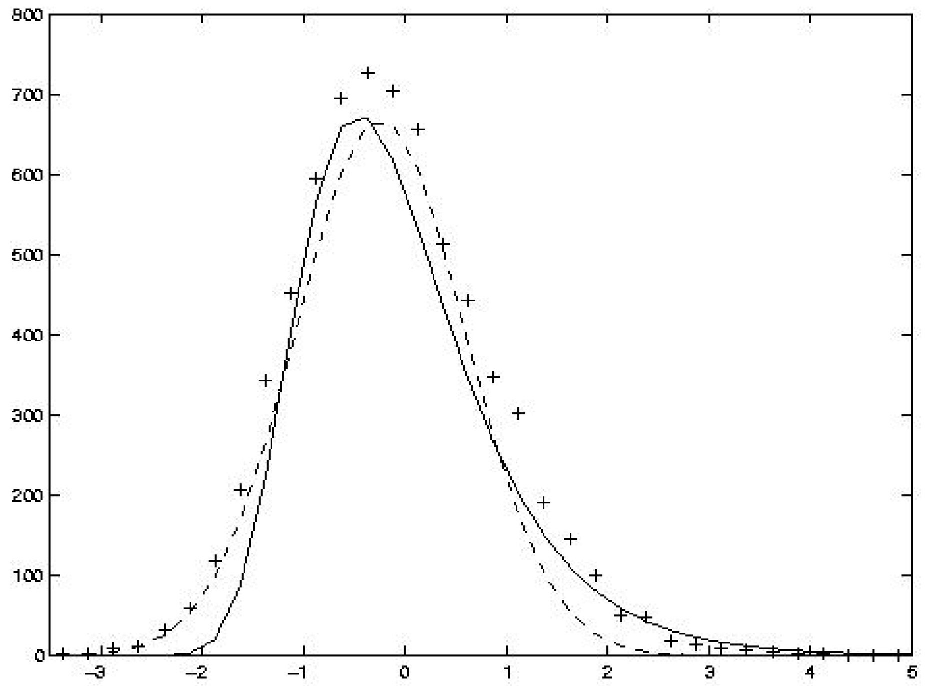
- BLAST approach: gapless seeds (High Scoring Pairs with well defined confidence measures), DP extensions:



Statistical verification of predictions:

- Scoring according to an energy may be insufficient (e.g. need to validate good matches due to similar length or composition).
- Z-score: a convenient measure of the strength of a match in terms of distribution of energies for random alignments.
- Such distribution is not normal !!

Distribution of Z-scores for false positives



$Z= -(E-<E>)/\sigma$

Dashed line - fit to a normal distribution

Solid line - fit to an extreme value distribution:

$p(Z)=\exp(-Z'-\exp(-Z'))/\sigma$

$Z'=(Z-a)/\sigma$

From analytical fit we get for example that $P(Z>4)=0.003$

Multiple alignment problem:

- DP search gives optimal solution scaling exponentially with the number of sequences K, $O(n^K)$, not practical for more than 3,4 sequences.
- Standard heuristics start from pairwise alignments (e.g. PsiBLAST, Clustalw)
- Hidden Markov Model approach to family profiles (profile HMM) as an alternative with pre-fixed parameters, trained separately for each family. Some initial multiple alignments necessary for training.

Summary:

- Problem of finding an optimal alignment of two sequences results in a global optimization problem that is solved by the dynamic programming algorithm in polynomial time for specific (local) scoring function and piecewise decomposable problem.
- For large scale comparisons heuristic methods that find approximate solutions are used, e.g. BLAST.

DP is GREAT !!

From a simple example to probabilistic linguistic models: a very brief introduction to Hidden Markov Models

From speech recognition
to gene and protein families recognition.

HMM’s and their applications:

- Problems with grammatical structure, such as gene finding, family profiles and protein function prediction, transmembrane protein fragments prediction
- In general, one may think of different biases in different fragments of the sequence (due to functional role for example) or of different states emitting these fragments using different probability distributions

Probabilistic models of biological sequences:

For any probabilistic model the total probability of observing a sequence $a_1a_2\ldots a_n$ may be written as:

$$P(a_1a_2\ldots a_n) = P(a_n | a_{n-1}\ldots a_1)\, P(a_{n-1} | a_{n-2}\ldots a_1) \ldots P(a_1)$$

In Markov chain models we simply have:

$$P(a_1a_2\ldots a_n) = P(a_n | a_{n-1})\, P(a_{n-1} | a_{n-2}) \ldots P(a_1)$$

HMM’s are different from Markov chain models since some “hidden” states that “emit” sequence symbols are introduced.

Probabilistic models of biological sequences:

HMM’s may be in fact regarded as probabilistic, finite automata that generate certain “languages”: sets of words (sentences etc.) with specific “grammatical” strcuture.

For example, promotor, start, exon, splice junction, intron, stop “states” will appear in a linguistic model of a gene, whereas column (sequence position), insert and deletion states will be employed in a linguistic model of a (protein) family profile.

Biologically relevant example: simple profile HMM.

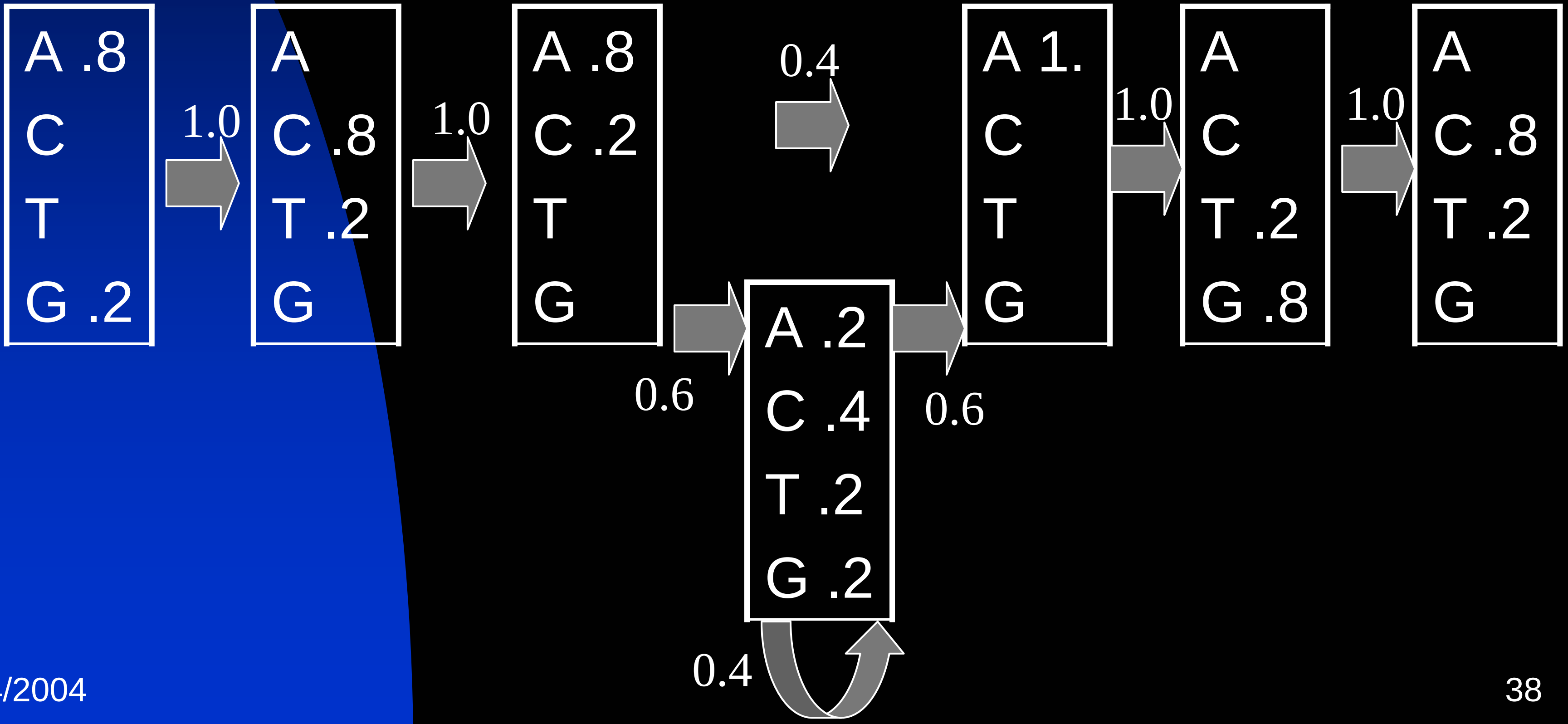
```
ACA---ATG
TCAACTATC
ACAC--AGC
AGA---ATC
ACCG--ATC
```

- Grep approach to motif search: find the following regular expression: [AT][CG][AC][ACGT]*A[TG][GC]
- Importance of finite automata in string parsing and searching.

Biologically relevant example: simple profile HMM.

P(ACACATC)=.8*1*.8*1*.8*.6*.4*.6*1*1*.8*1*.8=0.047

Probability in random model Q(ACACATC)=$(1/4)^7$ Score=log(P/Q)



Profile (and other) HMM method(s):

- Given a training set of aligned sequences maximize probability of observing the training sequences, choosing appropriate transition and emission probabilities – Expectation Maximization algorithm
- In recognition phase, having the optimized probabilities, we ask what is the likelihood that a new sequence belongs to a family i.e. it is generated by the HMM with sufficiently high probability. The Viterbi algorithm, which is fact another incarnation of dynamic programming in a suitable formulation, is used to find an optimal path through the states, which defines in this case alignment