Course organization

- Introduction (Week 1-2)
  - Course introduction
  - A brief introduction to molecular biology
  - A brief introduction to sequence comparison
- Part I: Algorithms for Sequence Analysis (Week 3 - 11)
  - Chapter 1-3, Models and theories
    - » Probability theory and Statistics (Week 4)
    - » Algorithm complexity analysis (Week 5)
    - » Classic algorithms (Week 6)
    - » Lab: Linux and Perl
  - Chapter 4, Sequence alignment (week 7)
  - Chapter 5, Hidden Markov Models (week 8)
  - Chapter 6. Multiple sequence alignment (week 10)
  - Chapter 7. Motif finding (week 11)
  - Chapter 8. Sequence binning (week 11)
- Part II: Algorithms for Network Biology (Week 12 - 16)

Chapter 3: Dynamic Programming

Chaochun Wei

Fall 2014

Contents

- Reading materials
- Introduction
  - Dynamic programming
  - Greedy algorithm

Reading

Cormen book:

Thomas, H. ,Cormen, Charles, E., Leiserson, and Ronald, L., Rivest . Introduction to Algorithms, The MIT Press.

(read Chapter 16 and 17, page 299-355).

Dynamic programming

- Find an optimal solution to a problem
- Four steps to develop a dynamic programming algorithm
  1. Characterize the structure of an optimal solution
  2. Recursive formula for an optimal solution
  3. Compute the value of an optimal solution
  4. Construct an optimal solution from the computed information

Elements of dynamic programming

- Two elements are required
  1. Optimal substructure
     - An optimal solution contains within it optimal solutions to the subproblems
  2. Overlapping subproblems
     - Recursive formula exists

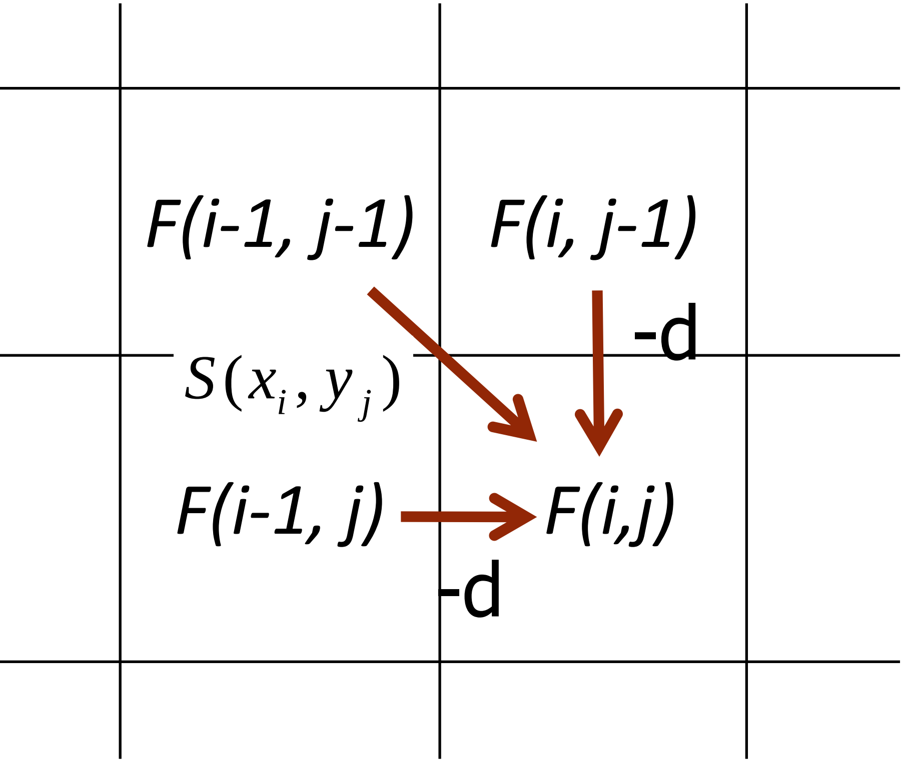
Needleman/Wunsch global alignment (1970)

- Two sequences $X = x_1 \ldots x_n$ and $Y = y_1 \ldots y_m$
- Let $F(i, j)$ be the optimal alignment score *of* $X_{1\ldots i}$ of X up to x_i and $Y_{1\ldots j}$ of Y up to Y_j *($0 \le i \le n$, $0 \le j \le m$), then we have*

$$F(0,0) = 0$$

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases}$$

Needleman/Wunsch global alignment (1970)



$$F(0,0) = 0$$

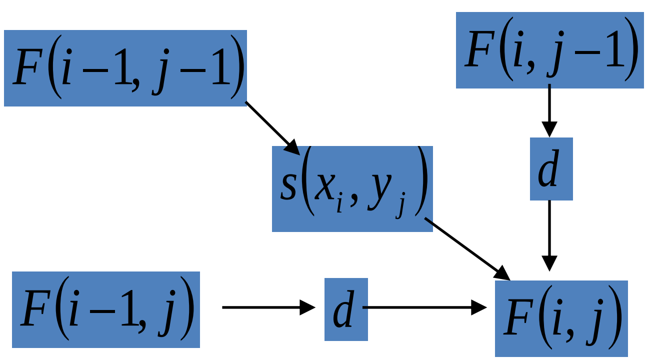
$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases}$$

A simple example

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Find the optimal alignment of AAG and AGC.
Use a gap penalty of d=-5.

		A	A	G
A				
G				
C				

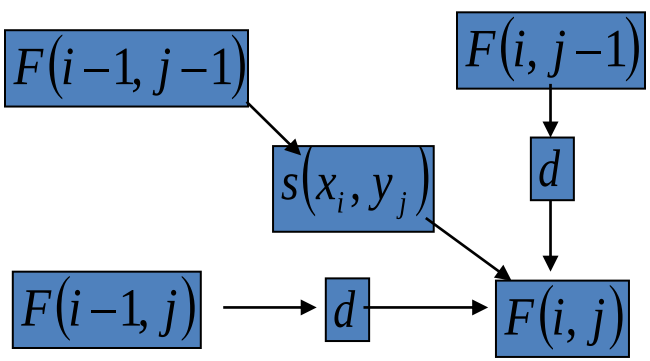


A simple example

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Find the optimal alignment of AAG and AGC.
Use a gap penalty of d=-5.

		A	A	G
	0			
A				
G				
C				

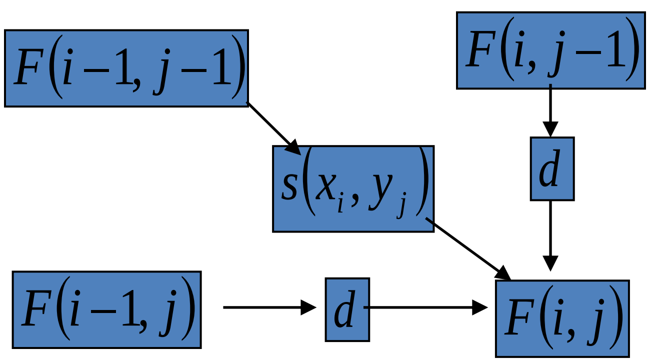


A simple example

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Find the optimal alignment of AAG and AGC.
Use a gap penalty of d=-5.

		A	A	G
	0 →	-5 →	-10 →	-15
A	↓ -5			
G	↓ -10			
C	↓ -15			

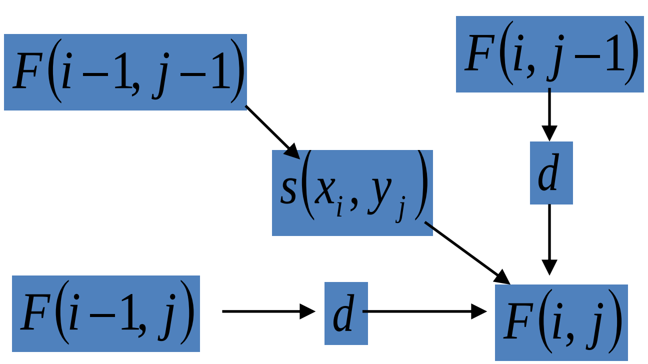


A simple example

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Find the optimal alignment of AAG and AGC.
Use a gap penalty of d=-5.

		A	A	G
	0	-5	-10	-15
A	-5	2	-3	-8
G	-10	-3	-3	-1
C	-15	-8	-8	**-6**



Traceback

1. Start from the lower right corner and trace back to the upper left.
2. Each arrow introduces one character at the end of each aligned sequence.
3. A <u>horizontal</u> move puts a gap in the <u>left</u> sequence.
4. A <u>vertical</u> move puts a gap in the <u>top</u> sequence.
5. A diagonal move uses one character from each sequence.

A simple example

1. Start from the lower right corner and trace back to the upper left.
2. Each arrow introduces one character at the end of each aligned sequence.
3. A horizontal move puts a gap in the left sequence.
4. A vertical move puts a gap in the top sequence.
5. A diagonal move uses one character from each sequence.

Find the optimal alignment of AAG and AGC.
Use a gap penalty of d=-5.

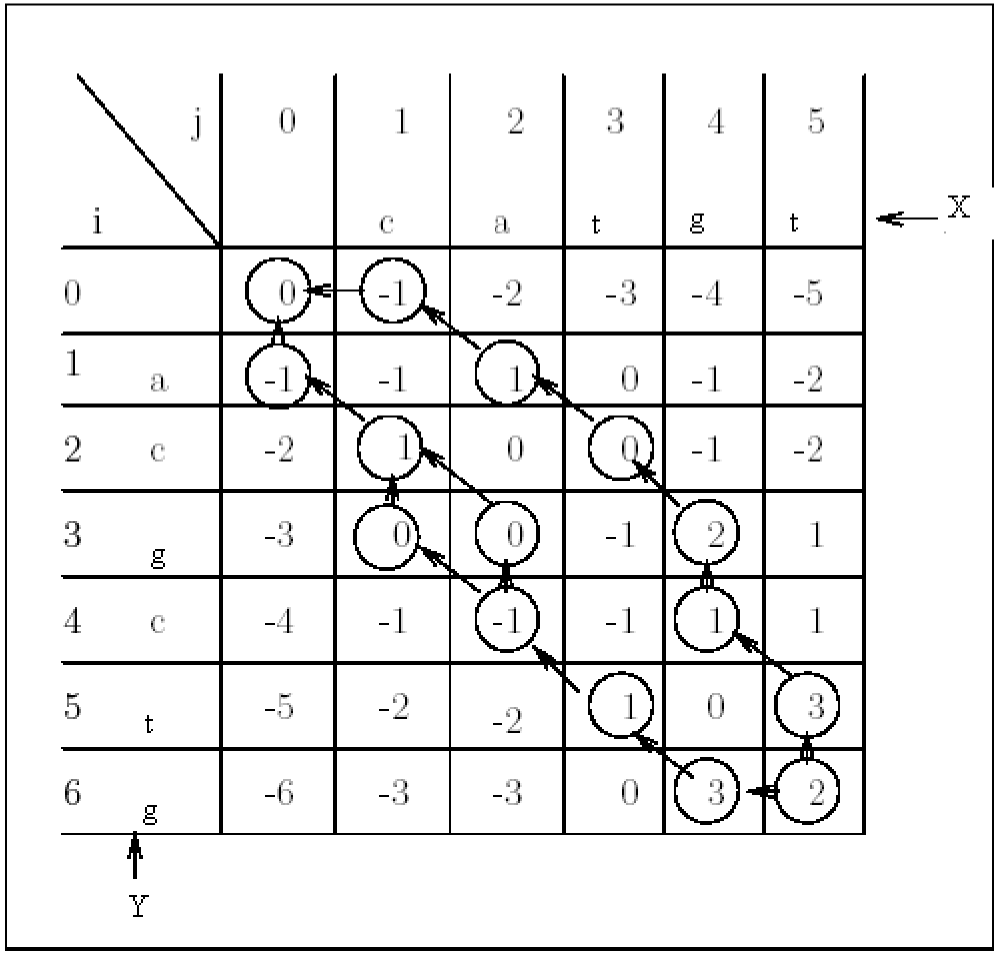
		A	A	G
	0	-5		
A		2	-3	
G				-1
C				**-6**

```
AAG-     AAG-
-AGC     A-GC
```

Exercise

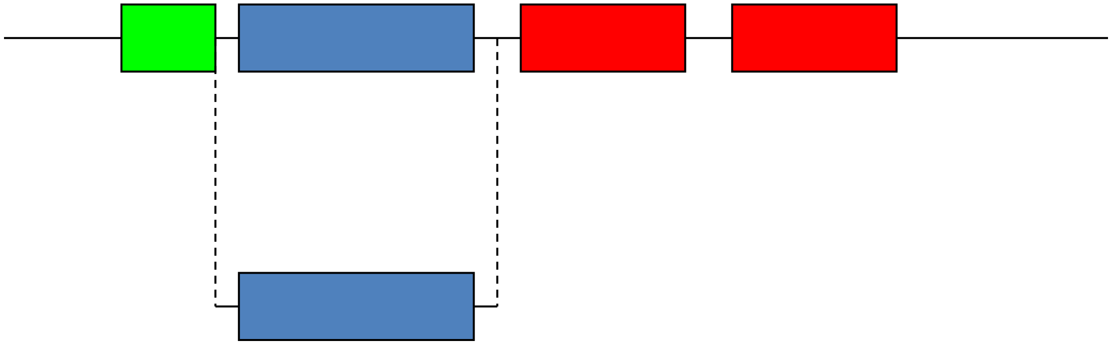
- Find Global alignment
  - X=catgt
  - Y=acgctg
  - Score: d=-1 mismatch=-1 match=2

Answer



j 0 1 2 3 4 5
i c a t g t
X
0 0 -1 -2 -3 -4 -5
1 a -1 -1 1 0 -1 -2
2 c -2 1 0 0 -1 -2
3 g -3 0 0 -1 2 1
4 c -4 -1 -1 -1 1 1
5 t -5 -2 -2 1 0 3
6 g -6 -3 -3 0 3 2
Y

Local alignment



- A single-domain protein may be homologous to a region within a multi-domain protein.
- Usually, an alignment that spans the complete length of both sequences is not required.

Smith/Waterman local alignment (1981)

- Two sequences $X = x_1 \dots x_n$ and $Y = y_1 \dots y_m$
- Let $F(i, j)$ be the optimal alignment score *of* $X_{1\dots i}$ of X up to x_i and $Y_{1\dots j}$ of Y up to Y_j *(*$0 \le i \le n$, $0 \le j \le m$*), then we have*

$$F(0,0) = 0$$

$$F(i, j) = \max \begin{cases} 0 \\ F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) - d \\ F(i, j-1) - d \end{cases}$$

Local alignment

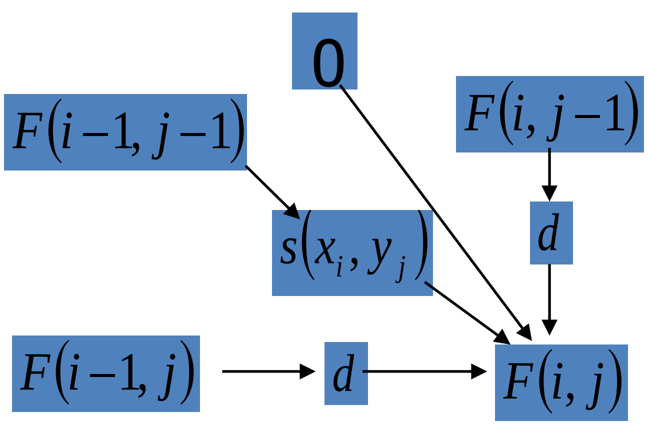
- Two differences with respect to global alignment:
  - No score is negative.
  - Traceback begins at the highest score in the matrix and continues until you reach 0.
- Global alignment algorithm: *Needleman-Wunsch*.
- Local alignment algorithm: *Smith-Waterman*.

A simple example

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Find the optimal local alignment of AAG and AGC.
Use a gap penalty of d=-5.

		A	A	G
A				
G				
C				

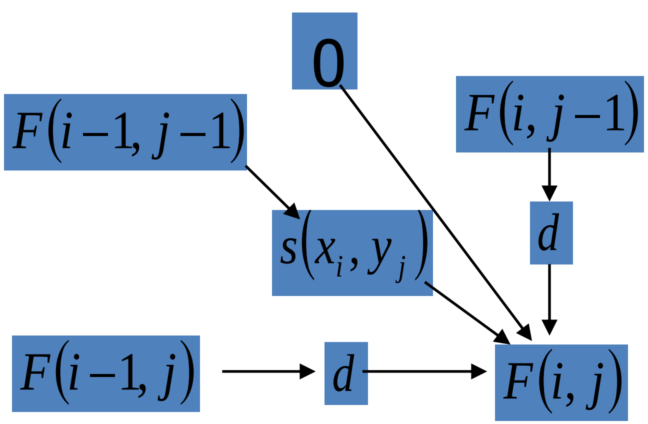


A simple example

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Find the optimal local alignment of AAG and AGC.
Use a gap penalty of d=-5.

		A	A	G
	0	0	0	0
A	0			
G	0			
C	0			

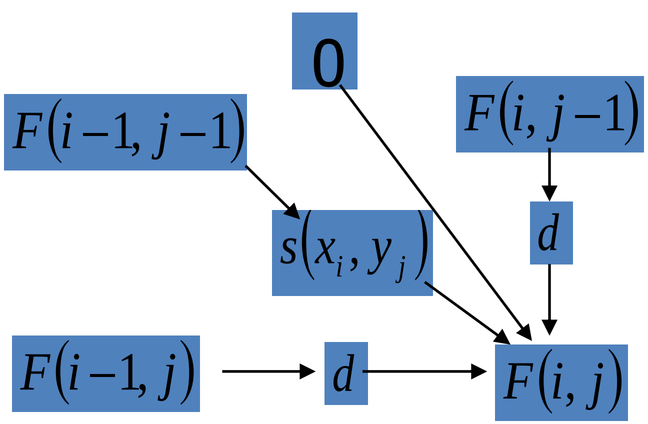


A simple example

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Find the optimal local alignment of AAG and AGC.
Use a gap penalty of d=-5.

		A	A	G
	0	0	0	0
A	0	2	2	0
G	0	0	0	4
C	0	0	0	0

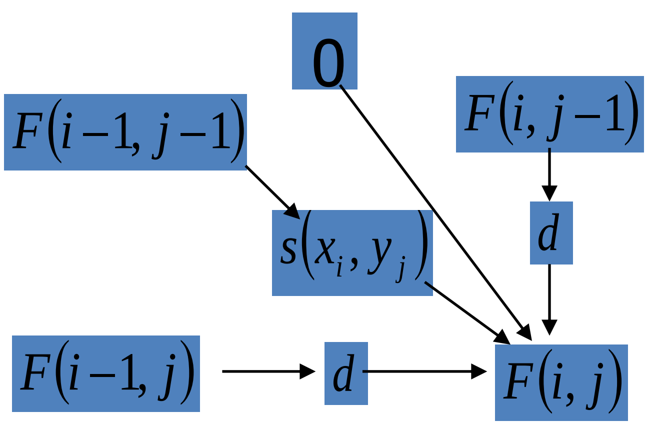


A simple example

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Find the optimal local alignment of AAG and AGC.
Use a gap penalty of d=-5.

		A	A	G
	0	0	0	0
A	0	2	2	0
G	0	0	0	4
C	0	0	0	0



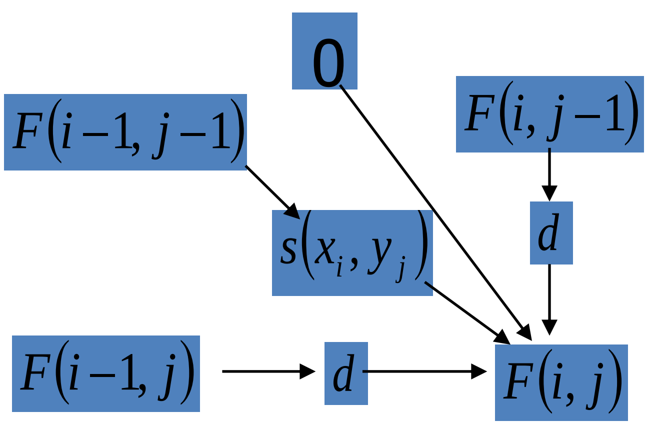
AG
AG

Local alignment

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Find the optimal local alignment of AAG and GAAGGC.
Use a gap penalty of d=-5.

		A	A	G
	0	0	0	0
G	0			
A	0			
A	0			
G	0			
G	0			
C	0			

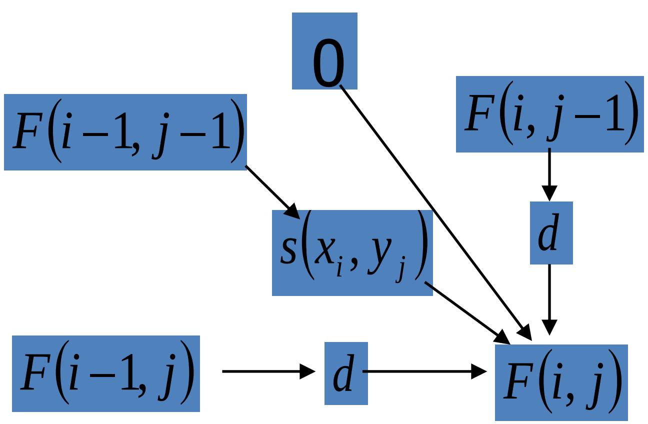


Local alignment

	A	C	G	T
A	2	-7	-5	-7
C	-7	2	-7	-5
G	-5	-7	2	-7
T	-7	-5	-7	2

Find the optimal local alignment of AAG and GAAGGC.
Use a gap penalty of d=-5.

		A	A	G
	0	0	0	0
G	0	0	0	2
A	0	2	2	0
A	0	2	4	0
G	0	0	0	6
G	0	0	0	2
C	0	0	0	0



Greedy algorithm: Choose the best at the moment

- Not always produce the optimal result
- Two elements are required to find an optimal solution by greedy algorithm
  1. Greedy-choice property
     - Global optimal can be reached by local optimal (greedy)
  2. Optimal substructure
     - An optimal solution contains within it optimal solutions to the subproblems

Greedy Algorithm

- Example: Activity-selection problem
  - N activities: S={1, 2, …, N}. Only one can be held at a time. Select the maximum number of mutually compatible activities

Let s_i and f_i be the start time and finish time for activity i.
Assume $f_1 \le f_2 \le \ldots \le f_N$

```
GREEDY_ACTIVITY_SELECTION(s,f)
N ← length[s]
A← {1}
J← 1
for i ← 2 to N
        do if  s_i>= f_j
            then A← A U {i}
                  j← i
Return A
```

Acknowledgement

PPTs for examples in dynamic programming are kindly provided by Dr. Qi Liu.