CS 466
Introduction to Bioinformatics
Lecture 9

Mohammed El-Kebir

October 1, 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

Homework 2 due Oct. 5 by 11:59pm

Midterm on Oct. 10, 7-9pm, 1310 DCL

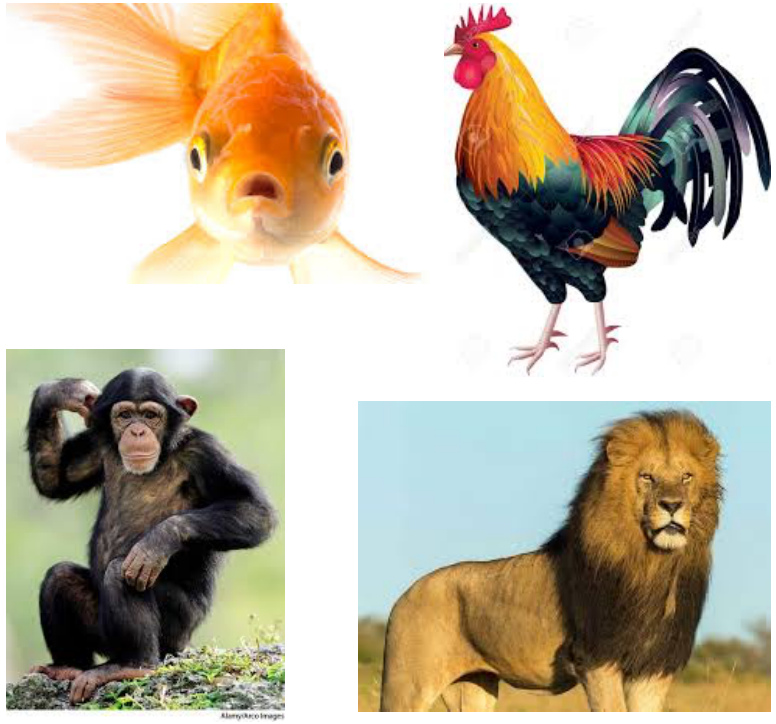
Outline

- Multiple sequence alignment
- Exact algorithm
- Sum-of-pairs (SP) score
- Carillo-Lipman
- Heuristic approaches

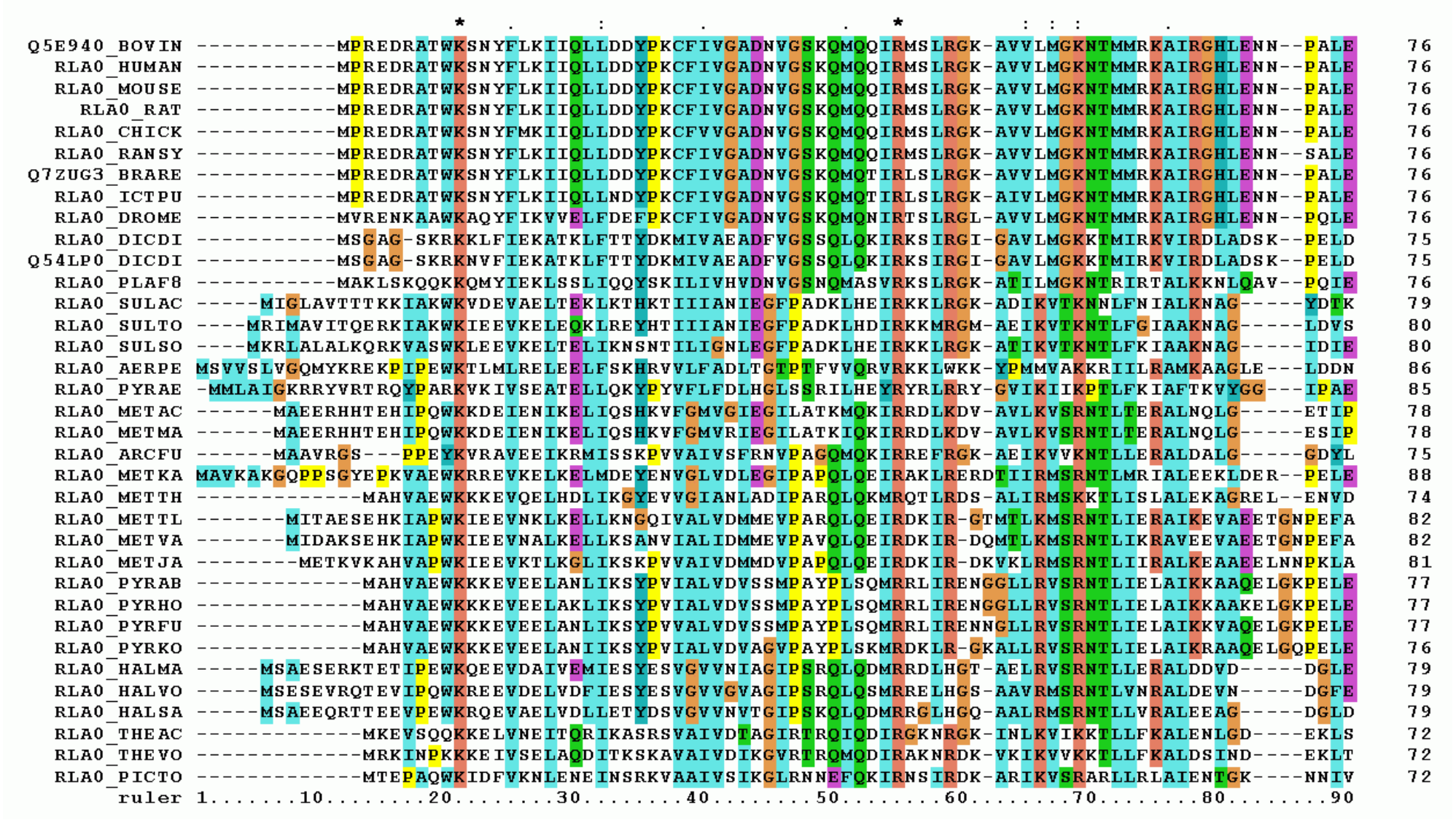
Reading:

- Material based on Chapter 14.6 in book “Algorithms on Strings, Trees and Sequences” by Dan Gusfield

Motivation



Simultaneous alignment of multiple (> 2) sequences enables inference of subtle similarities that are conserved in more than two species



"Pairwise alignment whispers ... multiple alignment shouts out loud".
Hubbard, Lesk, Tramontano, Nature Structural Biology 1996.

Multiple Sequence Alignment (MSA)

A **multiple sequence alignment** $\mathcal{M}$ between k strings $\mathbf{v}_1, \ldots, \mathbf{v}_k$ is a $k \times q$ matrix, where $q = \{\max\{\, |\mathbf{v}_i| : i \in [k] \,\}, \ldots, \sum_{i=1}^{k} |\mathbf{v}_i|\}$ such that the i-th row contains the characters of $\mathbf{v}_i$ in order with spaces '-' interspersed and no column contains k spaces

$\mathbf{v}_1$	A	T	–	G	C	G	–
$\mathbf{v}_2$	A	–	C	G	T	–	C
$\mathbf{v}_3$	A	T	C	A	C	–	A

Question: How to score a multiple sequence alignment?

Scoring a Multiple Sequence Alignment

$\mathbf{v}_1$	A	T	–	G	C	G	–
$\mathbf{v}_2$	A	–	C	G	T	–	C
$\mathbf{v}_3$	A	T	C	A	C	–	A

Question: How to score a multiple sequence alignment?

Pairwise scoring function:

$$\delta : (\Sigma \cup \{-\}) \times (\Sigma \cup \{-\}) \rightarrow \mathbb{R}$$

Scoring a Multiple Sequence Alignment

$\mathbf{v}_1$	A	T	–	G	C	G	–
$\mathbf{v}_2$	A	–	C	G	T	–	C
$\mathbf{v}_3$	A	T	C	A	C	–	A

Question: How to score a multiple sequence alignment?

Pairwise scoring function:

$$\delta : (\Sigma \cup \{-\}) \times (\Sigma \cup \{-\}) \to \mathbb{R}$$

k-wise scoring function:

$$\delta : (\Sigma \cup \{-\})^k \to \mathbb{R}$$

Outline

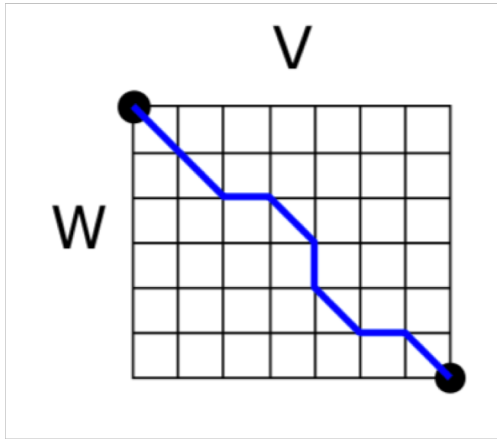
- Multiple sequence alignment
- Exact algorithm
- Sum-of-pairs (SP) score
- Carillo-Lipman
- Heuristic approaches

Reading:

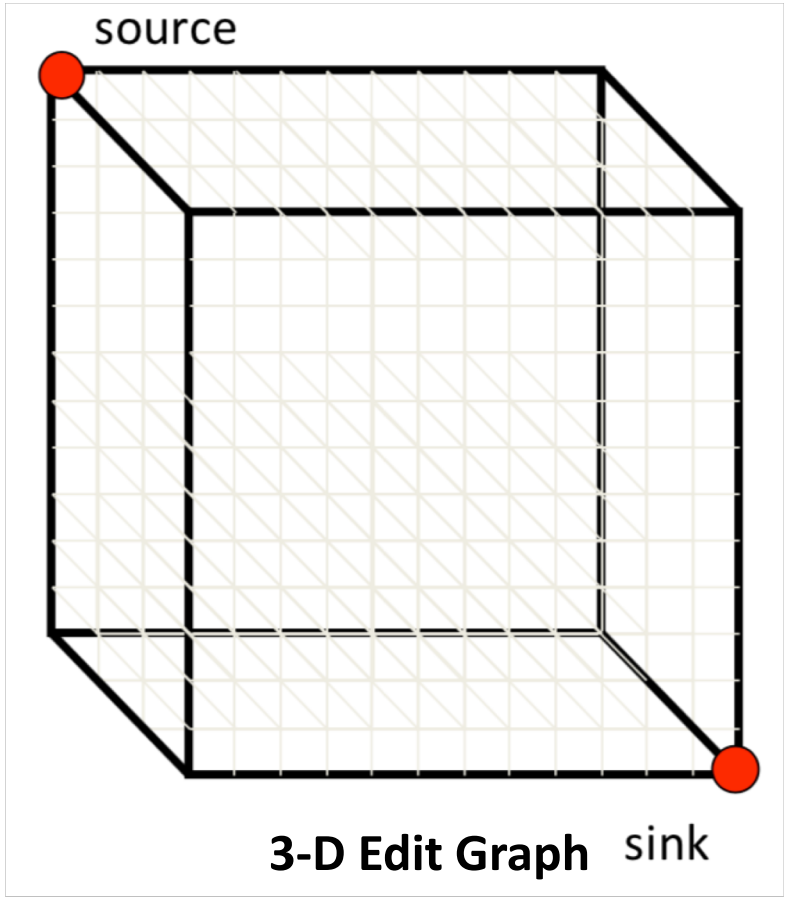
- Material based on Chapter 14.6 in book “Algorithms on Strings, Trees and Sequences” by Dan Gusfield

Aligning Three Sequences

- Same strategy as pairwise edit distance
- Use 3-D cube, with each axis representing an input sequence
- Alignment is a path from source to sink

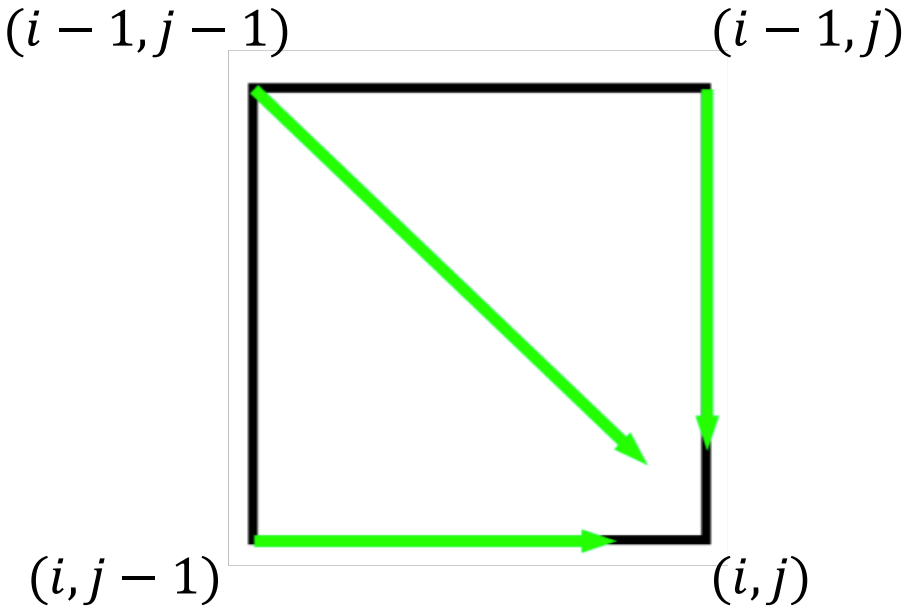


2-D Edit Graph

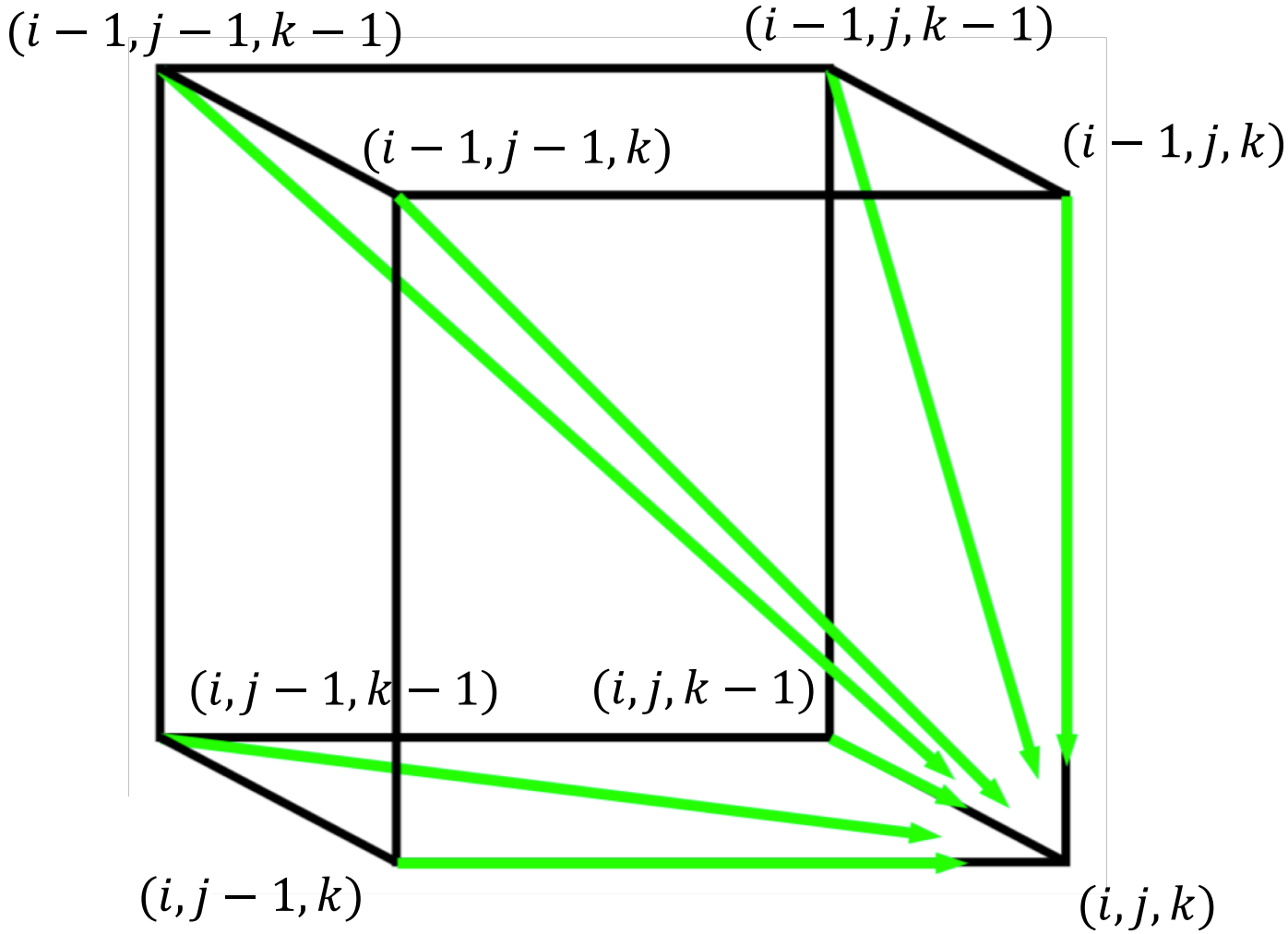


3-D Edit Graph

2-D vs 3-D Vertex Neighborhood



2-D Neighborhood
(3 edges)

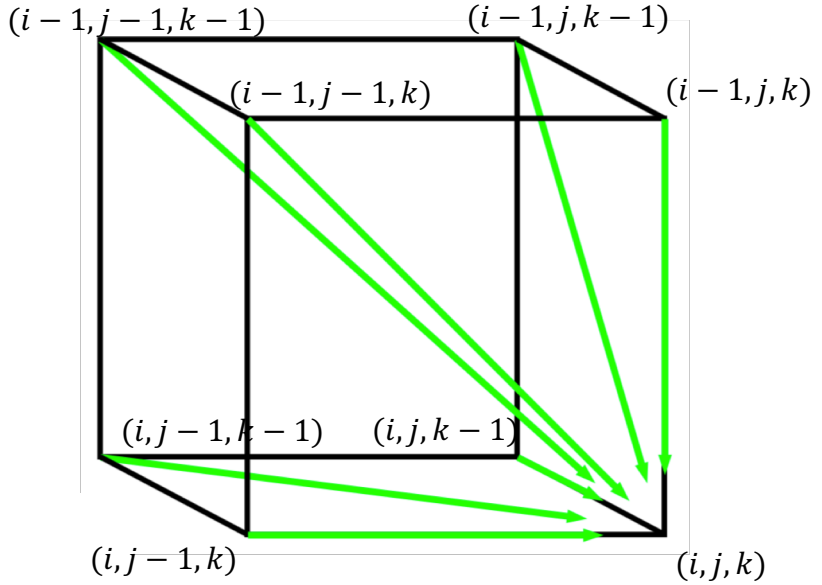


3-D Neighborhood
(7 edges)

3-D Sequence Alignment

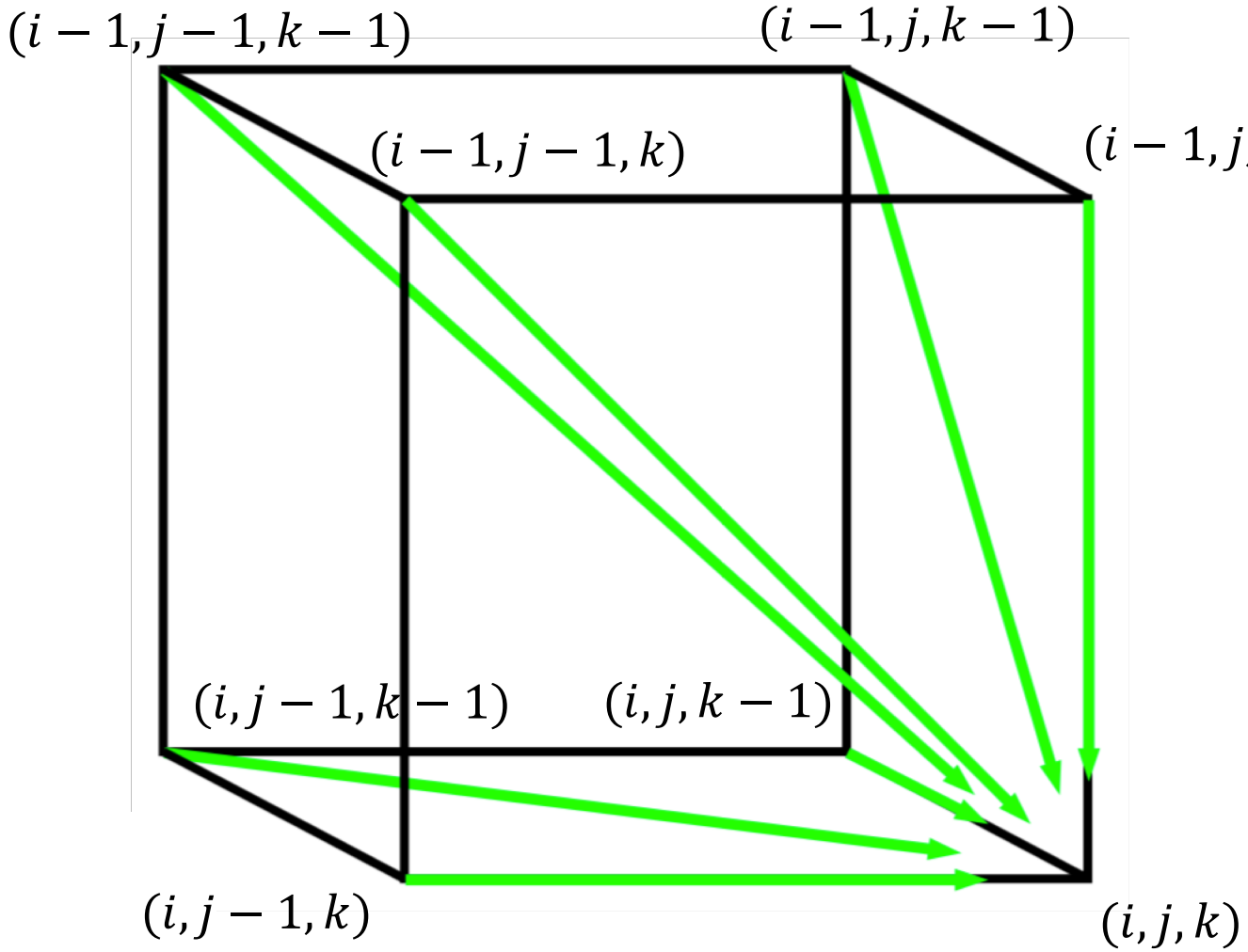
$\delta(x, y, z)$ is an entry in 3-D scoring matrix

Given three sequences each of length n, running time: $O(n^3)$



$$s[i,j,k] = \max \begin{cases} s[i-1,j-1,k-1] + \delta(v_i, w_j, u_k), & \text{no gaps} \\ s[i-1,j-1,k] + \delta(v_i, w_j, -), & \\ s[i-1,j,k-1] + \delta(v_i, -, u_k), & \text{one gap} \\ s[i,j-1,k-1] + \delta(-, w_j, u_k), & \\ s[i-1,j,k] + \delta(v_i, -, -), & \\ s[i,j-1,k] + \delta(-, w_j, -), & \text{two gaps} \\ s[i,j,k-1] + \delta(-, -, u_k), & \end{cases}$$

3-D vs k-D Vertex Neighborhood



3-D Neighborhood
(7 edges)

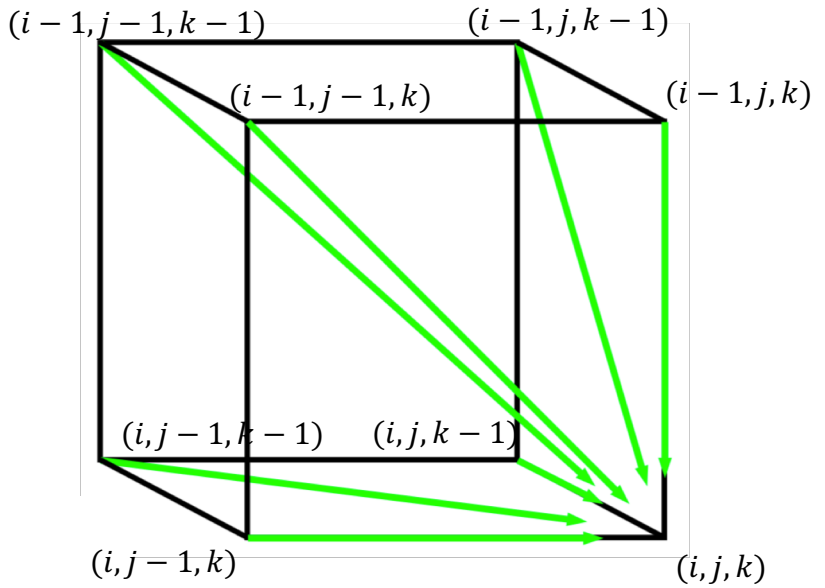
$$(i_1 - 1, i_2 - 1, \dots, i_{k-1} - 1, i_k - 1)$$
$$(i_1 - 1, i_2 - 1, \dots, i_{k-1} - 1, i_k)$$
$$\dots$$
$$(i_1, i_2 - 1, \dots, i_{k-1} - 1, i_k - 1)$$
$$\dots$$
$$(i_1 - 1, i_2, \dots, i_{k-1}, i_k)$$
$$\dots$$
$$(i_1, i_2, \dots, i_{k-1}, i_k - 1)$$

k-D Neighborhood
($2^k - 1$ edges)

k-D Sequence Alignment

$\delta(x_1, \ldots, x_k)$ is an entry in k-D scoring matrix

Given k sequences each of length n,
running time: $O(2^k n^k)$



$$s[i_1, i_2, \ldots, i_{k-1}, i_k] = \max \begin{cases} s[i_1-1, i_2-1, \ldots, i_{k-1}-1, i_k-1] + \delta(\mathbf{v_1}[i_1], \mathbf{v_2}[i_2], \ldots, \mathbf{v_{k-1}}[i_{k-1}], \mathbf{v_k}[i_k]) \\ s[i_1-1, i_2-1, \ldots, i_{k-1}-1, i_k] + \delta(\mathbf{v_1}[i_1], \mathbf{v_2}[i_2], \ldots, \mathbf{v_{k-1}}[i_{k-1}], -) \\ \quad \vdots \\ s[i_1, i_2-1, \ldots, i_{k-1}-1, i_k-1] + \delta(-, \mathbf{v_2}[i_2], \ldots, \mathbf{v_{k-1}}[i_{k-1}], \mathbf{v_k}[i_k]) \\ \quad \vdots \\ s[i_1-1, i_2, \ldots, i_{k-1}, i_k] + \delta(\mathbf{v_1}[i_1], -, \ldots, -, -) \\ \quad \vdots \\ s[i_1, i_2, \ldots, i_{k-1}, i_k-1] + \delta(-, -, \ldots, -, \mathbf{v_k}[i_k]) \end{cases}$$

no gaps

one gap

$k-1$ gaps

Multiple Sequence Alignment – Running Time

Given 2 sequences each of length n, running time: $O(n^2)$

Given 3 sequences each of length n, running time: $O(n^3)$

Given k sequences each of length n, running time: $O(2^k n^k)$

Multiple Sequence Alignment – Running Time

Given 2 sequences each of length n, running time: $O(n^2)$

Given 3 sequences each of length n, running time: $O(n^3)$

Given k sequences each of length n, running time: $O(2^k n^k)$

Question: Can we align k sequences each of length n in time $O(\text{poly}(n))$?

Multiple Sequence Alignment – Running Time

Given 2 sequences each of length n, running time: $O(n^2)$

Given 3 sequences each of length n, running time: $O(n^3)$

Given k sequences each of length n, running time: $O(2^k n^k)$

Question: Can we align k sequences each of length n in time $O(\text{poly}(n))$?

Let's look at a more wieldy scoring function

Outline

- Multiple sequence alignment
- Exact algorithm
- Sum-of-pairs (SP) score
- Carillo-Lipman
- Heuristic approaches

Reading:

- Material based on Chapter 14.6 in book “Algorithms on Strings, Trees and Sequences” by Dan Gusfield

Multiple Alignment Induces Pairwise Alignments

v_1	A	T	-	G	C	G	-
v_2	A	-	C	G	T	-	C
v_3	A	T	C	A	C	-	A

v_1	A	T	-	G	C	G	-
v_2	A	-	C	G	T	-	C

v_1	A	T	-	G	C	G	-
v_3	A	T	C	A	C	-	A

v_2	A	-	C	G	T	-	C
v_3	A	T	C	A	C	-	A

Resulting columns with -/- are removed

Sum-of-Pairs (SP) Score

$\mathbf{v}_1$	A	T	–	G	C	G	–
$\mathbf{v}_2$	A	–	C	G	T	–	C
$\mathbf{v}_3$	A	T	C	A	C	–	A

Multiple sequence alignment $\mathcal{M}$

$S(\mathbf{v}_i, \mathbf{v}_j)$ is score of induced pairwise alignment of sequences $(\mathbf{v}_i, \mathbf{v}_j)$

$\mathbf{v}_1$	A	T	–	G	C	G	–
$\mathbf{v}_2$	A	–	C	G	T	–	C

$\mathbf{v}_1$	A	T	–	G	C	G	–
$\mathbf{v}_3$	A	T	C	A	C	–	A

$\mathbf{v}_2$	A	–	C	G	T	–	C
$\mathbf{v}_3$	A	T	C	A	C	–	A

$$\text{SP-score}(\mathcal{M}) = \sum_{i=1}^{k} \sum_{j=i+1}^{k} S(\mathbf{v}_i, \mathbf{v}_j)$$

Sum-of-Pairs (SP) Score – Example

$\mathbf{v}_1$	A	T	G	–	C
$\mathbf{v}_2$	A	–	G	–	C
$\mathbf{v}_3$	A	T	C	C	C

Multiple sequence alignment $\mathcal{M}$

Match score: 3
Mismatch score: 1
Gap score: $-\sigma$

Question: Calculate
SP-score$(\mathcal{M}) =$
$\sum_{i=1}^{k} \sum_{j=i+1}^{k} S(\mathbf{v}_i, \mathbf{v}_j)$

Sum-of-Pairs (SP) Score – Example

$\mathbf{v}_1$	A	T	G	–	C
$\mathbf{v}_2$	A	–	G	–	C
$\mathbf{v}_3$	A	T	C	C	C

Multiple sequence alignment $\mathcal{M}$

Match score: 3
Mismatch score: 1
Gap score: $-\sigma$

Question: Calculate
SP-score$(\mathcal{M}) =$
$\sum_{i=1}^{k} \sum_{j=i+1}^{k} S(\mathbf{v}_i, \mathbf{v}_j)$

We can sum over scores for the columns, ignoring -/-

Multiple Sequence Alignment Problem w/ SP-Score

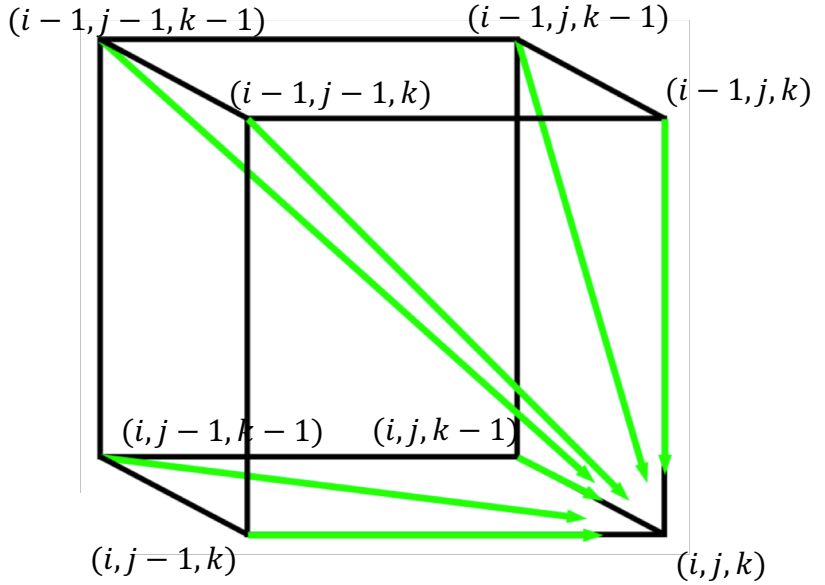
A **multiple sequence alignment** $\mathcal{M}$ between k strings $\mathbf{v}_1, \ldots, \mathbf{v}_k$ is a $k \times q$ matrix, where $q = \{\max\{ |\mathbf{v}_i| : i \in [k]\}, \ldots, \sum_{i=1}^{k} |\mathbf{v}_i|\}$ such that the i-th row contains the characters of $\mathbf{v}_i$ in order with spaces '-' interspersed and no column contains k spaces

MSA-SP problem: Given strings strings $\mathbf{v}_1, \ldots, \mathbf{v}_k$ find multiple sequence alignment $\mathcal{M}^*$ with **minimum** value of SP-score$(\mathcal{M}^*) = \sum_{i=1}^{k} \sum_{j=i+1}^{k} S(\mathbf{v}_i, \mathbf{v}_j)$ where $S(\mathbf{v}_i, \mathbf{v}_j)$ is the score of the induced pairwise alignment of $(\mathbf{v}_i, \mathbf{v}_j)$ in $\mathcal{M}^*$

3-D MSA-SP

$\delta(x, y, z)$ is an entry in 3-D scoring matrix

Given three sequences each of length n, running time: $O(n^3)$

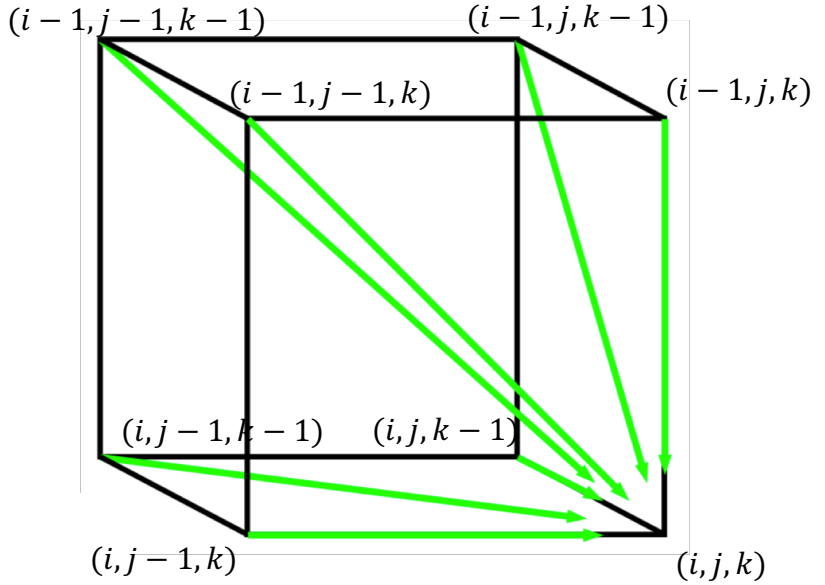


$$d[i_1, i_2, i_3] = \min \begin{cases} d[i_1-1, i_2-1, i_3-1] + \delta(\mathbf{v_1}[i_1], \mathbf{v_2}[i_2]) + \delta(\mathbf{v_1}[i_1], \mathbf{v_3}[i_3]) + \delta(\mathbf{v_2}[i_2], \mathbf{v_3}[i_3]) & \text{no gaps} \\ d[i_1-1, i_2-1, i_3] + \delta(\mathbf{v_1}[i_1], \mathbf{v_2}[i_2]) + 2\sigma \\ d[i_1-1, i_2, i_3-1] + \delta(\mathbf{v_1}[i_1], \mathbf{v_3}[i_3]) + 2\sigma \\ d[i_1, i_2-1, i_3-1] + \delta(\mathbf{v_2}[i_2], \mathbf{v_3}[i_3]) + 2\sigma & \text{one gap} \\ d[i_1-1, i_2, i_3] + 2\sigma \\ d[i_1, i_2-1, i_3] + 2\sigma \\ d[i_1, i_2, i_3-1] + 2\sigma & \text{two gaps} \end{cases}$$

k-D MSA-SP

Computing SP-score in each case: $O(k^2)$ time

Given k sequences each of length n, running time: $O(k^2 2^k n^k)$

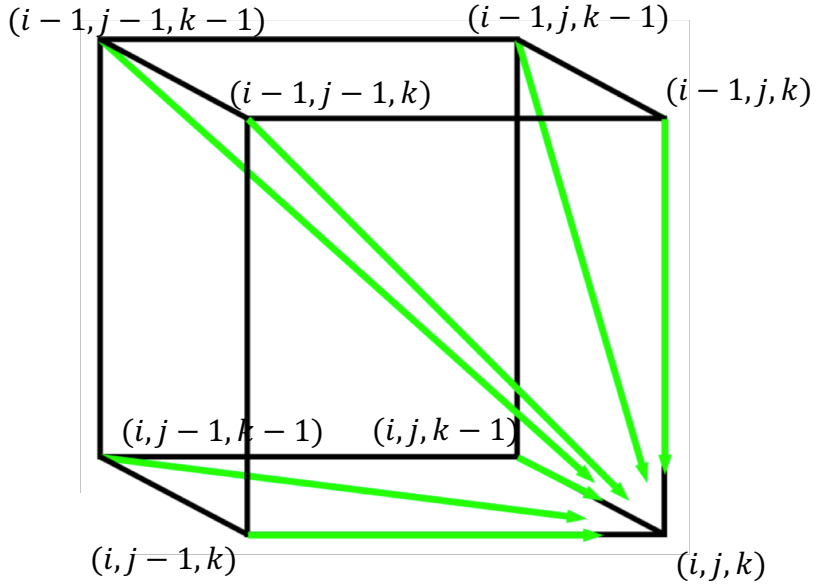


$$d[i_1,i_2,\ldots,i_{k-1},i_k] = \min \begin{cases} s[i_1-1,i_2-1,\ldots,i_{k-1}-1,i_k-1] + \sum_{p=1}^{k}\sum_{q=p+1}^{k}\delta(\mathbf{v_p}[i_p],\mathbf{v_q}[i_q]) & \text{no gaps} \\ s[i_1-1,i_2-1,\ldots,i_{k-1}-1,i_k] + (k-1)\sigma + \sum_{p=1}^{k-1}\sum_{q=p+1}^{k-1}\delta(\mathbf{v_p}[i_p],\mathbf{v_q}[i_q]) & \\ \quad\vdots & \text{one gap} \\ s[i_1,i_2-1,\ldots,i_{k-1}-1,i_k-1] + (k-1)\sigma + \sum_{p=2}^{k}\sum_{q=p+1}^{k}\delta(\mathbf{v_p}[i_p],\mathbf{v_q}[i_q]) & \\ \quad\vdots & \\ s[i_1-1,i_2,\ldots,i_{k-1},i_k] + (k-1)\sigma & \\ \quad\vdots & k-1 \text{ gaps} \\ s[i_1,i_2,\ldots,i_{k-1},i_k-1] + (k-1)\sigma & \end{cases}$$

k-D MSA-SP

Computing SP-score in each case: $O(k^2)$ time

Given k sequences each of length n, running time: $O(k^2 2^k n^k)$



$$d[i_1,i_2,\ldots,i_{k-1},i_k] = \min \begin{cases} s[i_1-1,i_2-1,\ldots,i_{k-1}-1,i_k-1] + \sum_{p=1}^{k}\sum_{q=p+1}^{k}\delta(\mathbf{v_p}[i_p],\mathbf{v_q}[i_q]) & \text{no gaps} \\ s[i_1-1,i_2-1,\ldots,i_{k-1}-1,i_k] + (k-1)\sigma + \sum_{p=1}^{k-1}\sum_{q=p+1}^{k-1}\delta(\mathbf{v_p}[i_p],\mathbf{v_q}[i_q]) & \\ \quad\vdots & \text{one gap} \\ s[i_1,i_2-1,\ldots,i_{k-1}-1,i_k-1] + (k-1)\sigma + \sum_{p=2}^{k}\sum_{q=p+1}^{k}\delta(\mathbf{v_p}[i_p],\mathbf{v_q}[i_q]) & \\ \quad\vdots & \\ s[i_1-1,i_2,\ldots,i_{k-1},i_k] + (k-1)\sigma & \\ \quad\vdots & k-1 \text{ gaps} \\ s[i_1,i_2,\ldots,i_{k-1},i_k-1] + (k-1)\sigma & \end{cases}$$

Question: How many times gap penalty with 2 gaps?

Multiple Sequence Alignment Problem w/ SP-Score

MSA-SP problem: Given strings strings $\mathbf{v}_1, \ldots, \mathbf{v}_k$ find multiple sequence alignment $\mathcal{M}^*$ with **minimum** value of SP-score$(\mathcal{M}^*) = \sum_{i=1}^{k} \sum_{j=i+1}^{k} S(\mathbf{v}_i, \mathbf{v}_j)$ where $S(\mathbf{v}_i, \mathbf{v}_j)$ is the score of the induced pairwise alignment of $(\mathbf{v}_i, \mathbf{v}_j)$ in $\mathcal{M}^*$

Question: Can we align k sequences each of length n in time $O(\text{poly}(n))$?

Multiple Sequence Alignment Problem w/ SP-Score

MSA-SP problem: Given strings strings $\mathbf{v}_1, \ldots, \mathbf{v}_k$ find multiple sequence alignment $\mathcal{M}^*$ with **minimum** value of SP-score$(\mathcal{M}^*) = \sum_{i=1}^{k} \sum_{j=i+1}^{k} S(\mathbf{v}_i, \mathbf{v}_j)$ where $S(\mathbf{v}_i, \mathbf{v}_j)$ is the score of the induced pairwise alignment of $(\mathbf{v}_i, \mathbf{v}_j)$ in $\mathcal{M}^*$

Question: Can we align k sequences each of length n in time $O(\text{poly}(n))$?

No, MSA-SP is NP-hard.

[WANG, L., & JIANG, T. (2009). On the Complexity of Multiple Sequence Alignment. *Journal of Computational Biology*, *1*(4), 337–348. http://doi.org/10.1089/cmb.1994.1.337]

Outline

- Multiple sequence alignment
- Exact algorithm
- Sum-of-pairs (SP) score
- Carillo-Lipman
- Heuristic approaches

Reading:

- Material based on Chapter 14.6 in book “Algorithms on Strings, Trees and Sequences” by Dan Gusfield

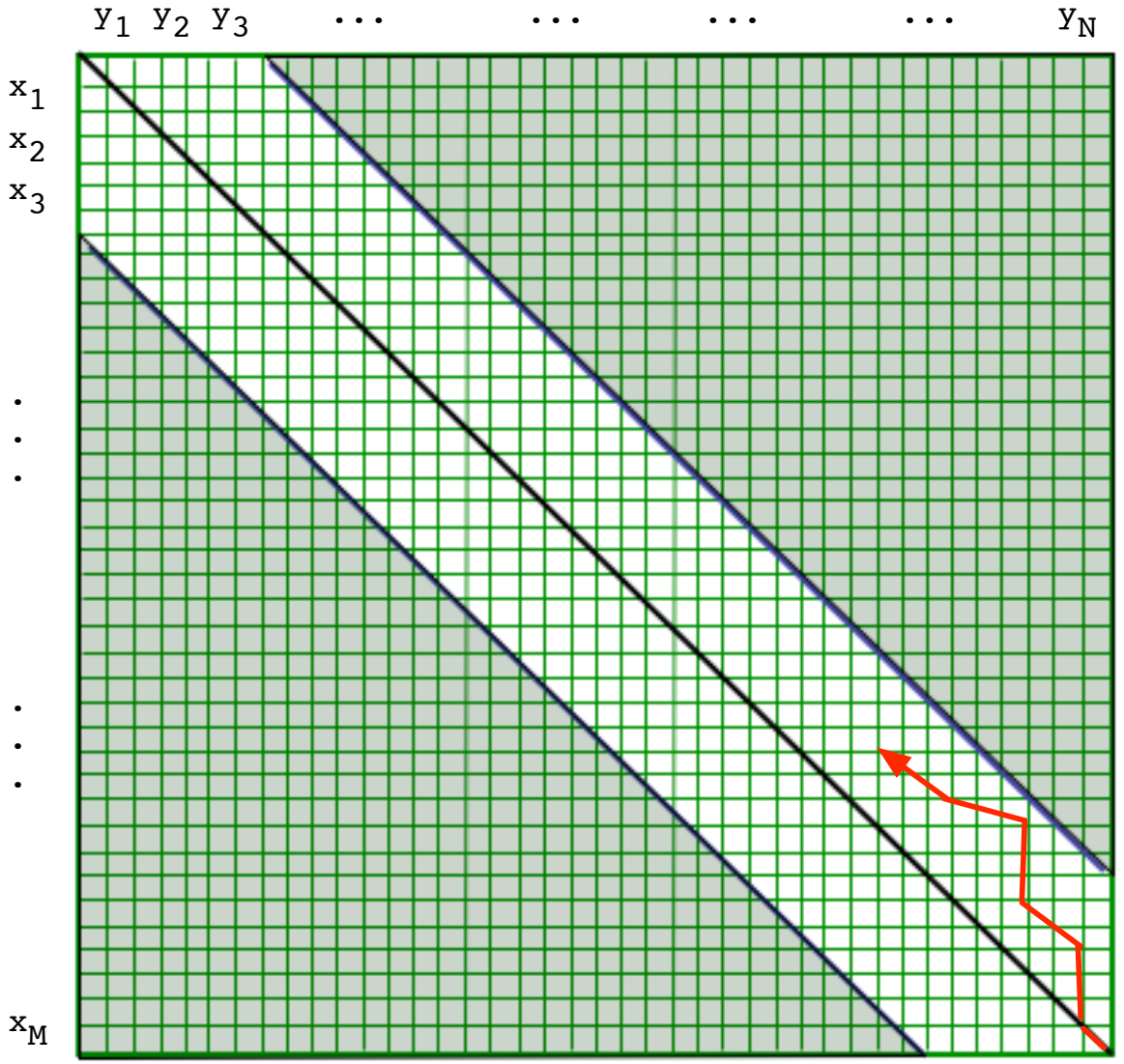
Recall: Banded Alignment

Alignment is a path from source $(0, 0)$ to target (m, n) in edit graph

Constraint path to band of width k around diagonal

Running time: $O(nk)$

Question: Alternative ways of constraining search space?



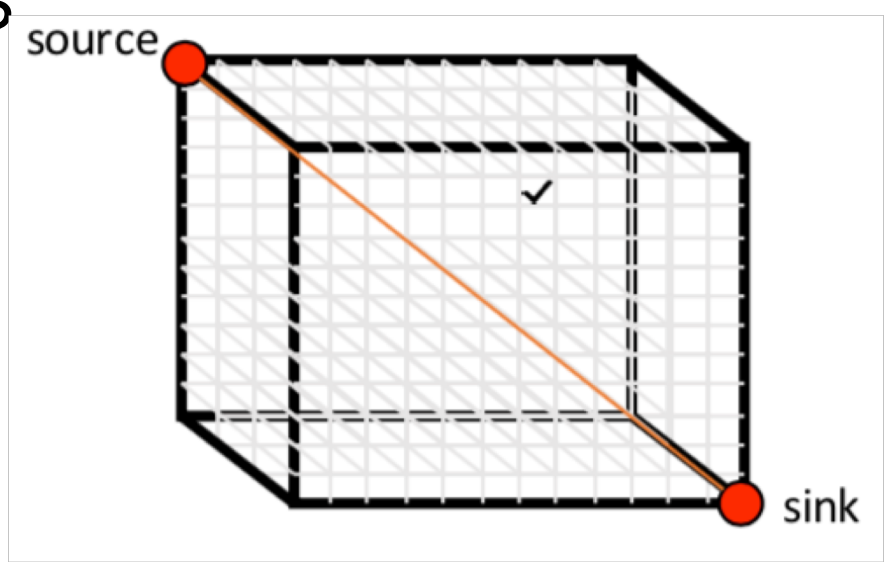
Constrain traceback to band of DP matrix (penalize big gaps)

Figure source: http://jinome.stanford.edu/stat366/pdfs/stat366_win0607_lecture04.pdf

Forward Dynamic Programming

Banded alignment: constraint path to polyhedron around diagonal

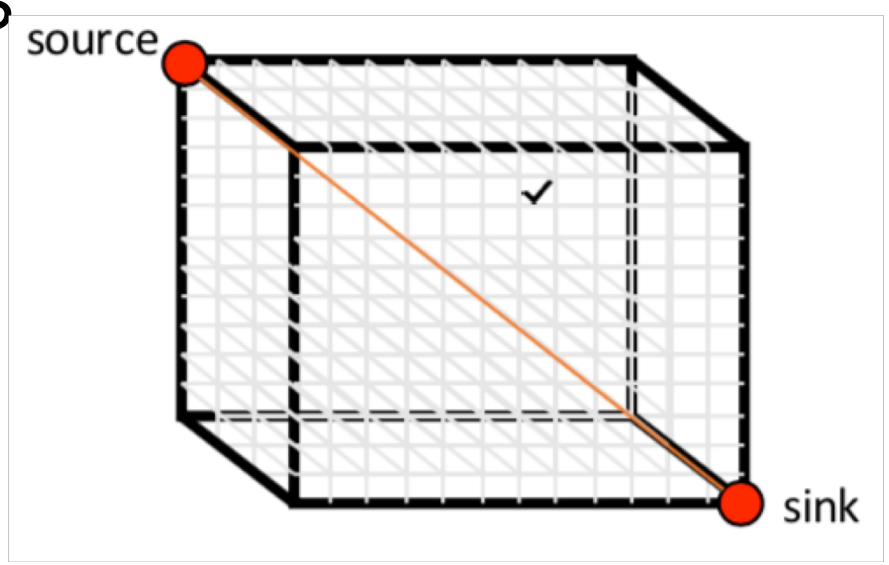
Alternatively: Stop computing when remaining alignment will be suboptimal

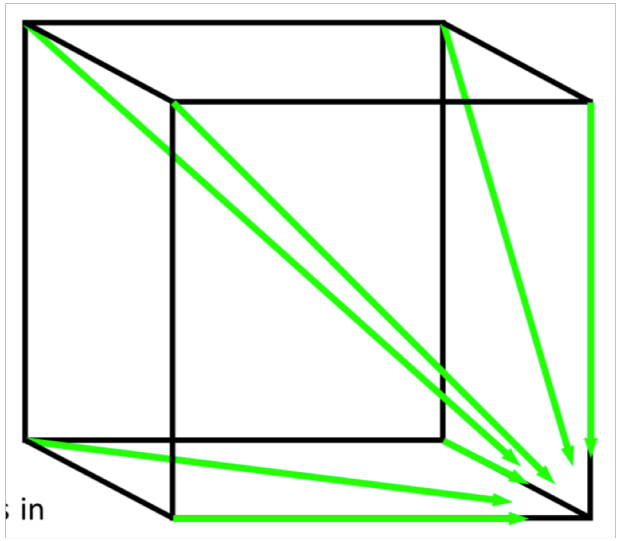


Forward Dynamic Programming

Banded alignment: constraint path to polyhedron around diagonal

Alternatively: Stop computing when remaining alignment will be suboptimal





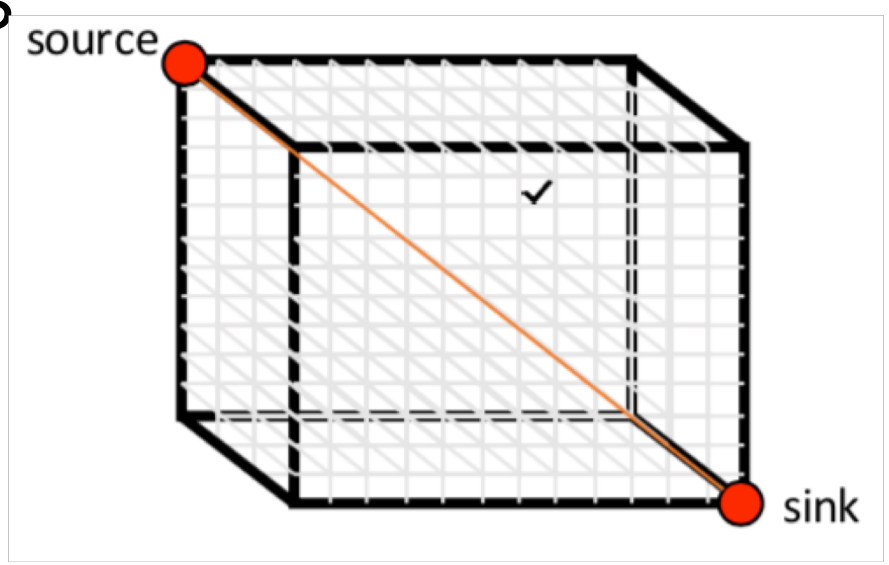
Forward dynamic programming – think of Dijkstra's algorithm:

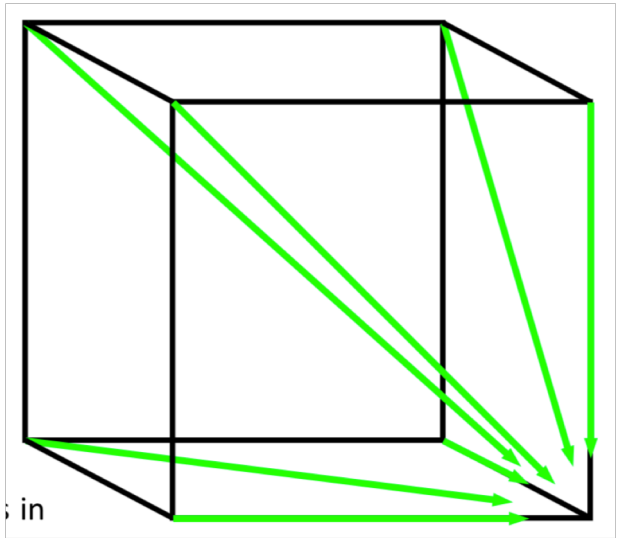
- Queue of unvisited vertices
- Maintain $p[i,j,k]$ shortest distance yet found from $(0,0,0)$ to (i,j,k).
- For each directed edge (i,j,k) to (i',j',k') with cost w, set $p[i',j',k'] = \min\{p[i',j',k'], p[i,j,k] + w\}$

Forward Dynamic Programming

Banded alignment: constraint path to polyhedron around diagonal

Alternatively: Stop computing when remaining alignment will be suboptimal





Forward dynamic programming – think of Dijkstra's algorithm:

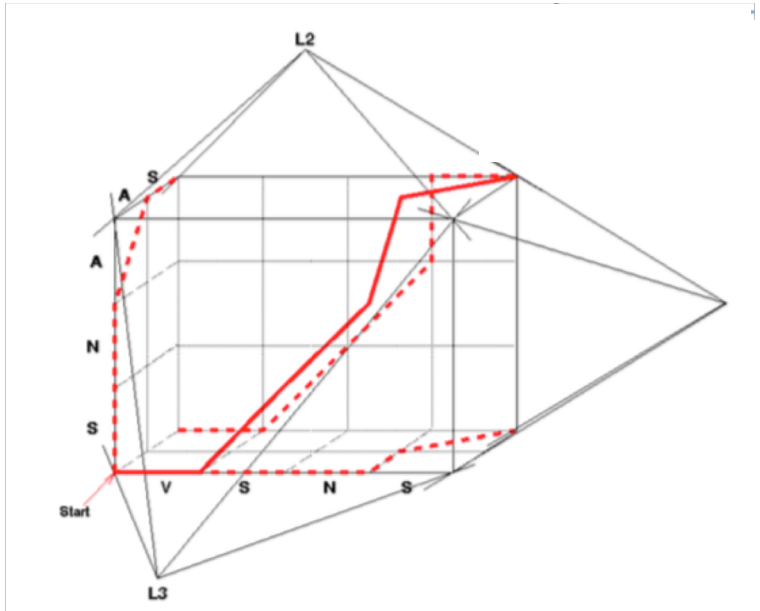
- Queue of unvisited vertices
- Maintain $p[i,j,k]$ shortest distance yet found from $(0,0,0)$ to (i,j,k).
- For each directed edge (i,j,k) to (i',j',k') with cost w, set $p[i',j',k'] = \min\{p[i',j',k'], p[i,j,k] + w\}$

Question: Can we remove vertices from consideration based on alignment score of prefix?

Alignment Projection and SP-score

Sequences $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ each of length n

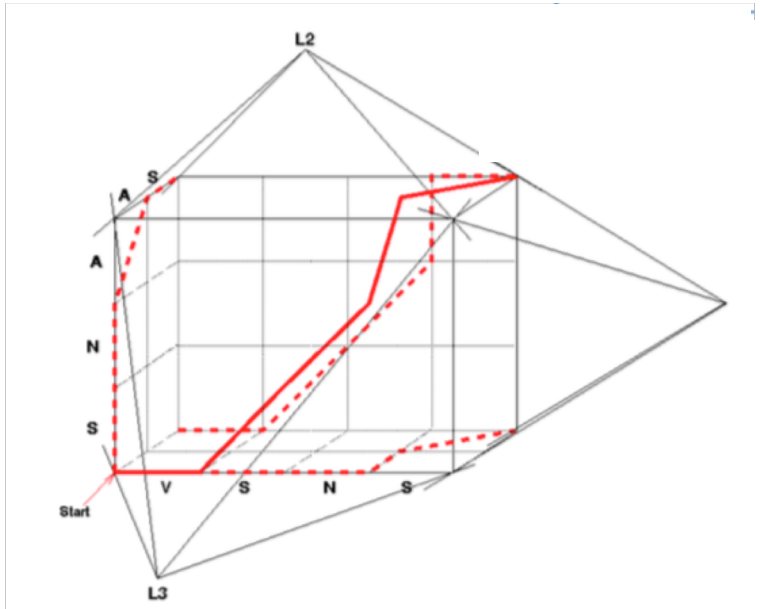
- $D(i,j,k)$ is min SP-cost of aligning $\mathbf{v}_1[1..i], \mathbf{v}_2[1..j], \mathbf{v}_3[1..k]$
- $d_{p,q}(i,j)$ is cost of induced alignment of $\mathbf{v}_p[1..i], \mathbf{v}_q[1..j]$
- $D_{p,q}(i,j)$ is min cost of aligning $\mathbf{v}_p[1..i], \mathbf{v}_q[1..j]$



Alignment Projection and SP-score

Sequences $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ each of length n

- $D(i,j,k)$ is min SP-cost of aligning $\mathbf{v}_1[1..i], \mathbf{v}_2[1..j], \mathbf{v}_3[1..k]$
- $d_{p,q}(i,j)$ is cost of induced alignment of $\mathbf{v}_p[1..i], \mathbf{v}_q[1..j]$
- $D_{p,q}(i,j)$ is min cost of aligning $\mathbf{v}_p[1..i], \mathbf{v}_q[1..j]$

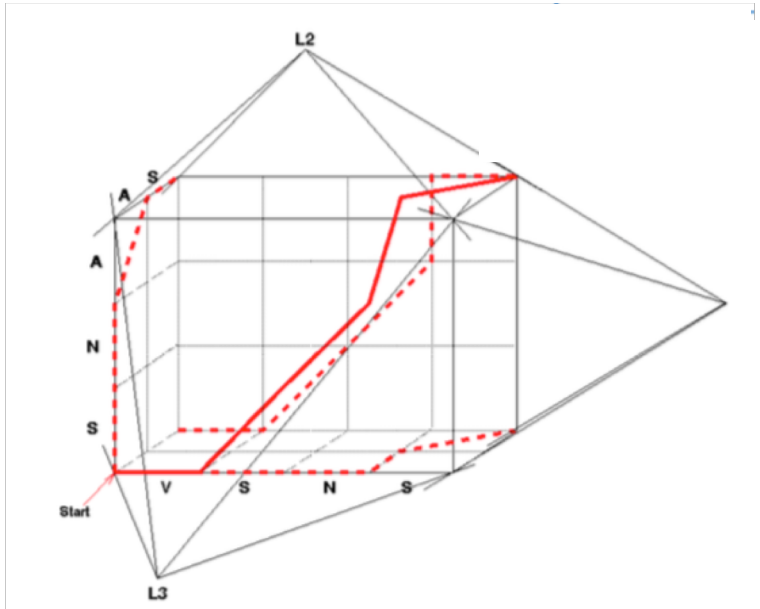


$$d_{p,q}(i,j) \geq D_{p,q}(i,j)$$

Alignment Projection and SP-score

Sequences $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ each of length n

- $D(i,j,k)$ is min SP-cost of aligning $\mathbf{v}_1[1..i], \mathbf{v}_2[1..j], \mathbf{v}_3[1..k]$
- $d_{p,q}(i,j)$ is cost of induced alignment of $\mathbf{v}_p[1..i], \mathbf{v}_q[1..j]$
- $D_{p,q}(i,j)$ is min cost of aligning $\mathbf{v}_p[1..i], \mathbf{v}_q[1..j]$

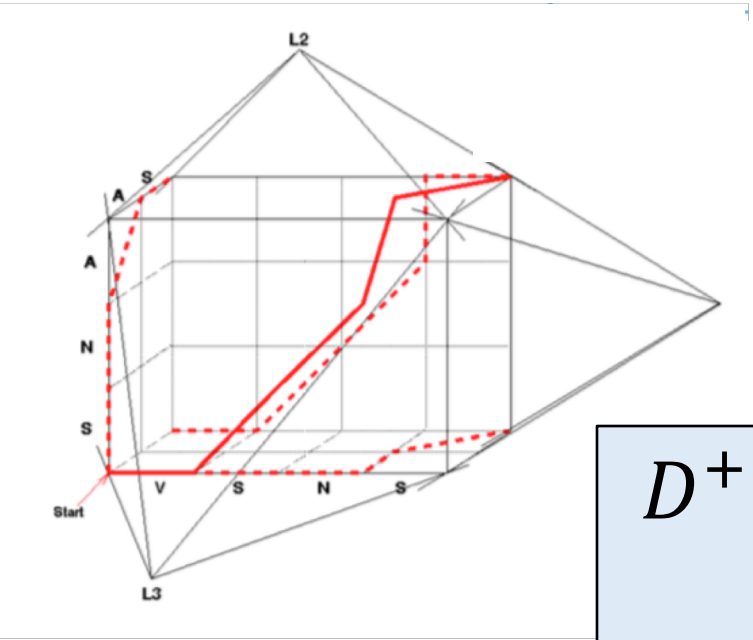


$$d_{p,q}(i,j) \geq D_{p,q}(i,j)$$

$$\begin{aligned} D(i,j,k) &= d_{1,2}(i,j) + d_{1,3}(i,k) + d_{2,3}(j,k) \\ &\geq D_{1,2}(i,j) + D_{1,3}(i,k) + D_{2,3}(j,k) \end{aligned}$$

Carillo-Lipman Method

- $D^+(i,j,k)$ is min SP-cost of alignment of **suffix** $\mathbf{v}_1[i..n], \mathbf{v}_2[j..n], \mathbf{v}_3[k..n]$
- $d^+_{p,q}(i,j)$ is cost of induced alignment of **suffix** $\mathbf{v}_p[i..n], \mathbf{v}_q[j..n]$
- $D^+_{p,q}(i,j)$ is min cost of alignment of **suffix** $\mathbf{v}_p[i..n], \mathbf{v}_q[j..n]$



$$d^+_{p,q}(i,j) \geq D^+_{p,q}(i,j)$$

$$D^+(i,j,k) = d^+_{1,2}(i,j) + d^+_{1,3}(i,k) + d^+_{2,3}(j,k) \geq D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$$

Carillo-Lipman Method

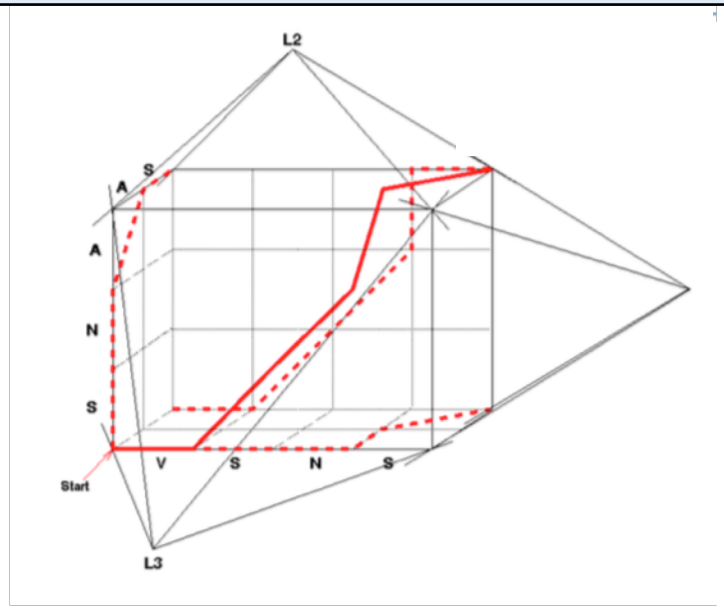
$$D^+(i,j,k) = d^+_{1,2}(i,j) + d^+_{1,3}(i,k) + d^+_{2,3}(j,k) \geq D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$$

$$D(i,j,k) + D^+(i,j,k) \geq D(i,j,k) + D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$$

Carillo-Lipman Method

$$D^+(i,j,k) = d^+_{1,2}(i,j) + d^+_{1,3}(i,k) + d^+_{2,3}(j,k) \geq D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$$

$$D(i,j,k) + D^+(i,j,k) \geq D(i,j,k) + D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$$

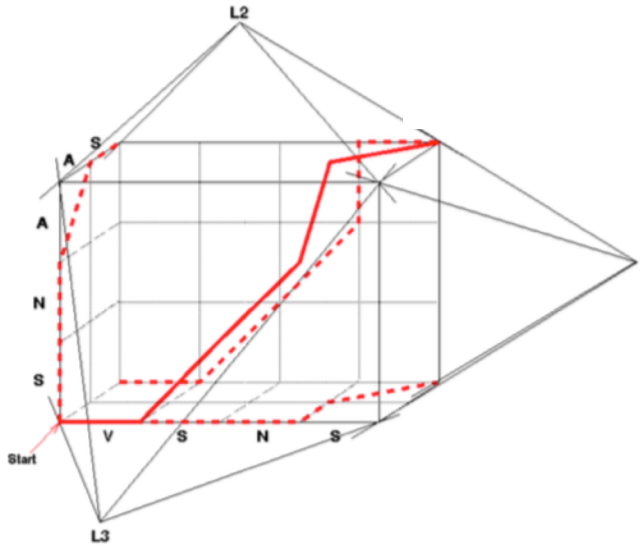


Question: What if we have an alignment with cost z?

Carillo-Lipman Method

$$D^+(i,j,k) = d^+_{1,2}(i,j) + d^+_{1,3}(i,k) + d^+_{2,3}(j,k) \geq D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$$

$$D(i,j,k) + D^+(i,j,k) \geq D(i,j,k) + D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$$



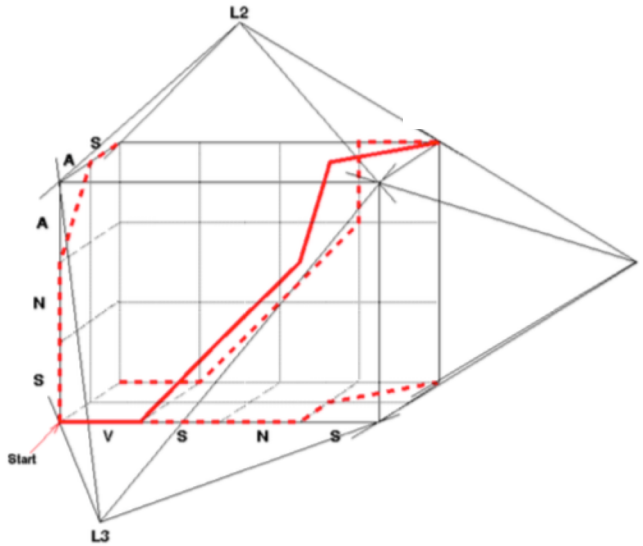
Question: What if we have an alignment with cost z?

If $z < D(i,j,k) + D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$
then (i,j,k) not on optimal path => **Prune**!

Carillo-Lipman Method

$$D^+(i,j,k) = d^+_{1,2}(i,j) + d^+_{1,3}(i,k) + d^+_{2,3}(j,k) \geq D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$$

$$D(i,j,k) + D^+(i,j,k) \geq D(i,j,k) + D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$$



Question: What if we have an alignment with cost z?

Question: How to find this alignment?

If $z < D(i,j,k) + D^+_{1,2}(i,j) + D^+_{1,3}(i,k) + D^+_{2,3}(j,k)$
then (i,j,k) not on optimal path => **Prune**!

Outline

- Multiple sequence alignment
- Exact algorithm
- Sum-of-pairs (SP) score
- Carillo-Lipman
- Heuristic approaches

Reading:

- Material based on Chapter 14.6 in book “Algorithms on Strings, Trees and Sequences” by Dan Gusfield

Inverse Problem: From Pairwise to Multiple Alignment

v_1	A	C	G	C	T	G	G	-	C
v_2	A	C	G	C	-	-	G	A	G

v_1	A	C	-	G	C	T	G	G	-	C
v_3	G	C	C	G	C	A	-	G	A	G

v_2	A	C	-	G	C	-	G	A	G
v_3	G	C	C	G	C	A	G	A	G

Question: Can we construct a multiple alignment that induces the above three pairwise alignments?

Inverse Problem: From Pairwise to Multiple Alignment

v_1	A	C	G	C	T	G	G	-	C
v_2	A	C	G	C	-	-	G	A	G

v_1	A	C	-	G	C	T	G	G	-	C
v_3	G	C	C	G	C	A	-	G	A	G

v_2	A	C	-	G	C	-	G	A	G
v_3	G	C	C	G	C	A	G	A	G

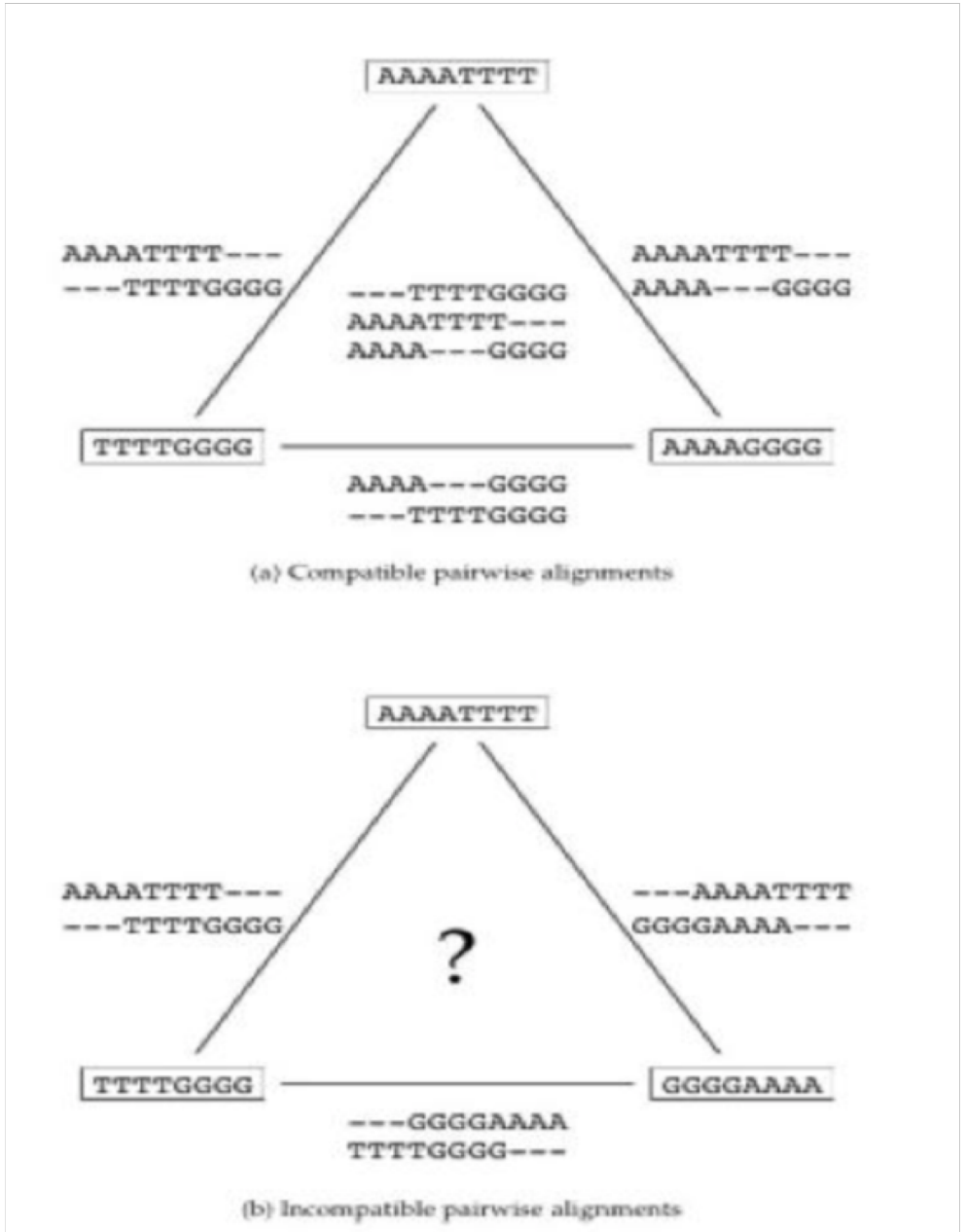
Question: Can we construct a multiple alignment that induces the above three pairwise alignments?

Not always!

Compatibility

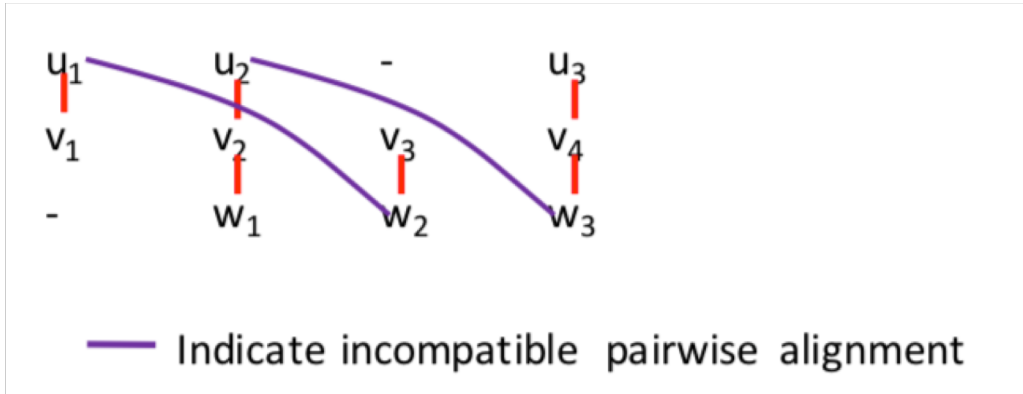
Compatible: Pairwise alignments can be combined into multiple alignment

Incompatible: Pairwise alignments *cannot* be combined into multiple alignment

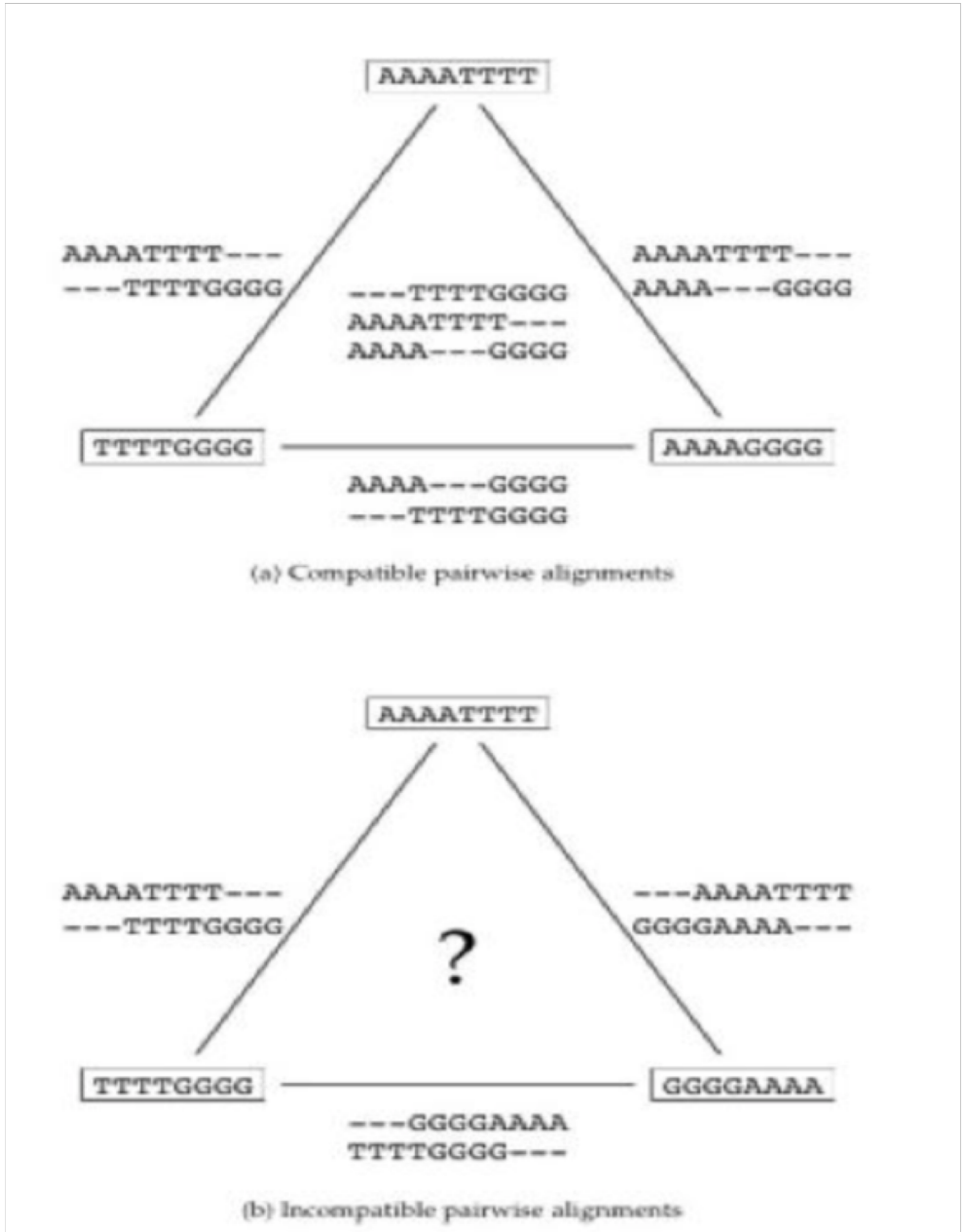


Compatibility

Compatible: Pairwise alignments can be combined into multiple alignment



Incompatible: Pairwise alignments *cannot* be combined into multiple alignment



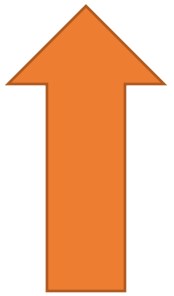
From Compatible Pairwise to Multiple Alignment

Optimal multiple alignment

Easy

Pairwise alignments between *all* pairs of sequences, but they are *not* necessarily optimal

(Sub)optimal multiple alignment



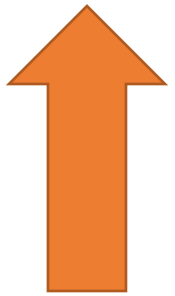
Challenging

Good (or optimal) *compatible* pairwise alignments between all sequences

From Compatible Pairwise to Multiple Alignment

(Sub)optimal multiple alignment

Iterative/progressive multiple sequence alignment: Merge pairwise alignments

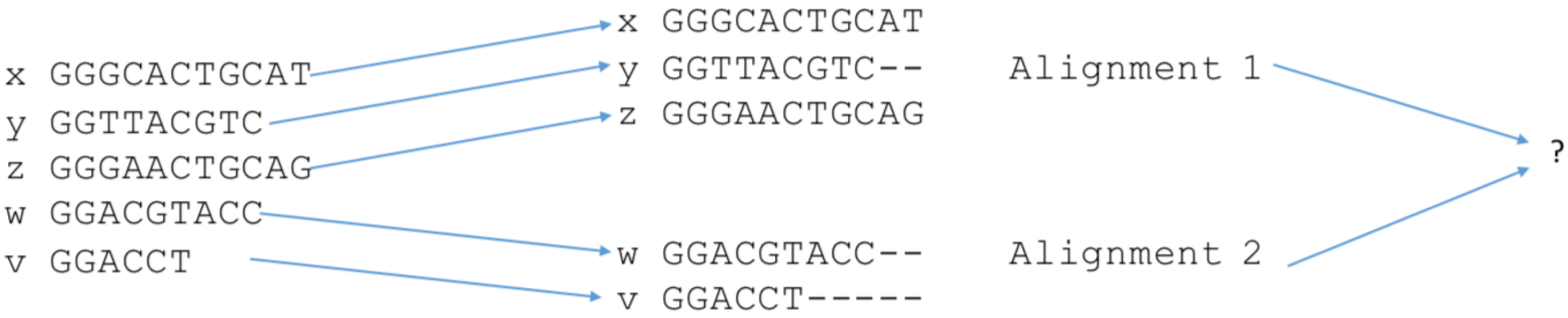


Challenging

Good (or optimal) *compatible* pairwise alignments between all sequences

Heuristic: Iterative/Progressive Alignment

Iteratively add strings (or alignments) to existing alignment(s).



Issues:

1. How to merge alignments?
2. What order to use in merging strings/alignments?

Heuristic Approach: Merge Pairwise Alignments

```
x GGGCACTGCAT
y GGTTACGTC--    Alignment 1
z GGGAACTGCAG

w GGACGTACC--    Alignment 2
v GGACCT-----
```

Question:
Can we align two alignments?

Need a way to summarize an alignment and score merged alignments

Profile Representation of Multiple Alignment

```
     -  A  G  G  C  T  A  T  C  A  C  C  T  G
     T  A  G  -  C  T  A  C  C  A  -  -  -  G
     C  A  G  -  C  T  A  C  C  A  -  -  -  G
     C  A  G  -  C  T  A  T  C  A  C  -  G  G
     C  A  G  -  C  T  A  T  C  G  C  -  G  G

A       1           1        .8
C    .6       1        .4  1     .6 .2
G          1 .2                 .2        .4  1
T    .2             1     .6                .2
-    .2        .8                      .4 .8 .4
```

A **profile** $P = [p_{i,j}]$ is a $(|\Sigma| + 1) \times l$ matrix,
where $p_{i,j}$ is the frequency of i-th letter in j-th position of alignment

Profile Representation of Multiple Alignment

We know how to align sequence against sequence

```
  -  A  G  G  C  T  A  T  C  A  C  C  T  G
  T  A  G  -  C  T  A  C  C  A  -  -  -  G
  C  A  G  -  C  T  A  C  C  A  -  -  -  G
  C  A  G  -  C  T  A  T  C  A  C  -  G  G
  C  A  G  -  C  T  A  T  C  G  C  -  G  G

A     1              1        .8
C  .6                 1     .4 1     .6 .2
G        1  .2                    .2        .4  1
T  .2                    1     .6              .2
-  .2           .8                       .4  .8 .4
```

Question: Can we align sequence against profile?

Question: Can we align profile against profile?

Aligning String to Profile

A **profile** $P = [p_{i,j}]$ is a $(|\Sigma| + 1) \times n$ matrix,
where $p_{i,j}$ is the frequency of i-th letter in j-th position of alignment

Given: Sequences $\mathbf{v} = v_1, \ldots, v_m$ and profile P with n columns

- $s[i, j]$ is optimal alignment of $v_1, \ldots, v_m$ and first j columns of P
- $\delta(x, y)$ is score for aligning characters x and y
- $\tau(x, j)$ is score for aligning character x and column j of P

Aligning String to Profile

$$\tau(x, j) = \sum_{y \in \Sigma \cup \{-\}} p_{y,j} \cdot \delta(x, y)$$

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

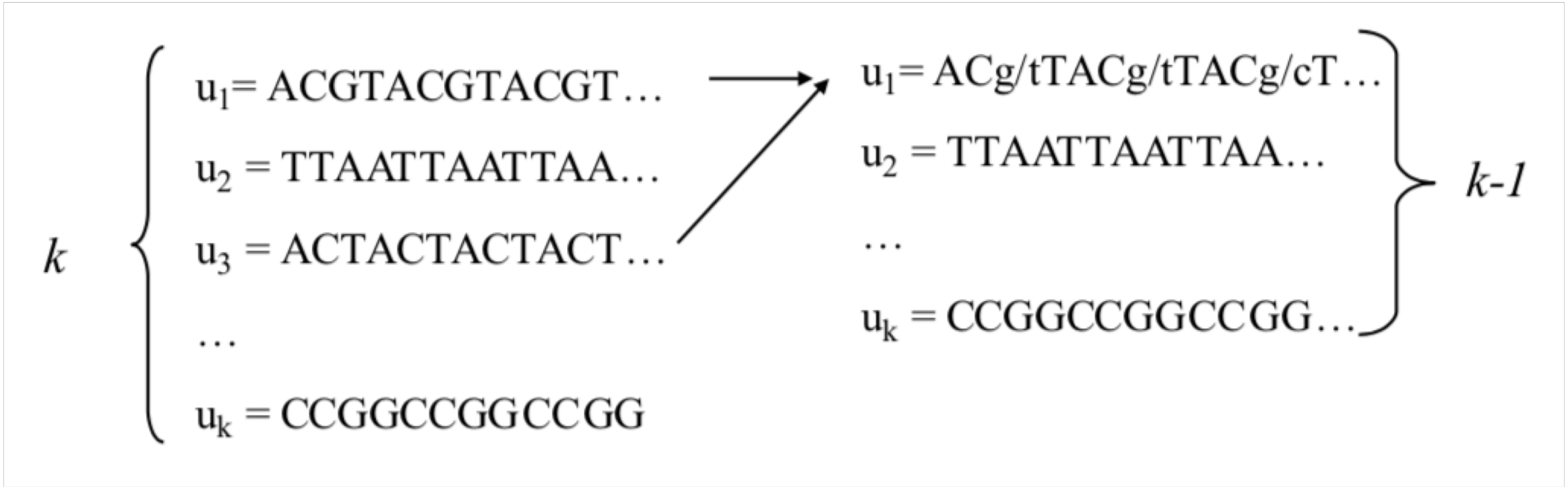
Insert space in profile

Insert space in string

- $s[i,j]$ is optimal alignment of $v_1, \ldots, v_m$ and first j columns of P
- $\delta(x, y)$ is score for aligning characters x and y
- $\tau(x, j)$ is score for aligning character x and column j of P

Progressive Multiple Alignment: Greedy Algorithm

Choose most similar pair among ***k*** input strings, combine into a profile. This reduces the original problem to alignment of ***k-1*** sequences to a profile. Repeat.



Example

Score of +1 for matches, -1 otherwise.

```
s2  GTCTGA
s4  GTCAGC (score = 2)

s1  GAT-TCA
s2  G-TCTGA (score = 1)

s1  GAT-TCA
s3  GATAT-T (score  = 1)

s1  GATTCA--
s4  G-T-CAGC(score = 0)

s2  G-TCTGA
s3  GATAT-T (score = -1)

s3  GAT-ATT
s4  G-TCAGC (score = -1)
```

Example

Score of +1 for matches, -1 otherwise.

```
s2   GTCTGA
s4   GTCAGC (score = 2)

s1   GAT-TCA
s2   G-TCTGA (score = 1)

s1   GAT-TCA
s3   GATAT-T (score  = 1)

s1   GATTCA--
s4   G-T-CAGC(score = 0)

s2   G-TCTGA
s3   GATAT-T (score = -1)

s3   GAT-ATT
s4   G-TCAGC (score = -1)
```

Question: Any theoretical guarantees on optimality?

No guarantees!

Summary

- Multiple sequence alignment
- Exact algorithm
- Sum-of-pairs (SP) score
- Carillo-Lipman
- Heuristic approaches

Homework 2 due Oct. 5 by 11:59pm

Midterm on Oct. 10, 7-9pm, 1310 DCL

Reading:

- Material based on Chapter 14.6 in book “Algorithms on Strings, Trees and Sequences” by Dan Gusfield