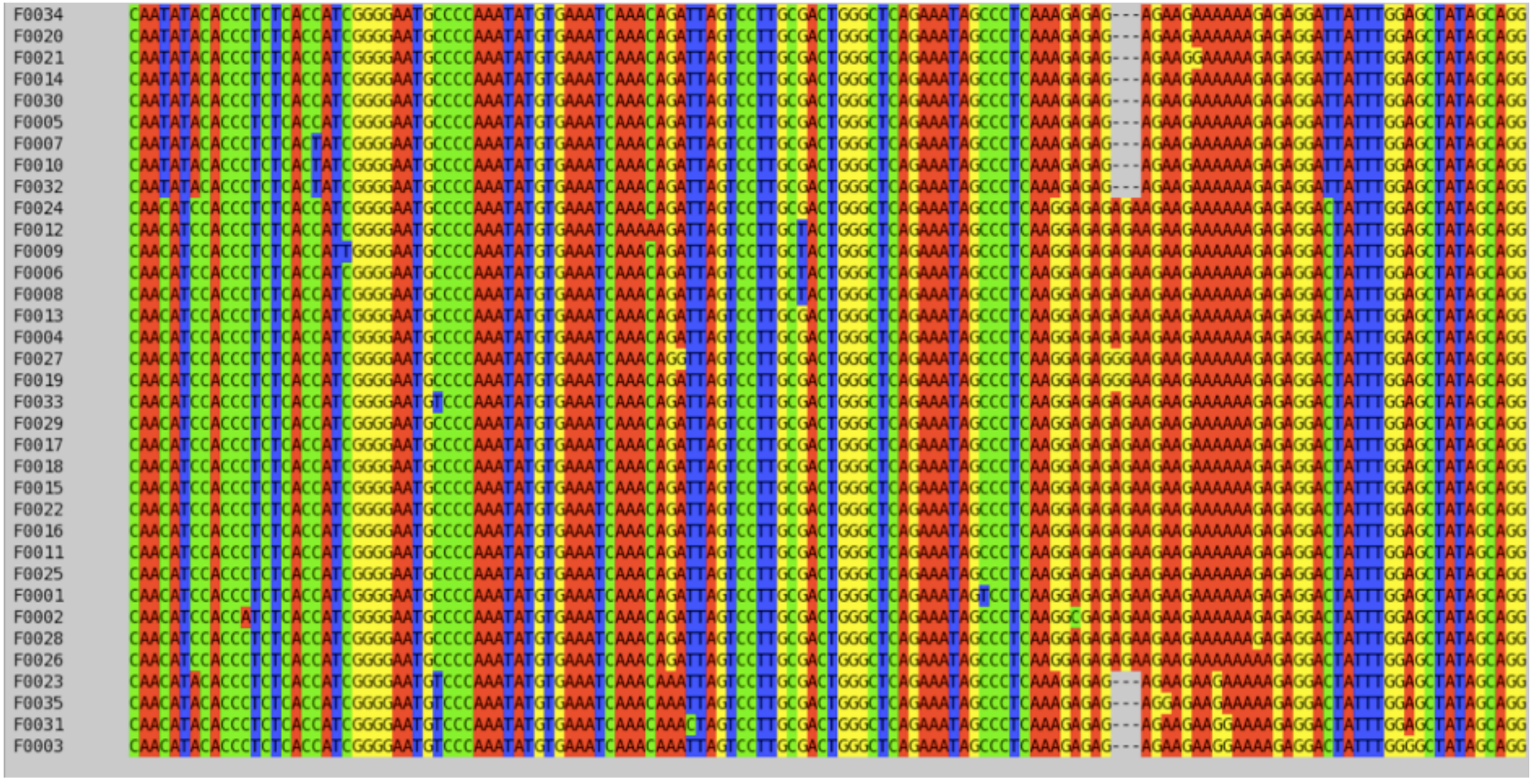
Multiple Sequence Alignment

CMSC 423

Multiple Sequence Alignment



Multiple sequence alignment: find more subtle patterns & find common patterns between all sequence.

MSA

- The multiple sequence alignment problem:

Input: Sequences: $S_1, S_2, ..., S_m$

Let M be a MSA between these sequences.

Let $d_M(S_i, S_j)$ be the score of the alignment between S_i and S_j implied by M.

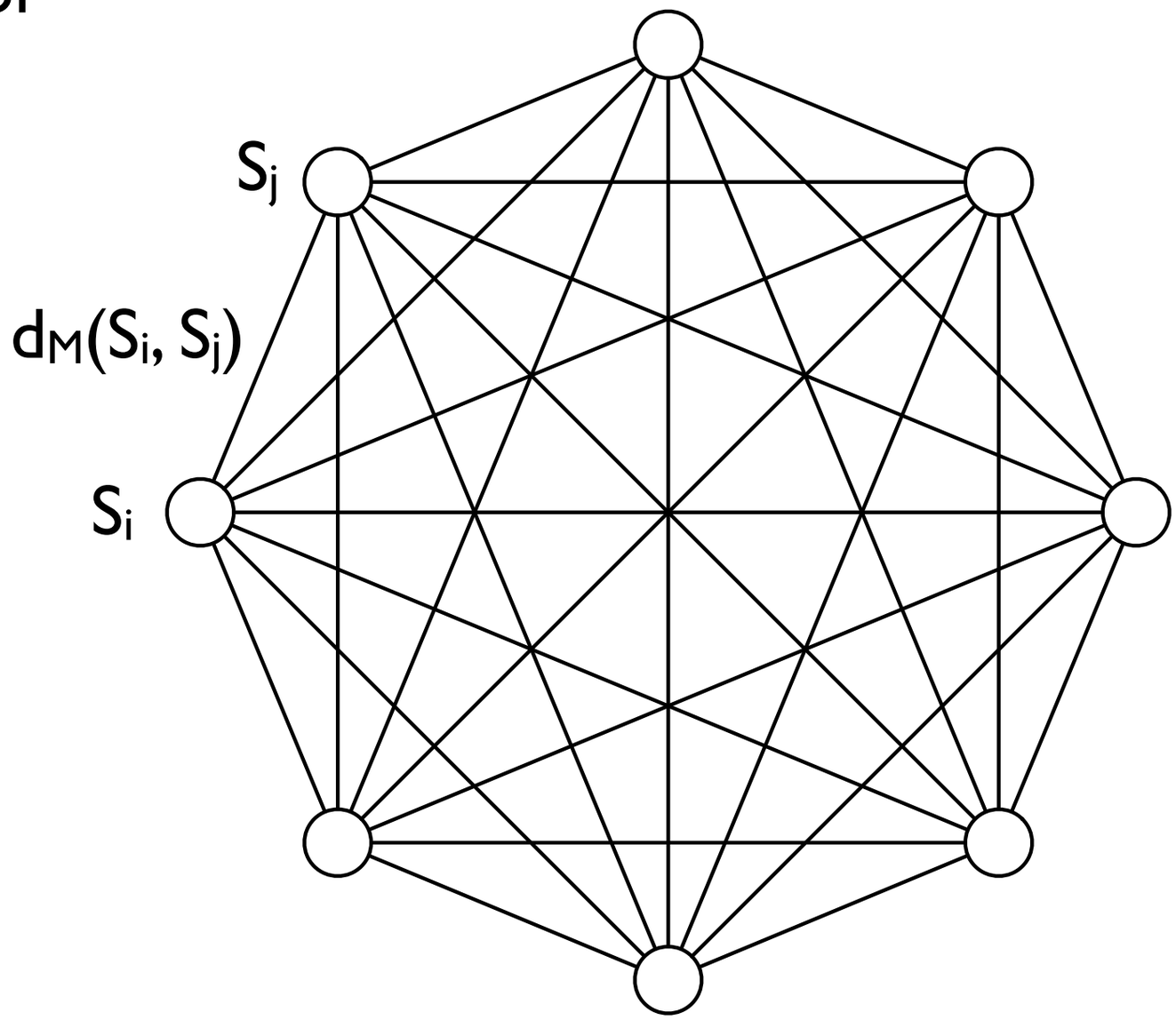
SP-Score(M) = $\sum_{i,j} d_M(S_i, S_j)$ = Sum of all pairwise alignment scores.

- **Goal**: find M to **minimize** SP-Score(M).
- But this is NP-hard.

SP-Score in a Picture

SP-Score(M) = $\sum_{i,j} d_M(S_i, S_j)$

= sum of all the scores of the pairwise alignments implied by M.



MSA

- A multiple sequence alignment (MSA) implies a pairwise alignment between every pair of sequences.
- This implied alignment need not be optimal, however:

match = -1, a mismatch = 1, gap = 2

Sequences: AT, A, T, AT, AT

Optimal MSA:

```
AT
A-
-T
AT
AT
+2 +2 = 4
```

Optimal Alignment between A and T:

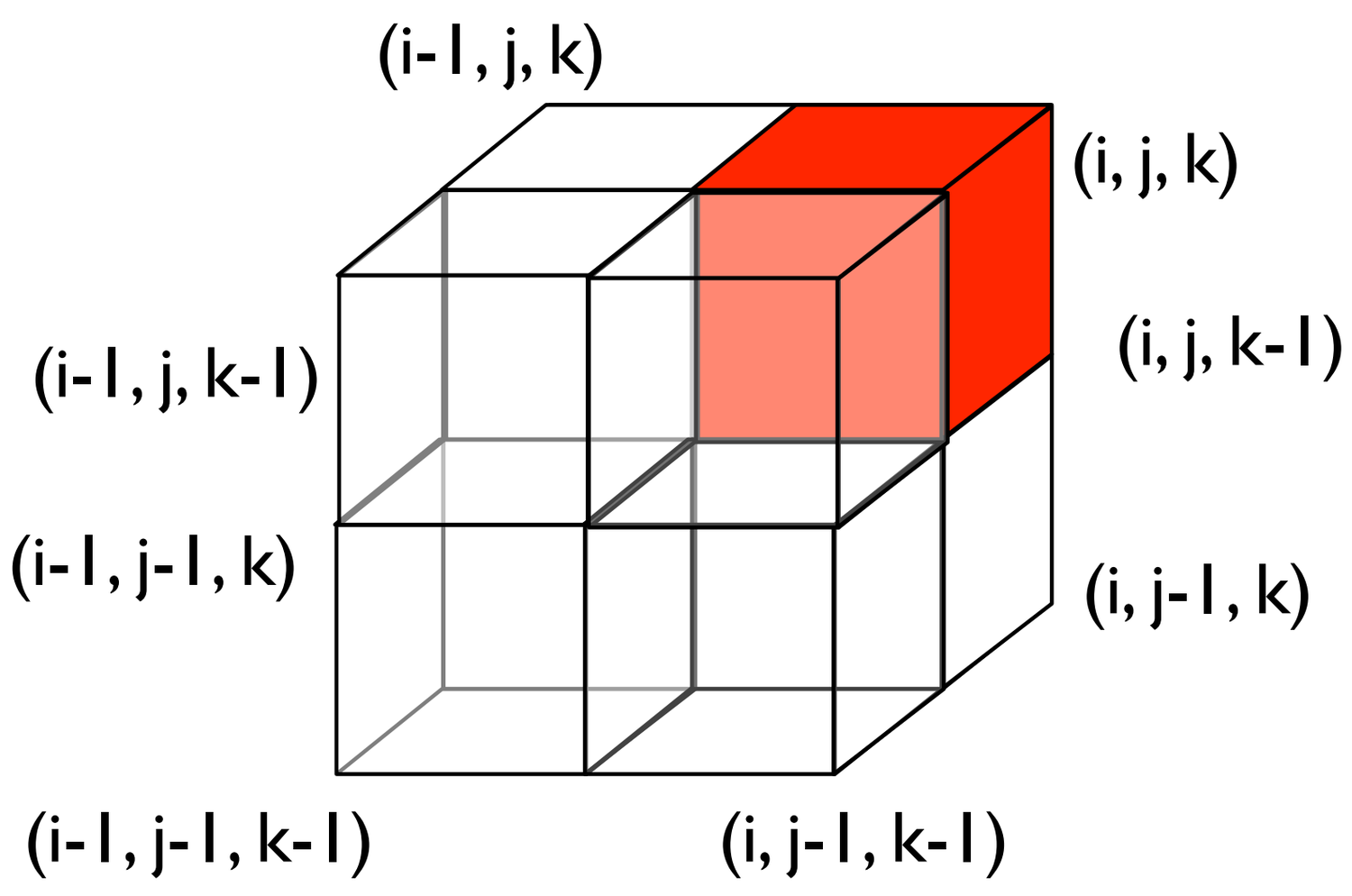
```
A
T
+1
```

(A,A), (A,-), (A,A), (A,A), (A, -), (A,A), (A,A) (-,A), (-,A), (A,A)
-1 + 2 -1 -1 +2 -1 -1 +2 +2 -1 = +2

Slow Dynamic Programming

Suppose you had just 3 sequences.

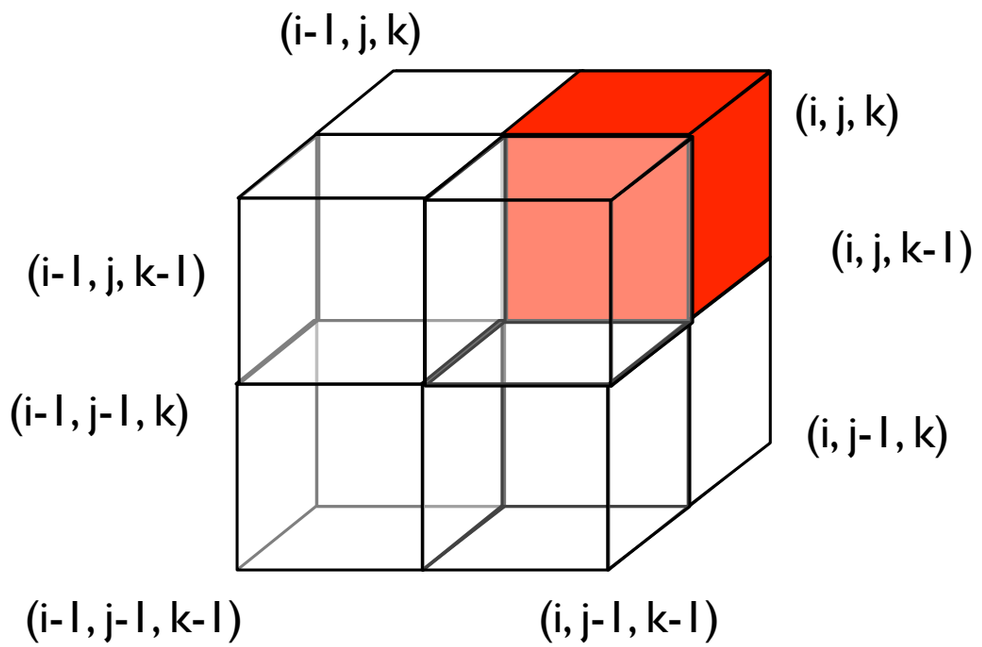
Apply the same DP idea as sequence alignment for 2 sequences, but now with a 3-dimensional matrix



DP Recurrence for 3 sequences

$$A[i,j,k] = \min \begin{cases} \text{cost}(x_i, y_j, z_k) + A[i-1, j-1, k-1] \\ \text{cost}(x_i, -, -) + A[i-1, j, k] \\ \text{cost}(x_i, y_j, -) + A[i-1, j-1, k] \\ \text{cost}(-, y_j, z_k) + A[i, j-1, k-1] \\ \text{cost}(-, y_j, -) + A[i, j-1, k] \\ \text{cost}(x_i, -, z_k) + A[i-1, j, k-1] \\ \text{cost}(-, -, z_k) + A[i, j, k-1] \end{cases}$$

Every possible 2^k pattern for the gaps.



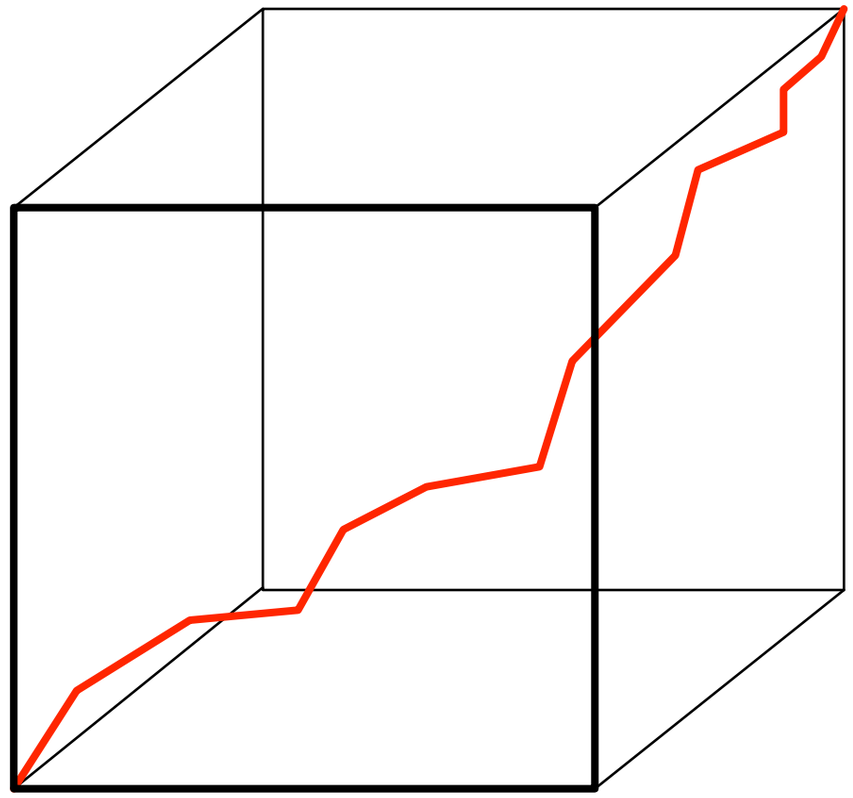
Running time

- n^3 subproblems, each takes 2^3 time
  $\Rightarrow O(n^3)$ time.

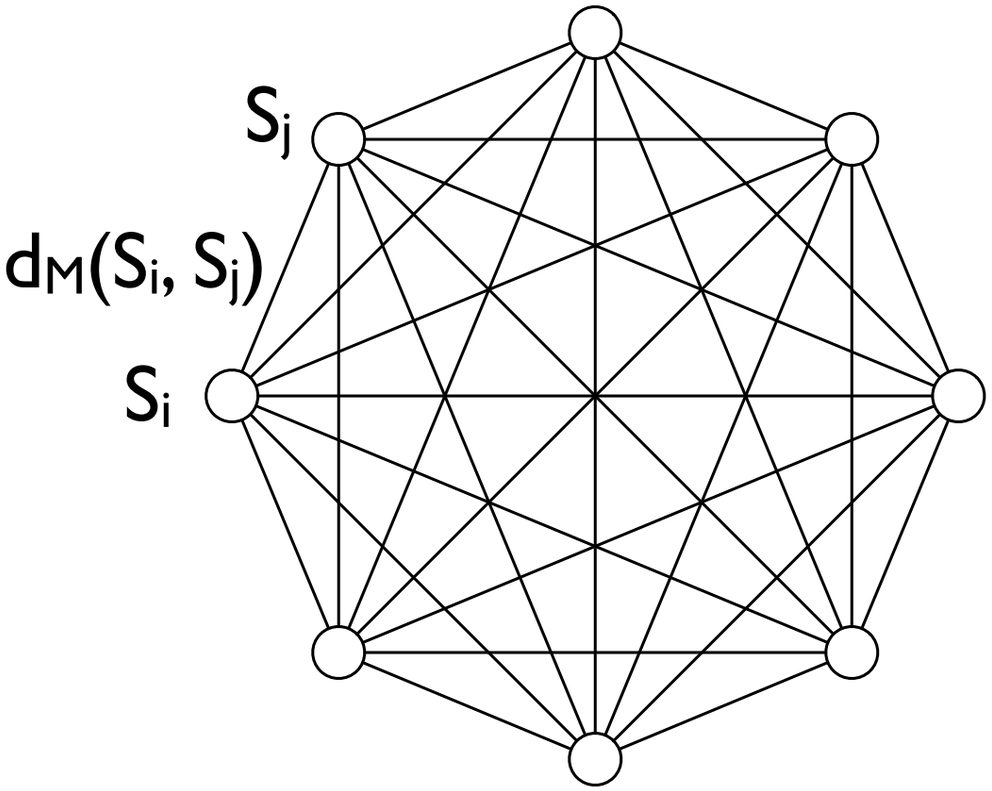
- For k sequences: n^k subproblems, each takes 2^k time for the max and k^2 to compute cost() $\Rightarrow O(k^2 n^k 2^k)$

- Even $O(n^3)$ is often too slow for the length of sequences encountered in practice.

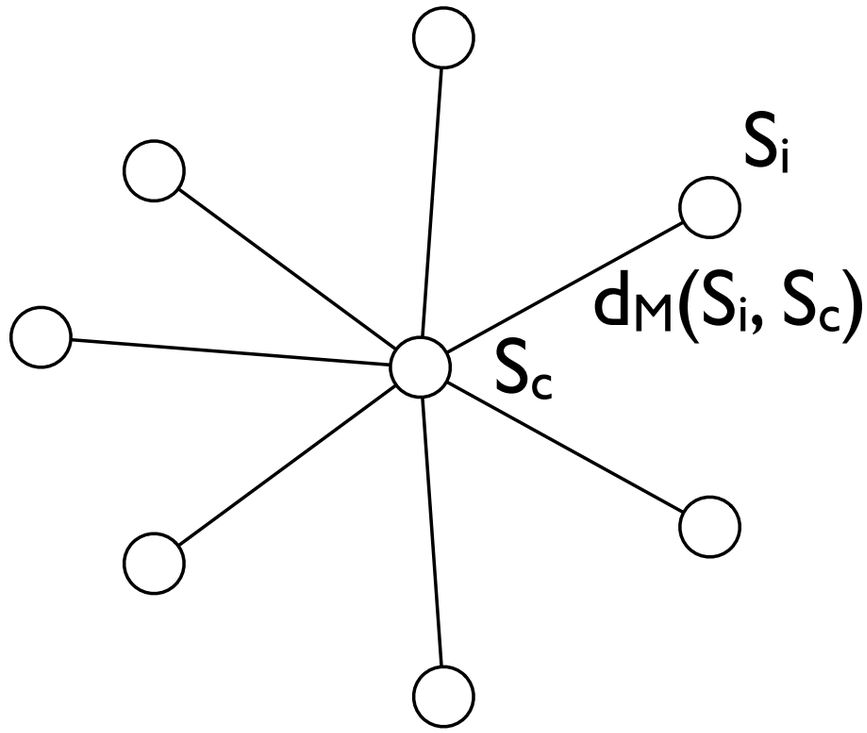
- One solution: approximation algorithm.



Star Alignment Approximation



SP-Score



Star-Score =

$\sum_i d_M(S_i, S_c)$

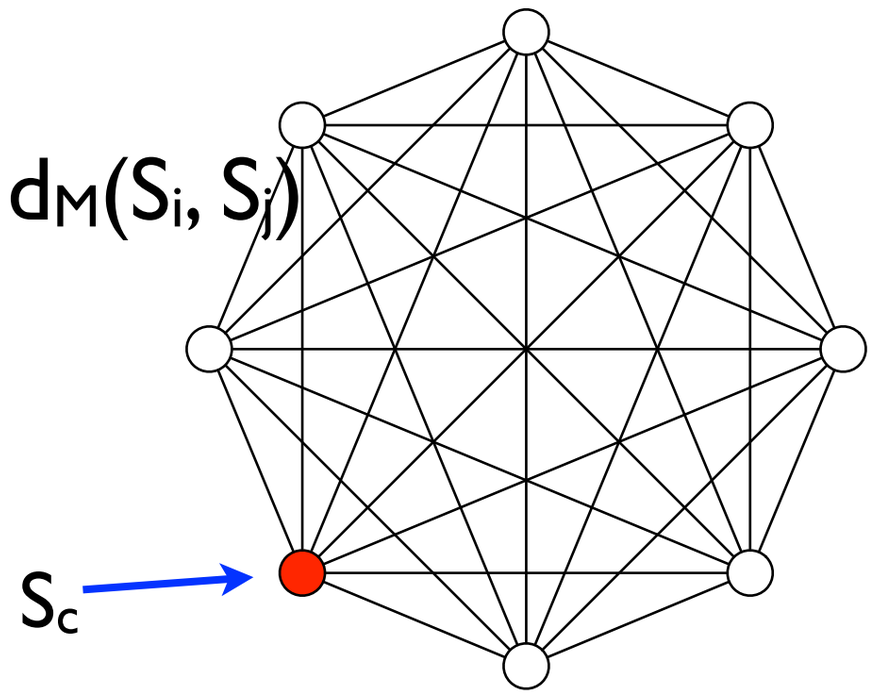
Star Alignment Algorithm

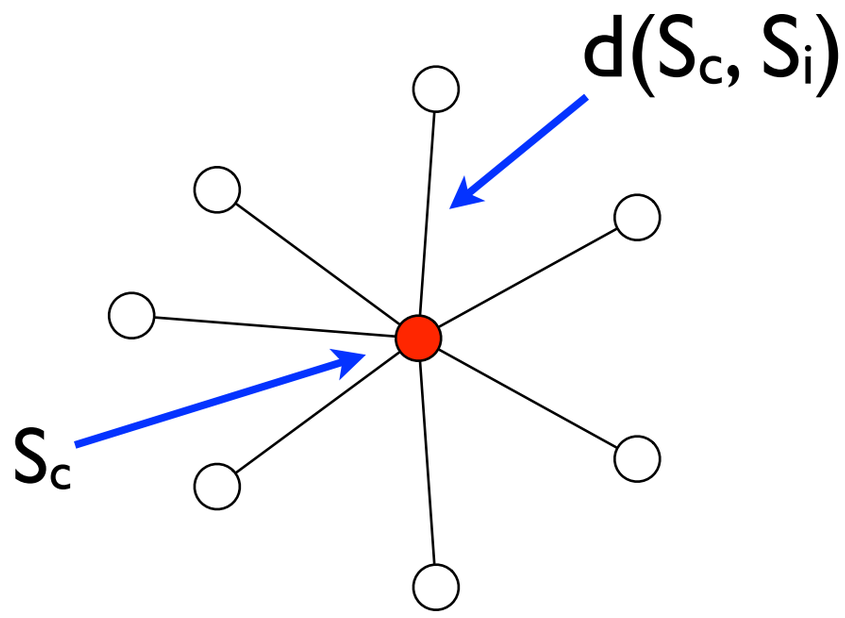
Input: sequences $S_1, S_2, ..., S_k$

- Build all $O(k^2)$ pairwise alignments.
- Let S_c = the sequence in $S_1, S_2, ..., S_k$ that is closest to the others. That is, choose S_c to minimize:

$$\sum_{i \neq c} d(S_c, S_i)$$

- *Iteratively align* all other sequences to S_c.





Iterative Alignment

- Build a multiple sequence alignment up from pairwise alignments.

Start with an alignment between S_c and some other sequence:

```
SC YFPHFDLSHGSAQVKAHGKKVGDALTLAVGHLDDLPGAL
S1 YFPHFDLSHG-AQVKG--KKVADALTNAVAHVDDMPNAL
```

Add 3rd sequence, say S2, and use the SC - S2 alignment as a guide, adding spaces into the MSA as needed.

SC - S2 alignment:

```
SC YFPHF-DLS-----HGSAQVKAHGKKVGDALTLAVGHL----DDLPGAL
S2 FFPKFKGLTTADQLKKSADVRWHAERII----NAVNDAVASMDDTEKMS
```

New {SC, S1, S2} alignment (red gaps added in S1):

```
SC YFPHF-DLS-----HGSAQVKAHGKKVGDALTLAVGHL----DDLPGAL
S1 YFPHF-DLS-----HG-AQVKG--KKVADALTNAVAHV----DDMPNAL
S2 FFPKFKGLTTADQLKKSADVRWHAERII----NAVNDAVASMDDTEKMS
```

Continue with S3, S4, ...

Performance

Assume the cost function satisfies the triangle inequality:

$$\text{cost}(x,y) \leq \text{cost}(x, z) + \text{cost}(z,y)$$

Example: $\underbrace{\text{cost}(A, C)}_{\substack{\text{cost of 1 mutation from } A \to C}} \leq \underbrace{\text{cost}(A, T) + \text{cost}(T, C)}_{\substack{\text{cost of a mutation from } A \to T \text{ and then from } T \to C}}$

STAR = cost of star alignment under SP-score

OPT = cost of optimal multiple sequence alignment (under SP-score)

Theorem. If cost satisfies the triangle inequality, then STAR $\leq$ 2×OPT.

Example: if optimal alignment has cost 10, the star alignment will have cost $\leq$ 20.

Proof (1)

Theorem. If cost satisfies the triangle inequality, then STAR $\leq$ 2OPT.

$$\frac{\mathrm{STAR}}{\mathrm{OPT}} \leq 2$$

For some *B* we will prove the 2 statements:

$$\mathrm{STAR} \leq 2B$$
$$\mathrm{OPT} \geq B$$

This will imply:

$$\Longrightarrow \frac{\mathrm{STAR}}{\mathrm{OPT}} \leq \frac{2B}{B} = 2$$

Proof (2)

Theorem. If cost satisfies the triangle inequality, then STAR ≤ 2OPT.

$$2 \cdot \mathrm{STAR} = \sum_{ij} d_{\mathrm{STAR}}(S_i, S_j)$$ defn of SP-score

by triangle inequality
$$\leq \sum_{ij} (d_{\mathrm{STAR}}(S_i, S_c) + d_{\mathrm{STAR}}(S_c, S_j))$$

because STAR alignment is optimal for pairs involving Sc
$$= \sum_{ij} (d(S_i, S_c) + d(S_c, S_j))$$

distribute ∑
$$= \sum_{ij} d(S_i, S_c) + \sum_{ij} d(S_c, S_j)$$

$$\leq 2k \sum_{i} d(S_i, S_c)$$

sums are the same and each term appears ≤ k (# of sequences) times.

Proof (3)

Theorem. If cost satisfies the triangle inequality, then STAR ≤ 2OPT.

$$2 \cdot \mathrm{OPT} = \sum_{ij} d_{\mathrm{OPT}}(S_i, S_j)$$ defn of SP-score

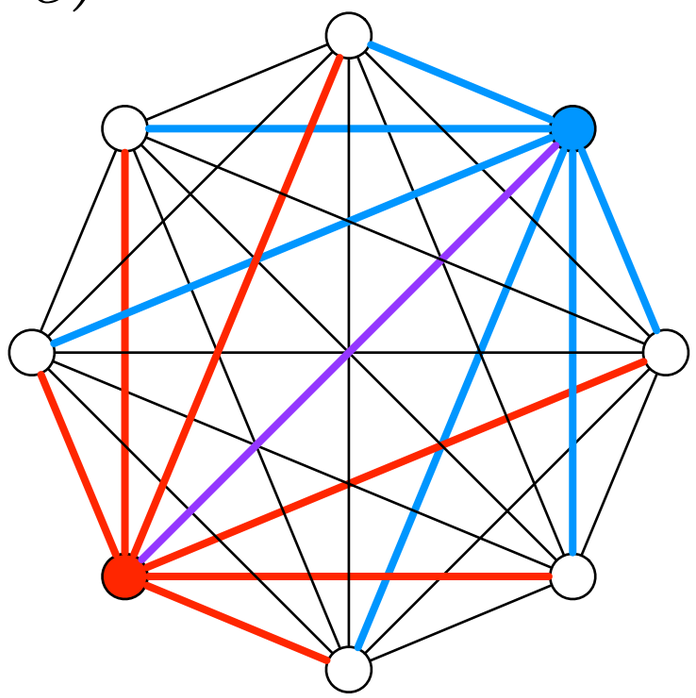
optimal pairwise alignment is ≤ pairwise alignment induced by any MSA

$$\geq \sum_{ij} d(S_i, S_j)$$

$$\geq k \sum_{i} d(S_i, S_c)$$

sum of all cost of all pairwise alignments is = the sum of k different stars.

We chose S_c because it was the lowest-cost star.



End of Proof

For some *B* we will prove the 2 statements:

$$\begin{aligned} \mathrm{STAR} &\leq 2B \\ \mathrm{OPT} &\geq B \end{aligned}$$

This will imply:

$$\Longrightarrow \frac{\mathrm{STAR}}{\mathrm{OPT}} \leq \frac{2B}{B} = 2$$

$$\begin{aligned} 2 \cdot STAR &\leq 2k \sum_i d(S_i, S_c) \\ 2 \cdot OPT &\geq k \sum_i d(S_i, S_c) \end{aligned}$$

$$\Longrightarrow \frac{\mathrm{STAR}}{\mathrm{OPT}} \leq \frac{2k \sum_i d(S_i, S_c)}{k \sum_i d(S_i, S_c)} = 2$$

Consensus Sequence

For every column j, choose c ∈ ∑ that minimizes $\sum_i$ cost(c, S_i[j])

(typically this means the most common letter)

```
S1 YFPHF-DLS-----HGSAQVKAHGKKVG-----DALTLAVAHLDDLPGAL
S2 YFPHF-DLS-----HG-AQVKG—GKKVA-----DALTNAVAHVDDMPNAL
S3 FFPKFKGLTTADQLKKSADVRWHAERII-----NAVNDAVASMDDTEKMS
S4 LFSFLKGTSEVP--QNNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATL
CO YFPHFKDLS-----HGSAQVKAHGKKVG-----DALTLAVAHVDDTPGAL
```

- Consensus is a summarization of the whole alignment.
- Consensus sequence is sometimes used as an estimate for the ancestral sequence.
- Sometimes the MSA problem is formulated as: find MSA M that minimizes:

$$\sum_i d_M(CO, S_i)$$

Profiles

- Another way to summarize an MSA:

```
S1 ACG-TT-GA
S2 ATC-GTCGA
S3 ACGCGA-CC
S4 ACGCGT-TA
```

Column in the alignment

Character	1	2	3	4	5	6	7	8	9
A	1	0	0	0	0	0.25	0	0	0.75
C	0	0.75	0.25	0.5	0	0	0.25	0.25	0.25
G	0	0	0.75	0	0.75	0	0	0.5	0
T	0	0.25	0	0	0.25	0.75	0	0.25	0
-	0	0	0	0.5	0	0	0.75	0	0

Call this profile matrix R

Fraction of time given column had the given character

Profile-based Alignment

gap in profile introduced to better fit sequence

R =

	1	2	3	4	5	6	7	8	9
A	1	0	0	0	0	0.25	0	0	0.75
C	0	0.75	0.25	0.5	0	0	0.25	0.25	0.25
G	0	0	0.75	0	0.75	0	0	0.5	0
T	0	0.25	0	0	0.25	0.75	0	0.25	0
-	0	0	0	0.5	0	0	0.75	0	0

A C C - A G A C G A

Score of matching character x with column j of the profile:

$$P(x, j) = \sum_{c \in \Sigma} \mathrm{sim}(x, c) \times R[c, j]$$

sim(x,c) = how similar character x is to character c.

$$A[i, j] = \max \begin{cases} A[i-1, j-1] + P(x_i, j) & \text{align } x_i \text{ to column } j \\ A[i-1, j] + \text{gap} & \text{introduce gap into profile} \\ A[i, j-1] + P(\text{“\_“}, j) & \text{introduce gap into } x \end{cases}$$

Recap

- Multiple sequence alignments (MSAs) are a fundamental tool. They help reveal subtle patterns, compute consistent distances between sequences, etc.

- Quality of MSAs often measured using the SP-score: sum of the scores of the pairwise alignments implied by the MSA.

- Same DP idea as pairwise alignment leads to exponentially slow algorithm for MSA.

- 2-approximation obtainable via star alignments.

- MSAs often used to create profiles summarizing a family of sequences. Profile-sequence alignments solvable via dynamic programming.