Dynamic Programming
Scoring matrices
BLAST

Section 3

Griffin Weber

Oct. 7th, 2003

Outline

- Recursion
- More Dynamic Programming
- Scoring Matrices
- BLAST

Recursion

General Idea: Solve a problem by solving related sub-problems and combining the results.

Advantage: Sub-problems are easier to solve and their total run time can be much less than original problem.

Example: Merge Sort – To sort a large list, split the list in half, sort each half, and merge the two sorted lists.

Example: Factorial(x)

x! = x*(x-1)*(x-2)*…*1

5! = 5*4*3*2*1 = 120

Recursion - Factorial

Iterative Algorithm:

```
$x = 5;
for ($n = $x-1 ; $n > 0 ; $n--) {
   $x *= $n;
}
```

Recursive Algorithm:

```
$x = factorial(5);
sub factorial {
     my($n) = $_[0];
     if ($n <= 1) {
          return 1;
     } else {
          return $n*factorial($n-1);
     }
}
```

base case

recursive case

Dynamic Programming

General Idea: Solve a problem by solving related sub-problems and combining the results.

Key Point: In recursive algorithms (e.g., Merge Sort), the sub-problems are independent. In dynamic programming the sub-problems share sub-sub-problems.

Implementation: Solve each sub-sub-problem only once and store the answers in an array.

Dynamic Programming – Step 1

Characterize the structure of the optimal solution.

Example: Sequence Alignment

Three choices at each step:
1. Align a base in X with a base in Y
2. Align a base in X with a gap
3. Align a base in Y with a gap

Goal:
Maximize the total score

Dynamic Programming – Step 2

Recursively define the value of an optimal solution.

Example: Needleman-Wunsch (Global Alignment)

$$F(i,j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) , \\ F(i-1, j) - d , \\ F(i, j-1) - d \end{cases}$$

Dynamic Programming – Step 3

Compute the optimal solution, starting from the easiest sub-sub-problems and saving the results.

	-	A	C	A	C	T
-	0	-3	-6			
A	-3	2				
G	-6					
C						
A						
C						

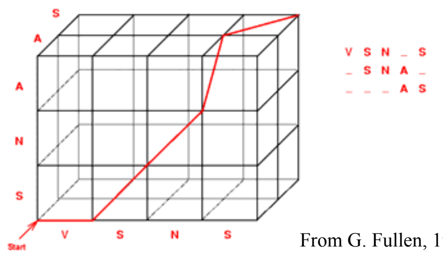
Dynamic Programming – Step 4

Construct an optimal solution from the computed information. (Perform a trace-back.)

	-	A	C	A	C	T
-	0	-3	-6	-9	-12	-15
A	-3	2		-4	-7	-10
G	-6	-1	1		-5	-8
C	-9	-4	1	0		-3
A	-12	-7	-2	3	0	
C	-15	-10	-5	0	5	2

Multiple Sequence Alignment

Dynamic programming in many dimensions!!!



From G. Fullen, 1996.

Dynamic Programming Summary

Advantage: Guaranteed to find the mathematically optimal solution (highest scoring alignment). Much faster than trying all possible alignments.

Disadvantage: For large problems (long sequences or many sequences), dynamic programming can be very slow and require lots of computer memory.

Scoring Matrices

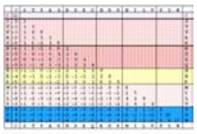
Definition: A table (matrix) of values that describe the probability of a residue (amino acid or base) pair occurring in an alignment of related sequences.

Why a matrix? Evolution tends to favor different sets of substitutions. Thus, a single probability does not accurately describe all residue pairs

Example: PAM

PAM = Percent Accepted Mutation

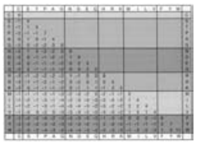
- Ranks a.a. substitutions on how well they're "accepted" by natural selection, without changing the function of the protein.
- Derived from global alignments of closely related sequences (85% identical).
- PAM1 models the mutation rate in 1 "PAM Unit" of evolutionary time.
- PAMn = $(\text{PAM1})^n$ = mutation rate in n PAM Units
- Examples: PAM40, PAM100



Example: BLOSUM

BLOSUM = Blocks Amino Acid Substitution Matrices

- Derived from large data sets of local, ungapped alignments of biochemically related sequences.
- Each matrix is derived from different data sets.
- BLOSUMn means no pair of sequences in the data set used to derive the matrix had more than n% similarity.
- BLOSUM62 indicates the substitution frequencies for sequences that are up to 62% similar



PAM vs BLOSUM

- BLOSUM better than PAM for local similarity
- BLOSUM better in similarity searches in databases
- PAM incorporates an evolutionary model representing species divergence
- BLOSUM tolerant of hydrophobic changes and of cystein and tryptophan mismatches
- PAM tolerant of change to/from hydrophilic a.a.

BLOSUM 80	**BLOSUM 62**	**BLOSUM 45**
PAM 1	**PAM 120**	**PAM 250**
Less Divergent	$\longleftrightarrow$	**More Divergent**

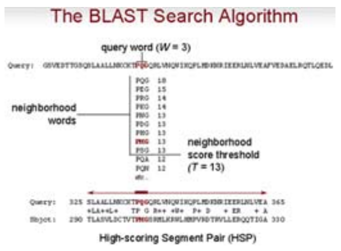
BLAST

BLAST = Basic Local Alignment Search Tool

- At least 50 times faster than dynamic programming
- Not guaranteed to find the optimal solution, but…
  - Individual alignments need not be perfect, you can always fine-tune later.
  - Most sequences will be completely unrelated to the query.
  - Will always find related sequences in a database that meet some similarity criteria.
  - BLAST is a heuristic algorithm (usually works well)
- Always translate DNA into protein sequences (if possible) before running BLAST.

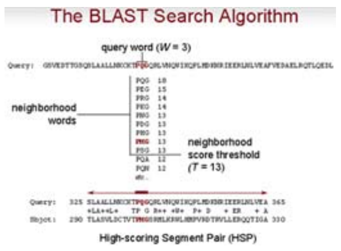
BLAST

Assumption: True alignments are very likely to contain a short stretch of identities, or very high scoring matches.



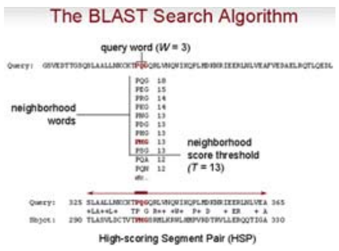
BLAST : K-mer Words

- Create a list of all "words" of length K (default is 3 amino acids or 11 nucleotides)
- Example: TPQGQR → TPQ, PQG, QGQ, GQR



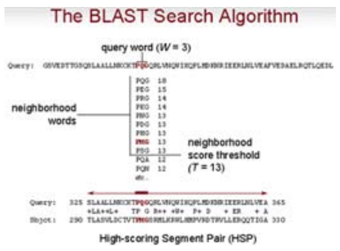
BLAST : K-mer Hashing

- For each k-mer, list all locations where it occurs in the database.
- Match the query to k-mers (or “neighborhood words” above a threshold) in the list. This forms seeds.


The BLAST Search Algorithm
query word (W = 3)
neighborhood words
neighborhood score threshold (T = 13)
High-scoring Segment Pair (HSP)

BLAST : Extending Seeds

- Connect “close” seeds into single long sequences.
- Extend seeds until the cumulative score drops.
- High-scoring Segment Pair = locally maximal segment.


The BLAST Search Algorithm
query word (W = 3)
neighborhood words
neighborhood score threshold (T = 13)
High-scoring Segment Pair (HSP)

BLAST : Evaluating HSPs

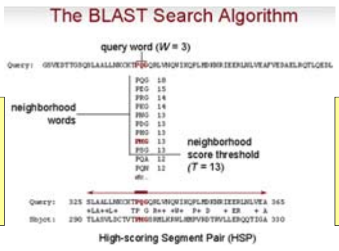
- Bit Score: independent of the scoring matrix used.
- E-value: “Expected number of High-Scoring Segment Pairs”. The smaller the number, the better the alignment.
- E-values of 0.1 or 0.05 are typically used as cutoffs.

Good

High Bit Score

=

Low E-value


The BLAST Search Algorithm
query word (W = 3)
neighborhood words
neighborhood score threshold (T = 13)
High-scoring Segment Pair (HSP)

Bad

Low Bit Score

=

High E-value

Next Week

- Microarrays
- Sequencing

Acknowledgement / References

This handout includes material written by Suzanne Komili, Yonatan Grad, Doug Selinger, and Zhou Zhu.

Mount, Bioinformatics – Sequence and Genome Analysis; Durbin et al., Biological Sequence Analysis; Cormen et al., Introduction to Algorithms

http://www.ncbi.nlm.nih.gov/BLAST