Hirschberg's algorithm

MPA-PRG: Programming in Bioinformatics

Exercise 5

Sequence alignment

- inputs:
  - two sequences ($A = a_1a_2...a_m$ and $B = b_1b_2...b_n$)
  - scoring system – match/mismatch scores and gap penalty

- global alignment – Needleman-Wunsch algorithm
- local alignment – Smith-Waterman algorithm

- time complexity $O(mn)$
- space complexity $O(mn)$

Scoring matrix

B

	S	j	A	A	G	G	T	A	T	G	A	A	T	C
	i	0	1	2	3	4	5	6	7	8	9	10	11	12
A	A	1												
	A	2												
	C	3												
	G	4												
	T	5												
	T	6												
	G	7												
	A	8												

best alignment between prefix a1..a4
and prefix b1..b4

Linear space alignment

		B											
	S	*j*	A	A	G	G	T	A	T	G	A	A	T	C
	i	0	1	2	3	4	5	6	7	8	9	10	11	12
A	A	1												
	A	2												
	C	3												
	G	4												
	T	5												
	T	6												
	G	7												
	A	8												

We need only three values:
$S(i - 1, j)$
$S(i - 1, j - 1)$
$S(i, j - 1)$

Therefore we only need to keep previous and current row in memory.

Linear space alignment

- Given sequences A and B we can in linear time score alignment of:
  - A with every prefix of B
  - every prefix of A with B
  - a particular prefix of A with every prefix of B
  - a particular suffix of A with every sufix of B

Alignment between prefixes of A and B

			B											
	S	*j*	**A**	**A**	**G**	**G**	**T**	**A**	**T**	**G**	**A**	**A**	**T**	**C**
	i	0	1	2	3	4	5	6	7	8	9	10	11	12
A	**A**	1												
	A	2												
	C	3												
	G	4												
	T	5												
	T	6												
	G	7												
	A	8												

Best scores between
A and all prefixes of B

Alignment between prefixes of A and B

			B											
	S	*j*	**A**	**A**	**G**	**G**	**T**	**A**	**T**	**G**	**A**	**A**	**T**	**C**
	i	0	1	2	3	4	5	6	7	8	9	10	11	12
A	**A**	1												
	A	2												
	C	3												
	G	4												
	T	5												
	T	6												
	G	7												
	A	8												

Best scores between all prefixes A and B

Alignment between prefixes of A and B

		B											
	S	*j*	**A**	**A**	**G**	**G**	**T**	**A**	**T**	**G**	**A**	**A**	**T**	**C**
	i	0	1	2	3	4	5	6	7	8	9	10	11	12
A	**A**	1												
	A	2												
	C	3												
	G	4												
	T	5												
	T	6												
	G	7												
	A	8												

Score of optimal alignment
between all prefixes A and a prefix of B

Best scores between
all prefixes A and B

Alignment between suffixes of A and B

			B												
	S	*j*	A	A	G	G	T	A	T	G	A	A	T	C	
	i	0	1	2	3	4	5	6	7	8	9	10	11	12	
A	A	1													8
	A	2													7
	C	3													6
	G	4													5
	T	5													4
	T	6													3
	G	7													2
	A	8													1
			12	11	10	9	8	7	6	5	4	3	2	1	0

The same principle as for scoring of prefixes

Best alignment between suffix a5..a8 and suffix b5..b12

Path through scoring matrix

B $ymid = 4$

	S	j	**A**	**A**	**G**	**G**	**T**	**A**	**T**	**G**	**A**	**A**	**T**	**C**
	i	0	1	2	3	4	5	6	7	8	9	10	11	12
A	**A**	1												
	A	2												
	C	3												
$\frac{m}{2}$	**G**	4												
	T	5												
	T	6												
	G	7												
	A	8												

Path goes through middle row

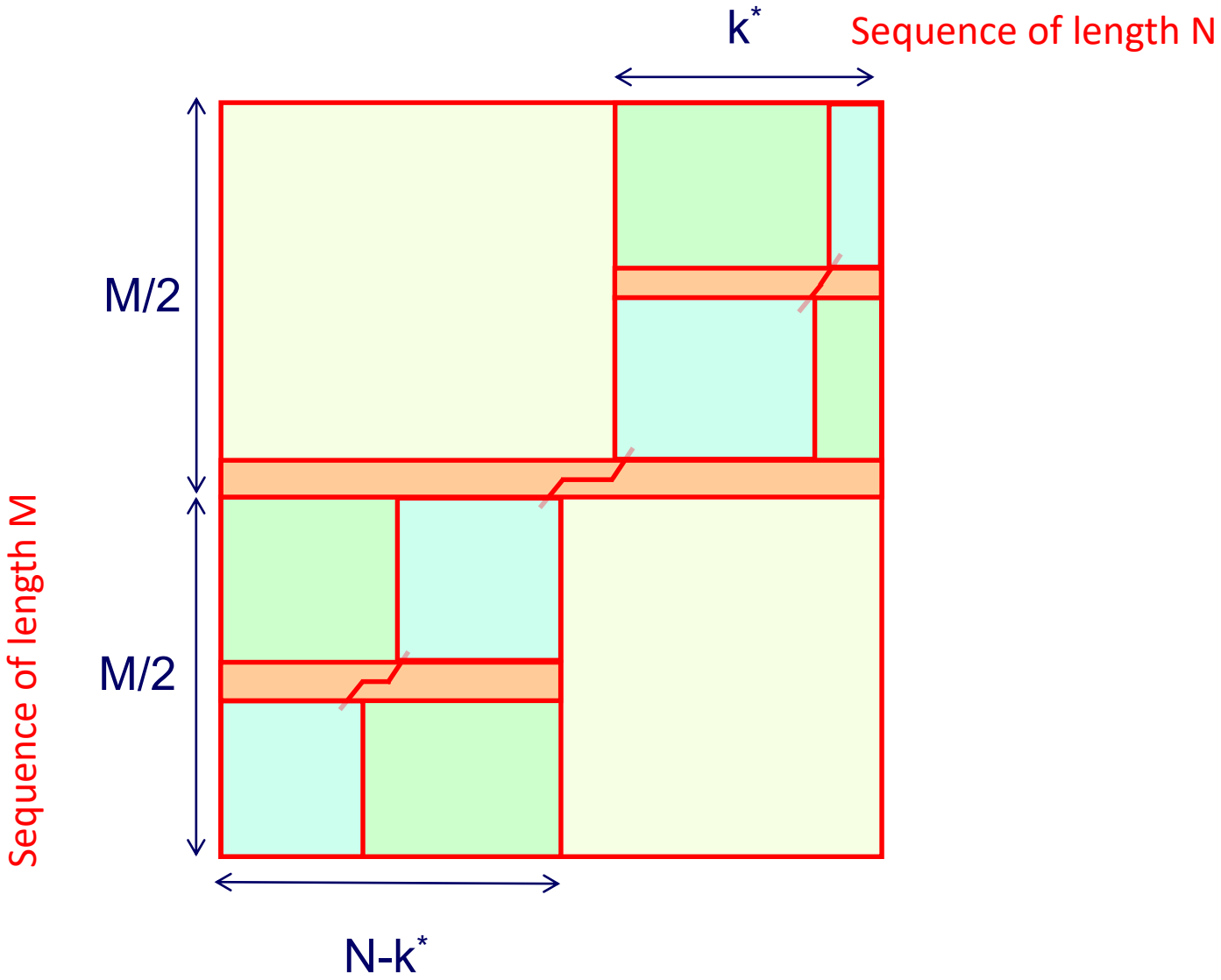
Path through scoring matrix

B

	S	j	A	A	G	G	T	A	T	G	A	A	T	C
	i	0	1	2	3	4	5	6	7	8	9	10	11	12
A	A	1												
	A	2												
	C	3												
	G	4												
	T	5												
	T	6												
	G	7												
	A	8												

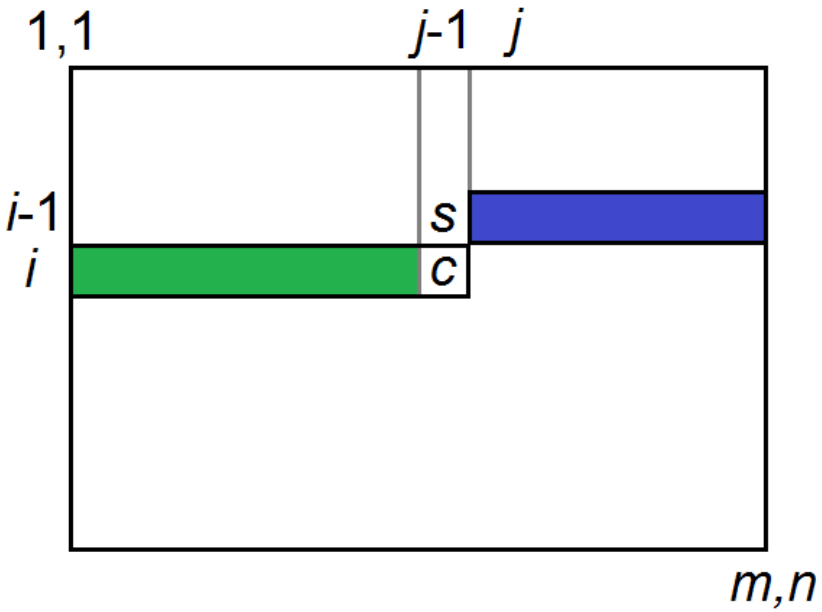
Finding the actual alignment is equivalent to finding all cells that optimal path passes through

Hirschberg's algorithm



Hirschberg's algorithm

- for global alignment of two sequnces
- uses divide and conquer, dynamic programming and linear-space



Complexity

Algorithm	**Time complexity**	**Space complexity**
Needleman-Wunsch	$O(m \cdot n)$	$O(m \cdot n)$
Hirschberg	$O(m \cdot n)$	$O(m + n)$

Example

- sequence 1: AGTACGCA
- sequence 2: TATGC
- match = 2
- mismatch = -1
- gap = -2

Example

- sequence 1: TACGAGGCA
- sequence 2: ACGGA
- match = 3
- mismatch = 1
- gap = -3

Task

- Implement Hirschberg's algorithm