CS 466
Introduction to Bioinformatics
Lecture 4

Mohammed El-Kebir

September 10, 2018

Course Announcements

Instructor:

- Mohammed El-Kebir (melkebir)
- Office hours: Mondays, 3:15-4:15pm -- (Unavailable on Sept. 10)

TA:

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

Class is canceled on Sept. 12

Homework 1: Released on Sept. 10, due Sept. 17 (11:59pm)

Outline

1. Global alignment recap
2. Fitting alignment
3. Local alignment
4. Gapped alignment

Reading:

- Jones and Pevzner. Chapters 6.7-6.9
- Lecture notes

Why $n! = O(n^n)$ but not $n^n = O(n!)$?

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 $n > 0$

Why $n! = O(n^n)$ but not $n^n = O(n!)$?

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: Is $n^n = O(n!)$?

That is, are there $c, n_0 > 0$ s.t.
$n^n \leq c\, n!$ for all $n \geq n_0$?

Why $n! = O(n^n)$ but not $n^n = O(n!)$?

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: Is $n^n = O(n!)$?

That is, are there $c, n_0 > 0$ s.t. $n^n \leq c\, n!$ for all $n \geq n_0$?

No! No constant c is able to accommodate $n^{n-1}/((n-1)!)$ asymptotically

$$\lim_{n\to\infty} \frac{n^n}{n!} = \lim_{n\to\infty} \frac{n^n}{n(n-1)(n-2)\dots(1)}$$

$$= \lim_{n\to\infty} \frac{n^{n-1}}{(n-1)(n-2)\dots(1)}$$

$$= \lim_{n\to\infty} \prod_{i=1}^{n-1} \frac{n}{n-i}$$

$$= \infty$$

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) $a_{i,j} \in \Sigma \cup \{-\}$, (iii) there is no $j \in [k]$ where $a_{1,j} = a_{2,j} = -$ and (iv) removing gaps from first (second) row of $\mathbf{A}$ yields $\mathbf{v}$ ($\mathbf{w}$).

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

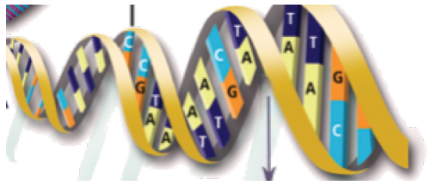
Outline

1. Global alignment recap
2. Fitting alignment
3. Local alignment
4. Gapped alignment

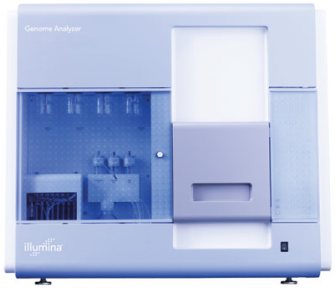
Reading:

- Jones and Pevzner. Chapters 6.7-6.9
- Lecture notes

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:

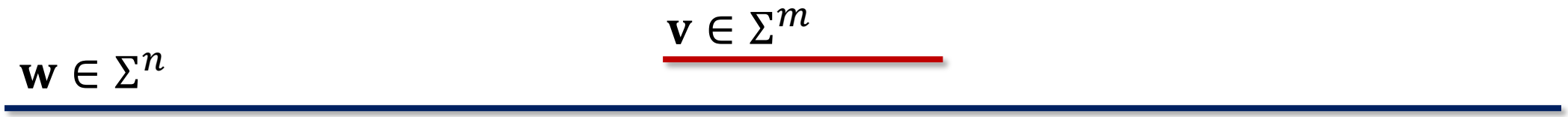
- Sequencing errors
- Polymorphisms/mutations in reference genome

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 an 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 $\mathbf{w}$

Fitting Alignment – Naive Approach

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an 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 $\mathbf{w}$

$\mathbf{v} \in \Sigma^m$

$\mathbf{w} \in \Sigma^n$

- Consider all contiguous non-empty substrings of $\mathbf{w}$, defined by $1 \leq i \leq j \leq n$
- How many?

Fitting Alignment – Naive Approach

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an 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 $\mathbf{w}$

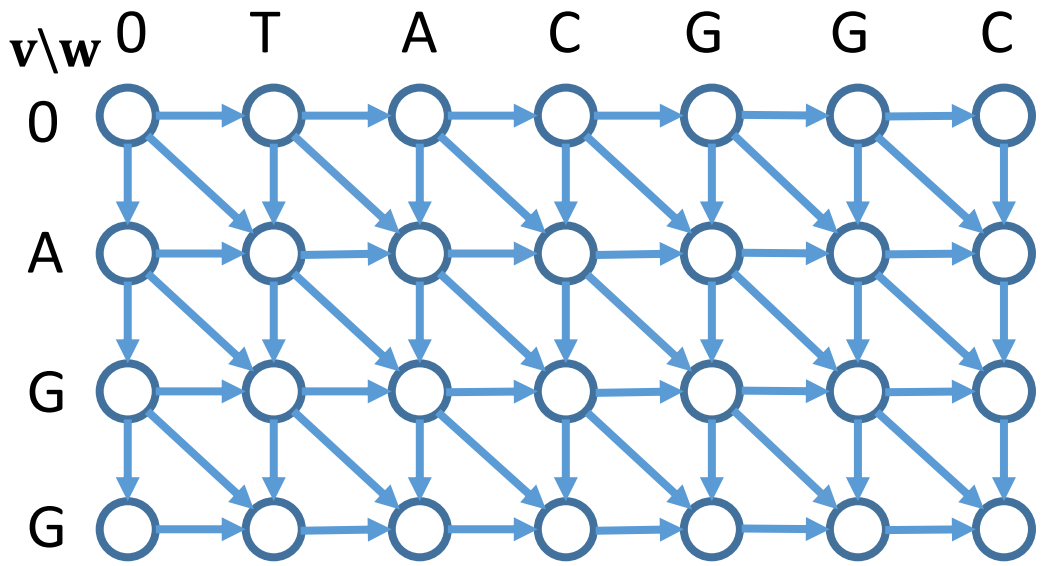
$\mathbf{v} \in \Sigma^m$

$\mathbf{w} \in \Sigma^n$

- Consider all contiguous non-empty substrings of $\mathbf{w}$, defined by $1 \leq i \leq j \leq n$
- How many? Answer: $n + \binom{n}{2}$
- What are their total lengths?
- What is the running time?

Fitting Alignment – Dynamic Programming

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an 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 $\mathbf{w}$

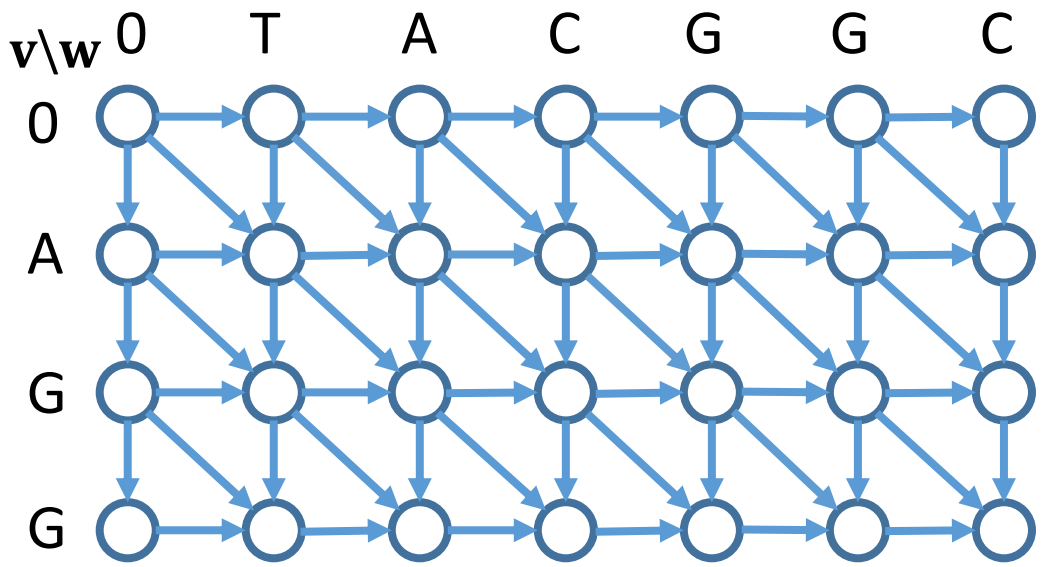


$$s[i,j] = \max \begin{cases} 0, & \text{if } i = 0, \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } i > 0 \text{ and } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \max\{s[m,0], \ldots, s[m,n]\}$$

Fitting Alignment – Dynamic Programming

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an 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 $\mathbf{w}$

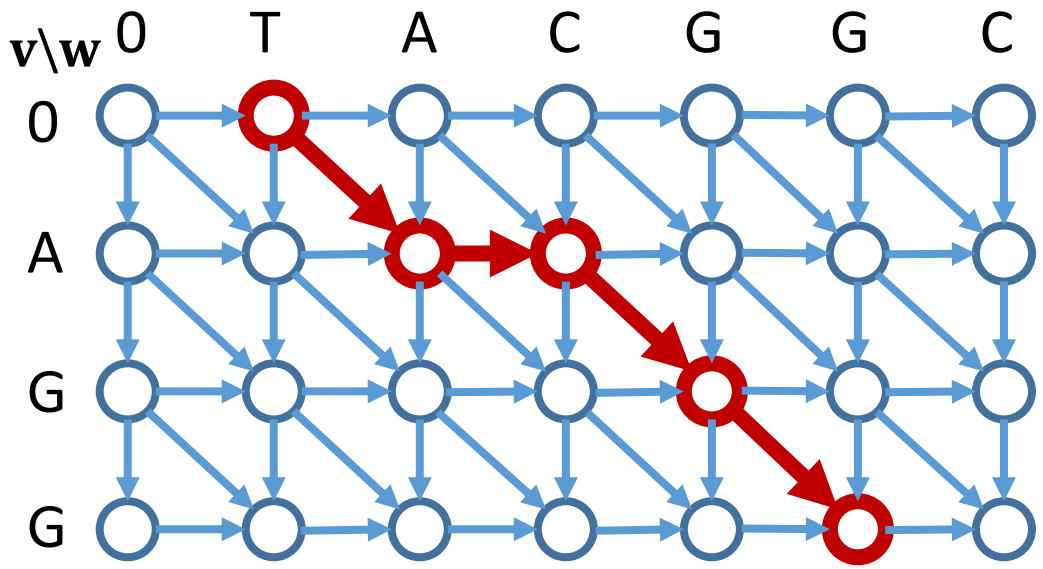


$$s[i,j] = \max \begin{cases} 0, \text{ Start anywhere on first row} & \text{if } i = 0, \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } i > 0 \text{ and } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \max\{s[m,0], \ldots, s[m,n]\} \quad \text{End anywhere on last row}$$

Fitting Alignment – Dynamic Programming

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an 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 $\mathbf{w}$



v	**-**	**A**	**-**	**G**	**G**	**-**
w	**T**	**A**	**C**	**G**	**G**	**C**

$$s[i,j] = \max \begin{cases} 0, \text{ Start anywhere on first row} & \text{if } i = 0, \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } i > 0 \text{ and } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

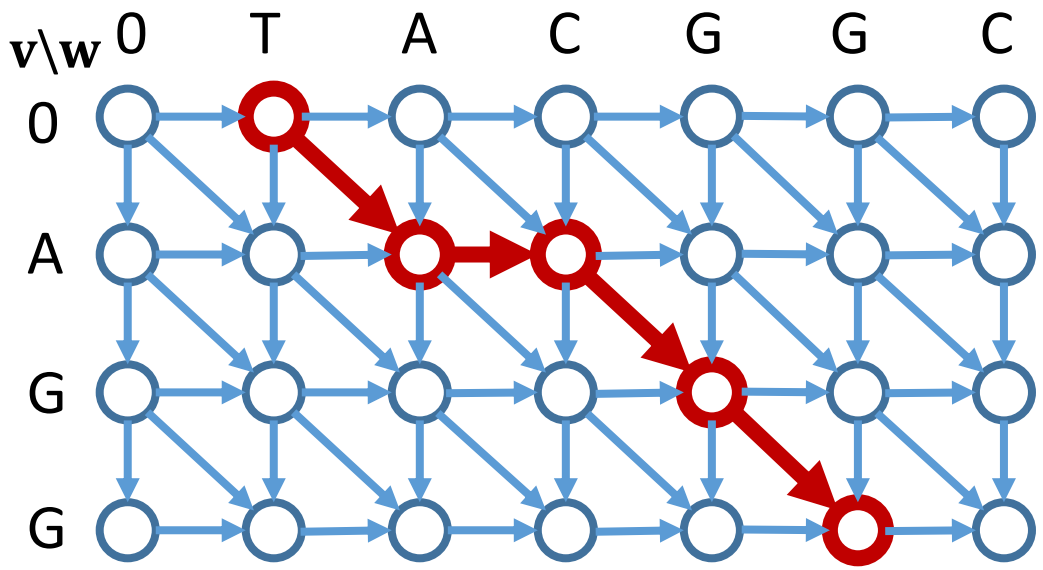
$$s^* = \max\{s[m,0], \ldots, s[m,n]\} \quad \text{End anywhere on last row}$$

Question: Let match score be 1, mismatch/indel score be -1. What is s^*?

Question: Same scores. What is optimal global alignment and score?

Fitting Alignment – Dynamic Programming

- Online: https://valiec.github.io/AlignmentVisualizer/index.html



v	**-**	**A**	**-**	**G**	**G**	**-**
w	**T**	**A**	**C**	**G**	**G**	**C**

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

$$s^* = \max\{s[m,0], \ldots, s[m,n]\}$$

Question: Let match score be 1, mismatch/indel score be -1. What is s^*?

Question: Same scores. What is optimal global alignment and score?

Outline

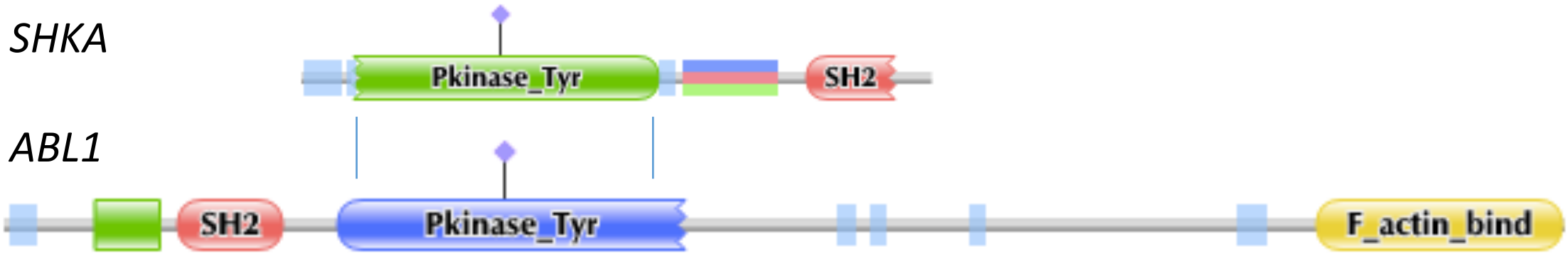
1. Global alignment recap
2. Fitting alignment
3. Local alignment
4. Gapped alignment

Reading:

- Jones and Pevzner. Chapters 6.7-6.9
- Lecture notes

Local Alignment – Biological Motivation

Proteins are composed of functional units called domains. Such domains may occur in different proteins even across species.



From Pfam database (http://pfam.sanger.ac.uk/)

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

Global, Fitting and Local Alignment

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

Fitting Alignment problem: Given strings $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ and scoring function δ, find an 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 $\mathbf{w}$

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

Local Alignment – Naive Algorithm

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

Brute force:

1. Generate all pairs $(\mathbf{v}', \mathbf{w}')$ of substrings of $\mathbf{v}$ and $\mathbf{w}$
2. For each pair $(\mathbf{v}', \mathbf{w}')$, solve global alignment problem.

Question: There are $\binom{m}{2}\binom{n}{2}$ pairs of substrings.
But they have different lengths. What is the running time?

Key Idea

Global alignment:

- Start at $(0,0)$ and end at (m,n)

Local alignment:

- Start and end anywhere

```
--T--CC-C-AGT--TATGT-CAGGGGACACG--A-GCATGCAGA-GAC
  |  || |  ||  | | | |||    || |  | |  | ||||  |
AATTGCCGCC-GTCGT-T-TTCAG----CA-GTTATG--T-CAGAT--C
```

```
                  tccCAGTTATGTCAGgggacacgagcatgcagagac
                    ||||||||||||
aattgccgccgtcgttttcagCAGTTATGTCAGatc
```

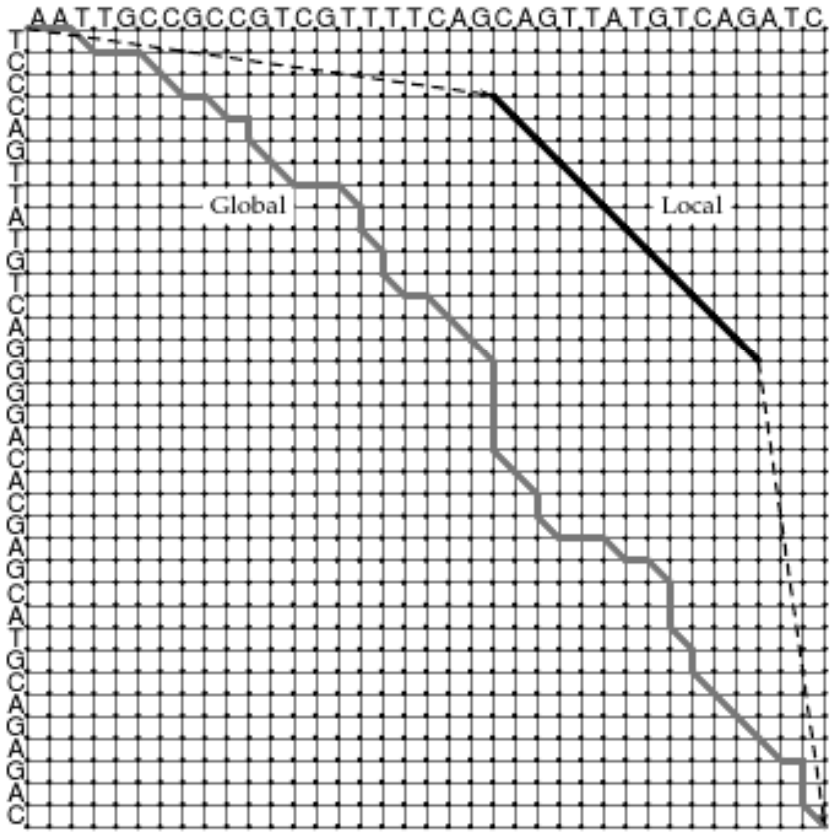


Figure 6.16 (a) Global and (b) local alignments of two hypothetical genes that each have a conserved domain. The local alignment has a much worse score according to the global scoring scheme, but it correctly locates the conserved domain.

Local Alignment Recurrence

```
--T--CC-C-AGT--TATGT-CAGGGGACACG--A-GCATGCAGA-GAC
  |  || |  ||  | | | |||    || |  | |  | ||||   |
AATTGCCGCC-GTCGT-T-TTCAG----CA-GTTATG--T-CAGAT--C

                    tccCAGTTATGTCAGgggacacgagcatgcagagac
                      ||||||||||||
aattgccgccgtcgttttcagCAGTTATGTCAGatc
```

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

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

$$s^* = \max_{i,j} s[i,j]$$

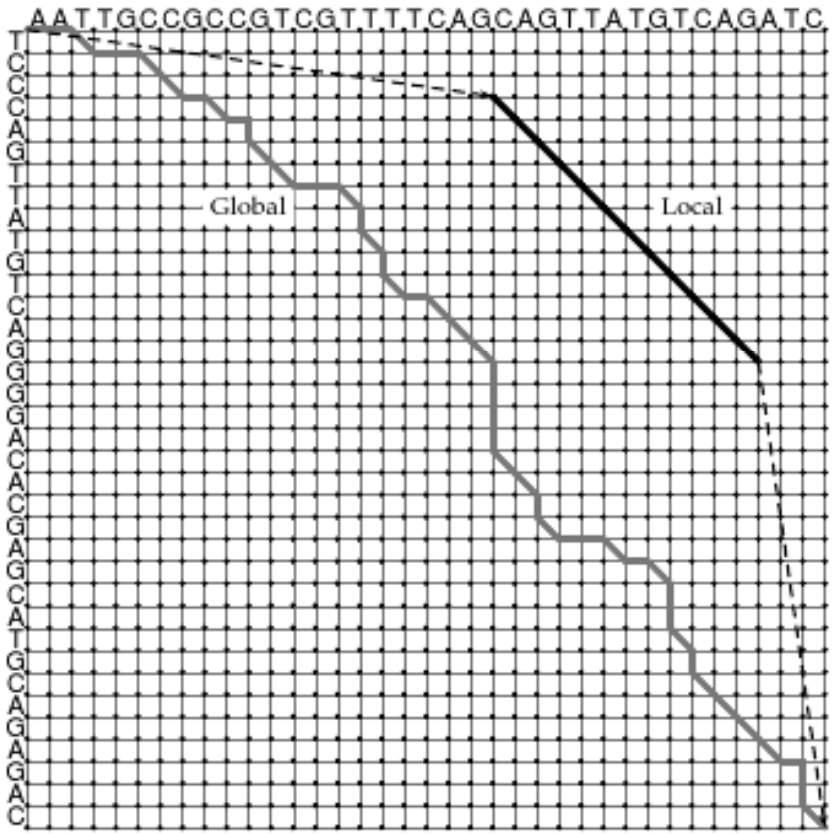


Figure 6.16 (a) Global and (b) local alignments of two hypothetical genes that each have a conserved domain. The local alignment has a much worse score according to the global scoring scheme, but it correctly locates the conserved domain.

Local Alignment Recurrence

```
--T--CC-C-AGT--TATGT-CAGGGGACACG--A-GCATGCAGA-GAC
  |  || |  ||  | | | |||    || |  | |  | ||||  |
AATTGCCGCC-GTCGT-T-TTCAG----CA-GTTATG--T-CAGAT--C

                    tccCAGTTATGTCAGgggacacgagcatgcagagac
                      ||||||||||||
aattgccgccgtcgttttcagCAGTTATGTCAGatc
```

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

$$s[i,j] = \max \begin{cases} 0, & \text{Start anywhere} \quad \text{~~if } i = 0 \text{ and } j = 0\text{~~,} \\ s[i-1,j] + \delta(v_i, -), & \text{if } i > 0, \\ s[i,j-1] + \delta(-, w_j), & \text{if } j > 0, \\ s[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \max_{i,j} s[i,j] \quad \text{End anywhere}$$

Running time: $O(mn)$

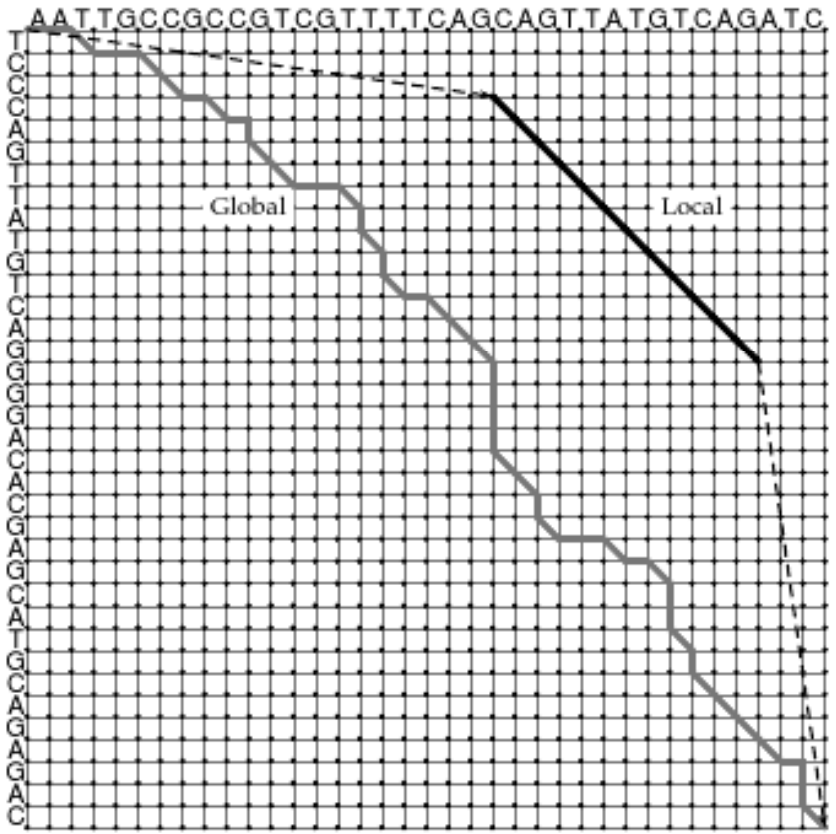
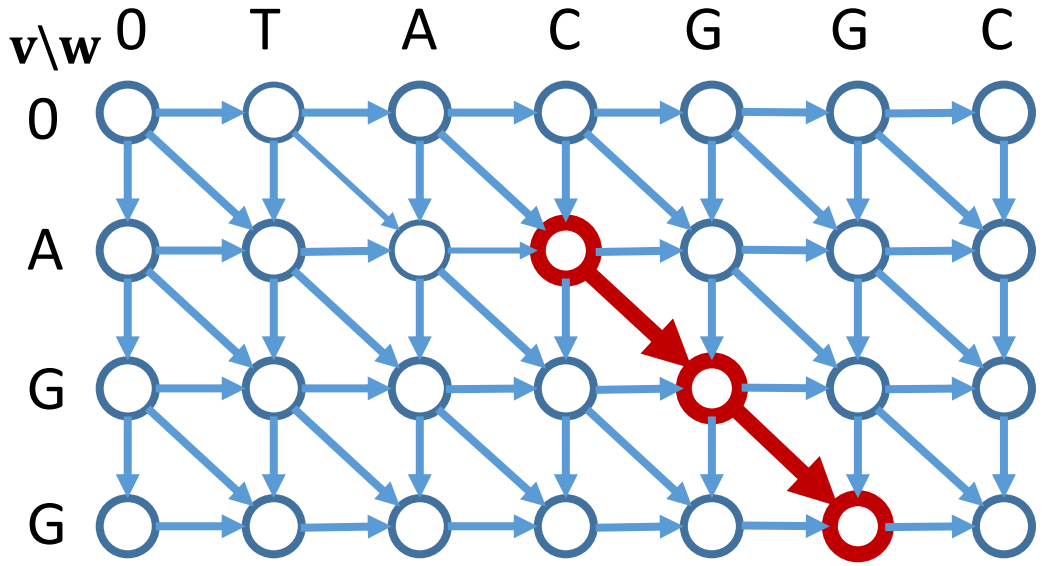


Figure 6.16 (a) Global and (b) local alignments of two hypothetical genes that each have a conserved domain. The local alignment has a much worse score according to the global scoring scheme, but it correctly locates the conserved domain.

Local Alignment – Dynamic Programming

- Online: https://valiec.github.io/AlignmentVisualizer/index.html



v	**G**	**G**
w	**G**	**G**

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

$$s^* = \max_{i,j} s[i,j]$$

Question: Let match score be 2, mismatch score be -2 and indel be -4. What is s^*?

Outline

1. Global alignment recap
2. Fitting alignment
3. Local alignment
4. Gapped alignment

Reading:

- Jones and Pevzner. Chapters 6.7-6.9
- Lecture notes

Scoring Gaps

Let $\mathbf{v} = \text{AAC}$ and $\mathbf{w} = \text{ACAGGC}$

Match $\delta(c, c) = 1$;
Mismatch $\delta(c, d) = -1$ (where $c \neq d$); Indel $\delta(c, -) = \delta(-, c) = -2$

v	**A**	**-**	**-**	**A**	**C**
w	**A**	**C**	**A**	**A**	**C**

v	**A**	**-**	**A**	**-**	**C**
w	**A**	**C**	**A**	**A**	**C**

Both alignments have 3 matches and 2 indels.
Score: $(3 * 1) + (2 * -2) = -1$

Scoring Gaps

Let $\mathbf{v} = \text{AAC}$ and $\mathbf{w} = \text{ACAGGC}$

Match $\delta(c, c) = 1$;
Mismatch $\delta(c, d) = -1$ (where $c \neq d$); Indel $\delta(c, -) = \delta(-, c) = -2$

v	**A**	**-**	**-**	**A**	**C**
w	**A**	**C**	**A**	**A**	**C**

v	**A**	**-**	**A**	**-**	**C**
w	**A**	**C**	**A**	**A**	**C**

Both alignments have 3 matches and 2 indels.
Score: $(3 * 1) + (2 * -2) = -1$

Question: Which alignment is better?

Scoring Gaps – Affine Gap Penalties

Desired: Lower penalty for consecutive gaps than interspersed gaps.

Why: Consecutive gaps are more likely due to slippage errors in DNA replication (2-3 nucleotides), codons for protein sequences, etc.

v	**A**	**-**	**-**	**A**	**C**
w	**A**	**C**	**A**	**A**	**C**

v	**A**	**-**	**A**	**-**	**C**
w	**A**	**C**	**A**	**A**	**C**

Scoring Gaps – Affine Gap Penalties

Desired: Lower penalty for consecutive gaps than interspersed gaps.

Why: Consecutive gaps are more likely due to slippage errors in DNA replication (2-3 nucleotides), codons for protein sequences, etc.

v	**A**	**-**	**-**	**A**	**C**
w	**A**	**C**	**A**	**A**	**C**

v	**A**	**-**	**A**	**-**	**C**
w	**A**	**C**	**A**	**A**	**C**

Affine gap penalty: Two penalties: (i) gap open penalty $\rho \geq 0$ and (ii) gap extension penalty $\sigma \geq 0$. Stretch of k consecutive gaps has score $-(\rho + \sigma k)$.

Scoring Gaps – Affine Gap Penalties

Desired: Lower penalty for consecutive gaps than interspersed gaps.

Why: Consecutive gaps are more likely due to slippage errors in DNA replication (2-3 nucleotides), codons for protein sequences, etc.

v	**A**	**-**	**-**	**A**	**C**
w	**A**	**C**	**A**	**A**	**C**

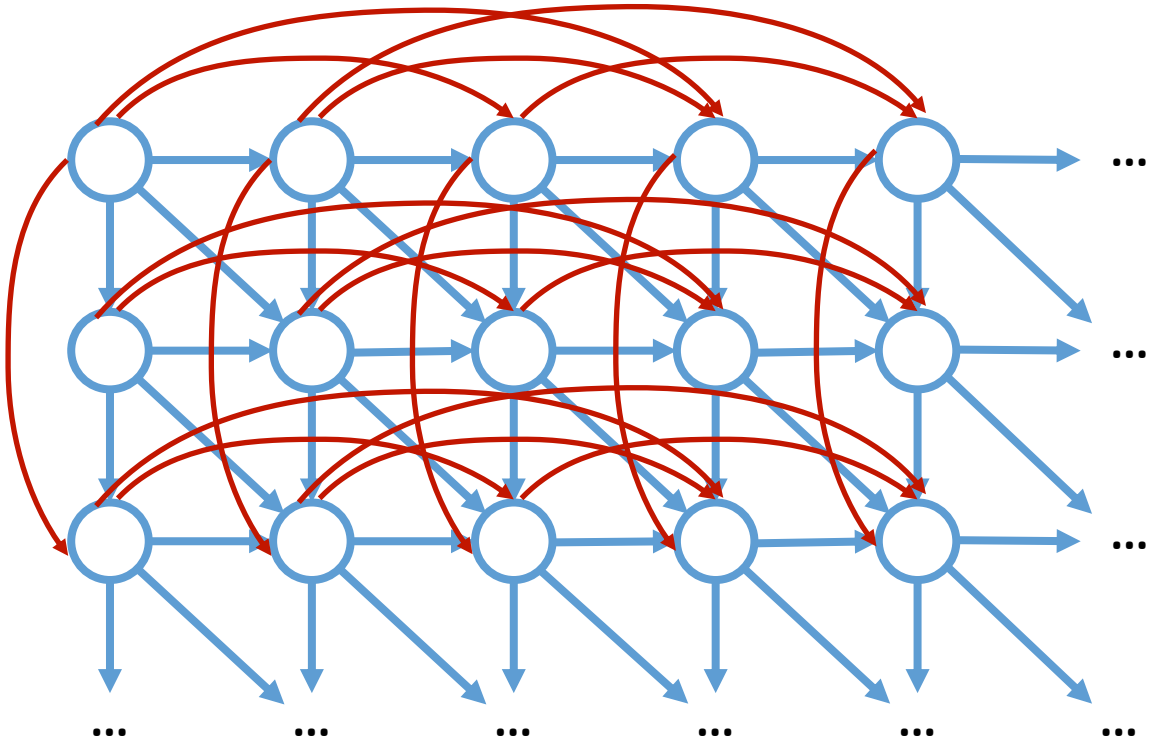
v	**A**	**-**	**A**	**-**	**C**
w	**A**	**C**	**A**	**A**	**C**

Affine gap penalty: Two penalties: (i) gap open penalty $\rho \geq 0$ and (ii) gap extension penalty $\sigma \geq 0$. Stretch of k consecutive gaps has score $-(\rho + \sigma k)$.

Let $\rho = 10$ and $\sigma = 1$. Left: $(3 * 1) - (10 + 1 * 2) = -9$.
Right: $(3 * 1) - (10 + 1 * 1) - (10 + 1 * 1) = -19$.

Affine Gap Penalty Alignment – Naive Approach

Affine gap penalty: Two penalties: (i) gap open penalty $\rho \geq 0$ and (ii) gap extension penalty $\sigma \geq 0$. Stretch of k consecutive gaps has score $-(\rho + \sigma k)$.



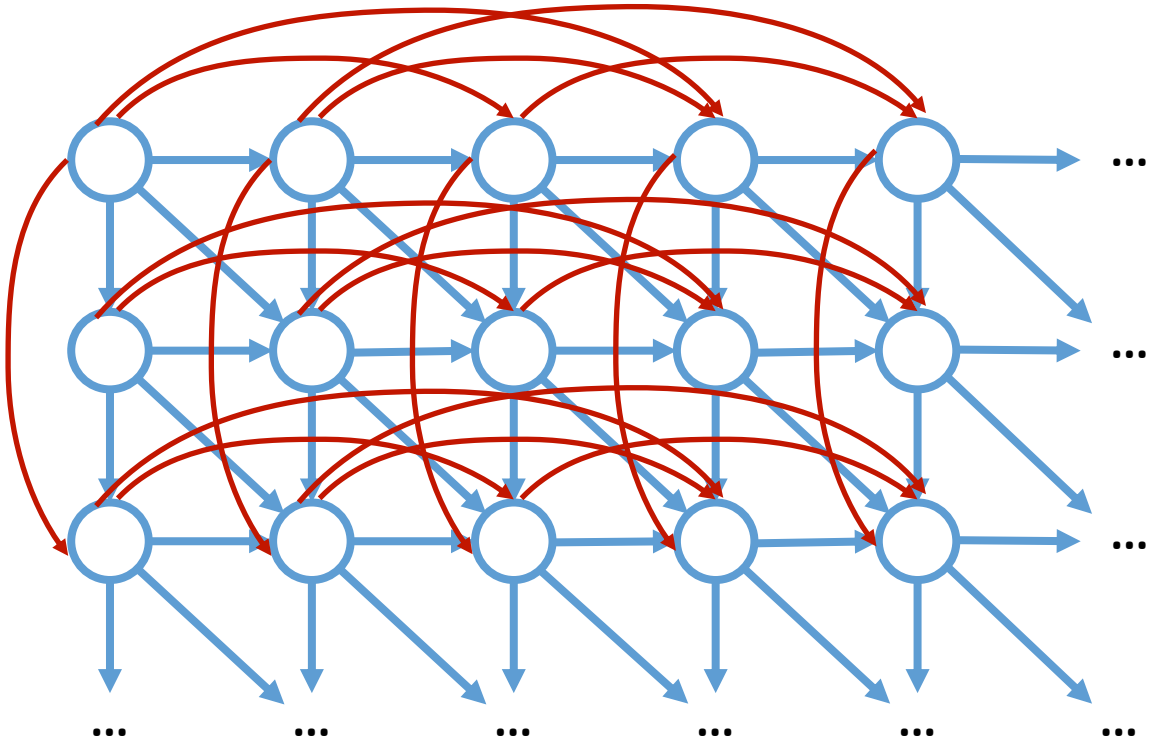
Idea: Insert horizontal (deletion) and vertical (insertion) edges spanning $k > 1$ gaps with score $-(\rho + \sigma k)$.

new edges

old edges

Affine Gap Penalty Alignment – Naive Approach

Affine gap penalty: Two penalties: (i) gap open penalty $\rho \geq 0$ and (ii) gap extension penalty $\sigma \geq 0$. Stretch of k consecutive gaps has score $-(\rho + \sigma k)$.



Idea: Insert horizontal (deletion) and vertical (insertion) edges spanning $k > 1$ gaps with score $-(\rho + \sigma k)$.

new edges

old edges

Question: What's the recurrence?

Question: What's the running time?

Affine Gap Penalty Alignment

Idea: Three separate recurrences:

(i) Gap in first sequence $s^{\rightarrow}[i,j]$

(ii) Match/mismatch $s^{\searrow}[i,j]$

(iii) Gap in second sequence $s^{\downarrow}[i,j]$

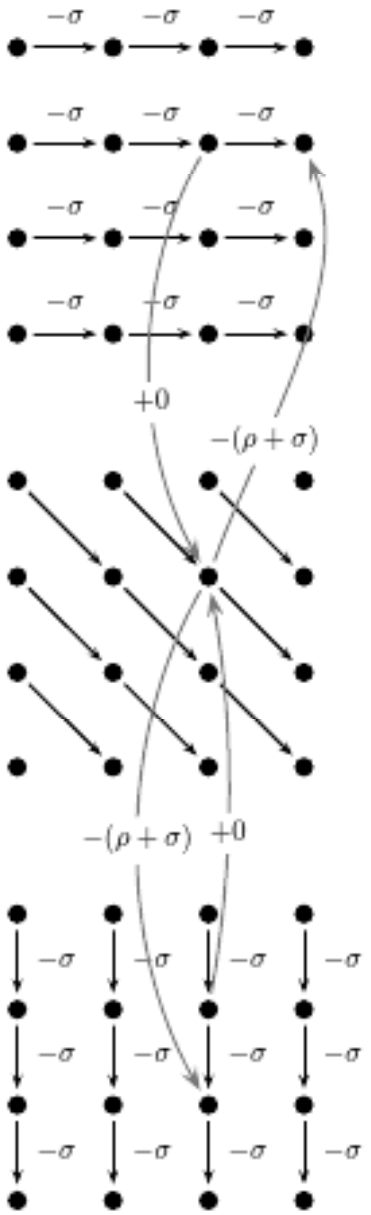


Figure 6.18 A three-level edit graph for alignment with affine gap penalties. Every vertex (i, j) in the middle level has one outgoing edge to the upper level, one outgoing edge to the lower level, and one incoming edge each from the upper and lower levels.

Affine Gap Penalty Alignment

Idea: Three separate recurrences:

(i) Gap in first sequence $s^{\rightarrow}[i,j]$

(ii) Match/mismatch $s^{\searrow}[i,j]$

(iii) Gap in second sequence $s^{\downarrow}[i,j]$

$$s^{\rightarrow}[i,j] = \max \begin{cases} s^{\rightarrow}[i,j-1] - \sigma, & \text{if } j > 1, \\ s^{\searrow}[i,j-1] - (\sigma + \rho), & \text{if } j > 0, \end{cases}$$

$$s^{\searrow}[i,j] = \max \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ s^{\rightarrow}[i,j], & \text{if } j > 0, \\ s^{\downarrow}[i,j], & \text{if } i > 0, \\ s^{\searrow}[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0, \end{cases}$$

$$s^{\downarrow}[i,j] = \max \begin{cases} s^{\downarrow}[i-1,j] - \sigma, & \text{if } i > 1, \\ s^{\searrow}[i-1,j] - (\sigma + \rho), & \text{if } i > 0. \end{cases}$$

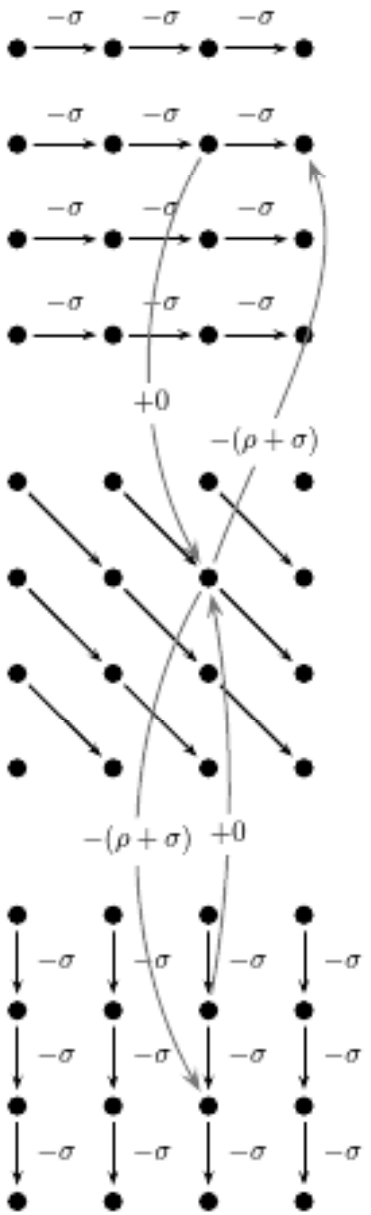


Figure 6.18 A three-level edit graph for alignment with affine gap penalties. Every vertex (i, j) in the middle level has one outgoing edge to the upper level, one outgoing edge to the lower level, and one incoming edge each from the upper and lower levels.

Affine Gap Penalty Alignment

Idea: Three separate recurrences:
(i) Gap in first sequence $s^{\rightarrow}[i,j]$
(ii) Match/mismatch $s^{\searrow}[i,j]$
(iii) Gap in second sequence $s^{\downarrow}[i,j]$

$$s^{\rightarrow}[i,j] = \max \begin{cases} s^{\rightarrow}[i,j-1] - \sigma, & \text{if } j > 1, \\ s^{\searrow}[i,j-1] - (\sigma + \rho), & \text{if } j > 0, \end{cases}$$

$$s^{\searrow}[i,j] = \max \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ s^{\rightarrow}[i,j], & \text{if } j > 0, \\ s^{\downarrow}[i,j], & \text{if } i > 0, \\ s^{\searrow}[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0, \end{cases}$$

$$s^{\downarrow}[i,j] = \max \begin{cases} s^{\downarrow}[i-1,j] - \sigma, & \text{if } i > 1, \\ s^{\searrow}[i-1,j] - (\sigma + \rho), & \text{if } i > 0. \end{cases}$$

Running time: $O(mn)$

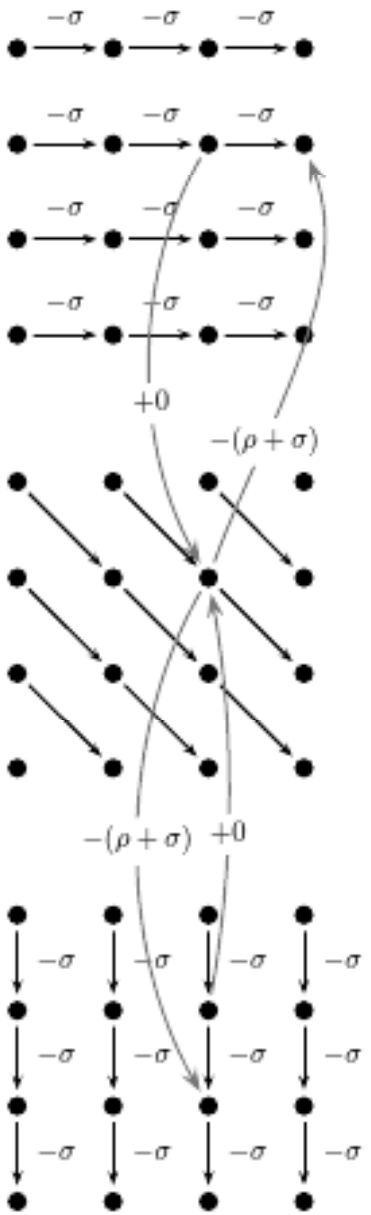


Figure 6.18 A three-level edit graph for alignment with affine gap penalties. Every vertex (i, j) in the middle level has one outgoing edge to the upper level, one outgoing edge to the lower level, and one incoming edge each from the upper and lower levels.

Affine Gap Penalty Alignment – Example

Let $\rho = 10$ and $\sigma = 1$. Match = 1. Mismatch = -1

$\mathbf{v} = \text{AAC}$ $\mathbf{w} = \text{ACAAC}$

$$s^{\rightarrow}[i,j] = \max \begin{cases} s^{\rightarrow}[i,j-1] - \sigma, & \text{if } j > 1, \\ s^{\searrow}[i,j-1] - (\sigma + \rho), & \text{if } j > 0, \end{cases}$$

$$s^{\searrow}[i,j] = \max \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, \\ s^{\rightarrow}[i,j], & \text{if } j > 0, \\ s^{\downarrow}[i,j], & \text{if } i > 0, \\ s^{\searrow}[i-1,j-1] + \delta(v_i, w_j), & \text{if } i > 0 \text{ and } j > 0, \end{cases}$$

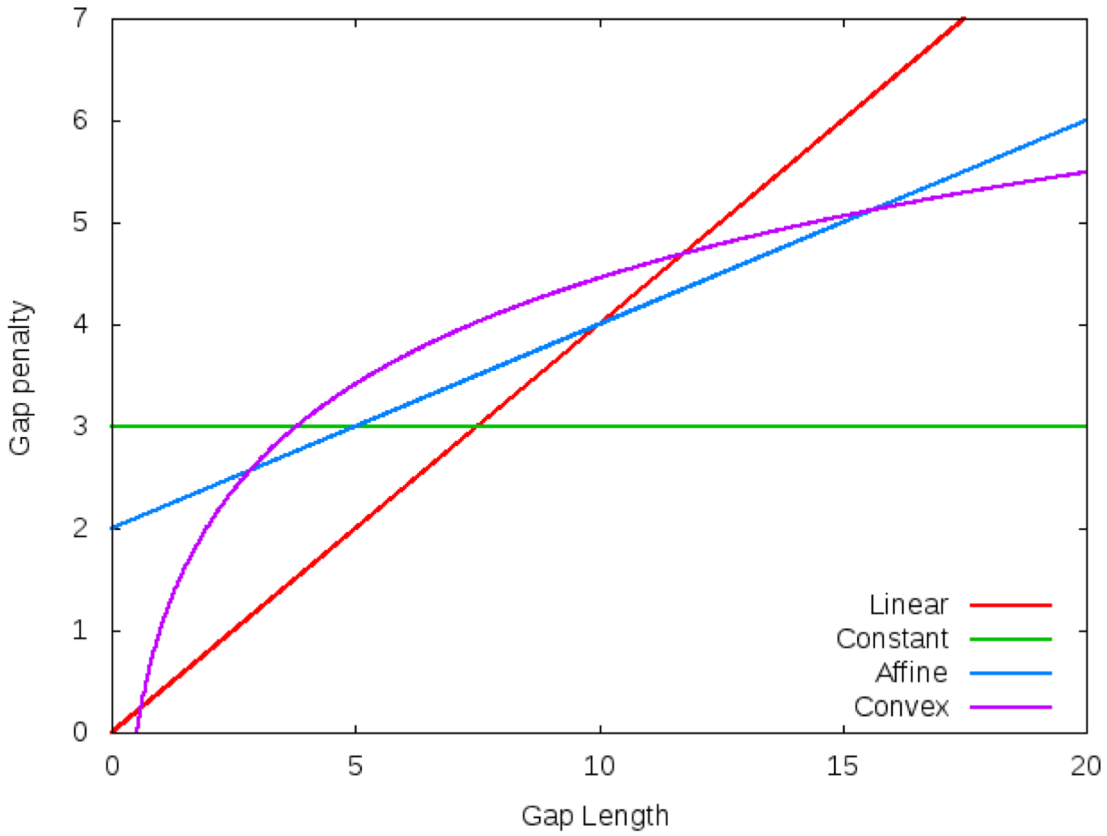
$$s^{\downarrow}[i,j] = \max \begin{cases} s^{\downarrow}[i-1,j] - \sigma, & \text{if } i > 1, \\ s^{\searrow}[i-1,j] - (\sigma + \rho), & \text{if } i > 0. \end{cases}$$

Gapped Alignment – Additional Insights

- Naive approach supports arbitrary gap penalties given two sequences $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$. This results in an $O(mn(m+n))$ algorithm.

- Alignment with **convex gap penalties** given two sequences $\mathbf{v} \in \Sigma^m$ and $\mathbf{w} \in \Sigma^n$ can be computed in $O(mn \log m)$ time.

See: Dan Gusfield. 1997. Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology. Cambridge University Press, New York, NY, USA.



Take Home Messages

1. Global alignment
2. Fitting alignment
3. Local alignment
4. Gapped alignment

Global alignment is longest path in DAG

Small tweaks enable different extensions

Reading:

- Jones and Pevzner. Chapters 6.7-6.9
- Lecture notes