Lecture 22-23: Sequence alignment (edit distance) (15.4)

CS303: Algorithms

Last update: March 3, 2014

1 Review

1. Dynamic programming problem
   - Benefit: reduce from exponential to polynomial for optimization problems
   - Requirements: overlapping subproblem and optimal substructure
   - Tool: memoization
2. Examples: rod cutting, matrix chain multiplication

2 Genetic Background

It is a biological problem: If we know one gene and another gene, what is their relationship?

Use the NCBI website to show what sequence alignment is (http://www.ncbi.nlm.nih.gov/) (blast human hexosaminidase A gene NM_000520.4 which is associated with the tay-sachs disease. the best match is with a mouse gene NM_010421.4). The following figure is from Emboss Needle site generated alignment (only part of the result is shown).

```
NM_010421.4       1 ------------------------------------------------     0

NM_000520.4       1 AGTTGCCGACGCCCGGCACAATCCGCTGCACGTAGCAGGAGCCTCAGGTC    50

NM_010421.4       1 ------------------------------------------------     0

NM_000520.4      51 CAGGCCGGAAGTGAAAGGGCAGGGTGTGGGTCCTCCTGGGGTCGCAGGCG   100

NM_010421.4       1 ------------------------------------------------     0

NM_000520.4     101 CAGAGCCGCCTCTGGTCACGTGATTCGCCGATAAGTCACGGGGGCGCCGC   150

NM_010421.4       1 --AGCTGACC-GGGGCTCACGTGGGCTCAGCCTGCT--GGAAGGGGAGCT    45
                      |.|||||| |||.|||||||||  .||||..||  |..||||||
NM_000520.4     151 TCACCTGACCAGGGTCTCACGTGG--CCAGCCCCCTCCGAGAGGGGA---   195

NM_010421.4      46 GGCCGGTGGGCCATGGCCGGCTGCAGGCTCTGGGTTTCGCTGCTGCTGGC    95
                    |.||.|.||||||||.|..|||.|||||.|||.||||||||||||||||
NM_000520.4     196 GACCAGCGGGCCATGACAAGCTCCAGGCTTTGGTTTTCGCTGCTGCTGGC   245

NM_010421.4      96 GGCGGCGTTGGCTTGCTTGGCCACGGCACTGTGGCCGTGGCCCCAGTACA   145
                    |||.|||||.||..|...|||.||||||.||.||||||.|||||.|||.||.
NM_000520.4     246 GGCAGCGTTCGCAGGACGGGCGACGGCCCTCTGGCCCTGGCCTCAGAACT   295

NM_010421.4     146 TCCAAACCTACCACCGGCGCTACACCCTGTACCCCAACAACTTCCAGTTC   195
                    |||||||||.|.|||.||||||||..|||.|||||.||||||||.||.|||
NM_000520.4     296 TCCAAACCTCCGACCAGCGCTACGTCCTTTACCCGAACAACTTTCAATTC   345

NM_010421.4     196 CGGTACCATGTCAGTTCGGCCGCGCGCAGGCGGGCTGCGTC-GTCCTCGACG   244
                    |.||||.|||||||.||||||||||||.|.|||||| || ||||||||||
NM_000520.4     346 CAGTACGATGTCAGCTCGGCCGCGCAGCCCGGCTGC-TCAGTCCTCGACG   394

NM_010421.4     245 AGGCCTTTCGACGCTACCGTAACCTGCTCTTCGGTTCCGGCTCTTGGCCC   294
                    |||||||.|..|||||.|||.|||||||.||||||||||.|||||||||
NM_000520.4     395 AGGCCTTCCAGCGCTATCGTGACCTGCTTTTCGGTTCCGGGTCTTGGCCC   444

NM_010421.4     295 CGACCCAGCTT--CTCAAA--TAAACAGCAAACGTTGGGGAAGAACATTC   340
                    ||.|   |||  ||||.|  .||||.|||.||..|||.||||||..|..
NM_000520.4     445 CGTC----CTTACCTCACAGGGAAACGGCATACACTGGAGAAGAATGTGT   490
```

3 What is sequence alignment

An (pairwise) alignment is a set of columns of three types:

- Match: bases are the same
- Mismatch: bases are different
- Gap: one base is given, paired with a blank

That is, for sequences $X = x_1...x_m$ and $Y = y_1...y_n$, each column (pair) may or may not have a character from X and may or may not have a character from Y (but must have at least one character total).

So, given a pair (x_i, y_j), characters $x_1...x_{i-1}$ must appear before x_i , and characters $y_1...y_{j-1}$ must appear before y_j. That is, X and Y are both in order, so there are no transpositions in this model.

4 How to evaluate an alignment

Given sequences of DNA or protein, how does one identify a "good" alignment?

Intuition: A perfect match in a column is good.
That is, for pair (x_i, y_j), $x_i = y_j$ is good and $x_i \neq y_j$ is not. In order to define what "good" means, we use the edit distance, or Levenshtein distance.

Let the valid set of operations on a sequence be to mutate, delete, or insert a character. The goal is to turn $X \rightarrow Y$ by way of some intermediate operations. The number of operations required is the edit, or Levenshtein, distance between the two sequences. Note that the Levenshtein distance is a general purpose string comparison algorithm and was not specifically developed for DNA or protein alignment problems. As noted in Wikipedia, "edit distance can be considered a generalization of the Hamming distance, which is used for strings of the same length and only considers substitution edits."

5 How to determine distance between sequence

In nature, there are usually three kinds of changes that may happen to a sequence (edit that may happen to a sequence)

- Insert a nucleotide into the gene
  Original: GGCCATT → GGCCATATT
- Delete a nucleotide from the gene
  Original: GGCCATT → GGCATT
- Mutate a nucleotide in the gene (most common)
  Original: GGCCATT → GGCTATT

Our goal is to find out what is the edit distance with the minimum number of required operations. We could say a perfect match has a cost of zero, while a mismatch / insertion / deletion costs one. Instead, though, we will say that a mismatch costs some $\mu > 0$ and insertion / deletion costs some $v > 0$. With this approach, instead of minimizing distance, we will maximize similarity.

- perfect match = +1
- mismatch = $-\mu$
- insertion / deletion = $-v$

Let's define a "similarity matrix" for a DNA sequence as follows, with dashes denoting gaps:

		x_i				
		A	**C**	**G**	**T**	**-**
y_j	**A**	+1	-μ	-μ	-μ	-*v*
	C	-μ	+1	-μ	-μ	-*v*
	G	-μ	-μ	+1	-μ	-*v*
	T	-μ	-μ	-μ	+1	-*v*
	-	-*v*	-*v*	-*v*	-*v*	UD

Now we can talk about similarity between two sequences relative to some particular alignment. Let the function S be similarity score, defined as follows:

$$S(x_i, y_j) = \begin{cases} +1 & if\ x_i = y_j \\ -\mu & if\ x_i \neq y_j \end{cases}$$
$$S(x_i, -) = S(-, y_j) = -v$$

The total similarity score between sequences is the sum of similarity scores of all columns.

For example, let x = ACC, y = CATT, $-\mu$ = -1, and $-v$ = -2.

-	A	C	C	-
C	A	-	T	T
-2	+1	-2	-1	-2

The total similarity score of the above alignment is -6

Q:/ Which (valid) alignment yields the largest similarity score, and what is the score?
A:/ Now that we have some methodology for computing the similarity score of an alignment, the question remains: which alignment yields the highest score? What is an algorithm for determining that alignment / score?

1. Brute force
   (a) Shift one sequence against the other.
   (b) Compute similarity scores for all of these combinations.
   (c) Pick the combination with the best score. Its efficiency is exponential because we can also insert gaps.
2. In 1970, the global alignment problem was proposed by Needleman and Wunsch. The solution they provided was for a more generalized problem and runs in cubic time. In 1982, Gotoh provided a simplified and optimized version of the algorithm that runs in quadratic time. A brief description and intuition of the Gotoh algorithm is given as follows:

   Define the i^{th}-prefix of X as $x_1...x_i$ and the j^{th}-prefix of Y as $y_1...y_j$. Let the value function $V(i, j)$ be the score of the optimal alignment of the i^{th}-prefix of X with the

j^{th}-prefix of Y.

The desired goal is $V(m, n)$, since the m^{th}-prefix of X is all of X and the n^{th}-prefix of Y is all of Y. A dynamic programming strategy can be used to progress from $V(1, 1) \rightarrow V(m, n)$.

Intuition: Initialize the grid somehow, do the core of the algorithm, terminate.

If we define the grid as being m by n cells containing V of that cell's indices, then termination is just to return the contents of the bottom right cell.

The intuition behind the core of the algorithm is as follows: for each $V(i, j)$,

$$V(i,j) = max \begin{cases} S(x_i, y_j) + V(x_{i-1}, y_{j-1}) & if\ x_i = y_j \\ S(-, y_j) + V(x_i, y_{j-1}) & if\ a\ gap\ pairs\ y_j \\ S(x_i, -) + V(x_{i-1}, y_j) & if\ a\ gap\ pairs\ x_i \end{cases}$$

6 Global sequence alignment

6.1 Initialization

In this global sequence alignment algorithm, the alignment matrix should be initialized with particular seed values and values for the scoring, i.e. match or mismatch and gap penalty, parameters should be established.

Q:/ Given that sequence x is placed vertically and sequence y is placed horizontally on the global alignment matrix, what are the values for the first row and first column?

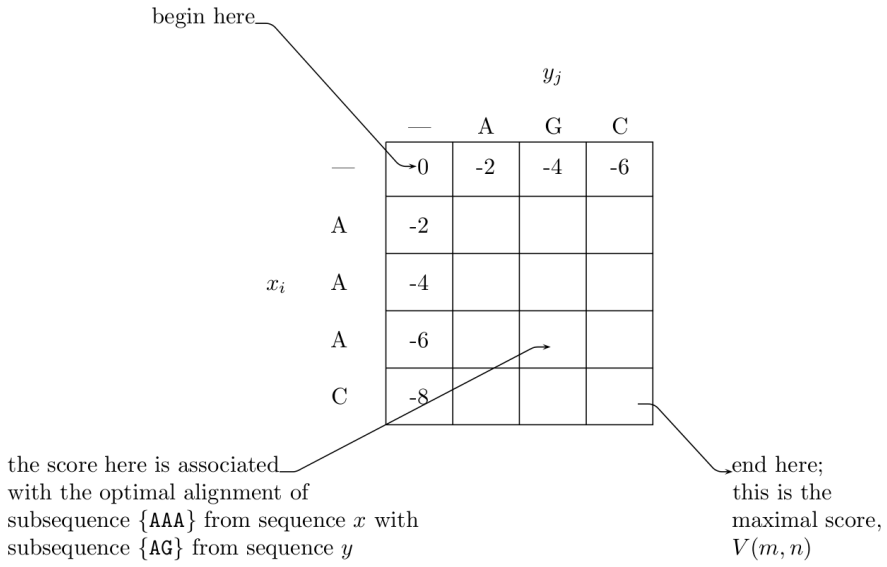
For the first column,
$V(i, 0) \forall i \in \{1, ..., m\}$ is required, that is the first i characters from sequence X with nothing from sequence Y.
Thus, the alignment for this case would look like:

Sequence X	x1	x2	x3	x4	x5
Sequence Y	-	-	-	-	-

The first column is very similar where characters from Y is aligned with gaps.
Thus, the matrix is ready to be initialized with the bases of sequence X written along the leftmost column, sequence Y written above the uppermost row, $V(i, 0) = -i \times v$ in the first column, $V(0, j) = -j \times v$ in the first row, and $V(0, 0) = 0$ in the uppermost, leftmost cell.
For example, globally align the following two sequences:
sequence X: AAAC
sequence Y: AGC
with the following values for the scoring parameters:

$$S(x_i, y_j) = \begin{cases} +1 & \text{if } x_i = y_j \\ -1 & \text{if } x_i \neq y_j \end{cases}$$

$$-g = -2$$

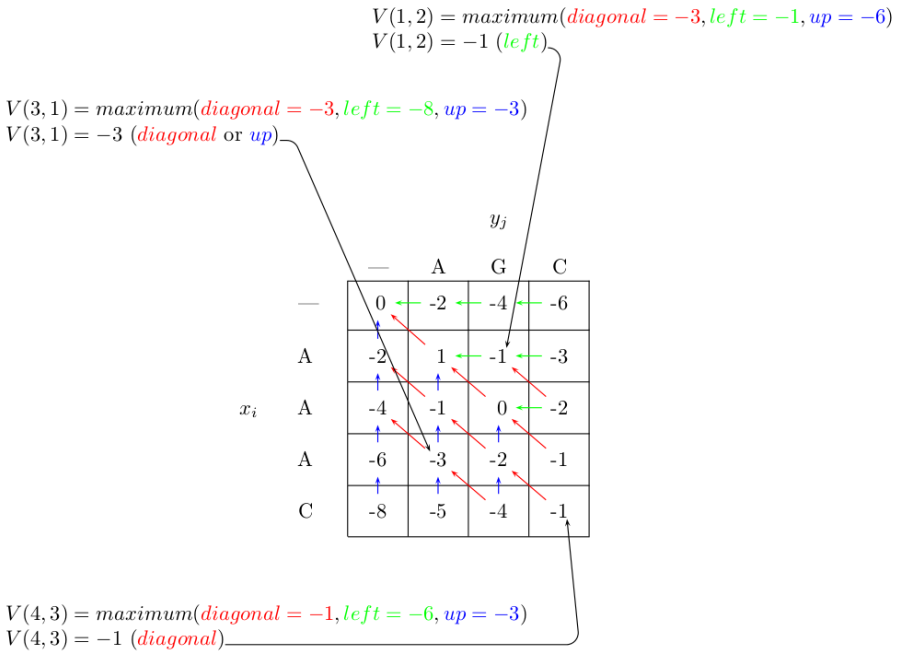


6.2 Calculate and fill in the scores

Now that the global alignment matrix has been initialized, the algorithm proceeds to calculate possible scores and fill with the maximum possible score for each cell in the matrix.

Note that this process proceeds from the top left corner to the bottom right corner of the matrix. However, again as a matter of convention, the arrows point from the cell having a specific score to the cell that permitted such a score to ease the traceback process once the bottom right corner is reached. Also, note that if the maximum score is due to more than one type of traversal, both traversals are kept and indicated. (See examples below.)

$$V(i,j) = maximum \begin{cases} V(i-1,j-1) + S(x_i, y_i) & \text{match or mismatch in the diagonal traversal} \\ V(i,j-1) - g & \text{gap in sequence } x\text{, i.e. traversing left} \\ V(i-1,j) - g & \text{gap in sequence } y\text{, i.e. traversing up} \end{cases}$$



6.3 Termination

After calculating and filling maximum possible scores into the cells, the algorithm terminates by returning the maximal score $V(m, n)$ found in the lowermost, rightmost cell.

7 How to retrieve the optimal alignment

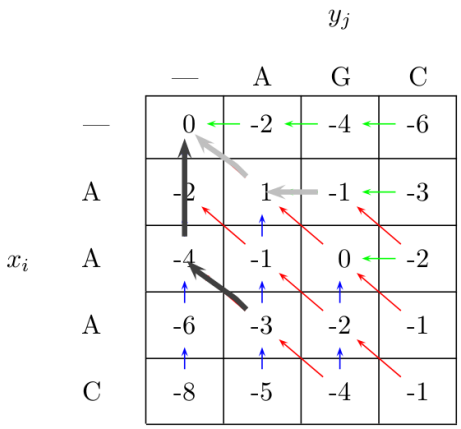
In this global sequence alignment algorithm thus far, the maximal score has been determined. However, the global sequence alignments that correspond to this maximal score await determination. In fact, many possible scores for optimal alignments of sequence x with sequence y can be determined. Starting at an arbitrarily given cell in the global sequence alignment matrix and retracing the path taken to reach that cell from the uppermost, leftmost cell, i.e. a traceback, will yield the associated optimal alignment for that starting cell and score. That is, given an arbitrary starting cell in the alignment matrix i, j with score $V(i, j)$, the traceback will yield the alignment of 0 through i bases of sequence X with 0 through j bases of sequence Y.

Thus, in order to determine the best global sequence alignments, the starting cell for the traceback(s) should be the lowermost, rightmost cell, $V(m, n)$, so that all bases with indices $1 \leq i \leq m$ of sequence X are globally aligned with all bases indices $1 \leq j \leq n$ of sequence Y.

Before determining the best global sequence alignment(s) of sequence X with sequence Y in this example, it is instructive to examine some of the other optimal sequence alignments available from an arbitrarily given cell and associated score, $V(i, j)$, after initializing, calculating and filling align-

ment scores, and terminating the global sequence alignment matrix.

Using the earlier examples of $V(1,2)$ and $V(3,1)$, below are associated tracebacks and corresponding alignments. The third earlier example of $V(4,3)$ will be discussed next, as it is the maximal score and will yield the best global sequence alignments.



$V(1,2) = -1$ (traceback is in light gray), which corresponds to alignment:

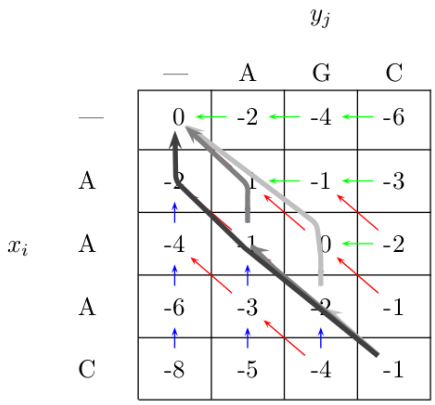
sequence x:	A	—
sequence y:	A	G

$V(3,1) = -3$ (traceback is in dark gray), which corresponds to alignment:

sequence x:	A	A	A
sequence y:	—	—	A

Now the best global sequence alignment(s) are determined by starting not from an arbitrarily given cell but rather the lowermost, rightmost cell which contains the maximal score, $V(m,n)$.

Note that in the remaining earlier example of $V(4,3)$ there are three possible tracebacks and corresponding alignments from the lowermost, rightmost cell to the uppermost, leftmost cell. They are considered the upmost, middle, and downmost alignments. Also note that all three best global alignments have the same maximal score of -1 since they start from the same cell $V(4,3)$.



$V(4,3) = -1$ (upmost traceback is in light gray), which corresponds to the upmost alignment:

sequence x:	A	A	A	C
sequence y:	A	G	—	C

$V(4,3) = -1$ (middle traceback is in gray), which corresponds to the middle alignment:

sequence x:	A	A	A	C
sequence y:	A	—	G	C

$V(4,3) = -1$ (downmost traceback is in dark gray), which corresponds to the downmost alignment:

sequence x:	A	A	A	C
sequence y:	—	A	G	C

As an aside, should all the alignments be desired, a depth first search (DFS) and breadth first search (BFS) can be performed starting from $V(m,n)$.

7.1 Efficiency

The algorithm above has the following computational space and time requirements for scoring and global sequence alignment.

1. Space requirement:
   $(m+1) \times (n+1) \in O(mn)$

2. Time requirement:
   $O(mn)$ to report the maximal score, $O(m+n)$ to report the best alignment(s)

   The worst case analysis is to skirt the bottom of the global alignment matrix for one sequence giving $(m+n)$ work and skirt the side of the global alignment matrix for the other sequence giving $(m+n)$ for a total of $2(m+n)$ for the pair. Thus, $O(mn)$ is required to report the maximal score and global alignment.