COL 351 : Analysis & Design of Algorithms

LECTURE 26

DYNAMIC PROGRAMMING IV :

SEQUENCE ALIGNMENT (CONTD.) & BELLMAN FORD ALGORITHM

OCT 04, 2024 | ROHIT VAISH

SEQUENCE ALIGNMENT

input : strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

SEQUENCE ALIGNMENT

input : strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

output : an alignment of X and Y with minimum total penalty.

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

output: an alignment of X and Y with minimum total penalty.

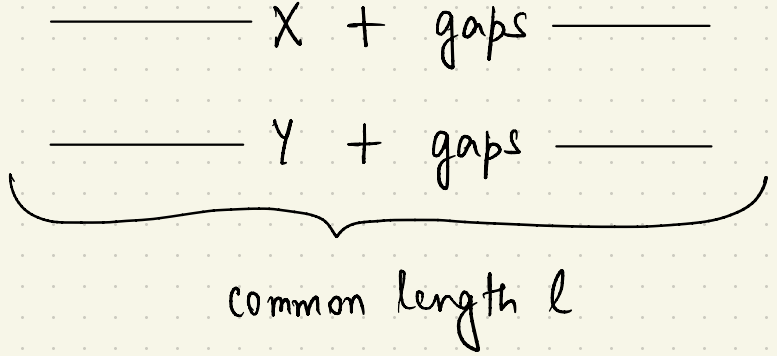
a way of inserting gaps into one or both strings so that they have equal length

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

output: an alignment of X and Y with minimum total penalty.

alignment: a way of inserting gaps into one or both strings so that they have equal length



SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

output: an alignment of X and Y with minimum total penalty.

a way of inserting gaps into one or both strings so that they have equal length

——— X + gaps ———

——— Y + gaps ———

common length l

e.g.,

O_currance

OCCurrence

SEQUENCE ALIGNMENT

input : strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

output : an alignment of X and Y with minimum total penalty.

$$X = A\ G\ G\ C\ T$$

$$Y = A\ C\ G\ G\ C\ C$$

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

output: an alignment of X and Y with minimum total penalty.

$$X = A\ G\ G\ C\ T$$

$$Y = A\ C\ G\ G\ C\ C$$

Which is better:

$$\begin{matrix} A & G & G & C & T & \_ \\ A & C & G & G & C & C \end{matrix} \quad \text{or} \quad \begin{matrix} A & \_ & G & G & C & \_ & T \\ A & C & G & G & C & C & \_ \end{matrix} \quad ?$$

SEQUENCE ALIGNMENT

input : strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

output : an alignment of X and Y with minimum total penalty.

Gap penalty α_{gap}

Mismatch penalty α_{xy}

where $x, y \in \Sigma$

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

output: an alignment of X and Y with minimum total penalty.

Gap penalty $\alpha_{gap} \geq 0$ Why?

Mismatch penalty α_{xy}

where $x, y \in \Sigma$

SEQUENCE ALIGNMENT

input : strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

output : an alignment of X and Y with minimum total penalty.

Gap penalty $\alpha_{gap} \geq 0$

Mismatch penalty α_{xy}

where $x, y \in \Sigma$

c _ a _ t

_ d _ o _ g

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Sum of all gap and mismatch penalties

Needleman - Wunsch (NW) score

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X = A\ G\ T\ A\ C\ G$

$Y = A\ C\ A\ T\ A\ G$

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X = A\ G\ T\ A\ C\ G$

$Y = A\ C\ A\ T\ A\ G$

NW score = ?

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X =$ A G T A C G

$Y =$ A C A T A G

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X =$ A G T A C G

$Y =$ A C A T A G

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X =$ A G T A C G

$Y =$ A C A T A G

No gaps $\Rightarrow$ penalty = 8

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X =$ A G T A C G

$Y =$ A C A T A G

A _ _ G T A C G

A C A _ T A _ G

Four gaps $\Rightarrow$ penalty $= 4$

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

Is penalty ≤ 3 possible?

$X =$ A G T A C G

$Y =$ A C A T A G

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

Is penalty ≤ 3 possible? No!

$X =$ A G T A C G

$Y =$ A C A T A G

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X =$ A G T A C G

$Y =$ A C A T A G

Is penalty ≤ 3 possible? No!

Strings of equal length $\Rightarrow$ #gaps is even

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X =$ A G T A C G

$Y =$ A C A T A G

Is penalty ≤ 3 possible? No!

Strings of equal length $\Rightarrow$ # gaps is even

If # gaps $= 0$, then # mismatches $= 4$

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ
a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X =$ A G T A C G

$Y =$ A C A T A G

Is penalty ≤ 3 possible? No!

Strings of equal length $\Rightarrow$ # gaps is even

If # gaps $= 0$, then # mismatches $= 4$

If # gaps $= 4$, then penalty ≥ 4

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X =$ A G T A C G

$Y =$ A C A T A G

Is penalty ≤ 3 possible? No!

Strings of equal length $\Rightarrow$ # gaps is even

If # gaps $= 0$, then # mismatches $= 4$

If # gaps $= 4$, then penalty ≥ 4

If # gaps $= 2$, then # mismatches ≥ 1.

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

Example: $\alpha_{gap} = 1$ $\alpha_{xy} = 2$ $\forall x, y \in \Sigma$

$X =$ A G T A C G

$Y =$ A C A T A G

NW score = 4.

SEQUENCE ALIGNMENT

input: strings $X = x_1 x_2 \cdots x_m$ $Y = y_1 y_2 \cdots y_n$ over alphabet Σ

a gap penalty $\alpha_{gap} \geq 0$, a mismatch penalty α_{xy} $\forall x, y \in \Sigma$

output: an alignment of X and Y with minimum total penalty.

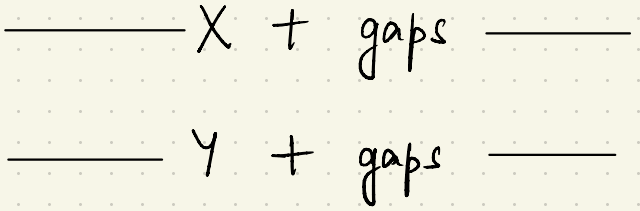
(Exercise)

alignments between X and Y: exponential in $(n + m)$.

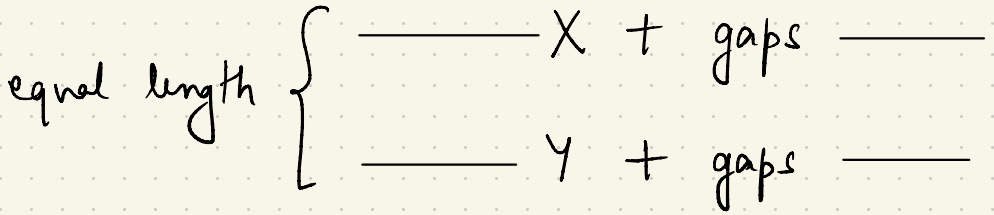
Brute force is prohibitive ☹️

OPTIMAL SUBSTRUCTURE

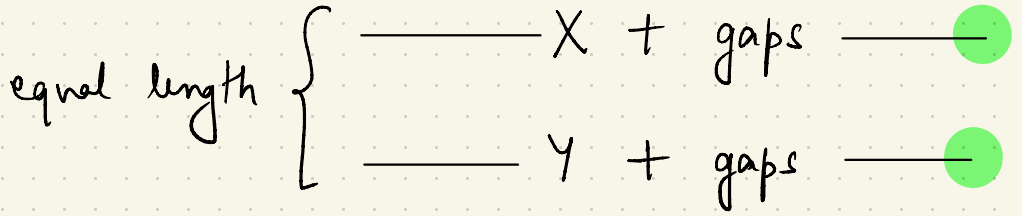
OPTIMAL SUBSTRUCTURE



OPTIMAL SUBSTRUCTURE

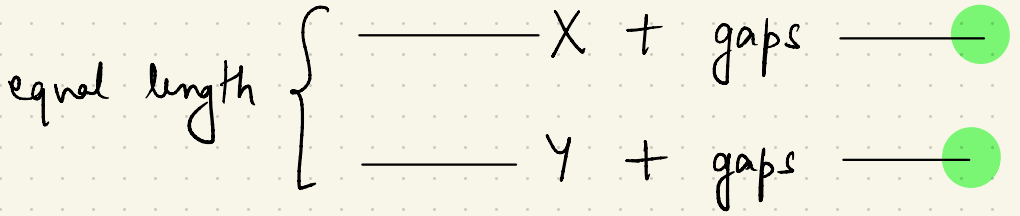


OPTIMAL SUBSTRUCTURE



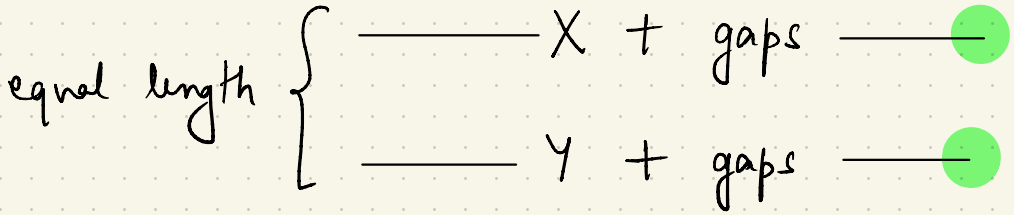
OPTIMAL SUBSTRUCTURE

How many relevant cases ?



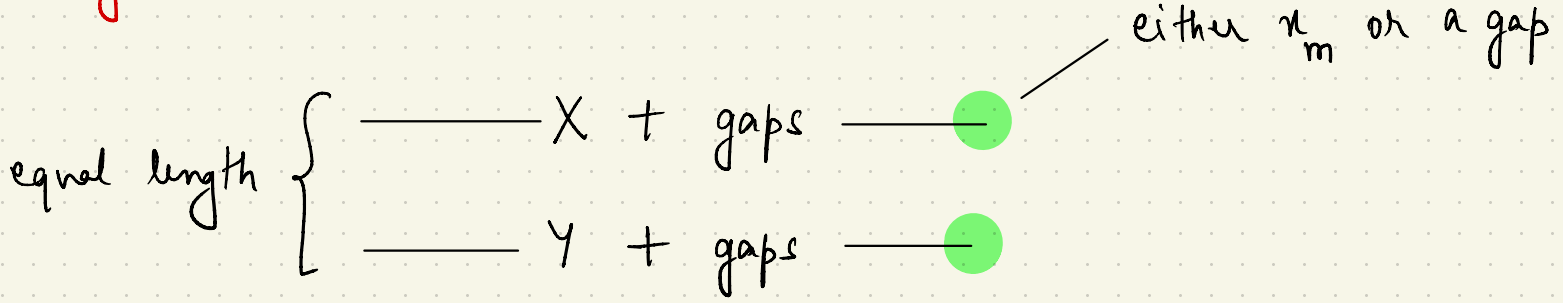
OPTIMAL SUBSTRUCTURE

How many relevant cases ? Three



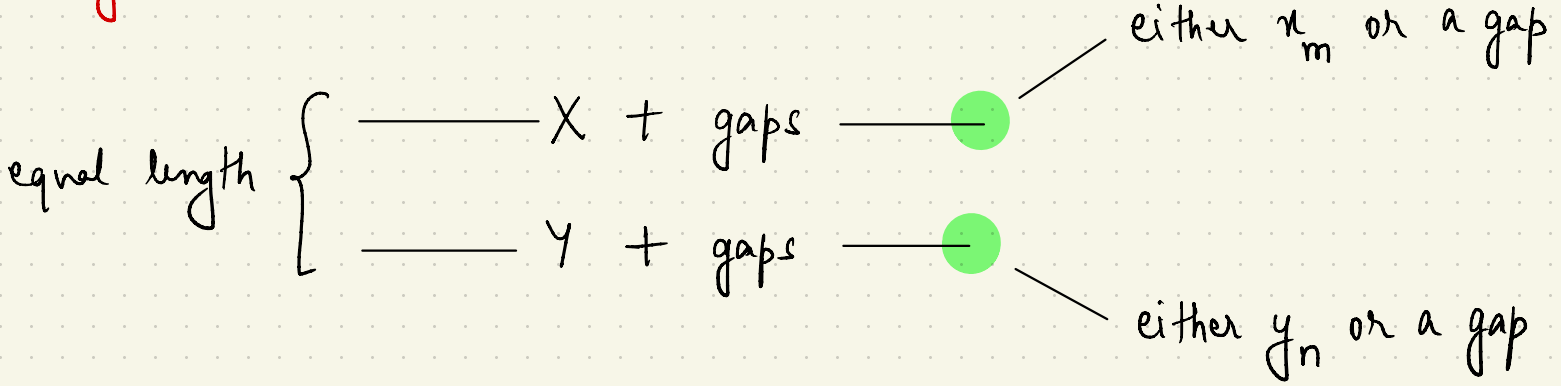
OPTIMAL SUBSTRUCTURE

How many relevant cases ? Three



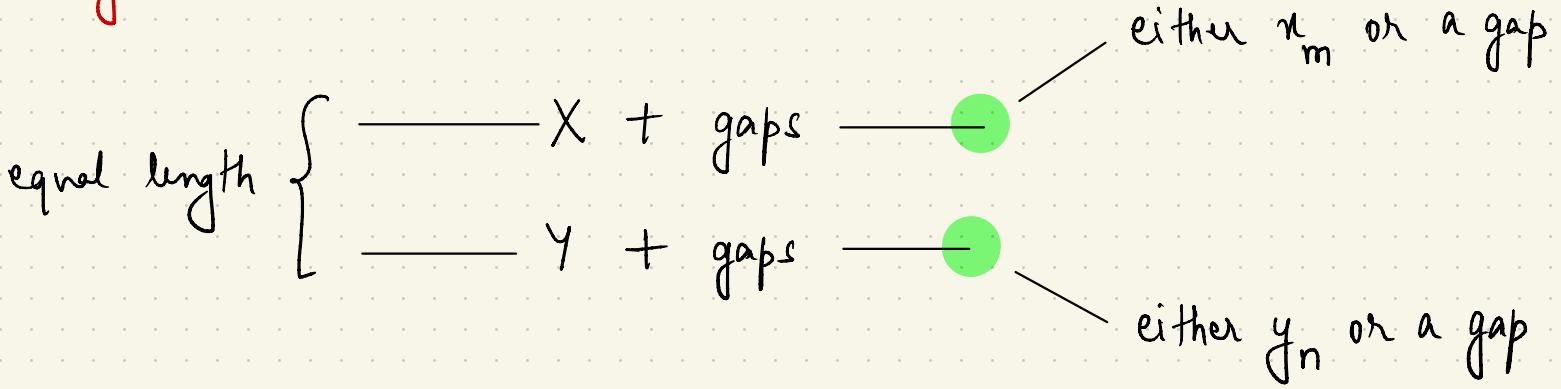
OPTIMAL SUBSTRUCTURE

How many relevant cases ? Three



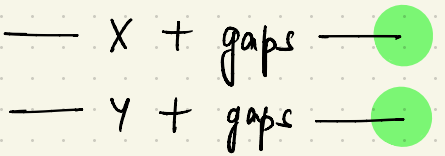
OPTIMAL SUBSTRUCTURE

How many relevant cases ? Three



$\alpha_{gap} \geq 0 \Rightarrow$ both can't be gaps !

OPTIMAL SUBSTRUCTURE



OPTIMAL SUBSTRUCTURE

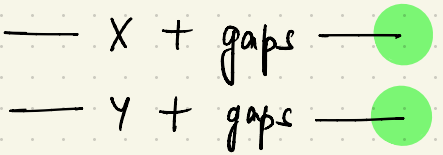
Let $X' := X - x_m$ $\quad Y' := Y - y_n$

— X + gaps —

— Y + gaps —

OPTIMAL SUBSTRUCTURE

Let $X' := X - x_m$ $\quad Y' := Y - y_n$



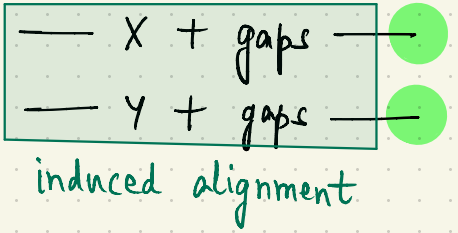
Case I: Final characters are x_m and y_n

Case II: Final characters are x_m and gap

Case III: Final characters are gap and y_n

OPTIMAL SUBSTRUCTURE

Let $X' := X - x_m$ $\quad Y' := Y - y_n$



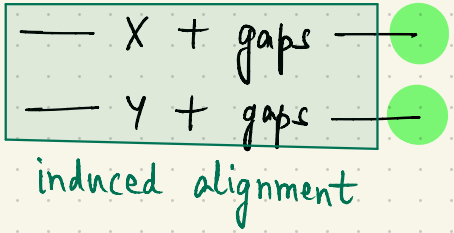
Case I: Final characters are x_m and y_n

Case II: Final characters are x_m and gap

Case III: Final characters are gap and y_n

OPTIMAL SUBSTRUCTURE

Let $X' := X - x_m$ $\quad Y' := Y - y_n$



Case I: Final characters are x_m and y_n

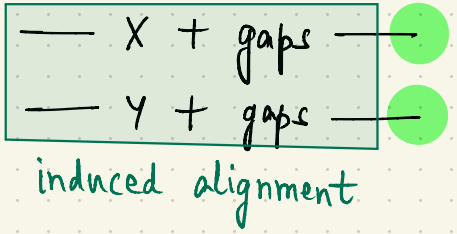
$\Rightarrow$ induced alignment of X' and Y' is optimal.

Case II: Final characters are x_m and gap

Case III: Final characters are gap and y_n

OPTIMAL SUBSTRUCTURE

Let $X' := X - x_m$ $\quad Y' := Y - y_n$



Case I: Final characters are x_m and y_n

$\Rightarrow$ induced alignment of X' and Y' is optimal.

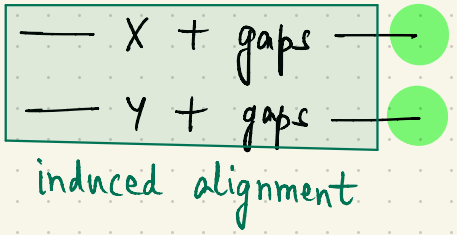
Case II: Final characters are x_m and gap

$\Rightarrow$ induced alignment of X' and Y is optimal.

Case III: Final characters are gap and y_n

OPTIMAL SUBSTRUCTURE

Let $X' := X - x_m$ $\quad Y' := Y - y_n$



Case I: Final characters are x_m and y_n

$\Rightarrow$ induced alignment of X' and Y' is optimal.

Case II: Final characters are x_m and gap

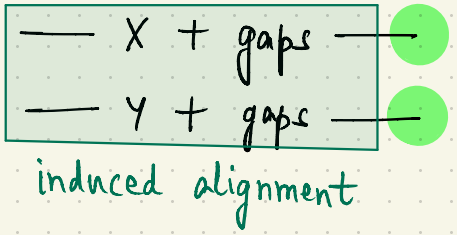
$\Rightarrow$ induced alignment of X' and Y is optimal.

Case III: Final characters are gap and y_n

$\Rightarrow$ induced alignment of X and Y' is optimal.

OPTIMAL SUBSTRUCTURE

Let $X' := X - x_m$ $\quad Y' := Y - y_n$



Case I : Final characters are x_m and y_n

$\Rightarrow$ induced alignment of X' and Y' is optimal.

Case II : Final characters are x_m and gap

$\Rightarrow$ induced alignment of X' and Y is optimal.

Case III : Final characters are gap and y_n

$\Rightarrow$ induced alignment of X and Y' is optimal.

Exercise : Prove formally.

Relevant Subproblems

Subproblems are two-dimensional (like knapsack)

X_i := first i letters of X

Y_j := first j letters of Y

The Recurrence

The Recurrence

Define $P_{i,j}$:= penalty of optimal alignment of X_i and Y_j

THE RECURRENCE

Define $P_{i,j}$:= penalty of optimal alignment of X_i and Y_j

For all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$

THE RECURRENCE

Define $P_{i,j}$:= penalty of optimal alignment of X_i and Y_j

For all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$

$P_{i,j} =$

// case I

// case II

// case III

The Recurrence

Define $P_{i,j}$:= penalty of optimal alignment of X_i and Y_j

For all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$

$$P_{i,j} = \begin{cases} \alpha_{x_i y_j} + P_{(i-1),(j-1)} & \text{// case I} \\ & \text{// case II} \\ & \text{// case III} \end{cases}$$

THE RECURRENCE

Define $P_{i,j}$:= penalty of optimal alignment of X_i and Y_j

For all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$

$$P_{i,j} = \begin{cases} \alpha_{x_i y_j} + P_{(i-1),(j-1)} & \text{// case I} \\ \alpha_{gap} + P_{(i-1),j} & \text{// case II} \\ & \text{// case III} \end{cases}$$

THE RECURRENCE

Define $P_{i,j}$:= penalty of optimal alignment of X_i and Y_j

For all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$

$$P_{i,j} = \begin{cases} \alpha_{x_i y_j} + P_{(i-1),(j-1)} & \text{// case I} \\ \alpha_{gap} + P_{(i-1),j} & \text{// case II} \\ \alpha_{gap} + P_{i,(j-1)} & \text{// case III} \end{cases}$$

THE RECURRENCE

Define $P_{i,j}$:= penalty of optimal alignment of X_i and Y_j

For all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$

$$P_{i,j} = \min \begin{cases} \alpha_{x_i y_j} + P_{(i-1),(j-1)} & \text{// case I} \\ \alpha_{gap} + P_{(i-1),j} & \text{// case II} \\ \alpha_{gap} + P_{i,(j-1)} & \text{// case III} \end{cases}$$

The Recurrence

Define $P_{i,j}$:= penalty of optimal alignment of x_i and y_j

For all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$

$$P_{i,j} = \min \begin{cases} \alpha_{x_i y_j} + P_{(i-1),(j-1)} & \text{// case I} \\ \alpha_{gap} + P_{(i-1),j} & \text{// case II} \\ \alpha_{gap} + P_{i,(j-1)} & \text{// case III} \end{cases}$$

Base cases: $P_{i,0} := ?$

The Recurrence

Define $P_{i,j}$:= penalty of optimal alignment of X_i and Y_j

For all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$

$$P_{i,j} = \min \begin{cases} \alpha_{x_i y_j} + P_{(i-1),(j-1)} & \text{// case I} \\ \alpha_{gap} + P_{(i-1),j} & \text{// case II} \\ \alpha_{gap} + P_{i,(j-1)} & \text{// case III} \end{cases}$$

Base cases: $P_{i,0} := i \cdot \alpha_{gap}$

THE RECURRENCE

Define $P_{i,j}$:= penalty of optimal alignment of X_i and Y_j

For all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$

$$P_{i,j} = \min \begin{cases} \alpha_{x_i y_j} + P_{(i-1),(j-1)} & \text{// case I} \\ \alpha_{gap} + P_{(i-1),j} & \text{// case II} \\ \alpha_{gap} + P_{i,(j-1)} & \text{// case III} \end{cases}$$

Base cases: $P_{i,0} := i \cdot \alpha_{gap}$ $\quad P_{0,j} := j \cdot \alpha_{gap}$

THE ALGORITHM

THE ALGORITHM

A = two-dimensional array of size $(m+1) \times (n+1)$

THE ALGORITHM

A = two-dimensional array of size $(m+1) \times (n+1)$

// base cases

$A[i, 0] = i \cdot \alpha_{gap} \quad \forall\ i$

$A[0, j] = j \cdot \alpha_{gap} \quad \forall\ j$

THE ALGORITHM

A = two-dimensional array of size $(m+1) \times (n+1)$

// base cases

$A[i,0] = i \cdot \alpha_{gap} \quad \forall\ i$

$A[0,j] = j \cdot \alpha_{gap} \quad \forall\ j$

// main loop

for $i = 1, 2, \cdots, m$

    for $j = 1, 2, \cdots, n$

THE ALGORITHM

A = two-dimensional array of size $(m+1) \times (n+1)$

// base cases

$A[i,0] = i \cdot \alpha_{gap} \quad \forall\ i$

$A[0,j] = j \cdot \alpha_{gap} \quad \forall\ j$

// main loop

for $i = 1, 2, \cdots, m$

  for $j = 1, 2, \cdots, n$

$$A[i,j] = \min \begin{cases} A[i-1, j-1] + \alpha_{x_i, y_j} \\ A[i-1, j] + \alpha_{gap} \\ A[i, j-1] + \alpha_{gap} \end{cases}$$

THE ALGORITHM

Correctness : By induction (Exercise)

THE ALGORITHM

Correctness : By induction (Exercise)

Running time : $O(m \cdot n)$ $O(1)$ work each of $O(mn)$ subproblems

THE ALGORITHM

Correctness : By induction (Exercise)

Running time : $O(m \cdot n)$ $O(1)$ work each of $O(mn)$ subproblems

Reconstruction : Easy Case I, II, III explicitly tells us where a gap is inserted.

REVISITING SINGLE SOURCE SHORTEST PATHS

SINGLE-SOURCE SHORTEST PATH PROBLEM

input: A directed graph $G = (V, E)$, starting vertex s, a non negative length l_e for each edge $e \in E$

output: dist (s, v) for every vertex $v \in V$.

SINGLE-SOURCE SHORTEST PATH PROBLEM

input: A directed graph $G = (V, E)$, starting vertex s, a non negative length l_e for each edge $e \in E$

output: dist (s, v) for every vertex $v \in V$.

length of the shortest path from s to v if $s \rightsquigarrow v$ path exists

∞ otherwise

ON DIJKSTRA'S ALGORITHM

ON DIJKSTRA'S ALGORITHM

👍 $O(m \log n)$ running time with heaps

ON DIJKSTRA'S ALGORITHM

👍 $O(m \log n)$ running time with heaps

👎 not always correct with negative edge lengths

(e.g., if edges $\leftrightarrow$ financial transactions)

ON DIJKSTRA'S ALGORITHM

👍 $O(m \log n)$ running time with heaps

👎 not always correct with negative edge lengths

(e.g., if edges $\leftrightarrow$ financial transactions)

👎 highly centralized

(need distributed algorithm for internet routing)

ON DIJKSTRA'S ALGORITHM

👍 $O(m \log n)$ running time with heaps

👎 not always correct with negative edge lengths

(e.g., if edges $\leftrightarrow$ financial transactions)

👎 highly centralized

(need distributed algorithm for internet routing)

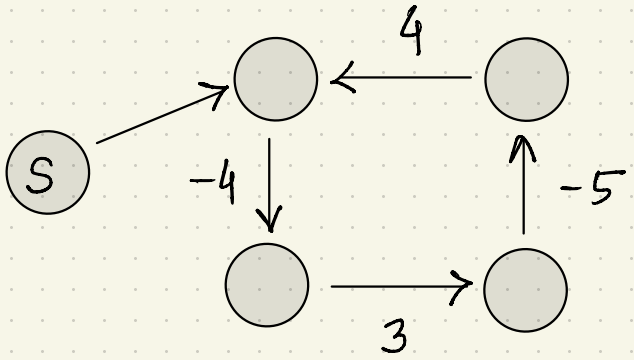
Solution: Bellman-Ford algorithm

NEGATIVE CYCLES

How to define shortest paths in the presence of negative cycles?

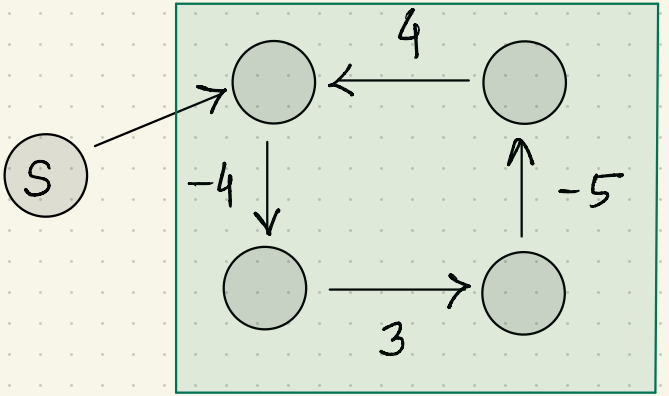
NEGATIVE CYCLES

How to define shortest paths in the presence of negative cycles?



NEGATIVE CYCLES

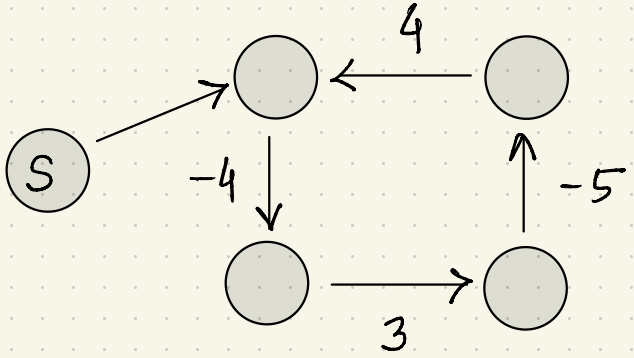
How to define shortest paths in the presence of negative cycles?



cycle with weight -2

NEGATIVE CYCLES

How to define shortest paths in the presence of negative cycles?

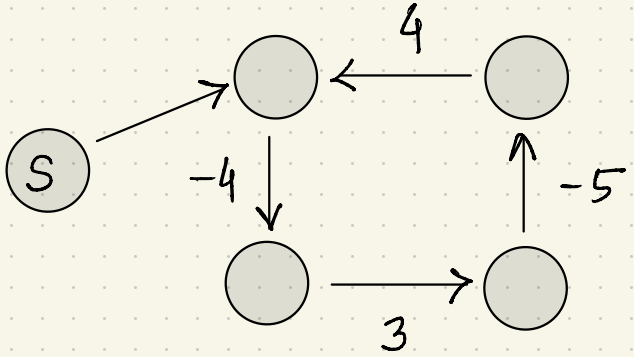


If we include cycles in shortest paths

then length of the shortest path is undefined (or $-\infty$).

NEGATIVE CYCLES

How to define shortest paths in the presence of negative cycles?

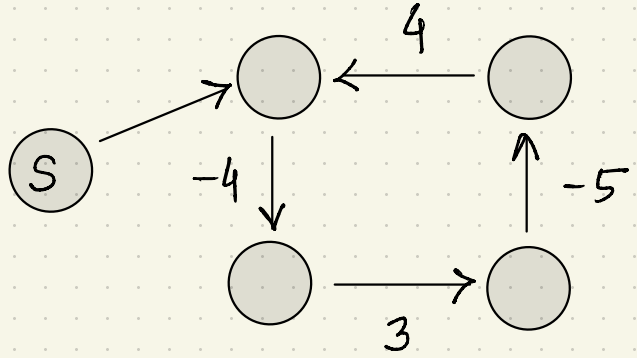


If we require shortest cycle-free path

then the problem is NP-complete (no known poly-time algo.)

NEGATIVE CYCLES

How to define shortest paths in the presence of negative cycles?



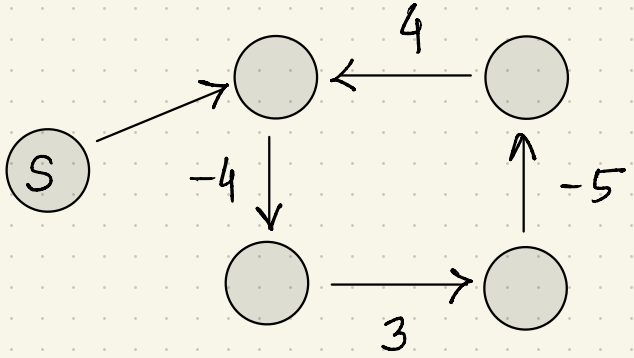
If we require shortest cycle-free path

then the problem is NP-complete (no known poly-time algo.)

encodes "longest path" problem

NEGATIVE CYCLES

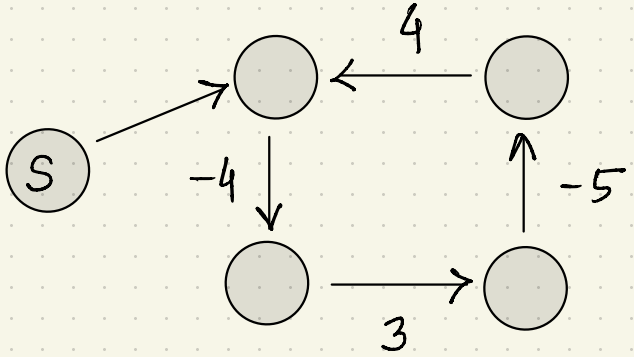
How to define shortest paths in the presence of negative cycles?



Solution: Allow negative length edges but not negative cycles.

NEGATIVE CYCLES

How to define shortest paths in the presence of negative cycles?



Solution: Allow negative length edges but not negative cycles.

Will show how to quickly check this