BLAST and FASTA Heuristics in pairwise sequence alignment

(based on materials of Christoph Dieterich
Department of Evolutionary Biology
Max Planck Institute for Developmental Biology)

Heuristics for large-scale database searching

- Pairwise alignment is used to detect homologies between different protein or DNA sequences, e.g. as global or local alignments.
- Problem solved using dynamic programming in $O(nm)$ time and $O(n)$ space.
- This is **too slow** for searching current databases.
- In practice algorithms are used that run much faster, at the expense of possibly missing some significant hits due to the heuristics employed.
- Such algorithms are usually *seed-and-extend* approaches in which first small exact matches are found, which are then extended to obtain long inexact ones.

Preprocessing

- Preprocessing should save time for subsequent searches, but the databases are changing – they are split into fixed and a dynamic part. Fixed part is preprocessed and the results of the preprocessing is stored in aproriate structures, e.g. hash tables.
- Information about substrings of length n can be stored in a hash table. For an alphabet Σ there are $|\Sigma|^n$ different substrings of length n.
- We will describe two mathods
  - BLAST
  - FASTA

BLAST

- BLAST, the Basic Local Alignment Search Tool (Altschul et al., 1990), is an alignment heuristic that determines “local alignments” between a query and a database. It is based on Smith-Waterman algorithm.
- BLAST consists of two components:
  - a search algorithm and
  - a computation of the statistical significance of solutions.

BLAST

- Let q be the query and d the database. A *segment* is simply a substring s of q or d.
- A segment-pair (s, t) (or hit) consists of two segments, one in q and one d, of the same length.

  Example:

  ```
  V A L L A R
  P A M M A R
  ```

- We think of s and t as being aligned without gaps and score this alignment using a substitution score matrix, e.g. BLOSUM or PAM in the case of protein sequences.
- The alignment score for (s, t) is denoted by $\sigma(s, t)$.

BLAST

- A *locally maximal segment pair (LMSP)* is any segment pair (s,t) whose score cannot be improved by shortening or extending the segment pair.
- A *maximum segment pair (MSP)* is any segment pair (s, t) of maximal alignment score $\sigma(s, t)$.
- Given a cutoff score S, a segment pair (s, t) is called a *high-scoring segment pair (HSP)*, if it is locally maximal and $\sigma(s, t) \geq S$.
- Finally, a *word* is simply a short substring of fixed length w.

BLAST – goal

- **Goal**: Find all HSPs for a given cut-off score.

- Three parameters:
  - a word size w,
  - a word similarity threshold T and
  - a minimum cut-off score S.
- We are looking for a segment pair with a score of at least S that contains at least one word pair of length w with score at least T.

BLAST: Preprocessing

1. For the query q, generate all subwords of length w.
2. Generate a list of all w-mers of length w over the alphabet Σ that have similarity > T to some subword in the query sequence q.

Example: For the query sequence RQCSAGW the list of words of length $w = 2$ with a score $T > 8$ using the BLOSUM62 matrix are:

word	2-mer with score > 8
RQ	RQ
QC	QC, RC, EC, NC, DC, KC, MC, SC
CS	CS,CA,CN,CD,CQ,CE,CG,CK,CT
SA	-
AG	AG
GW	GW,AW,RW,NW,DW,QW,EW,HW,KW,PW,SW,TW,WW

BLAST: Searching

- **Localization** of the hits: The database sequence d is scanned for all hits t of w-mers s in the list, and the position of the hit is saved.
- **Detection** of hits: First all pairs of hits are searched that have a distance of at most A (think of them lying on the same diagonal in the matrix of the SW-algorithm).
- **Extension** to HSPs: Each such seed (s, t) is extended in both directions until its score $\sigma(s, t)$ cannot be enlarged (LMSP). Then all best extensions are reported that have score $\geq S$, these are the HSPs.

- In practice, $w = 3$ and $A = 40$ for proteins.
- Originally the extension did not include gaps, a newer BLAST2 algorithm allows insertion of gaps.

BLAST: Searching

- The list L of all words of length w that have similarity $> T$ to some word in the query sequence q can be produced in $O(|L|)$ time.
- These are placed in a “keyword tree” and then, for each word in the tree, all exact locations of the word in the database d are detected in time linear to the length of d.
- As an alternative to storing the words in a tree, a finite-state machine can be used.

BLAST: Extension

- As BLAST does not allow indels at that stage, hit extension is very fast.
- Use of seeds of length w and the termination of extensions with fading scores (score dropoff threshold X) are both steps that speed up the algorithm, but also imply that BLAST is not guaranteed to find all HSPs (after all it is a heuristic).

- Recent improvements (BLAST 2.0):
  - Two word hits must be found within a window of A residues.
  - Explicit treatment of gaps.
  - Position-specific iterative BLAST (PSI-BLAST).

BLAST for DNA

For DNA sequences, BLAST operates as follows:

- The list of all words of length w in the query sequence q is generated. In practice, $w = 12$ for DNA.
- The database d is scanned for all hits of words in this list.
- Blast uses a two-bit encoding for DNA. This saves space and also search time, as four bases are encoded per byte.
- Note that the "T" parameter dictates the speed and sensitivity of the search.

Different versions of BLAST

- **BLASTN**: compares a DNA query sequence to a DNA sequence database;
- **BLASTP**: compares a protein query sequence to a protein sequence database;
- **TBLASTN**: compares a protein query sequence to a DNA sequence database (6 frames translation);
- **BLASTX**: compares a DNA query sequence (6 frames translation) to a protein sequence database.
- **TBLASTX**: compares a DNA query sequence (6 frames translation) to a DNA sequence database (6 frames translation).

BLAST – statistical analysis

- **Problem:**

  Given an HSP (s, t) with score $\sigma(s, t)$. How significant is this match (i.e., local alignment)?

- For a given HSP (s, t) the raw score S is transformed into a bit-score.
- Such bit-scores can be compared between different BLAST searches, as the parameters of the given scoring systems are subsumed in them.
- For each match an expectation values E is computed (it is printed in scientific notation – an exponent only) the smaller the number, i.e. the closer it is to 0, the more significant the match is. Expectation values show us how often we could expect that particular alignment match to occur merely by chance alone in a search of that size database.

FASTA

- FASTA (pronounced fast-ay) is a heuristic for finding significant matches between a query string q and a database string d. It is the older of the two heuristics introduced in the lecture.
- FASTA's general strategy is to find the most significant diagonals in the dot-plot or dynamic programming matrix.
- The algorithm consists of four phases:

  Phase 1: Hashing,

  Phase 2: 1st scoring,

  Phase 3: 2nd scoring,

  Phase 4: alignment.

FASTA Phase 1: hashing

- The first step of the algorithm is to determine all exact matches of length k (word-size) between the two sequences, called hot-spots.
- A hot-spot is given by (i, j), where i and j are the locations (i.e., start positions) of an exact match of length k in the query and database sequence respectively.
- Any such hot-spot (i, j) lies on the diagonal $(i - j)$ of the dot-plot or dynamic programming matrix.
- Using this scheme, the main diagonal has number 0 ($i = j$), whereas diagonals above the main one have positive numbers ($i < j$), the ones below negative ($i < j$).
- A diagonal run is a set of hot-spots that lie in a consecutive sequence on the same diagonal. It corresponds to a gapless local alignment.
- A score is assigned to each diagonal run. This is done by giving a positive score to each match (using e.g. the PAM250 match score matrix in the case of proteins) and a negative score for gaps in the run, the latter scores decrease with increasing length of the gap between hot-spots.
- The algorithm then locates the ten best diagonal runs.

FASTA Phase 2+3: scoring

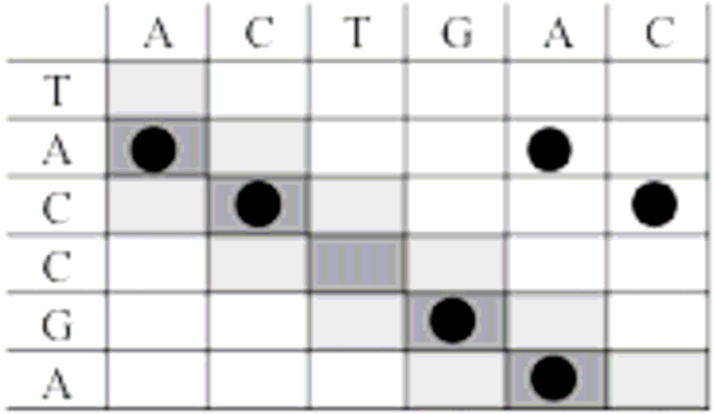
- Each of the ten diagonal runs with highest score are further processed. Within each of these scores an optimal local alignment is computed using the match score substitution matrix. These alignments are called initial regions.
- The score of the best sub-alignment found in this phase is reported as **init1**.
- The next step is to combine high scoring sub-alignments into a single larger alignment, allowing the introduction of gaps into the alignment. The score of this alignment is reported as **initn.**

FASTA Phase 4: alignment

- Finally, a banded Smith-Waterman dynamic program is used to produce an optimal local alignment along the best matched regions. The center of the band is determined by the region with the score init1, and the band has width 8. The score of the resulting alignment is reported as **opt**.
- In this way, FASTA determines a highest scoring region, not all high scoring alignments between two sequences. Hence, FASTA may miss instances of repeats or multiple domains shared by two proteins.
- After all sequences of the databases have thus been searched a statistical significance similar to the BLAST statistics is computed and reported.

FASTA example

- Two sequences ACTGAC and TACCGA: The hot spots for $k = 2$ are marked as pairs of black bullets, a diagonal run is shaded in dark grey. An optimal sub-alignment in this case coincides with the diagonal run. The light grey shaded band of width 3 around the sub-alignment denotes the area in which the optimal local alignment is searched.



Comparing BLAST and FASTA

- BLAST: individual seeds are found and then extended without indels.
- FASTA: individual seeds contained in the same diagonal are merged and the resulting segments are then connected using a banded Smith-Waterman alignment.