Multiple Sequence Alignment

Solon P. Pissis

CWI & VU, Amsterdam

November 12, 2024

Contents

Introduction: Why?

- PSAs for distantly related sequences are not reliable:
  - they depend on gap penalty scores, scoring matrices, etc;
  - many alignments with the same score – which one is right?
- Multiple sequence alignment (MSA) has also other applications:
  - discovering conserved motifs in a protein family;
  - phylogeny reconstruction.

	0	1	2	3	4	5	6	7	8
$x =$	G	C	G	A	C	G	T	C	C
$y =$	G	C	G	A	–	–	T	A	C
$z =$	G	T	G	A	C	G	T	–	C

Figure: An MSA of $x =$ GCGACGTCC, $y =$ GCGATAC, and $z =$ GTGACGTC.

Multiple sequence alignment (MSA): How do we score?

Definition

Given a fixed MSA M of sequences $S_1, \ldots, S_k$ and a cost function, let $d_M(S_i, S_j)$ denote the cost of the alignment between S_i and S_j **as implied by** M. Then the sum-of-pairs (SP) score is defined as:

$$\text{SP-score}(M) = \sum_{1 \le i < j \le k} d_M(S_i, S_j).$$

MSA implies a pairwise alignment between every pair of sequences.

The implied pairwise alignment may not be optimal! Aligning 5 seqs of length 2

Example

Let the cost of match be -1, of mismatch 1, and of gap 2. Let M be:

```
AT
A-
-T
AT
AT
```

Cost of column 1: $\text{cost}(\texttt{A}, \texttt{A}) + \text{cost}(\texttt{A}, -) + \cdots + \text{cost}(\texttt{A}, \texttt{A}) = 2$

Cost of column 2: $\text{cost}(\texttt{T}, -) + \text{cost}(\texttt{T}, \texttt{T}) + \cdots + \text{cost}(\texttt{T}, \texttt{T}) = 2$

SP-score$(M) = 2 + 2 = 4$

$d_M(\texttt{A}, \texttt{T}) = 4$ but the optimal alignment of A and T has cost 1!

Multiple sequence alignment (MSA): How do we score?

Definition

Given a fixed MSA M of sequences $S_1, \ldots, S_k$ and a cost function, let $d_M(S_i, S_j)$ denote the cost of the alignment between S_i and S_j **as implied by** M. Then the sum-of-pairs (SP) score is defined as:

$$\text{SP-score}(M) = \sum_{1 \le i < j \le k} d_M(S_i, S_j).$$

MSA implies a pairwise alignment between every pair of sequences. The implied pairwise alignment may not be optimal!

Example

Let the cost of match be -1, of mismatch 1, and of gap 2. Let M be:

AT
A-
-T
AT
AT

Cost of column 1: $\text{cost}(\texttt{A}, \texttt{A}) + \text{cost}(\texttt{A}, \texttt{-}) + \cdots + \text{cost}(\texttt{A}, \texttt{A}) = 2$

Cost of column 2: $\text{cost}(\texttt{T}, \texttt{-}) + \text{cost}(\texttt{T}, \texttt{T}) + \cdots + \text{cost}(\texttt{T}, \texttt{A}) = 2$

$\text{SP-score}(M) = 2 + 2 = 4$

$d_M(\texttt{A}, \texttt{T}) = 4$ but the optimal alignment of A and T has cost 1!

Multiple sequence alignment (MSA): How do we score?

Given an alignment, how do we score?

Definition

Given a fixed MSA M of sequences $S_1, \ldots, S_k$ and a cost function, let $d_M(S_i, S_j)$ denote the cost of the alignment between S_i and S_j **as implied by** M. Then the sum-of-pairs (SP) score is defined as:

$$\text{SP-score}(M) = \sum_{1 \le i < j \le k} d_M(S_i, S_j).$$

MSA implies a pairwise alignment between every pair of sequences. The implied pairwise alignment may not be optimal!

Example

Let the cost of match be -1, of mismatch 1, and of gap 2. Let M be:

AT
A-
-T
AT
AT

Cost of column 1: $\text{cost}(\texttt{A}, \texttt{A}) + \text{cost}(\texttt{A}, \texttt{-}) + \cdots + \text{cost}(\texttt{A}, \texttt{A}) = 2$

Cost of column 2: $\text{cost}(\texttt{T}, \texttt{-}) + \text{cost}(\texttt{T}, \texttt{T}) + \cdots + \text{cost}(\texttt{T}, \texttt{A}) = 2$

$\text{SP-score}(M) = 2 + 2 = 4$

$d_M(\texttt{A}, \texttt{T}) = 4$ but the optimal alignment of A and T has cost 1!

Multiple sequence alignment (MSA): How do we score?

Definition

Given a fixed MSA M of sequences $S_1, \ldots, S_k$ and a cost function, let $d_M(S_i, S_j)$ denote the cost of the alignment between S_i and S_j **as implied by** M. Then the sum-of-pairs (SP) score is defined as:

$$\text{SP-score}(M) = \sum_{1 \le i < j \le k} d_M(S_i, S_j).$$

MSA implies a pairwise alignment between every pair of sequences. The implied pairwise alignment may not be optimal!

Example

Let the cost of match be -1, of mismatch 1, and of gap 2. Let M be:

AT
A-
-T
AT
AT

Cost of column 1: $\text{cost}(\texttt{A}, \texttt{A}) + \text{cost}(\texttt{A}, \texttt{-}) + \cdots + \text{cost}(\texttt{A}, \texttt{A}) = 2$

Cost of column 2: $\text{cost}(\texttt{T}, \texttt{-}) + \text{cost}(\texttt{T}, \texttt{T}) + \cdots + \text{cost}(\texttt{T}, \texttt{A}) = 2$

$\text{SP-score}(M) = 2 + 2 = 4$

$d_M(\texttt{A}, \texttt{T}) = 4$ but the optimal alignment of A and T has cost 1!

Multiple sequence alignment (MSA): How do we score?

Definition

Given a fixed MSA M of sequences $S_1, \ldots, S_k$ the entropy-based (EB) score of M is defined as:

$$\text{EB-score}(M) = -\sum_j c_j/C \ln(c_j/C),$$

where c_j is the number of occurrences of letter j in the column, and C is the total number of letters in the column.

Example

```
AAAAA
AAAAI  Cost of column 1: 0
AAAAK  Cost of column 2: 0.44
AAAIL  Cost of column 3: 0.65
AAIIS  Cost of column 4: 0.69
AIIIW  Cost of column 5: 1.79
```

J: the number of characters in each column
Score calculated for each column

Contents

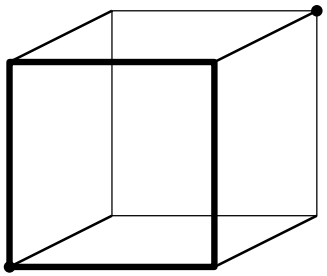
Dynamic-programming algorithm

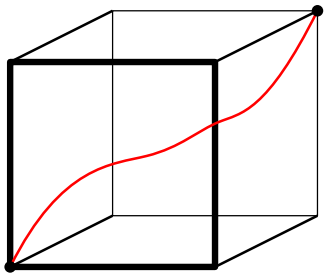
Given k input sequences $S_1, \ldots, S_k$:

Pair-wise sequence alignment

- How could we generalize the classic PSA DP algorithm?
- Therein we use a 2d matrix for $k = 2$
- We can generalize this solution if we use a kd matrix
- Let us see this for $k = 3$ (for sanity...)

Use matrix for 2 sequences, so use a 3-tensor for 3 sequences and so on...





Dynamic-programming algorithm

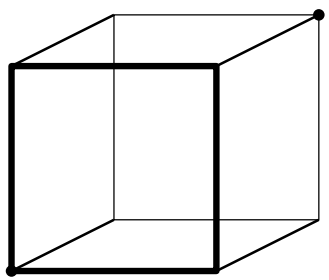
- Given 3 input sequences x, y, z the recurrence is:

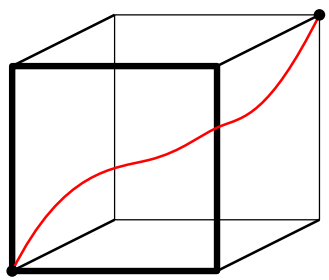
$$T[i,j,k] = \max \begin{cases} \text{cost}(x[i], y[j], z[k]) + T[i-1, j-1, k-1] \\ \text{cost}(x[i], -, -) + T[i-1, j, k] \\ \text{cost}(x[i], y[j], -) + T[i-1, j-1, k] \\ \text{cost}(-, y[j], z[k]) + T[i, j-1, k-1] \\ \text{cost}(-, y[j], -) + T[i, j-1, k] \\ \text{cost}(x[i], -, z[k]) + T[i-1, j, k-1] \\ \text{cost}(-, -, z[k]) + T[i, j, k-1] \end{cases}$$

The score matrix update rule generalized for 3-tensor

for all i, j, and k.

- Note that it is every possible pattern for the gaps

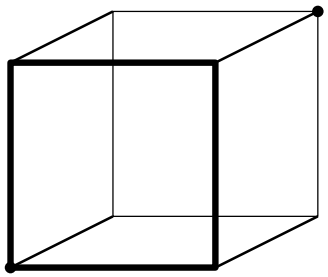


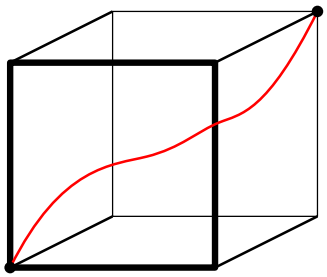


Dynamic-programming algorithm

- Assume, for simplicity, that x, y, z are all of length n
- "for all i, j, and k": we have $O(n^3)$ subproblems
- "every possible pattern for the gaps": we have $O(2^3)$ additions
- Every cost function takes $O(3^2)$ time to evaluate
- Time is $O(3^2 2^3 n^3)$ for $k = 3$
- Time is $O(k^2 2^k n^k)$ for $k > 3$
- Exponential in k is not practical!

Too many computations, does not work





Contents

The Star Alignment (SA) algorithm

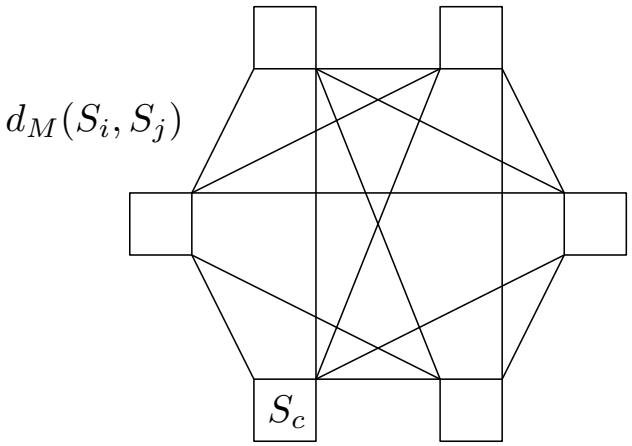
Given k input sequences $S_1, \ldots, S_k$, SA works as follows:

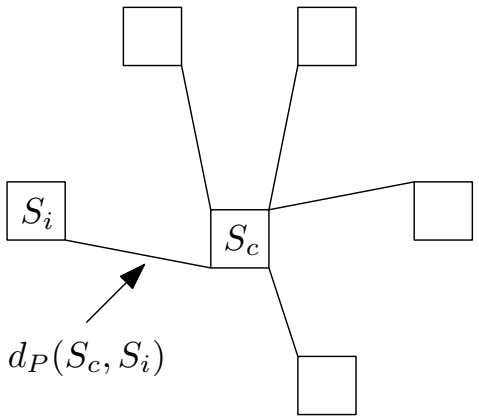
1. Construct all $O(k^2)$ pairwise alignments
2. Let S_c be the sequence in $\{S_1, \ldots, S_k\}$ closest to the others; namely, choose S_c that minimizes:

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

   Sc is one of the original sequences, Sc is the one that is closest to all the other sequences

   where d_P is the optimal cost of pairwise alignment.
3. Progressively align all other $k-1$ sequences onto S_c.





The Star Alignment (SA) algorithm

Progressive alignment means:

1. Construct an MSA made up from PSAs
2. Start with a PSA between S_c and some other sequence

$$\begin{aligned} S_c &= \texttt{YFPHFDLSHGSAQVKAHGKKVGDALTLAVGHLDDLPGAL} \\ S_1 &= \texttt{YFPHFDLSHG-AQVKG--KKVADALTNAVAHVDDMPNAL} \end{aligned}$$

3. Add S_2 using the (S_c, S_2) PSA as a guide:

$$\begin{aligned} S_c &= \texttt{YFPHF-DLS-----HGSAQVKAHGKKVGDALTLAVGHL----DDLPGAL} \\ S_2 &= \texttt{FFPKFKGLTTADQLKKSADVRWHAERII----NAVNDAVASMDDTEKMS} \end{aligned}$$

4. $\{S_c, S_1, S_2\}$ alignment (red gaps added in S_1):

$$\begin{aligned} S_c &= \texttt{YFPHF-DLS-----HGSAQVKAHGKKVGDALTLAVGHL----DDLPGAL} \\ S_1 &= \texttt{YFPHF-DLS-----HG-AQVKG--KKVADALTNAVAHV----DDMPNAL} \\ S_2 &= \texttt{FFPKFKGLTTADQLKKSADVRWHAERII----NAVNDAVASMDDTEKMS} \end{aligned}$$

5. Continue with the rest of the sequences.

The Star Alignment (SA) algorithm

We make the following standard assumptions:

- The cost function satisfies the triangle inequality:
  $\text{cost}(a, b) \leq \text{cost}(a, c) + \text{cost}(c, b)$,
  a, b, c are column elements.

Cost of AC is less than cost of aligned AG and GC

Example

$\text{cost}(\mathtt{A}, \mathtt{C}) \leq \text{cost}(\mathtt{A}, \mathtt{G}) + \text{cost}(\mathtt{G}, \mathtt{C})$.

- We use the SP-score.

Sum of pairs score

We will later prove the following theorem.

The score obtained using SA is at most 2 times worse than the optimal alignment. SA finds an approximate solution to MSA. SA is a greedy method, maximizing local score at each step. It may not lead to the optimal solution.

Theorem ([1])

For any instance $\mathcal{I}$ *of MSA,* $\frac{SA(\mathcal{I})}{\mathcal{OPT}(\mathcal{I})} \leq 2$.

Example

If for some instance $\mathcal{I}$ of MSA, $\mathcal{OPT}(\mathcal{I}) = 25$, then $SA(\mathcal{I}) \leq 50$.

Contents

Consensus Sequence: Summarization

This is the alignment criteria: the sequences, once aligned, should minimize the total alignment cost with the CO

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

In column j of the CO sequence, choose …

- In column j choose $c \in \Sigma$ that minimizes: $\sum_i \text{cost}(c, S_i[j])$.
- CO is used as an estimate for the ancestral sequence.
- Find MSA M that minimizes: $\sum_i d_M(\text{CO}, S_i)$.

Assuming the sequences being aligned are from a common ancestor

dM(CO,Si): Pairwise alignment score of the ancestral sequence and the other sequences

Profile: Summarization

$$
\begin{aligned}
S_1 &= \texttt{ACG-TT-GA} \\
S_2 &= \texttt{ATC-GTCGA} \\
S_3 &= \texttt{ACGCGA-CC} \\
S_4 &= \texttt{ACGCGT-TA}
\end{aligned}
$$

Profile R	0	1	2	3	4	5	6	7	8
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

- Fraction of time a given column has a certain letter.

CLUSTAL W

Given k input sequences $S_1, \ldots, S_k$, CLUSTAL W [3]:

1. Constructs all $O(k^2)$ PSAs.
2. Constructs a guide tree (we will see this later).
3. Aligns (summarized) sequences from the most similar ones to the least similar ones according to the tree in Step 2.

How can we do this alignment: profile vs. sequence?

Profile R	0	1	2	3	4	5	6	7	8
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

ACCAGGACA

CLUSTAL W

Profile R	0	1	2	3		4	5	6	7	8
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

cost(x,c): cost of aligning char x to char c

- Score of matching letter x with column j of the profile: $P(x,j) = \sum_{c\in\Sigma} \text{cost}(x,c)R[c,j]$.

R[c,j]: score in the profile R column j character c

- The recurrence formula is similar now to PSA:

$$T[i,j] = \max \begin{cases} T[i-1,j-1] + P(x_i,j) & \text{align letter to column} \\ T[i-1,j] + \text{gap-cost} & \text{gap into profile} \\ T[i,j-1] + P(-,j) & \text{gap into sequence.} \end{cases}$$

for all i and j.

CLUSTAL W

Profile R	0	1	2	3		4	5	6	7	8
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 letter x with column j of the profile: $P(x, j) = \sum_{c \in \Sigma} \text{cost}(x, c) R[c, j]$.
- The recurrence formula is similar now to PSA:

$$T[i,j] = \max \begin{cases} T[i-1, j-1] + P(x_i, j) & \text{align letter to column} \\ T[i-1, j] + \text{gap-cost} & \text{gap into profile} \\ T[i, j-1] + P(-, j) & \text{gap into sequence.} \end{cases}$$

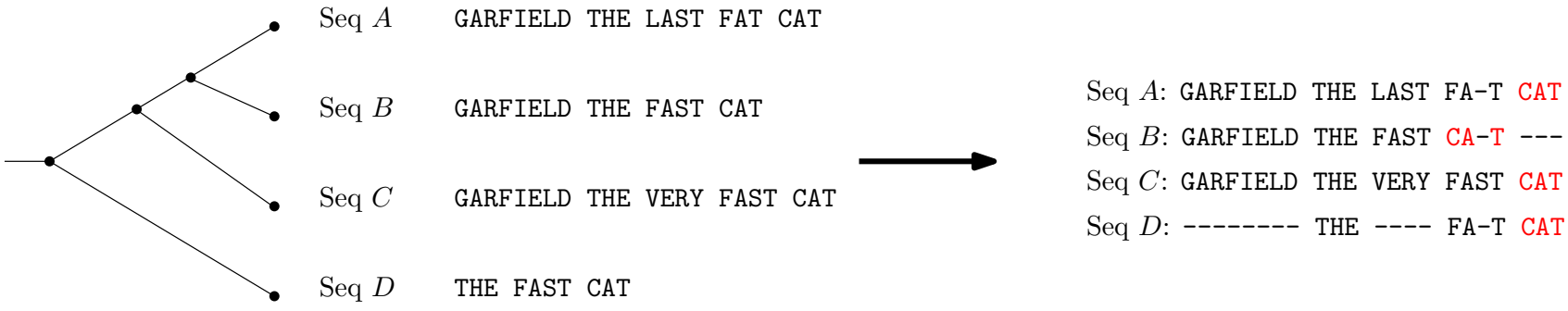
for all i and j.

Contents

T-coffee

CLUSTAL W is fast but it can be problematic:

- Strong dependence on the initial PSAs.
- Propagating errors from the initial PSAs.



Unfortunately `CAT` is not consistently aligned.

T-coffee

T-coffee [2] starts by constructing a library of PSAs:

- Each PSA is represented as a list of pairwise (mis)matches.
- The initial PSA weight = the percentage of identical matches.

Seq A: GARFIELD THE LAST FAT CAT
Seq B: GARFIELD THE FAST CAT ---
Prim. Weight = 88

Seq B: GARFIELD THE ---- FAST CAT
Seq C: GARFIELD THE VERY FAST CAT
Prim. Weight = 100

Seq A: GARFIELD THE LAST FA-T CAT
Seq C: GARFIELD THE VERY FAST CAT
Prim. Weight = 77

Seq B: GARFIELD THE FAST CAT
Seq D: -------- THE FA-T CAT
Prim. Weight = 100

Seq A: GARFIELD THE LAST FAT CAT
Seq D: -------- THE ---- FAT CAT
Prim. Weight = 100

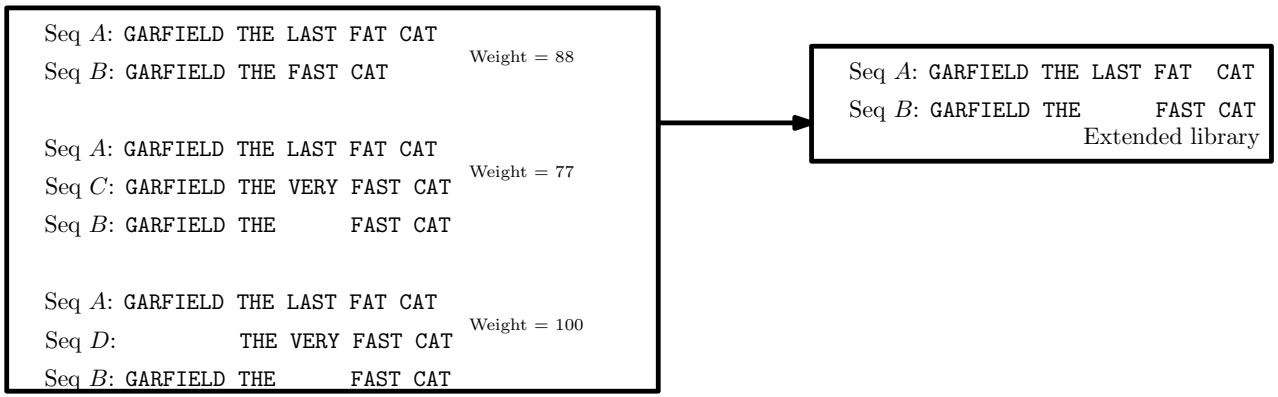
Seq C: GARFIELD THE VERY FAST CAT
Seq D: -------- THE ---- FA-T CAT
Prim. Weight = 100

We have colored red the non-identical matches.

T-coffee: Triplet strategy

T-coffee then constructs an extended library of PSAs:

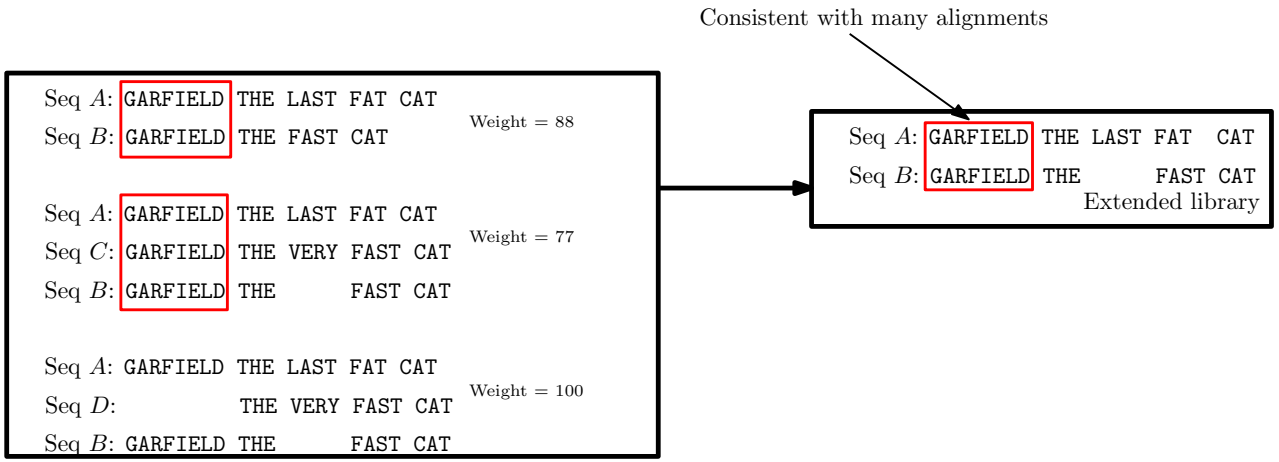
- For each PSA consider the alignment with a third sequence.
- Sum up the weights to get the extended library.



T-coffee: Triplet strategy

T-coffee then constructs an extended library of PSAs:

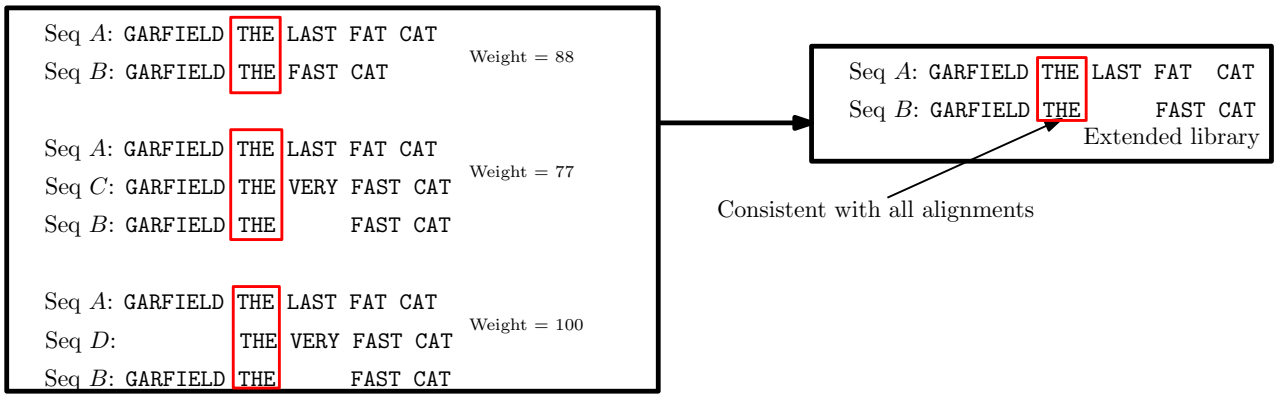
- For each PSA consider the alignment with a third sequence.
- Sum up the weights to get the extended library.



T-coffee: Triplet strategy

T-coffee then constructs an extended library of PSAs:

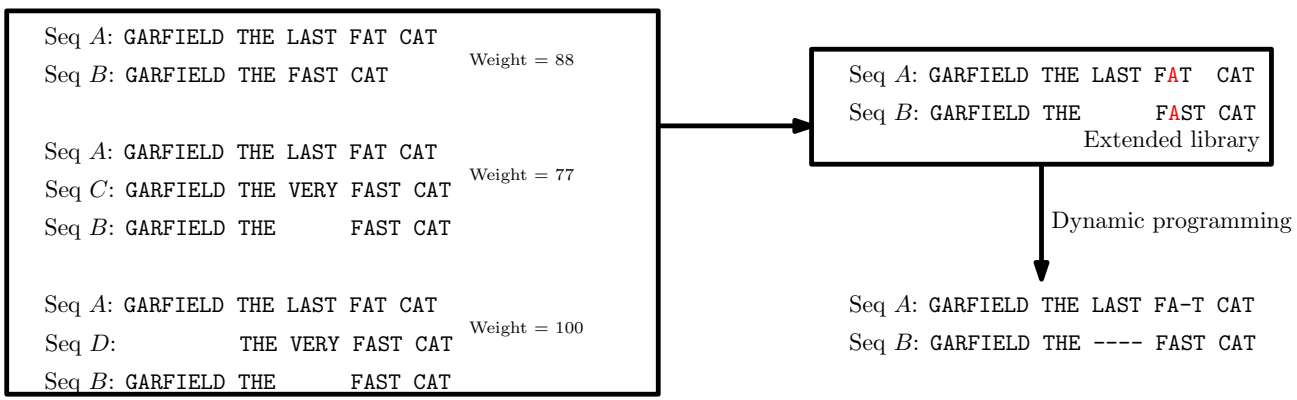
- For each PSA consider the alignment with a third sequence.
- Sum up the weights to get the extended library.



T-coffee

Finally, T-coffee performs the progressive alignment:

- It uses the tree but also the weights of the extended library!
- `A` in `FAST` has a high score with `A` in `FAT`.
- `A` in `FAST` has a low score with `A` in `LAST`.



Contents

Conclusion

- PSA cannot replace MSA
- MSA is the key computation in many biological applications
- Like in PSA, we must first define a scoring scheme!
- Unlike PSA, DP-based MSA is computationally expensive
- We use approximation algorithms or heuristics for MSA
  - Star-alignment (we will analyze this more later!)
    - Main idea: progressive alignment via choosing a center
  - CLUSTAL W
    - Main idea: profile alignment using a guide tree
  - T-coffee
    - Main idea: consistency via a triplet strategy

Bibliography I


T. Jiang, E. L. Lawler, and L. Wang.
Aligning sequences via an evolutionary tree: complexity and approximation.
In *STOC*, pages 760–769. ACM, 1994.


C. Notredame, D. G. Higgins, and J. Heringa.
T-coffee: a novel method for fast and accurate multiple sequence alignment.
Journal of Molecular Biology, 302(1):205–217, 2000.


J. D. Thompson, D. G. Higgins, and T. J. Gibson.
CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice.
Nucleic Acids Research, 22(22):4673–4680, 11 1994.