CS 3220 Final Project

Pairwise Sequence Alignment via Dynamic Programming

Allan Stewart
88121
CS 3220
Yingwei Wang
12/7/2017

Allan Stewart

CS 3220 Final Project

Pairwise Sequence Alignment via Dynamic Programming

Introduction

Pairwise sequence alignment is important not only for finding optimal global and local alignments when considering a pair of sequences, but also as part of more complicated algorithms seeking to align multiple sequences. One prominent and effective, if computationally expensive technique, is dynamic programming. Dynamic programming is of particular importance when the optimal alignment is desired, particularly in global alignment. Here we will consider an implementation of the foundational global and local alignment algorithms, the Needleman–Wunsch and Smith–Waterman algorithms, named BEJAT, standing for Basic Example Java Alignment Tool. Several example outputs will also be considered as demonstrations of the algorithms and their implementations as part of BEJAT.

Background

There are several techniques used to compare and align pairs of sequences. They range from the more qualitative, such as dot-matrix methods, to the exact, such as dynamic programming, to the less computationally expensive heuristic methods that are widely applied. Dot-matrix methods all involve variations on recurrence plots, where each dot in the diagram corresponds in some way to a correspondence between the two sequences being considered, such as exact matches, or in some versions, accepted mutations as well. It is also possible to use only one sequence, in which case the the diagram shows self correlations, such as repetitions. These methods are qualitative and simple, and are limited by being time consuming, prone to noise, and difficult to analyse. They also do not generalize to more than two sequences. They are however good for visualizing insertions, deletions, repetitions, and inversions, when noise is low. Dot-matrix plots were considered as a possible output for this implementation, but a satisfactory version was not completed in time to be included in the final version.

Most of the more widely used methods for larger sequences, and for multiple sequence alignment, are more heuristic. While these techniques are not guaranteed to be optimal, they are particularly good for local alignments and large scale database searches, where it is expected that only a small portion of the given sequences will align. This is because they find regions of similarity first, and then focus on aligning that region, saving the time that would have been taken aligning regions of little similarity. Two of the most well known implementations of such methods are EBI's FASTA, and NCBI's BLAST. These two implementations are similar, both based around fixed length word comparison. However BLAST is able to improve on the performance of FASTA by only evaluating the most significant word matches, as opposed to all matches, though this does sacrifice some accuracy.

Dynamic programming, also called dynamic optimization, is a common computational technique applied in many fields. At its core, dynamic programming is functionally a method that breaks a larger problem into a series of simpler sub-problems, in a possibly recursive fashion. The two foundational algorithms for paired sequence alignment are the Needleman–Wunsch algorithm for global alignment, and the Smith–Waterman algorithm for local alignment.

The Needleman–Wunsch algorithm, first published in 1970, was developed by Saul B. Needleman and Christian D. Wunsch and is still widely used for optimal global alignment, particularly when the quality of the global alignment is of the primary concern. It is however computationally expensive, having a space complexity of O(mn), and a time complexity of up to O(m^2n) with generalized gap penalties. There are however variations on this algorithm that are able to offer better performance, such as Hirschberg's algorithm which reduces the space complexity to Θ(min(m,n)), and approximations, such as bounded dynamic programming, which improve performance by restricting the searched parameter space.

The Smith–Waterman algorithm, first published in 1981, was developed by Temple F. Smith and Michael S. Waterman. It differs primarily from the Needleman–Wunsch algorithm by disallowing negative scores, thus focusing on the positively scored local alignments. As it similarly complex in both space and time, and often only a small portion of a given sequence will align locally, other methods are generally preferred. However, there have been variations on the method that reduce the space complexity to Θ(min(m,n)) similarly to Hirschberg's algorithm, mentioned above. The Smith-Waterman algorithm was chosen for local alignment in this program due to its foundational nature, and its being the analogous local alignment algorithm to the Needleman–Wunsch algorithm.

Implementation

BEJAT, this implementation of the Needleman–Wunsch and Smith–Waterman algorithms, is of O(nm) complexity in both space and time, as it restricts the gap penalties to at most affine (linear extension penalty) in complexity. It attempts to optimize the running time by implementing a bitflag based adjacency recording scheme as opposed to some of the more computationally complex implementations. This is demonstrated in detail in Example Run 1. It is estimated that this method increases the space requirements by less than about 10%. This implementation supports DNA, RNA and protein sequences. It also supports both simple match-mismatch scoring as well as scoring matrices, and, as mentioned above, supports both constant gap penalties, and affine (linear) gap penalties, which it takes in as negative values. The implementation allows the user to input an arbitrary scoring matrix for DNA and RNA alignment, and offers Blosum45, Blosum50, Blosum62, Blosum80, Blosum90, Pam30, Pam70, and Pam250 as options for protein sequences.

The program is designed to take in a list of input files either via the command line, or if none are provided, from the user. The first input file must start with the settings to be used, and all following entries must be sequences or the same type in FASTA format. The headers to the sequences are expected to begin with a '>' character, and the first word is taken as the name, and expected not to contain any characters that would be unsafe for filenames, as the output files are named based on these sequence names. If more than two sequences are provided, every possible pair will be aligned. The possible outputs from the program are a file containing all of the maximal alignments and their score, the scores matrices as a csv file, the adjacency bit flags for each entry as a csv file, and/or their equivalents as arrows also as a csv file.

Settings

Settings	Possibilities	Notes
GLOBAL=	TRUE, FALSE	Whether to do a global alignment.
LOCAL=	TRUE, FALSE	Whether to do a local alignment.
ALIGNMENT=	TRUE, FALSE	Whether to output the alignment(s). (Defaulted to true)
SCORES=	TRUE, FALSE	Whether to output the score(s).
ADJACENCY=	TRUE, FALSE	Whether to output the adjacency flags(s).
ARROWS=	TRUE, FALSE	Whether to output the adjacency as arrows.
TYPE=	DNA, RNA, PROTEIN	The type of sequences to use. (If the scoring method is only available for one type, this can be left out.)
VALUE=	Varies	See below.

Value Example	Meaning
Pam30,-4 or Blosum80,-10,-1	Name of scoring matrix to use, followed by one or two (negative) integers, representing the gap opening and extension penalties.
5,-3,-4 or 5,-3,-4,-2	Two integers, representing the match and mismatch scores, followed by one or two (negative) integers, representing the gap opening and extension penalties.
5,-3,-1,-4,-2	Five integers, representing a DNA or RNA scoring matrix with the first three values for similarity, transition, transversion, followed by 2 (negative) integers, representing the gap opening and extension penalties.
4,3,4,...,-4,-2	Eleven or twelve integers, representing the ten unique coefficients of the score matrix for DNA or RNA in the following order: GG, AA, CC, TT, GA, GC, GT, AC, AT, CT (or U for RNA); followed by one or two (negative) integers, representing the gap opening and extension penalties.

Results

The output from this program has been thoroughly tested, and four example input files have been provided. These four test runs will be used to demonstrate the performance of the program, and how the algorithms were implemented. The user is of course encouraged to create and test their own files.

Example Run 1

The first such file, Example Run 1, uses the DNA snippet example given in class as it is an illustrative example with a known solution. The results of the global alignment are discussed first. Comparing the output to what was presented in class, we can see that it is consistent. Here the raw adjacency bit flags have been presented to illustrate the feature. As can be seen, a total of three bits are used per cell, with the bits corresponding to: a gap in the first sequence (001), a match or mismatch (010), and a gap in the second sequence (100). Combinations of these flags thus produce values from 1 to 7, though 7 is not observed. To maximize memory efficiency, these flags are stored in blocks of 64 bits in length, the word size on most modern machines. As can be seen in the backtrack diagram, there are two equally good alignments with these parameters, and both are given as results.

Below the global results are the local alignment results, presented for comparison. As can be seen in the scores table, there are two maximal alignments with a score of 14, and as can be seen from the backtrack table, both alignments have two equally valid alignments for the TT section in the middle. This gives a total of 4 possible alignments that are equally optimal, and all 4 are presented as results.

Input File ExampleRun1.txt

```
TYPE=DNA
VALUE=5,-3,-4  #Match score of 5, mismatch of -3, and gap of -4
GLOBAL=TRUE    #Perform a global alignment
LOCAL=TRUE     #Perform a local alignment
SCORES=TRUE    #Output the scores
ADJACENCY=TRUE #Output the adjacency flags
ARROWS=TRUE    #Output the adjacency as arrows

>Test1
GAATTCAGTTA

>Test2
GGATCGA
```

Output Optimal Global Alignments (Score = 11)

```
G A A T T C A G T T A        G A A T T C A G T T A
|   |   | |   |     |        |   | |   |   |     |
G G A - T C - G - - A        G G A T - C - G - - A
```

Allan Stewart

CS 3220 Final Project

Pairwise Sequence Alignment via Dynamic Programming

Global Scores

S\F	-	G	A	A	T	T	C	A	G	T	T	A
-	0	-4	-8	-12	-16	-20	-24	-28	-32	-36	-40	-44
G	-4	5	1	-3	-7	-11	-15	-19	-23	-27	-31	-35
G	-8	1	2	-2	-6	-10	-14	-18	-14	-18	-22	-26
A	-12	-3	6	7	3	-1	-5	-9	-13	-17	-21	-17
T	-16	-7	2	3	12	8	4	0	-4	-8	-12	-16
C	-20	-11	-2	-1	8	9	13	9	5	1	-3	-7
G	-24	-15	-6	-5	4	5	9	10	14	10	6	2
A	-28	-19	-10	-1	0	1	5	14	10	11	7	11

With Global Backtrack Arrows

S\F	-	G	A	A	T	T	C	A	G	T	T	A
-	⟳0	←-4	←-8	←-12	←-16	←-20	←-24	←-28	←-32	←-36	←-40	←-44
G	↑-4	↖5	←1	←-3	←-7	←-11	←-15	←-19	↖←-23	←-27	←-31	←-35
G	↑-8	↑↖1	↖2	↖←-2	↖←-6	↖←-10	↖←-14	↖←-18	↖-14	←-18	←-22	←-26
A	↑-12	↑-3	↖6	↖7	←3	←-1	←-5	↖←-9	←-13	↖←-17	↖←-21	↖-17
T	↑-16	↑-7	↑2	↑↖3	↖12	↖←8	←4	←0	←-4	↖←-8	↖←-12	←-16
C	↑-20	↑-11	↑-2	↑↖-1	↑8	↖9	↖13	←9	←5	←1	←-3	←-7
G	↑-24	↑↖-15	↑-6	↑↖-5	↑4	↑↖5	↑9	↖10	↖14	←10	←6	←2
A	↑-28	↑-19	↑↖-10	↖-1	↑0	↑↖1	↑5	↖14	↑←10	↖11	↖←7	↖11

Raw Global Adjacency Flags

S\F	-	G	A	A	T	T	C	A	G	T	T	A
-	0	4	4	4	4	4	4	4	4	4	4	4
G	1	2	4	4	4	4	4	4	6	4	4	4
G	1	3	2	6	6	6	6	6	2	4	4	4
A	1	1	2	2	4	4	4	6	4	6	6	2
T	1	1	1	3	2	6	4	4	4	6	6	4
C	1	1	1	3	1	2	2	4	4	4	4	4
G	1	3	1	3	1	3	1	2	2	4	4	4
A	1	1	3	2	1	3	1	2	5	2	6	2

1 = ↑	2 = ↖	4 = ←	3 = ↑↖	5 =↑←	6 = ↖←

Output Optimal Local Alignments (Score = 14)

```
G A A T T C - A          G A A T T C - A
|   |   | |   |          |   | |   |   |
G G A - T C G A          G G A T - C G A

G A A T T C A G          G A A T T C A G
|   |   | |   |          |   | |   |   |
G G A - T C - G          G G A T - C - G
```

Local Scores

S\F	-	G	A	A	T	T	C	A	G	T	T	A
-	0	0	0	0	0	0	0	0	0	0	0	0
G	0	5	1	0	0	0	0	0	5	1	0	0
G	0	5	2	0	0	0	0	0	5	2	0	0
A	0	1	10	7	3	0	0	5	1	2	0	5
T	0	0	6	7	12	8	4	1	2	6	7	3
C	0	0	2	3	8	9	13	9	5	2	3	4
G	0	5	1	0	4	5	9	10	14	10	6	2
A	0	1	10	6	2	1	5	14	10	11	7	11

With Local Backtrack Arrows

S\F	-	G	A	A	T	T	C	A	G	T	T	A
-	⟳0	⟳0	⟳0	⟳0	⟳0	⟳0	⟳0	⟳0	⟳0	⟳0	⟳0	⟳0
G	⟳0	↖5	←1	⟳0	⟳0	⟳0	⟳0	⟳0	↖5	←1	⟳0	⟳0
G	⟳0	↖5	↖2	⟳0	⟳0	⟳0	⟳0	⟳0	↖5	↖2	⟳0	⟳0
A	⟳0	↑1	↖10	↖7	←3	⟳0	⟳0	↖5	↑←1	↖2	⟳0	↖5
T	⟳0	⟳0	↑6	↖7	↖12	↖←8	←4	↑1	↖2	↖6	↖7	←3
C	⟳0	⟳0	↑2	↑↖3	↑8	↖9	↖13	←9	←5	↑2	↑↖3	↖4
G	⟳0	↖5	←1	⟳0	↑4	↑↖5	↑9	↖10	↖14	←10	←6	←2
A	⟳0	↑1	↖10	↖←6	←2	↑↖1	↑5	↖14	↑←10	↖11	↖←7	↖11

Example Run 2

The second example file adds in a third sequence to demonstrate the results when more than two sequences are provided. The input file and global output alignments are presented below. As can be seen, the program aligns all possible pairs of the inputs, and the results for the first alignment are consistent with the above results from Example Run 1.

Input File ExampleRun2.txt

```
TYPE=DNA
VALUE=5,-3,-4  #Match score of 5, mismatch of -3, and gap of -4
GLOBAL=TRUE    #Perform a global alignment
LOCAL=TRUE     #Perform a local alignment
>Test1
GAATTCAGTTA
>Test2
GGATCGA
>Test3
GATTACA
```

Output Optimal Global Alignments Test1 & Test2 (Score = 11)

```
G A A T T C A G T T A            G A A T T C A G T T A
|   |   | |   |     |            |   | |   |   |     |
G G A - T C - G - - A            G G A T - C - G - - A
```

Output Optimal Global Alignments Test1 & Test3 (Score = 11)

```
G A A T T C A G T T A            G A A T T C A G T T A
|   | | |   |       |            |   | | |   |       |
G - A T T - A - - C A            G - A T T - A - C - A

G A A T T C A G T T A            G A A T T C A G T T A
|   | | |   |       |            | |   | |   |       |
G - A T T - A C - - A            G A - T T - A C - - A

G A A T T C A G T T A            G A A T T C A G T T A
| |   | |   |       |            | |   | |   |       |
G A - T T - A - C - A            G A - T T - A - - C A
```

Output Optimal Global Alignments Test2 & Test3 (Score = 9)

```
G G A - T - C G A                G G A T - - C G A
  | |   |   |   |                  | | |     |   |
- G A T T A C - A                - G A T T A C - A

G G A T - - C G A                G G A - T - C G A
|   | |     |   |                |   |   |   |   |
G - A T T A C - A                G - A T T A C - A
```

Example Run 3

The third example run file uses a selection of the protein sequence data used in the second assignment. Only the global alignment of the first pair is presented. It should be noted that because of the high degree of similarity between the two sequences, the global alignments have quite high scores, there are no gaps, and the local alignments are identical. It should also be noted that high degree of similarity is quite evident when looking at the score data, as the resulting matrix has its highest values clustered along the diagonal.

Input File ExampleRun3.txt

```
TYPE=PROTEIN
VALUE=BLOSUM62,-10,-1 #BLOSUM62 with an affine gap of -10 + -1 * length
GLOBAL=TRUE           #Perform a global alignment
LOCAL=TRUE            #Perform a local alignment
SCORES=TRUE           #Output the scores
ADJACENCY=TRUE        #Output the adjacency flags
ARROWS=TRUE           #Output the adjacency as arrows
>ALM05426 A/Addis Ababa/1514A07305892N/2013 2013/09/26 NA
MNPNQKIITIGSVSLTISTICFFMQIAILITTVTLHFKQYEFNSPPNNQVMLCEPTIIERNITEIVYLTN
TTIEKEICPKPAEYRNWSKPQCGITGFAPFSKDNSIRLSAGGDIWVTREPYVSCDPDKCYQFALGQGTTL
NNVHSNNTVRDRTPYRTLLMNELGVPFHLGTKQVCIAWSSSSCHDGKAWLHVCITGDDKNATASFIYNGR
LVDSVVSWSKDILRTQESECVCINGTCTVVMTDGSASGKADTKILFIEEGKIVHTSTLSGSAQHVEECSC
YPRYPGVRCVCRDNWKGSNRPIVDINIKDHSIVSSYVCSGLVGDTPRKNDSSGSSHCLDPNNEEGGHGVK
GWAFDDGNDVWMGRTINETSRLGYETFKVIEGWSNPKSKLQTNRQVIVDRGDRSGYSGIFSVEGKSCINR
CFYVELIRGRKEETEVLWTSNSIVVFCGTSGTYGTGSWPDGADLNLMPI
>BAD16642 A/Aichi/2/1968 1968// NA
MNPNQKIITIGSVSLTIATVCFLMQIAILVTTVTLHFKQYECDSPASNQVMPCEPIIIERNITEIVYLNN
TTIDKEKCPKVVEYRNWSKPQCQITGFAPFSKDNSIRLSAGGDIWVTREPYVSCDHGKCYQFALGQGTTL
DNKHSNDTIHDRIPHRTLLMNELGVPFHLGTRQVCIAWSSSSCHDGKAWLHVCITGDDKNATASFIYDGR
LVDSIGSWSQNILRTQESECVCINGTCTVVMTDGSASGRADTRILFIEEGKIVHISPLSGSAQHVEECSC
YPRYPGVRCICRDNWKGSNRPVVDINMEDYSIDSSYVCSGLVGDTPRNDDRSSNSNCRNPNNERGNQGVK
GWAFDNGDDVWMGRTISKDLRSGYETFKVIGGWSTPNSKSQINRQVIVDSDNRSGYSGIFSVEGKSCINR
CFYVELIRGRKQETRVWWTSNSIVVFCGTSGTYGTGSWPDGANINFMPI
>ABQ53724 A/Alabama/01/1998 1998// NA
MNPNQKIITIGSVSLTIATICFLMQIAILVTTVTLHFKQYECSSPPNNQVMLCEPTIIERNITEIVYLTN
TTIEKEICPKLAEYRNWSKPQCKITGFAPFSKDNSIRLSAGGDIWVTREPYVSCDPDKCYQFALGQGTTL
NNRHSNDTVHDRTPYRTLLMNELGVPFHLGTKQVCIAWSSSSCHDGKAWLHVCVTGHDENATASFIYDGR
LVDSIGSWSKKILRTQESECVCINGTCTVVMTDGSASGRADTKILFIEEGKIVHISPLSGSAQHVEECSC
YPQYPGVRCVCRDNWKGSNRPIVDINVKDYSIVSSYVCSGLVGDTPRKNDSSSSSHCLNPNNEEGGHGVK
GWAFDDGNDVWMGRTISEKFRSGYETFKVIEGWSKPNSKLQINRQVIVDRGNRSGYSGIFSVEGKSCINR
CFYVELIRGRKQETEVWWTSNSIVVFCGTSGTYGTGSWPDGADINLMPI
```

Output Optimal Global Alignment of ALM05426 & BAD16642 (Score = 2176)

```
MNPNQKIITIGSVSLTISTICFFMQIAILITTVTLHFKQYEFNSPPNNQVMLCEPTIIERNITEIVYLTNTTIEKEICPKPAEYRNWSKPQ
||||||||||||||||| | || |||||| |||||||||||  ||  |||| ||| |||||||||||| |||| || |||  |||||||||
MNPNQKIITIGSVSLTIATVCFLMQIAILVTTVTLHFKQYECDSPASNQVMPCEPIIIERNITEIVYLNNTTIDKEKCPKVVEYRNWSKPQ

CGITGFAPFSKDNSIRLSAGGDIWVTREPYVSCDPDKCYQFALGQGTTLNNVHSNNTVRDRTPYRTLLMNELGVPFHLGTKQVCIAWSSSS
| ||||||||||||||||||||||||||||||||  ||||||||||||| | ||| |  || | |||||||||||||||| ||||||||||
CQITGFAPFSKDNSIRLSAGGDIWVTREPYVSCDHGKCYQFALGQGTTLDNKHSNDTIHDRIPHRTLLMNELGVPFHLGTRQVCIAWSSSS

CHDGKAWLHVCITGDDKNATASFIYNGRLVDSVVSWSKDILRTQESECVCINGTCTVVMTDGSASGKADTKILFIEEGKIVHTSTLSGSAQ
||||||||||||||||||||||||| ||||||  |||  ||||||||||||||||||||||||||| ||| ||||||||||| | ||||||
CHDGKAWLHVCITGDDKNATASFIYDGRLVDSIGSWSQNILRTQESECVCINGTCTVVMTDGSASGRADTRILFIEEGKIVHISPLSGSAQ

HVEECSCYPRYPGVRCVCRDNWKGSNRPIVDINIKDHSIVSSYVCSGLVGDTPRKNDSSGSSHCLDPNNEEGGHGVKGWAFDDGNDVWMGR
|||||||||||||||| ||||||||||| ||||  | || ||||||||||||||  | |  | |  |||| |  |||||||| | ||||||
HVEECSCYPRYPGVRCICRDNWKGSNRPVVDINMEDYSIDSSYVCSGLVGDTPRNDDRSSNSNCRNPNNERGNQGVKGWAFDNGDDVWMGR

TINETSRLGYETFKVIEGWSNPKSKLQTNRQVIVDRGDRSGYSGIFSVEGKSCINRCFYVELIRGRKEETEVLWTSNSIVVFCGTSGTYGT
||    | |||||||| ||| | || | |||||||   ||||||||||||||||||||||||||||| || | ||||||||||||||||||
TISKDLRSGYETFKVIGGWSTPNSKSQINRQVIVDSDNRSGYSGIFSVEGKSCINRCFYVELIRGRKQETRVWWTSNSIVVFCGTSGTYGT

GSWPDGADLNLMPI
|||||||  | |||
GSWPDGANINFMPI
```

Example Run 4

The fourth and final example file presented here is a local alignment between two quite different sequences, a hemoglobin protein from Pseudoterranova decipiens, and a truncated record for hemoglobin in Tetrahymenidae, tetrahymena. As can be seen, from the optimal local alignment, there isn't much similarity between these protein sequences.

Input File ExampleRun4.txt

```
TYPE=PROTEIN
VALUE=BLOSUM80,-10,-1 #BLOSUM80 with an affine gap of -10 + -1 * length
GLOBAL=FALSE          #Don't perform a global alignment
LOCAL=TRUE            #Perform a local alignment
SCORES=TRUE           #Output the scores
ARROWS=TRUE           #Output the adjacency as arrows
>AAA29796.1 hemoglobin [Pseudoterranova decipiens]
MHSSIVLATVLFVAIASASKTRELCMKSLEHAKVGTSKEAKQDGIDLYKHMFEHYPAMKKYFKHRENYTP
ADVQKDPFFIKQGQNILLACHVLCATYDDRETFDAYVGELMARHERDHVKVPNDVWNHFWEHFIEFLGSK
TTLDEPTKHAWQEIGKEFSHEISHHGRHSVRDHCMNSLEYIAIGDKEHQKQNGIDLYKHMFEHYPHMRKA
FKGRENFTKEDVQKDAFFVNKDTRFCWPFVCCDSSYDDEPTFDYFVDALMDRHIKDDIHLPQEQWHEFWK
LFAEYLNEKSHQHLTEAEKHAWSTIGEDFAHEADKHAKAEKDHHEGEHKEEHH
>P17724.1 RecName: Full=Group 1 truncated hemoglobin;[...]
MNKPQTIYEKLGGENAMKAAVPLFYKKVLADERVKHFFKNTDMDHQTKQQTDFLTMLLGGPNHYKGKNMT
EAHKGMNLQNLHFDAIIENLAATLKELGVTDAVINEAAKVIEHTRKDMLGK
```

Output Optimal Local Alignment (Score = 28)

```
DHHEGEHKEEHH
 |  |    | |
NHYKGKNMTEAH
```

Conclusion

This implementation of the Needleman–Wunsch and Smith–Waterman algorithms seems to perform as expected in all tests, and for sequences of similar length it is able to perform the alignments quite rapidly. Though this tool is not a thoroughly optimized as many of the sequence alignment tools available both for purchase and made free as open source software, it is potentially useful as an educational tool due to its development in Java, and its relatively straightforward options.

Sources

Hirschberg, D. S. (1975). "A linear space algorithm for computing maximal common subsequences". Communications of the ACM. 18 (6): 341–343. doi:10.1145/360825.360861.

Miller, Webb; Myers, Eugene (1988). "Optimal alignments in linear space". Bioinformatics. 4: 11–17. doi:10.1093/bioinformatics/4.1.11.

Mount DM. (2004). Bioinformatics: Sequence and Genome Analysis (2nd ed.). Cold Spring Harbor Laboratory Press: Cold Spring Harbor, NY. ISBN 0-87969-608-7

Needleman, Saul B. & Wunsch, Christian D. (1970). "A general method applicable to the search for similarities in the amino acid sequence of two proteins". Journal of Molecular Biology. 48 (3): 443–53. doi:10.1016/0022-2836(70)90057-4.

Smith, Temple F. & Waterman, Michael S. (1981). "Identification of Common Molecular Subsequences" (PDF). Journal of Molecular Biology. 147: 195–197. doi:10.1016/0022-2836(81)90087-5.