CS 466
Introduction to Bioinformatics
Lecture 3

Mohammed El-Kebir

September 5, 2018

Course Announcements

Instructor:

- Mohammed El-Kebir (melkebir)
- Office hours: Mondays, 3:15-4:15pm

TA:

- Anusri Pampari (pampari2)
- Office hours: Thursdays, 11:00-11:59am in SC 4105

Piazza: (please sign up)

- https://piazza.com/class#fall2018/cs466

Outline

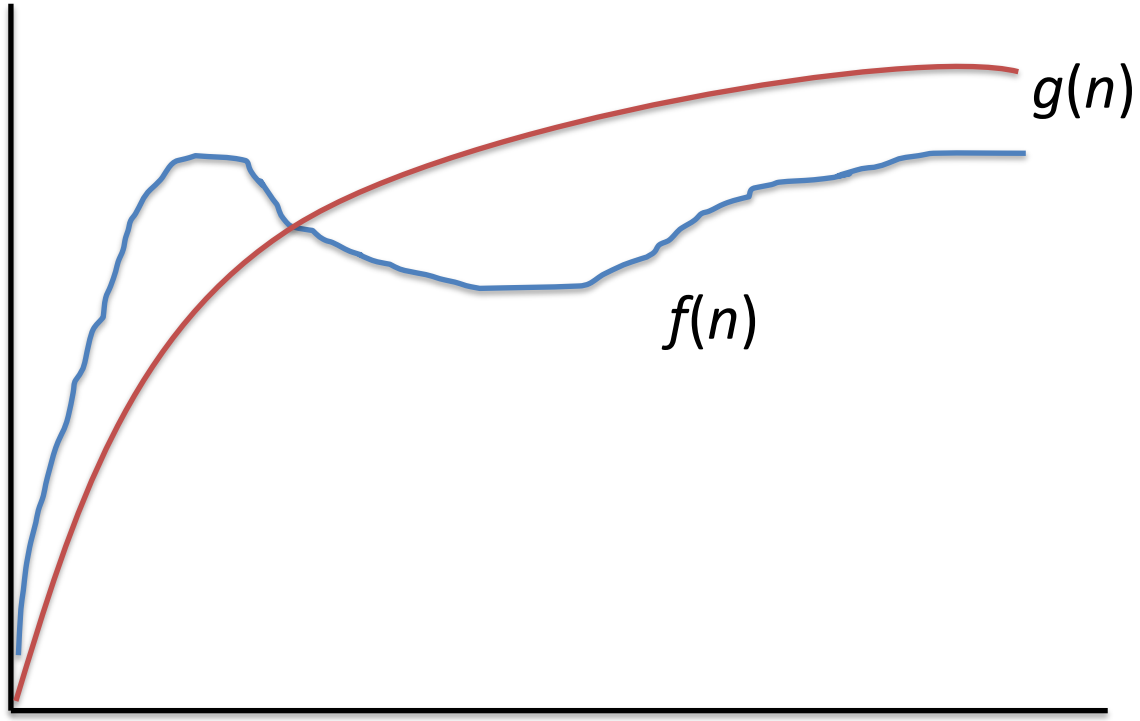
1. Running time recap
2. Edit distance recap
3. Global alignment
4. Fitting alignment
5. Gapped alignment

Reading:

- Jones and Pevzner. Chapters 6.6, 6.7 and 6.9
- Lecture notes on running time

Running Time Analysis

- The **running time** of an algorithm A for problem Π is the maximum number of steps that A will take on any instance of size $n = |X|$
- **Asymptotic running time** ignores constant factors using Big O notation

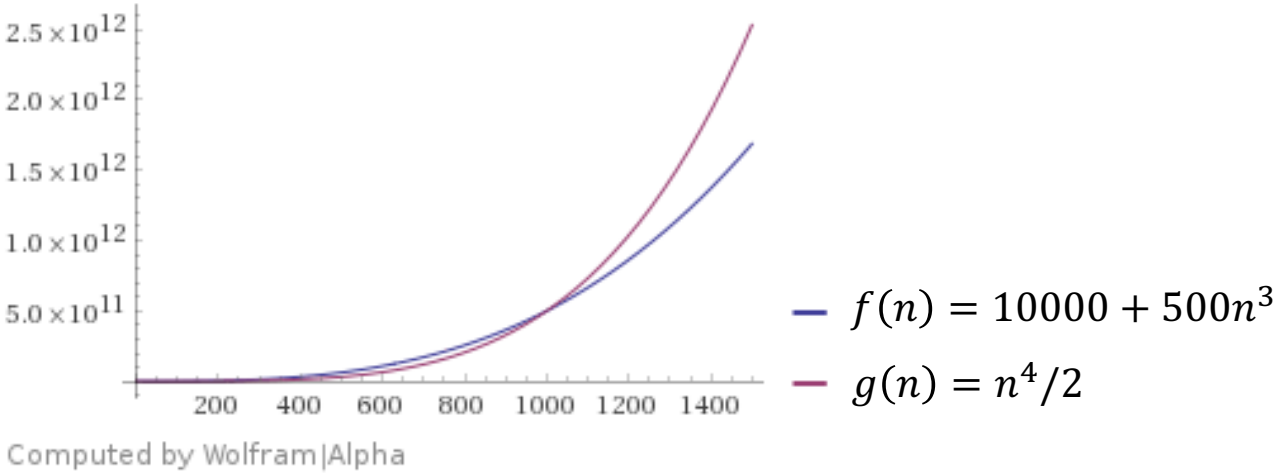


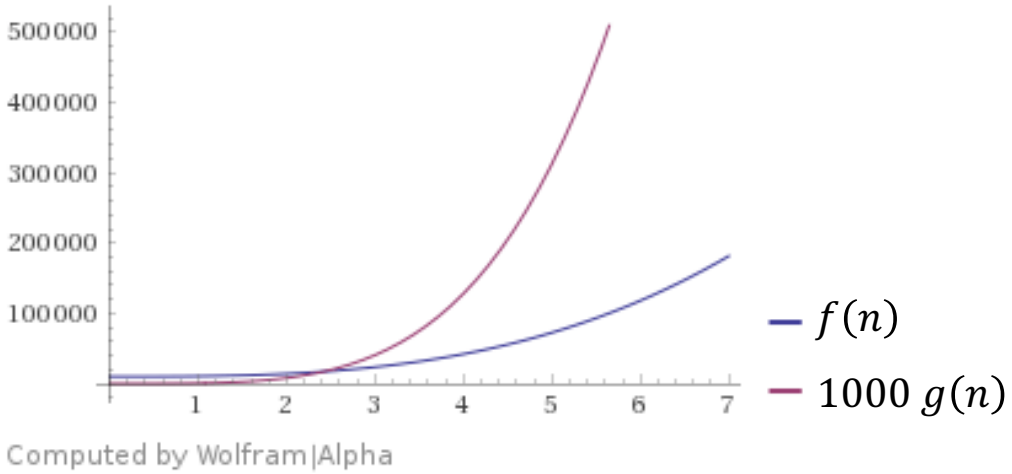
$f(n) = O(g(n))$ provided there exists $c > 0$ and $n_0 \geq 0$ such that $f(n) \leq c\ g(n)$ for all $n \geq n_0$

Note that $O(g(n))$ is a set of functions. Thus, $f(n) = O(g(n))$ actually means $f(n) \in O(g(n))$

Running Time Analysis – Example

$f(n)$ is $O(g(n))$ provided there exists $c > 0$ and $n_0 \geq 0$ such that $f(n) \leq c\ g(n)$ for all $n \geq n_0$





Pick $c = 1000$ and $n_0 = 3$. Then, $f(n) \leq cg(n)$ for all $n \geq n_0$.

Running Time Analysis – Guidelines

- $O(n^a) \subset O(n^b)$ for any positive constants $a < b$
- For any constants $a, b > 0$ and $c > 1$,

  $O(a) \subset O(\log n) \subset O(n^b) \subset O(c^n)$
- We can multiply to learn about other functions. For any constants $a, b > 0$ and $c > 1$,

  $O(an) = O(n) \subset O(n \log n) \subset O\left(n\, n^b\right) = O(n^{b+1}) \subset O(nc^n)$
- Base of the logarithm is a constant and can be ignored. For any constants $a, b > 1$,

  $O(\log_a n) = O(\log_b n / \log_b a) = O(1/(\log_b a) \log_b n) = O(\log_b n)$

Running Time Analysis – Guidelines

- $O(n^a) \subset O(n^b)$ for any positive constants $a < b$

- For any constants $a, b > 0$ and $c > 1$,

  $O(a) \subset O(\log n) \subset O(n^b) \subset O(c^n)$

Big Oh	Name
$O(1)$	Constant
$O(\log n)$	Logarithmic
$O(n)$	Linear
$O(n^2)$	Quadratic
$O(n^c) = O(\text{poly}(n))$	Polynomial
$O(2^{\text{poly}(n)})$	Exponential

- We can multiply to learn about other functions. For any constants $a, b > 0$ and $c > 0$,

  $O(an) = O(n) \subset O(n \log n) \subset O(n\, n^b) = O(n^{b+1}) \subset O(nc^n)$

- Base of the logarithm is a constant and can be ignored. For any constants $a, b > 0$,

  $O(\log_a n) = O(\log_b n / \log_b a) = O(1/(\log_b a) \log_b n) = O(\log_b n)$

Running Time Analysis – More Examples

- Recall that $n! = \prod_{i=1}^{n} i$

Question: What is $O(n!)$?

Running Time Analysis – More Examples

- Recall that $n! = \prod_{i=1}^{n} i$

Question: What is $O(n!)$?

Stirling's approximation: $n! \approx \sqrt{2\pi n}\left(\frac{n}{\mathrm{e}}\right)^n = \sqrt{2\pi}\,\frac{\sqrt{n}}{\exp(n)}\,n^n \overset{(*)}{=} O(n^n) = O(2^{n \log n})$

$(*) : \sqrt{n} \,/\, \exp(n) < 1$ for all $n > 0$

Question: What is $O(\log(n!))$?

Running Time Analysis – More Examples

- Recall that $n! = \prod_{i=1}^{n} i$

Question: What is $O(n!)$?

Stirling's approximation: $n! \approx \sqrt{2\pi n}\left(\frac{n}{\mathrm{e}}\right)^n = \sqrt{2\pi}\,\frac{\sqrt{n}}{\exp(n)}\,n^n \overset{(*)}{=} O(n^n) = O(2^{n \log n})$

$(*) : \sqrt{n} \,/\, \exp(n) < 1$ for all $n > 0$

Question: What is $O(\log(n!))$?

- For constant $k > 0$ it holds that $\binom{n}{k} = \mathrm{O}(n^k)$

Running Time Analysis – More Examples

- Recall that $n! = \prod_{i=1}^{n} i$

Question: What is $O(n!)$?

Stirling's approximation: $n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n = \sqrt{2\pi}\,\frac{\sqrt{n}}{\exp(n)}\,n^n \overset{(*)}{=} O(n^n) = O(2^{n \log n})$

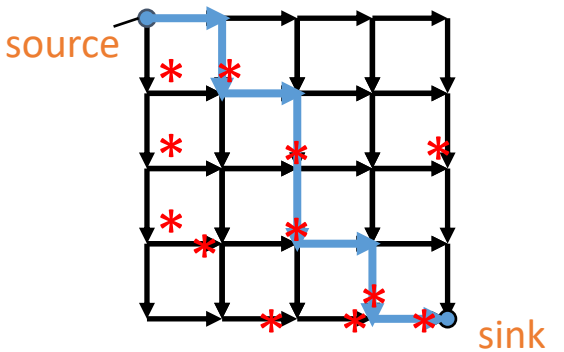
$(*): \sqrt{n} / \exp(n) < 1$ for all $n > 0$

Question: What is $O(\log(n!))$?

- For constant $k > 0$ it holds that $\binom{n}{k} = O(n^k)$

- Number of source-to-sink paths in the Manhattan Tourist Problem on a square $n \times n$ grid is $\binom{2n}{n}$

Question: What is $O\left(\binom{2n}{n}\right)$?



Running Time Analysis – More Examples

- Recall that $n! = \prod_{i=1}^{n} i$

Question: What is $O(n!)$?

Stirling's approximation: $n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n = \sqrt{2\pi}\frac{\sqrt{n}}{\exp(n)}n^n \overset{(*)}{=} O(n^n) = O(2^{n \log n})$

$(*): \sqrt{n} / \exp(n) < 1$ for all $n > 0$

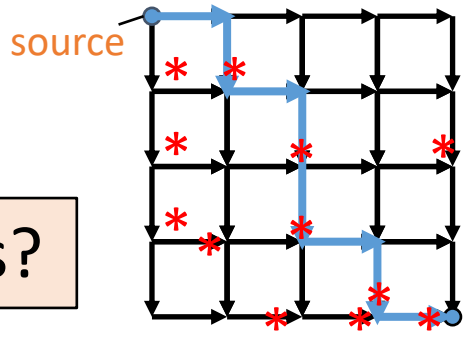
Question: What is $O(\log(n!))$?

- For constant $k > 0$ it holds that $\binom{n}{k} = O(n^k)$

- Number of source-to-sink paths in the Manhattan Tourist Problem on a square $n \times n$ grid is $\binom{2n}{n}$

Question: What is $O\left(\binom{2n}{n}\right)$?

When do we achieve this?



Outline

1. Running time recap
2. Edit distance recap
3. Global alignment
4. Fitting alignment
5. Gapped alignment

Reading:

- Jones and Pevzner. Chapters 6.6, 6.7 and 6.9
- Lecture notes on running time

Alignment

An **alignment** between two strings **v** (of m characters) and **w** (of n characters) is a $2 \times k$ matrix, where $k = \{\max(m, n), \ldots, m + n\}$ such that the first row contains the characters of **v** in order, the second row contains the characters of **w** in order, and spaces may be interspersed throughout each.

Input

v: KITTEN (m = 6)

w: SITTING (n = 7)

Output

	Insertion	Match	Mismatch	Insertion
v:	K	–	I	T	T	E	N	–
w:	S	I	–	T	T	I	N	G
	Mismatch	Deletion	Match	Match

Note: There is no **-/-**

Edit Distance

Edit Distance problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$, compute the minimum number $d(\mathbf{v}, \mathbf{w})$ of **elementary operations** to transform $\mathbf{v}$ into $\mathbf{w}$.

v: ATGTTAT

w: AGCGTAC

Elementary operations: deletion, insertion, mismatch, match

$i-1$ i

prefix of **v** of length i $\mathbf{v}_i$:

prefix of **w** of length j $\mathbf{w}_j$:

A	T	–	G	T	T	T
A	G	C	G	T	–	C

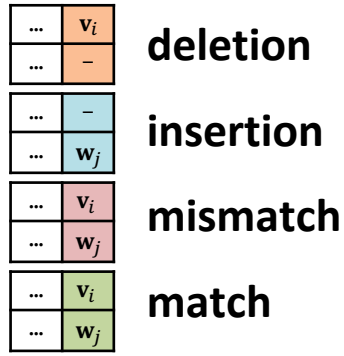
$j-1$ j

Optimal substructure:

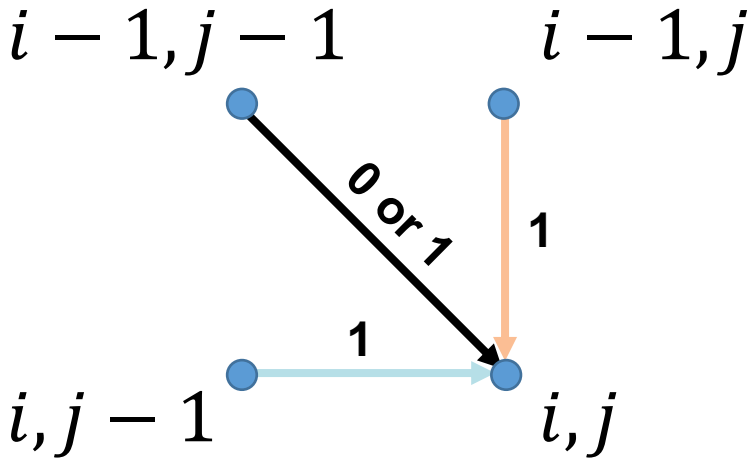
Edit distance obtained from edit distance of prefix of string.

Computing Edit Distance using Dynamic Programming

$$d[i,j] = \min \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ d[i-1,j]+1, & \text{if } i > 0, \\ d[i,j-1]+1, & \text{if } j > 0, \\ d[i-1,j-1]+1, & \text{if } i > 0,\ j > 0 \text{ and } v_i \neq w_j, \\ d[i-1,j-1], & \text{if } i > 0,\ j > 0 \text{ and } v_i = w_j. \end{cases}$$



V \ W		A	T	C	G
	0	1	2	3	4
0	0	1	2	3	4
A 1	1	0	1	2	3
T 2	2	1	0	1	2
G 3	3	2	1	1	1
T 4	4	3	2	2	2



Weighted Edit Distance – Practice Problem

- Compute weighted edit distance between $\mathbf{v} = \mathrm{AGT}$ and $\mathbf{w} = \mathrm{ATCT}$.

$$d[i,j] = \min \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ d[i-1,j] + 1, & \text{if } i > 0, \\ d[i,j-1] + 1, & \text{if } j > 0, \\ d[i-1,j-1] + 2, & \text{if } i > 0,\ j > 0 \text{ and } v_i \neq w_j, \\ d[i-1,j-1], & \text{if } i > 0,\ j > 0 \text{ and } v_i = w_j. \end{cases}$$

Edit Distance – Additional Insights

- An alignment corresponds to a series of elementary operations

Example

```
T-ACAT-
TGAT-AT
```

$$\text{TACAT} \xrightarrow{\text{ins}} \text{TGACAT} \xrightarrow{\text{subst}} \text{TGATAT} \xrightarrow{\text{del}} \text{TGATT} \xrightarrow{\text{subst}} \text{TGATA} \xrightarrow{\text{ins}} \text{TGATAT}$$

Examples from http://profs.scienze.univr.it/~liptak/ACB/files/StringDistance_6up.pdf

Edit Distance – Additional Insights

- An alignment corresponds to a series of elementary operations

Example

```
T-ACAT-
TGAT-AT
```

$$\text{TACAT} \xrightarrow{\text{ins}} \text{TGACAT} \xrightarrow{\text{subst}} \text{TGATAT} \xrightarrow{\text{del}} \text{TGATT} \xrightarrow{\text{subst}} \text{TGATA} \xrightarrow{\text{ins}} \text{TGATAT}$$

- But not every series of elementary operations corresponds to an alignment! Why?
  - $\text{TACAT} \xrightarrow{\text{subst}} \text{GACAT} \xrightarrow{\text{del}} \text{GAAT} \xrightarrow{\text{ins}} \text{TGAAT} \xrightarrow{\text{ins}} \text{TGATAT}$

```
-TAC-AT
TGA-TAT
```

  - $\text{TACAT} \xrightarrow{\text{ins}} \text{TGACAT} \xrightarrow{\text{subst}} \text{TGATAT}$

```
T-ACAT
TGATAT
```

  - $\text{TACAT} \xrightarrow{\text{ins}} \text{TGACAT} \xrightarrow{\text{subst}} \text{TGAGAT} \xrightarrow{\text{subst}} \text{TGATAT}$

???

Examples from http://profs.scienze.univr.it/~liptak/ACB/files/StringDistance_6up.pdf

Distance Function / Metric

A **distance function** (metric) on a set X is a function $d : X \times X \to \mathbb{R}$ s.t. for all $x, y, z \in X$:

i. $d(x, y) \geq 0$ [non-negativity]

ii. $d(x, y) = 0$ if and only if $x = y$ [identity of indiscernibles]

iii. $d(x, y) = d(y, x)$ [symmetry]

iv. $d(x, y) \leq d(x, z) + d(z, y)$ [triangle inequality]

Question: Is edit distance a distance function?

Edit Distance is a Distance Function

Edit distance $d(\mathbf{v}, \mathbf{w})$ is the minimum number of **elementary operations** to transform $\mathbf{v} \in \Sigma^*$ into $\mathbf{w} \in \Sigma^*$.

Claim: edit distance is a distance function.

Proof: Let $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \Sigma^*$.

i. $d(\mathbf{v}, \mathbf{w}) \geq 0$ [non-negativity]

Edit distance is defined by an alignment. This in turn uniquely determines a series of elementary operations, each with cost either 0 (match) or 1 (otherwise). Thus, $d(\mathbf{v}, \mathbf{w}) \geq 0$.

Edit Distance is a Distance Function

Edit distance $d(\mathbf{v}, \mathbf{w})$ is the minimum number of **elementary operations** to transform $\mathbf{v} \in \Sigma^*$ into $\mathbf{w} \in \Sigma^*$.

Claim: edit distance is a distance function.

Proof: Let $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \Sigma^*$.

ii. $d(\mathbf{v}, \mathbf{w}) = 0$ if and only if $\mathbf{v} = \mathbf{w}$ [identity of indiscernibles]

(=>) By the premise, $d(\mathbf{v}, \mathbf{w}) = 0$. By definition, the optimal alignment can only consist of operations with cost 0. That is, the alignment consist of only matches. Thus, $\mathbf{v} = \mathbf{w}$.

(<=) By the premise, $\mathbf{v} = \mathbf{w}$. Thus, there exists an alignment where every pair of columns is a match. This means that $|\mathbf{v}| = |\mathbf{w}|$ and each letter v_i equals w_i (where $i \in [|\mathbf{v}|]$). Moreover, only the match operations has cost 0, the other operations have cost 1. Hence, this is the optimal alignment with cost $d(\mathbf{v}, \mathbf{w}) = 0$.

Edit Distance is a Distance Function

Edit distance $d(\mathbf{v}, \mathbf{w})$ is the minimum number of **elementary operations** to transform $\mathbf{v} \in \Sigma^*$ into $\mathbf{w} \in \Sigma^*$.

Claim: edit distance is a distance function.

Proof: Let $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \Sigma^*$.

iii. $d(\mathbf{v}, \mathbf{w}) = d(\mathbf{w}, \mathbf{v})$ [symmetry]

Let $\mathbf{A} = [a_{i,j}]$ be the optimal alignment corresponding to $d(\mathbf{v}, \mathbf{w})$, i.e. $\mathbf{A}$ is an $2 \times k$ matrix where $k \in \{\max(|\mathbf{v}|, |\mathbf{w}|), \dots, |\mathbf{v}| + |\mathbf{w}|\}$. Define the function $f(\mathbf{A}) = \mathbf{B}$ such that $\mathbf{B}$ is obtained by interchanging the two rows of $\mathbf{A}$. Since the cost of any insertion, deletion and mismatch is 1, we have that alignment $\mathbf{B}$ has cost $d(\mathbf{v}, \mathbf{w})$. The existence of an alignment from $\mathbf{w}$ to $\mathbf{v}$ with cost less than $d(\mathbf{v}, \mathbf{w})$, yields a contradiction as it implies that $\mathbf{A}$ is not an optimal alignment from $\mathbf{v}$ to $\mathbf{w}$. Hence, $d(\mathbf{w}, \mathbf{v}) = d(\mathbf{v}, \mathbf{w})$.

Edit Distance is a Distance Function

Edit distance $d(\mathbf{v}, \mathbf{w})$ is the minimum number of **elementary operations** to transform $\mathbf{v} \in \Sigma^*$ into $\mathbf{w} \in \Sigma^*$.

Claim: edit distance is a distance function.

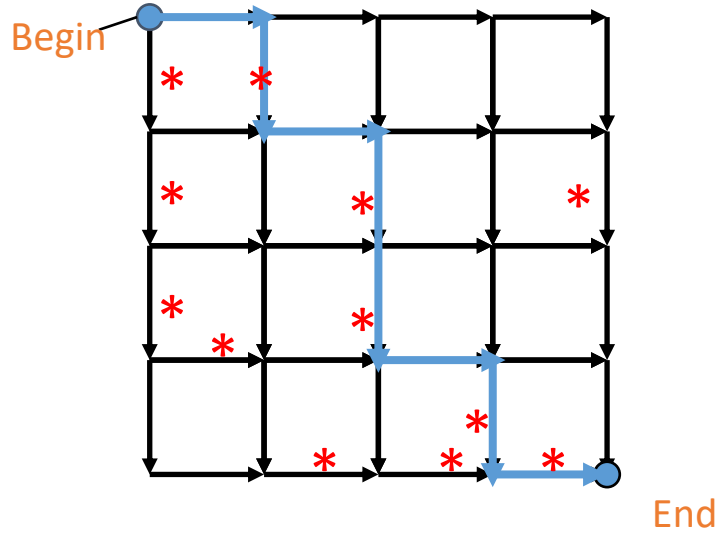
Proof: Let $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \Sigma^*$.

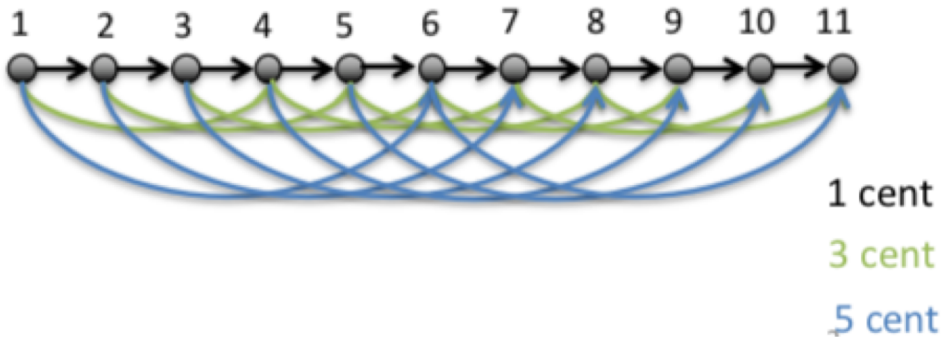
iv. $d(\mathbf{v}, \mathbf{w}) \leq d(\mathbf{v}, \mathbf{u}) + d(\mathbf{u}, \mathbf{w})$ [triangle inequality]

Assume for a contradiction that $d(\mathbf{v}, \mathbf{w}) > d(\mathbf{v}, \mathbf{u}) + d(\mathbf{u}, \mathbf{w})$. Let S be the sequence of elementary operations for transforming $\mathbf{v}$ into $\mathbf{u}$. Let S' be the sequence of elementary operations for transforming $\mathbf{u}$ into $\mathbf{w}$. Note that $d(\mathbf{v}, \mathbf{u}) = |S|$ and $d(\mathbf{u}, \mathbf{w}) = |S'|$. Concatenate S and S' and remove redundant operations, yielding sequence S''. By definition, $|S''| \leq |S| + |S'|$. We can obtain an alignment of $\mathbf{v}$ and $\mathbf{w}$ from S'' with cost $|S''| \leq d(\mathbf{v}, \mathbf{u}) + d(\mathbf{u}, \mathbf{w})$. This yields a contradiction with $d(\mathbf{v}, \mathbf{w}) > d(\mathbf{v}, \mathbf{u}) + d(\mathbf{u}, \mathbf{w})$ being the cost of the optimal alignment of $\mathbf{v}$ and $\mathbf{w}$.

Dynamic Programming as a Graph Problem

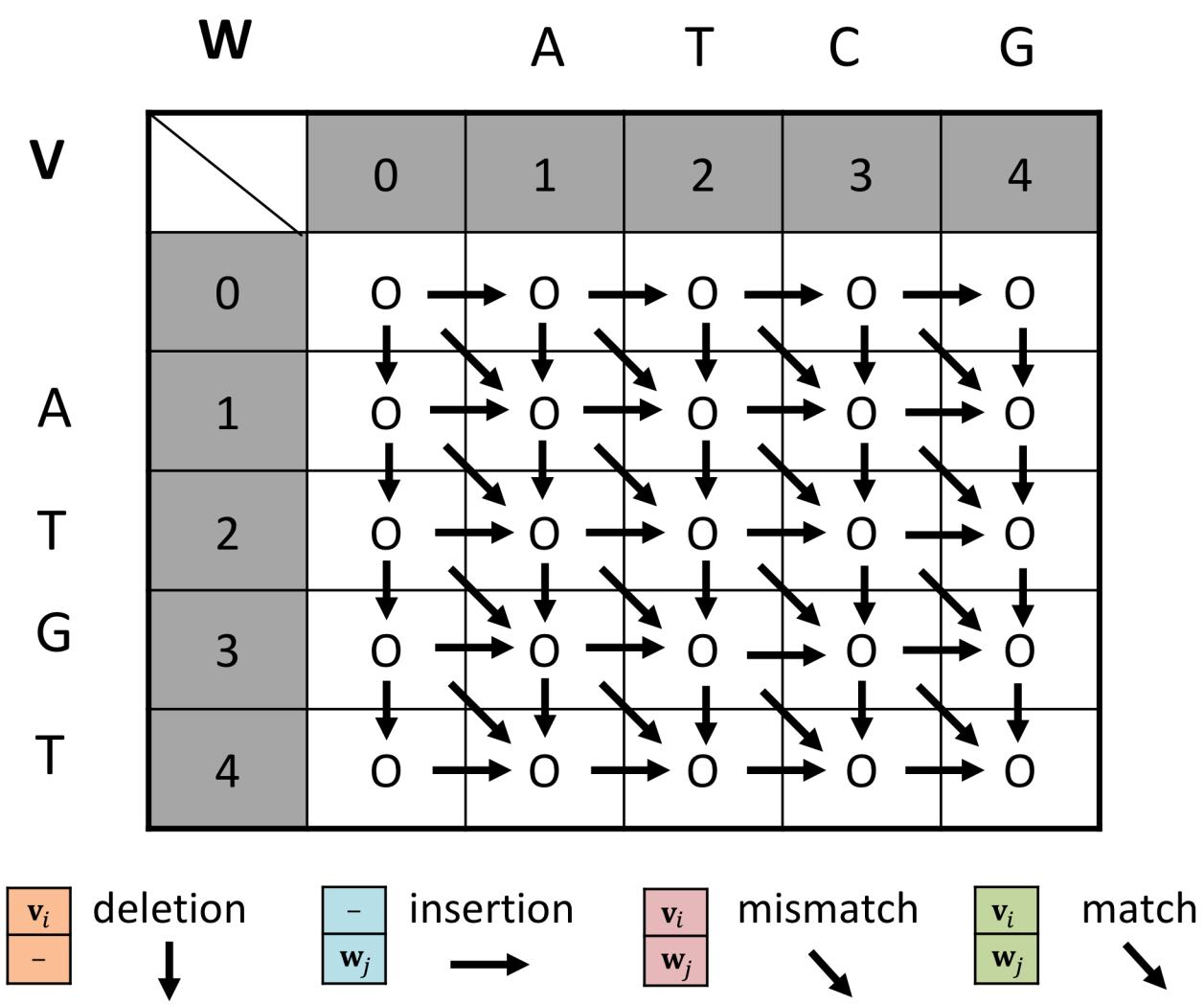
Manhattan Tourist Problem:
Every path in directed graph is a possible tourist path. Find maximum weight path.
Running time: $O(mn) = O(|E|)$





Change Problem: Make M cents using minimum number of coins $\mathbf{c} = (1, 3, 5)$.
Every path in directed graph is a possible change. Find shortest path.
Running time: $O(Mn) = O(|E|)$

Edit Distance as a Graph Problem



Edit Distance problem: Given edit graph $G = (V, E)$, with edge weights $c : E \to \{0,1\}$. Find shortest path from $(0, 0)$ to (m, n).

Alignment is a path from $(0, 0)$ to (m, n)

Edit graph is a weighed, directed grid graph $G = (V, E)$ with source vertex $(0, 0)$ and target vertex (m, n). Each edge (i, j) has weight $[i, j]$ corresponding to edit cost: deletion (1), insertion (1), mismatch (1) and match (0).

Outline

1. Running time recap
2. Edit distance recap
3. Global alignment
4. Fitting alignment
5. Gapped alignment

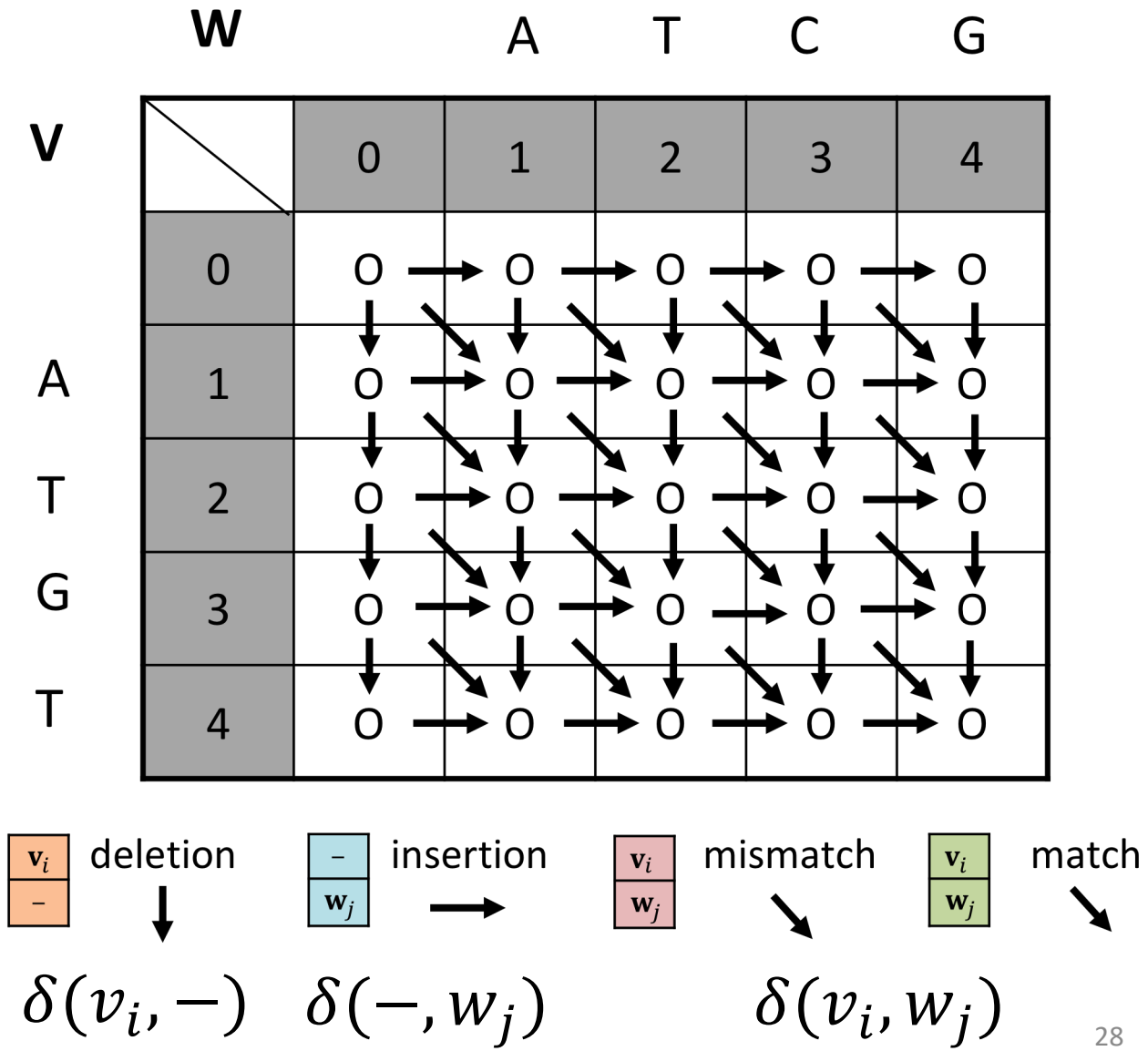
Reading:

- Jones and Pevzner. Chapters 6.6, 6.7 and 6.9
- Lecture notes on running time

Biological Sequence Alignment

- Weighted edit distance: find alignment with minimum distance
  - Shortest path in weighted edit graph
- Sequence alignment: find alignment with maximum similarity
  - Longest path in weighted edit graph
  - Score function: $\delta : (\Sigma \cup \{-\})^2 \to \mathbb{R}$

Question: What is an example of δ?



Scoring Matrices

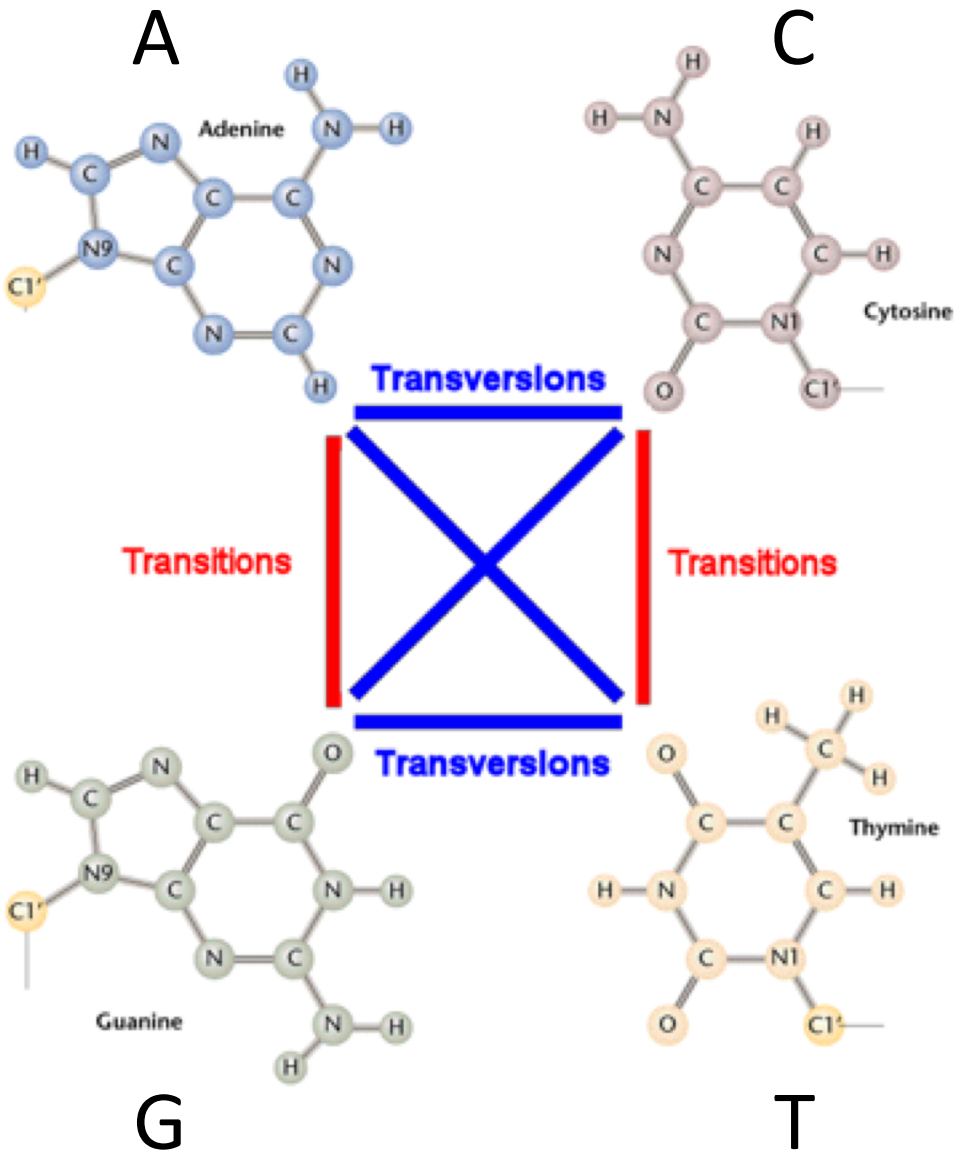
Transitions: interchanges among purines (two rings) or pyrimidines (one ring)

- A <--> G
- C <--> T

Transversions: interchanges between purines (two rings) and pyrimidines (one ring)

- A <--> C, A <--> T
- G <--> C, G <--> T

Transitions more likely than transversions!



Scoring Matrices

Transitions: interchanges among purines (two rings) or pyrimidines (one ring)

- A <--> G
- C <--> T

Transversions: interchanges between purines (two rings) and pyrimidines (one ring)

- A <--> C, A <--> T
- G <--> C, G <--> T

Transitions more likely than transversions!

δ	A	T	C	G	-
A	1	-2	-2	-1	-1
T	-2	1	-1	-2	-1
C	-2	-1	1	-2	-1
G	-1	-2	-2	1	-1
-	-1	-1	-1	-1	$-\infty$

Global Alignment – Needleman-Wunsch Algorithm

Global Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find alignment with maximum score.

- An alignment is a source-to-sink path in the edit graph
- An alignment $\mathbf{A} = [a_{i,j}]$ is a $2 \times k$ matrix s.t. (i) $k = \{\max(m,n), \ldots, m+n\}$, (ii) $a_{i,j} \in \Sigma \cup \{-\}$ and (iii) there is no $j \in [k]$ where $a_{1,j} = a_{2,j} = -$

$$s[i,j] = \max \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \quad \text{deletion} \\ s[i,j-1] + \delta(-, w_j), & \text{if } j > 0, \quad \text{insertion} \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \quad \text{match/ mismatch} \end{cases}$$

Demonstration

- http://alfehrest.org/sub/nwa/index.html

- $\mathbf{v} = \text{ATGTTAT}$ and $\mathbf{w} = \text{ATCGTAC}$.

δ	A	T	C	G	-
A	1	-2	-2	-1	-1
T	-2	1	-1	-2	-1
C	-2	-1	1	-2	-1
G	-1	-2	-2	1	-1
-	-1	-1	-1	-1	$-\infty$

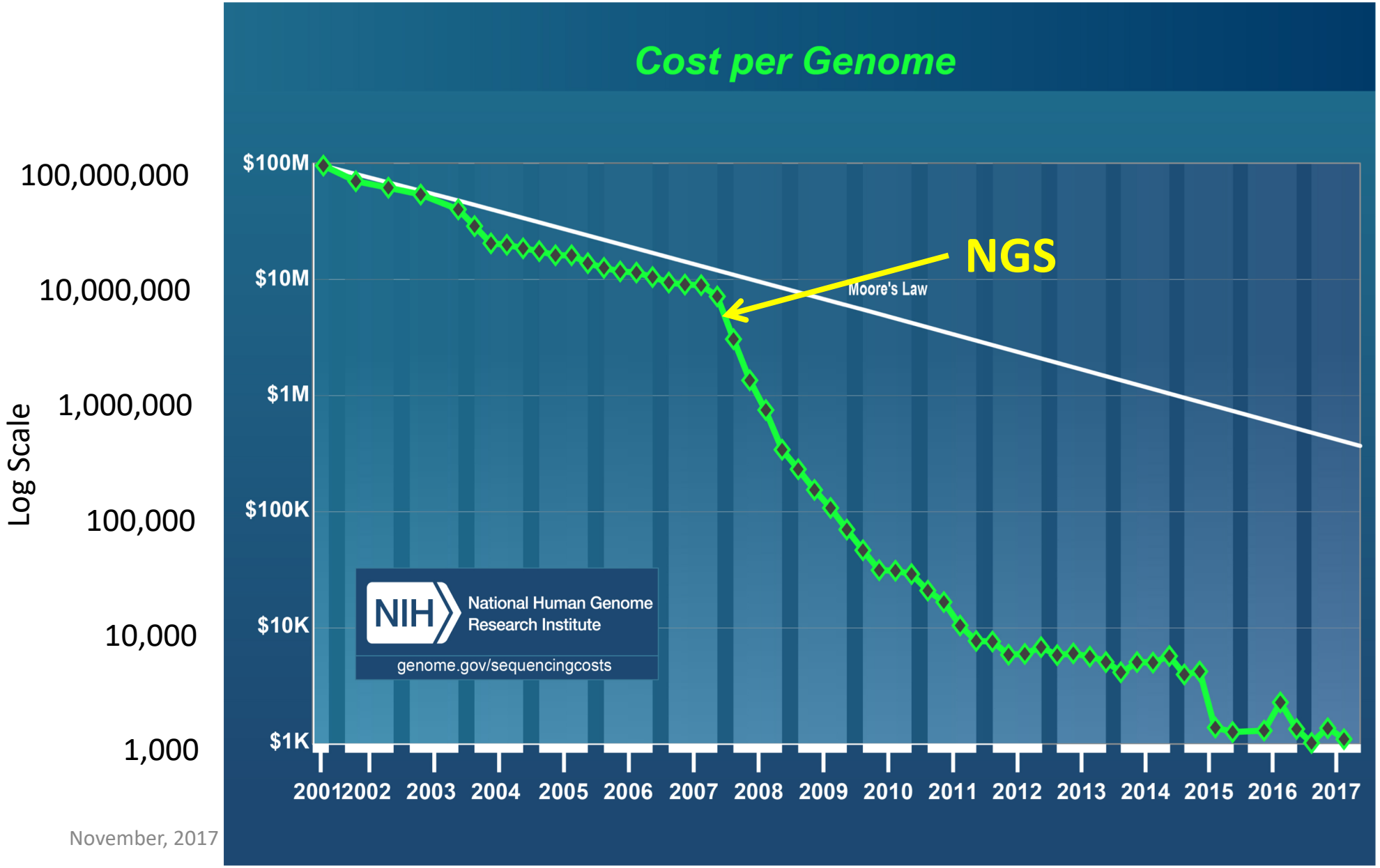
Outline

1. Running time recap
2. Edit distance recap
3. Global alignment
4. Fitting alignment
5. Gapped alignment

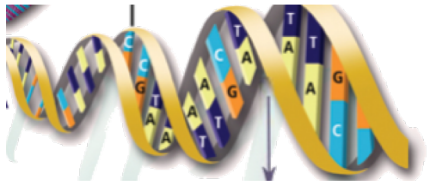
Reading:

- Jones and Pevzner. Chapters 6.6, 6.7 and 6.9
- Lecture notes on running time

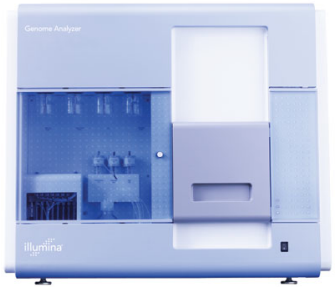
Next Generation Sequencing (NGS) Technology



NGS Characterized by Short Reads



Genome
Millions -billions nucleotides



Next-generation DNA sequencing

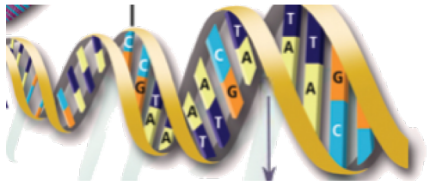
… CATTCAGTAG …
… AGCCATTAG …
… GGTAGTTAG …
… GGTAAACTAG …
… TATAATTAG …
… CGTACCTAG …

10-100's million ***short reads***
Short read*:* 100 nucleotides

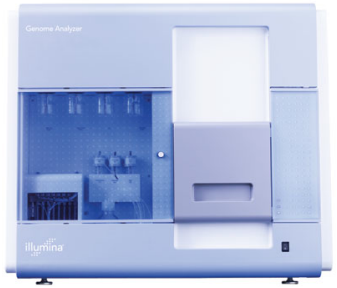
Allow for inexact matches due to:

- Sequencing errors
- Polymorphisms/mutations in reference genome

NGS Characterized by Short Reads



Genome
Millions -billions nucleotides



Next-generation DNA sequencing

… CATTCAGTAG …
… AGCCATTAG …
… GGTAGTTAG …
… GGTAAACTAG …
… TATAATTAG …
… CGTACCTAG …

10-100's million ***short reads***
Short read*:* 100 nucleotides

Allow for inexact matches due to:

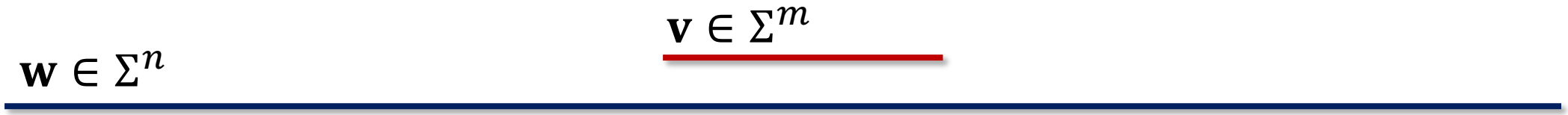
- Sequencing errors
- Polymorphisms/mutations in reference genome

Human reference genome is 3,300,000,000 nucleotides, while a short read is 100 nucleotides. Global sequence alignment will not work!

Question: How to account for discrepancy between lengths of reference and short read?

Fitting Alignment

For short read alignment, we want to align complete short read $\mathbf{v} \in \Sigma^m$ to substring of reference genome $\mathbf{w} \in \Sigma^n$. Note that $m \ll n$.



Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find a alignment of $\mathbf{v}$ and a substring of $\mathbf{w}$ with maximum global alignment score s^* among *all* global alignments of $\mathbf{v}$ and *all* substrings of $\mathbf{w}$

Take Home Messages

1. Running time recap

$$O(a) \subset O(\log n) \subset O(n^b) \subset O(c^n)$$

2. Edit distance recap

Edit distance is a distance function (metric)

3. Global alignment

Global alignment is longest path in DAG

Reading:

- Jones and Pevzner. Chapters 6.6, 6.7 and 6.9
- Lecture notes on running time