Alignment Algorithms & Blast

Oct 6, 2008
BENG 182 Biological Data Analysis
Winnie Yi Wen
Sequence Alignment Algorithm
**What is a sequence alignment?**

The process of lining up two or more sequences to achieve maximal levels of identity (and conservation, in the case of amino acid sequences) for the purpose of assessing the degree of similarity and the possibility of homology.

- to find similarity between unknowns
- to find difference between knowns
The biology behind sequence alignment

* Chromosomal recombination
* DNA damage and repair
* Insertion, deletion, point mutation, duplication,

```
<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>T</th>
<th>-</th>
<th>G</th>
<th>T</th>
<th>T</th>
<th>A</th>
<th>T</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>T</td>
<td>C</td>
<td>G</td>
<td>T</td>
<td>-</td>
<td>A</td>
<td>-</td>
<td>C</td>
<td></td>
</tr>
</tbody>
</table>
```

Insertion   deletion
Edit distance

* minimum number of editing operations (insertion, deletion, and substitution) needed to transform one string into another
* allows one to compare strings of different length

![Diagram showing edit operations from TGCATAT to ATCCGAT](image)

**Figure 6.11** Five edit operations can take TGCATAT into ATCCGAT.

**Figure 6.12** Four edit operations can also take TGCATAT into ATCCGAT.
Longest common subsequences (LCS)

Longest Common Subsequence Problem:
Find the longest subsequence common to two strings.

- **Input:** Two strings, v and w.
- **Output:** The longest common subsequence of v and w.

- simplest form of a sequence similarity analysis
- eliminates the operation of substitution; only allows deletion and insertion
V and W have a subsequence TCTA in common

Computing similarity \( s(V,W) = 4 \)

Computing distance \( d(V,W) = 5 \)

V can be transformed into W by deleting A,G,T and inserting G,A

Alignment:

\[
\begin{align*}
\text{A} & - \text{T} - \text{C} - \text{T} & \text{G} & \text{A} & \text{T} \\
\text{T} & - \text{G} & \text{C} & \text{A} & \text{T} - \text{A} \\
\end{align*}
\]
Dynamic programming (DP)

- The problem is broken into subproblems, and these subproblems are solved and the solutions remembered, in case they need to be solved again.
- Break the problem into smaller subproblems.
- Solve these problems optimally using this three-step process recursively.
- Use these optimal solutions to construct an optimal solution for the original problem.
Back to LCS

LCS(v, w)
1 for i ← 0 to n
2 \( s_{i,0} \leftarrow 0 \)
3 for j ← 1 to m
4 \( s_{0,j} \leftarrow 0 \)
5 for i ← 1 to n
6 for j ← 1 to m
7 \( s_{i,j} \leftarrow \max \left\{ \begin{array}{ll}
    s_{i-1,j} & \text{if } v_i = w_j \\
    s_{i,j-1} & \text{if } s_{i,j} = s_{i-1,j} \\
    s_{i-1,j-1} + 1, & \text{if } s_{i,j} = s_{i-1,j-1} + 1
\end{array} \right. \)
8 \( b_{i,j} \leftarrow \left\{ \begin{array}{ll}
    "\mid" & \text{if } s_{i,j} = s_{i-1,j} \\
    "\leftarrow" & \text{if } s_{i,j} = s_{i,j-1} \\
    "\triangleleft" & \text{if } s_{i,j} = s_{i-1,j-1} + 1
\end{array} \right. \)
9 return \((s_{n,m}, b)\)
(Global) Sequence alignment

Global Alignment Problem:
Find the best alignment between two strings under a given scoring matrix.

Input: Strings v, w and a scoring matrix δ.

Output: An alignment of v and w whose score (as defined by the matrix δ) is maximal among all possible alignments of v and w.

\[
s_{i,j} = \max \begin{cases} 
  s_{i-1,j} - \sigma \\
  s_{i,j-1} - \sigma \\
  s_{i-1,j-1} - \mu, \text{ if } v_i \neq w_j \\
  s_{i-1,j-1} + 1, \text{ if } v_i = w_j 
\end{cases}
\]
Local sequence alignment

Local Alignment Problem:
Find the best local alignment between two strings.

**Input:** Strings v and w and a scoring matrix δ.

**Output:** Substrings of v and w whose global alignment, as defined by δ, is maximal among all global alignments of all substrings of v and w.

\[
s_{i,j} = \max \begin{cases} 
0 & \\
 s_{i-1,j} + \delta(v_i,-) & \\
 s_{i,j-1} + \delta(-,w_j) & \\
 s_{i-1,j-1} + \delta(v_i,w_j) & 
\end{cases}
\]
Useful links:

* Wikipedia
  
    http://en.wikipedia.org/wiki/Sequence_alignment

* An introduction to bioinformatics algorithms
  
    http://bix.ucsd.edu/bioalgorithms/
Blast
What is Blast?

* Basic Local Alignment Search Tool

* blastp: p query p db (p)
  blastn: n query n db (n)
  blastx: n query, 6 reading frames, p db (p)
  tblastn: p query, n db 6 reading frames (p)
  tblastx: n query 6 reading frames, n db 6 reading frames (p)
Terms and biology

- **Similarity**: The extent to which nucleotide or protein sequences are related. The extent of similarity between two sequences can be based on percent sequence identity and/or conservation. In BLAST similarity refers to a positive matrix score.

- **Homology**: Similarity attributed to descent from a common ancestor.

![Diagram showing homologs, orthologs, paralogs, and gene duplication.](image)
Conservation: Changes at a specific position of an amino acid or (less commonly, DNA) sequence that preserve the physico-chemical properties of the original residue.
The input
### Algorithm parameters

#### General Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max target sequences</td>
<td>100</td>
</tr>
<tr>
<td>Short queries</td>
<td>✔</td>
</tr>
<tr>
<td>Expect threshold</td>
<td>10</td>
</tr>
<tr>
<td>Word size</td>
<td>3</td>
</tr>
</tbody>
</table>

#### Scoring Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Matrix</td>
<td>BLOSUM62</td>
</tr>
<tr>
<td>Gap Costs</td>
<td>Existence: 11 Extension: 1</td>
</tr>
<tr>
<td>Compositional adjustments</td>
<td>Conditional compositional score matrix adjustment</td>
</tr>
</tbody>
</table>

Assigns a score for aligning pairs of residues, and determines overall alignment score. [more...](#)

#### Filters and Masking

<table>
<thead>
<tr>
<th>Filter</th>
<th>Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low complexity regions</td>
<td>Mask for lookup table only</td>
</tr>
<tr>
<td>Mask lower case letters</td>
<td></td>
</tr>
</tbody>
</table>

Search database nr using Blastp (protein-protein BLAST)
The output
Job Title: lci7295 (508 letters)

Please, try our new design!

BLASTP 2.2.18+

Reference:

Reference for compositional score matrix adjustment:

RID: EGY4UVV6016

Database: All non-redundant GenBank CDS translations+PDB+SwissProt+PIR+PRF excluding environmental samples from WGS projects
7,103,639 sequences; 2,451,291,059 total letters

If you have any problems or questions with the results of this search please refer to the BLAST FAQs
Taxonomy reports

Query:
Length=508
Distribution of 100 Blast Hits on the Query Sequence

Mouse-over to show define and scores, click to show alignments

Color key for alignment scores

Query:
- <40
- 40-60
- 50-80
- 80-200
- >=200

Query sequence:

[Graph showing alignment scores distribution]
Score: The score of an alignment, $S$, calculated as the sum of substitution and gap scores. Substitution scores are given by a look-up table. Gap scores are typically calculated as the sum of $G$, the gap opening penalty and $L$, the gap extension penalty. For a gap of length $n$, the gap cost would be $G+Ln$. The choice of gap costs, $G$ and $L$ is empirical, but it is customary to choose a high value for $G$ (10-15) and a low value for $L$ (1-2).
Identities: The extent to which two (nucleotide or amino acid) sequences are invariant.

Expect: Expectation value. The number of different alignments with scores equivalent to or better than S that are expected to occur in a database search by chance. The lower the E value, the more significant the score.
gap: number of -
positive: number of identities and plus signs (aligned with positive scores)
Database: All non-redundant GenBank CDS translations+PDB+SwissProt+PIR+PRF excluding environmental samples from WGS projects
Posted date: Oct 2, 2008 5:56 PM
Number of letters in database: 1,843,676,233
Number of sequences in database: 7,103,639

<table>
<thead>
<tr>
<th>Lambda</th>
<th>K</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.309</td>
<td>0.125</td>
<td>0.368</td>
</tr>
</tbody>
</table>

Gapped

<table>
<thead>
<tr>
<th>Lambda</th>
<th>K</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.267</td>
<td>0.0410</td>
<td>0.140</td>
</tr>
</tbody>
</table>

Matrix: BLOSUM62
Gap Penalties: Existence: 11, Extension: 1
Number of Sequences: 7103639
Number of Hits to DB: 281065607
Number of extensions: 14839283
Number of successful extensions: 421715
Number of sequences better than 100: 12295
Number of HSPs better than 100 without gapping: 0
Number of HSPs gapped: 352539
Number of HSPs successfully gapped: 38279
Length of query: 508
Length of database: 2451291059
Length adjustment: 139
Effective length of query: 369
Effective length of database: 1463885238
Effective search space: 540173652822
Effective search space used: 540173652822
T: 11
A: 40
X1: 16 (7.1 bits)
X2: 30 (14.6 bits)
X3: 64 (24.7 bits)
S1: 42 (20.8 bits)
S2: 72 (32.3 bits)
Rules of thumb

- Matches that are more than 50% identical in a 20-40 amino acid region occur frequently by chance.

- Low complexity regions, transmembrane regions and coiled-coil regions frequently display significant similarity in the absence of homology. Low complexity regions can be filtered out using the default parameters of BLAST. Transmembrane and coiled-coil regions should be identified and masked (by eliminating these regions from the query) by the user.
Useful links

* NCBI Blast information