UNIVERSITÄT DES SAARLANDES

ZBI ZENTRUM FÜR BIOINFORMATIK

Multiple Sequence Alignment I

Algorithms for Sequence Analysis

Sven Rahmann

Summer 2021

Overview

Multiple sequence Alignment

- Basic Definitions
- Why Multiple sequence comparison?
- Multiple Alignment Problem
- Sum-of-Pairs Scores

Algorithms:

- Generalization of the Universal Alignment Algorithm
- The Center Star approximation

Basics

Definition

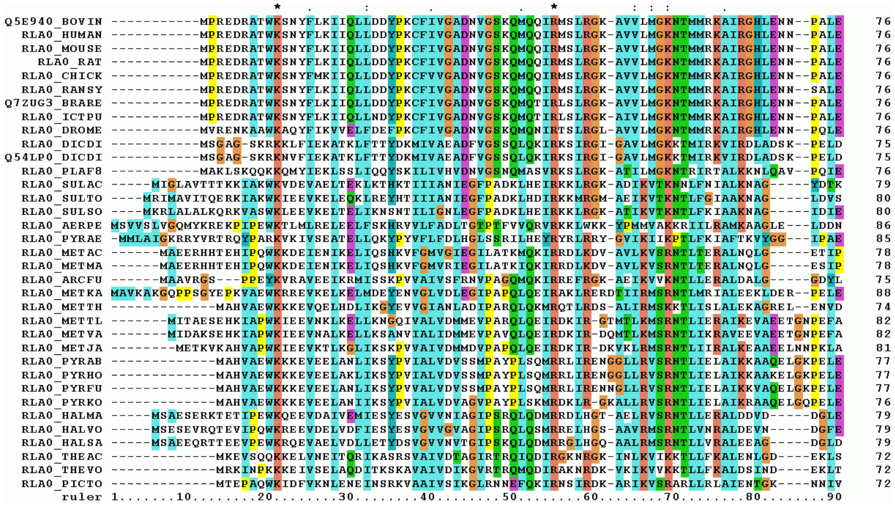
Multiple Sequence Alignment (MSA) is generalization of pairwise alignment with more than two sequences.

Applications

- Compute motifs (eg. transcription factor binding sites)
- Detect homologous residues and estimate their conservation
- Predict the secondary structures of proteins
- Infer the evolutionary history of the sequences

One or two homologous sequences whisper...
a full multiple alignment shouts out loud. (Hubbard et al., 1996)

MSA example (60S acidic ribosomal protein P0)



```
                                 *   .      :       .          .   *         : : :      .
Q5E940_BOVIN -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE  76
  RLA0_HUMAN -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE  76
  RLA0_MOUSE -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE  76
    RLA0_RAT -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE  76
  RLA0_CHICK -----------MPREDRATWKSNYFMKIIQLLDDYPKCFVVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE  76
  RLA0_RANSY -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQQIRMSLRGK-AVVLMGKNTMMRKAIRGHLENN--SALE  76
Q7ZUG3_BRARE -----------MPREDRATWKSNYFLKIIQLLDDYPKCFIVGADNVGSKQMQTIRLSLRGK-AVVLMGKNTMMRKAIRGHLENN--PALE  76
  RLA0_ICTPU -----------MPREDRATWKSNYFLKIIQLLNDYPKCFIVGADNVGSKQMQTIRLSLRGK-AIVLMGKNTMMRKAIRGHLENN--PALE  76
  RLA0_DROME -----------MVRENKAAWKAQYFIKVVELFDEFPKCFIVGADNVGSKQMQNIRTSLRGL-AVVLMGKNTMMRKAIRGHLENN--PQLE  76
  RLA0_DICDI -----------MSGAG-SKRKKLFIEKATKLFTTYDKMIVAEADFVGSSQLQKIRKSIRGI-GAVLMGKKTMIRKVIRDLADSK--PELD  75
Q54LP0_DICDI -----------MSGAG-SKRKNVFIEKATKLFTTYDKMIVAEADFVGSSQLQKIRKSIRGI-GAVLMGKKTMIRKVIRDLADSK--PELD  75
  RLA0_PLAF8 -----------MAKLSKQQKKQMYIEKLSSLIQQYSKILIVHVDNVGSNQMASVRKSLRGK-ATILMGKNTRIRTALKKNLQAV--PQIE  76
  RLA0_SULAC -----MIGLAVTTTKKIAKWKVDEVAELTEKLKTHKTIIIANIEGFPADKLHEIRKKLRGK-ADIKVTKNNLFNIALKNAG-----YDTK  79
  RLA0_SULTO ----MRIMAVITQERKIAKWKIEEVKELEQKLREYHTIIIANIEGFPADKLHDIRKKMRGM-AEIKVTKNTLFGIAAKNAG-----LDVS  80
  RLA0_SULSO ----MKRLALALKQRKVASWKLEEVKELTELIKNSNTILIGNLEGFPADKLHEIRKKLRGK-ATIKVTKNTLFKIAAKNAG-----IDIE  80
  RLA0_AERPE MSVVSLVGQMYKREKPIPEWKTLMLRELEELFSKHRVVLFADLTGTPTFVVQRVRKKLWKK-YPMMVAKKRIILRAMKAAGLE---LDDN  86
  RLA0_PYRAE -MMLAIGKRRYVRTRQYPARKVKIVSEATELLQKYPYVFLFDLHGLSSRILHEYRYRLRRY-GVIKIIKPTLFKIAFTKVYGG---IPAE  85
  RLA0_METAC ------MAEERHHTEHIPQWKKDEIENIKELIQSHKVFGMVGIEGILATKMQKIRRDLKDV-AVLKVSRNTLTERALNQLG-----ETIP  78
  RLA0_METMA ------MAEERHHTEHIPQWKKDEIENIKELIQSHKVFGMVRIEGILATKIQKIRRDLKDV-AVLKVSRNTLTERALNQLG-----ESIP  78
  RLA0_ARCFU ------MAAVRGS---PPEYKVRAVEEIKRMISSKPVVAIVSFRNVPAGQMQKIRREFRGK-AEIKVVKNTLLERALDALG-----GDYL  75
  RLA0_METKA MAVKAKGQPPSGYEPKVAEWKRREVKELKELMDEYENVGLVDLEGIPAPQLQEIRAKLRERDTIIRMSRNTLMRIALEEKLDER--PELE  88
  RLA0_METTH -------------MAHVAEWKKKEVQELHDLIKGYEVVGIANLADIPARQLQKMRQTLRDS-ALIRMSKKTLISLALEKAGREL--ENVD  74
  RLA0_METTL -------MITAESEHKIAPWKIEEVNKLKELLKNGQIVALVDMMEVPARQLQEIRDKIR-GTMTLKMSRNTLIERAIKEVAEETGNPEFA  82
  RLA0_METVA -------MIDAKSEHKIAPWKIEEVNALKELLKSANVIALIDMMEVPAVQLQEIRDKIR-DQMTLKMSRNTLIKRAVEEVAEETGNPEFA  82
  RLA0_METJA --------METKVKAHVAPWKIEEVKTLKGLIKSKPVVAIVDMMDVPAPQLQEIRDKIR-DKVKLRMSRNTLIIRALKEAAEELNNPKLA  81
  RLA0_PYRAB -------------MAHVAEWKKKEVEELANLIKSYPVIALVDVSSMPAYPLSQMRRLIRENGGLLRVSRNTLIELAIKKAAQELGKPELE  77
  RLA0_PYRHO -------------MAHVAEWKKKEVEELAKLIKSYPVIALVDVSSMPAYPLSQMRRLIRENGGLLRVSRNTLIELAIKKAAKELGKPELE  77
  RLA0_PYRFU -------------MAHVAEWKKKEVEELANLIKSYPVVALVDVSSMPAYPLSQMRRLIRENNGLLRVSRNTLIELAIKKVAQELGKPELE  77
  RLA0_PYRKO -------------MAHVAEWKKKEVEELANIIKSYPVIALVDVAGVPAYPLSKMRDKLR-GKALLRVSRNTLIELAIKRAAQELGQPELE  76
  RLA0_HALMA -----MSAESERKTETIPEWKQEEVDAIVEMIESYESVGVVNIAGIPSRQLQDMRRDLHGT-AELRVSRNTLLERALDDVD-----DGLE  79
  RLA0_HALVO -----MSESEVRQTEVIPQWKREEVDELVDFIESYESVGVVGVAGIPSRQLQSMRRELHGS-AAVRMSRNTLVNRALDEVN-----DGFE  79
  RLA0_HALSA -----MSAEEQRTTEEVPEWKRQEVAELVDLLETYDSVGVVNVTGIPSKQLQDMRRGLHGQ-AALRMSRNTLLVRALEEAG-----DGLD  79
  RLA0_THEAC -------------MKEVSQQKKELVNEITQRIKASRSVAIVDTAGIRTRQIQDIRGKNRGK-INLKVIKKTLLFKALENLGD----EKLS  72
  RLA0_THEVO -------------MRKINPKKKEIVSELAQDITKSKAVAIVDIKGVRTRQMQDIRAKNRDK-VKIKVVKKTLLFKALDSIND----EKLT  72
  RLA0_PICTO -------------MTEPAQWKIDFVKNLENEINSRKVAAIVSIKGLRNNEFQKIRNSIRDK-ARIKVSRARLLRLAIENTGK----NNIV  72
       ruler 1.......10........20........30........40........50........60........70........80........90
```

By Miguel Andrade, CC-BY-SA 3.0, wikipedia:`RPLP0 90 ClustalW aln.gif`

Definition

Multiple Alignment Alphabet

- Let Σ be the character alphabet.
- Then $\mathcal{A}(k) := (\Sigma \cup \{-\})^k \setminus \{(-)^k\}$ is the **multiple alignment alphabet** of k sequences.

Example

For $k = 3$ and $\Sigma = \{\texttt{A}, \texttt{G}\}$, there are 26 elements:

$$\mathcal{A}(3) = \begin{pmatrix} \texttt{A} & \texttt{A} & \texttt{A} & \texttt{A} & \texttt{A} & \texttt{A} & \texttt{A} & \texttt{A} & \texttt{A} & \dots \\ \texttt{A} & \texttt{A} & \texttt{A} & \texttt{G} & \texttt{G} & \texttt{G} & \texttt{-} & \texttt{-} & \texttt{-} & \dots \\ \texttt{A} & \texttt{G} & \texttt{-} & \texttt{A} & \texttt{G} & \texttt{-} & \texttt{A} & \texttt{G} & \texttt{-} & \dots \end{pmatrix}$$

Global MSA

Global Multiple Sequence Alignment (MSA)

A **global multiple alignment** A of $s_1, \ldots, s_k \in \Sigma^*$ is a sequence over $\mathcal{A}(k)$ with projections $\pi_{\{i\}}(A) = s_i$ for all $i = 1, \ldots, k$.

Definition: Projections

The **projection** $\pi_{\{i\}}$ of an alignment **column** c to the i^{th} sequence is the function $\mathcal{A}(k) \to \Sigma^? := \Sigma^0 \cup \Sigma^1$ with

$$\pi_{\{i\}}\left(c = \begin{pmatrix} a_1 \\ \vdots \\ a_k \end{pmatrix}\right) := \begin{cases} \mathrm{a}_i & \text{if } \mathrm{a}_i \neq -, \\ \epsilon & \text{if } \mathrm{a}_i = -. \end{cases}$$

The **projection** of a **multiple sequence alignment** $A = c_1 \cdots c_n$ to sequence i is the concatenation of the projections of the respective columns:

$$\pi_{\{i\}}(A = (c_1, \ldots, c_n)) := \pi_{\{i\}}(c_1) \cdots \pi_{\{i\}}(c_n) \, .$$

Projection to Index Set

Definition

The **projection** $\pi_{\mathcal{I}}$ of an alignment **column** c to the **index set** $\mathcal{I} = \{i_1, \ldots, i_q\}$ is the function $\mathcal{A}(k) \to \mathcal{A}(q)^?$ with:

$$\pi_{\mathcal{I}}\left(c = \begin{pmatrix} a_1 \\ \vdots \\ a_k \end{pmatrix}\right) := \begin{cases} \epsilon & \text{if } \begin{pmatrix} a_{i_1} \\ \vdots \\ a_{i_q} \end{pmatrix} = \begin{pmatrix} - \\ \vdots \\ - \end{pmatrix}, \\ \begin{pmatrix} a_{i_1} \\ \vdots \\ a_{i_q} \end{pmatrix} & \text{otherwise.} \end{cases}$$

The **projection** of a **multiple sequence alignment** $A = c_1 \cdots c_n$ to **index set** $\mathcal{I}$ is the concatenation of the projections of the respective columns:

$$\pi_{\mathcal{I}}(A) = \pi_{\mathcal{I}}(c_1) \cdots \pi_{\mathcal{I}}(c_n) .$$

Example

$$A = \begin{pmatrix} - & A & C & C & - & - & A & T & G \\ - & A & - & C & G & A & A & T & - \\ T & A & C & C & - & - & A & G & G \\ - & A & - & C & C & A & A & T & G \end{pmatrix}$$

Example

$$A = \begin{pmatrix} - & A & C & C & - & - & A & T & G \\ - & A & - & C & G & A & A & T & - \\ T & A & C & C & - & - & A & G & G \\ - & A & - & C & C & A & A & T & G \end{pmatrix}$$

$$\pi_{\{1,2\}}(A) = \begin{pmatrix} A & C & C & - & - & A & T & G \\ A & - & C & G & A & A & T & - \end{pmatrix}$$

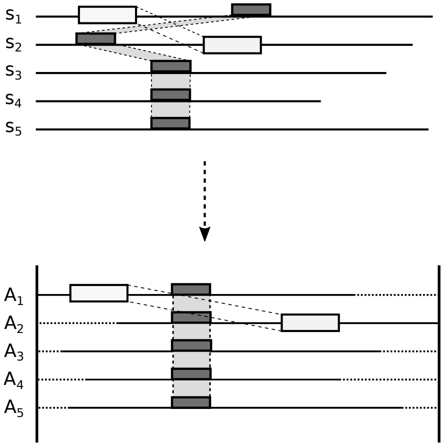
Example

$$A = \begin{pmatrix} - & A & C & C & - & - & A & T & G \\ - & A & - & C & G & A & A & T & - \\ T & A & C & C & - & - & A & G & G \\ - & A & - & C & C & A & A & T & G \end{pmatrix}$$

$$\pi_{\{1,2\}}(A) = \begin{pmatrix} A & C & C & - & - & A & T & G \\ A & - & C & G & A & A & T & - \end{pmatrix}$$

$$\pi_{\{3,4\}}(A) = \begin{pmatrix} T & A & C & C & - & - & A & G & G \\ - & A & - & C & C & A & A & T & G \end{pmatrix}$$

Why multiple sequence comparison



Why multiple sequence comparison

Pairiwse alignment ambiguities may be resolved by additional information.

Example

$s_1 = \texttt{VIEQLA}$ and $s_2 = \texttt{VINLA}$ may be aligned in the two different ways:

$$A_1 = \begin{pmatrix} V & I & E & Q & L & A \\ V & I & N & - & L & A \end{pmatrix}$$

and

$$A_2 = \begin{pmatrix} V & I & E & Q & L & A \\ V & I & - & N & L & A \end{pmatrix}$$

Additional sequence $s_3 = \texttt{VINQLA}$ shows that alignment A_1 is probably the correct one.

Multiple Alignment Problem

Multiple Sequence Alignment Problem

Given k sequences $s_1, s_2, \ldots, s_k$ and alignment score (cost) function S (D), find an alignment A^{opt} of $s_1, s_2, \ldots, s_k$ such that $S(A^{\text{opt}})$ is maximal ($D(A^{\text{opt}})$ is minimal) among all possible alignments of $s_1, s_2, \ldots, s_k$.

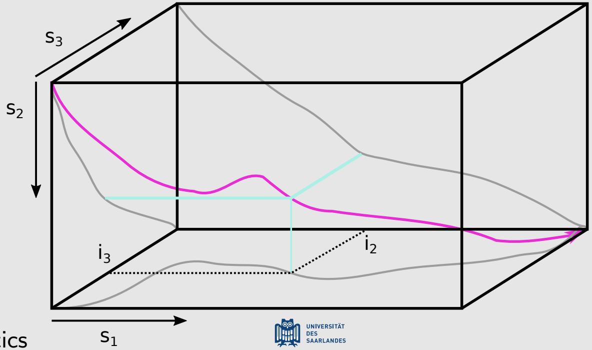
Such an alignment A^{opt} is called an **optimal alignment**, and
$S(s_1, s_2, \ldots, s_k) := S(A^{\text{opt}})$ is the **optimal alignment score** and
$D(s_1, s_2, \ldots, s_k) := D(A^{\text{opt}})$ is the **optimal alignment cost** of $s_1, s_2, \ldots, s_k$.

(Weighted) Sum-of-Pairs Score (Cost)

Definition

Score a multiple alignment by the sum of scores of all pairwise projections.

$$S_{\mathrm{[W]SP}}(A) := \sum_{1 \le p < q \le k} [w_{p,q}] \cdot S(\pi_{\{p,q\}}(A))$$



Example: Weighted Cost of an Alignment

[Weighted] Cost/Distance

$$D_{[\mathrm{W}]\mathrm{SP}}(A) := \sum_{1\le p<q\le k} [w_{p,q}] \cdot D(\pi_{\{p,q\}}(A))$$

Let $s_1 = \texttt{CGCTT}$, $s_2 = \texttt{ACGGT}$, $s_3 = \texttt{GCTGT}$.

$$A_2 = \begin{pmatrix} C & G & C & T & - & T \\ - & A & C & G & G & T \\ - & G & C & T & G & T \end{pmatrix}$$

Let D be the unit cost edit distance.
Then $D_{\mathrm{SP}}(A) = 4 + 2 + 2 = 8$,
and $D_{\mathrm{WSP}}(A) = 4w_{1,2} + 2w_{1,3} + 2w_{2,3}$.

Example: Sum-of-Pairs MSA

Let $s_1 = \texttt{CGCG}$, $s_2 = \texttt{ACGC}$ and $s_3 = \texttt{GCGA}$.
In a unit cost scenario, the (only) optimal alignment of s_1 and s_2 is:

$$A^{(1,2)} = \begin{pmatrix} - & C & G & C & G \\ A & C & G & C & - \end{pmatrix}$$

with cost $D(A^{(1,2)}) = 2$.
The (only) optimal alignment of s_1 and s_3 is

$$A^{(1,3)} = \begin{pmatrix} C & G & C & G & - \\ - & G & C & G & A \end{pmatrix}$$

with cost $D(A^{(1,3)}) = 2$.

Example: Sum-of-Pairs MSA

Combining the two alignments into one multiple alignment, using the common sequence s_1 as seed, yields the multiple alignment

$$A^{((1,2),(1,3))} = \begin{pmatrix} - & C & G & C & G & - \\ A & C & G & C & - & - \\ - & - & G & C & G & A \end{pmatrix}$$

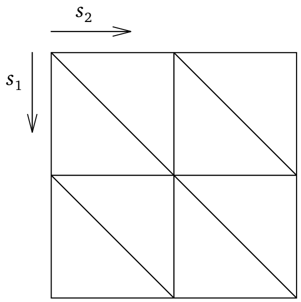
with cost $D(A^{((1,2),(1,3))}) = 2 + 2 + 4 = 8$.
However, this is not the sum-of-pairs optimal alignment, which is

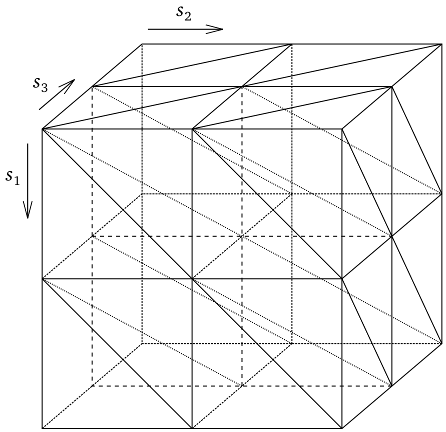
$$A^{\text{opt}} = \begin{pmatrix} - & C & G & C & G \\ A & C & G & C & - \\ G & C & G & A & - \end{pmatrix}$$

with cost $D(A^{\text{opt}}) = 2 + 3 + 2 = 7$.

Algorithms for Sum-of-Pairs Multiple Alignment

Alignment Graph: From 2 to k Dimensions





An Exact Solution: Universal Alignment Algorithm

The Needleman-Wunsch algorithm for pairwise global alignment can be generalized for the multiple alignment of k sequences $s_1, s_2, \ldots, s_k$ of lengths $n_1, n_2, \ldots, n_k$, respectively.

An Exact Solution: Universal Alignment Algorithm

The Needleman-Wunsch algorithm for pairwise global alignment can be generalized for the multiple alignment of k sequences $s_1, s_2, \ldots, s_k$ of lengths $n_1, n_2, \ldots, n_k$, respectively.

- k-dimensional weighted edit graph
- Edge e corresponds to a possible alignment column c
- Each edge weighted by its corresponding alignment score $w(e) = S(c)$
- Optimal alignment = maximum scoring path from source to sink.

An Exact Solution: Universal Alignment Algorithm

The Needleman-Wunsch algorithm for pairwise global alignment can be generalized for the multiple alignment of k sequences $s_1, s_2, \ldots, s_k$ of lengths $n_1, n_2, \ldots, n_k$, respectively.

- k-dimensional weighted edit graph
- Edge e corresponds to a possible alignment column c
- Each edge weighted by its corresponding alignment score $w(e) = S(c)$
- Optimal alignment = maximum scoring path from source to sink.

Minimization version

For each vertex v in the edit graph in topological order:

$$D(v) = \min\{D(v') + w(v' \rightarrow v) \mid v' \text{ is a predecessor of v}\}.$$

Universal Algorithm Explicitly

$D(0, 0, \ldots, 0) = 0$ and

$$D(\overbrace{i_1, i_2, \ldots, i_k}^{v}) = \min_{\substack{\Delta_1,\ldots,\Delta_k \in \{0,1\} \\ \Delta_1 + \cdots + \Delta_k \neq 0}} \left\{ D(\overbrace{i_1 - \Delta_1, i_2 - \Delta_2, \ldots, i_k - \Delta_k}^{\text{predecessor } v'}) + D_{\mathrm{SP}} \overbrace{\begin{pmatrix} \Delta_1 s_1[i_1 - 1] \\ \vdots \\ \Delta_k s_k[i_k - 1] \end{pmatrix}}^{alignm.col.} \right\} .$$

Notation: For $c \in \Sigma$, let $\Delta c := c$ if $\Delta = 1$ and $\Delta c = " - "$ if $\Delta = 0$.

Universal Algorithm Explicitly

$D(0, 0, \dots, 0) = 0$ and

$$D(\overbrace{i_1, i_2, \dots, i_k}^{v}) = \min_{\substack{\Delta_1,\dots,\Delta_k \in \{0,1\} \\ \Delta_1+\dots+\Delta_k \neq 0}} \left\{ D(\overbrace{i_1 - \Delta_1, i_2 - \Delta_2, \dots, i_k - \Delta_k}^{\text{predecessor } v'}) + D_{\mathrm{SP}} \overbrace{\begin{pmatrix} \Delta_1 s_1[i_1 - 1] \\ \vdots \\ \Delta_k s_k[i_k - 1] \end{pmatrix}}^{alignm.col.} \right\}.$$

Notation: For $c \in \Sigma$, let $\Delta c := c$ if $\Delta = 1$ and $\Delta c = " - "$ if $\Delta = 0$.

Observation: In general, a node has $2^k - 1$ predecessors.

Space and Time Complexity of Sum-of-Pairs Multiple Alignment

Space Complexity

The space complexity is the size of the k-dimensional edit graph:
$O(n_1 n_2 \ldots n_k) = O(n^k)$ if n is the maximum sequence length.

Space and Time Complexity of Sum-of-Pairs Multiple Alignment

Space Complexity

The space complexity is the size of the k-dimensional edit graph:
$O(n_1 n_2 \ldots n_k) = O(n^k)$ if n is the maximum sequence length.
We can save one dimension if we do not need traceback: $O(n^{k-1})$.

Space and Time Complexity of Sum-of-Pairs Multiple Alignment

Space Complexity

The space complexity is the size of the k-dimensional edit graph:
$O(n_1 n_2 \dots n_k) = O(n^k)$ if n is the maximum sequence length.
We can save one dimension if we do not need traceback: $O(n^{k-1})$.

Time Complexity

- $O(n^k)$ nodes to compute
- For each node: minimization/maximization over $O(2^k)$ predecessors
- For each predecessor: Compute edge cost/score: $O(k^2)$ (sum-of-pairs)

Total (for linear gap costs): $O((2n)^k \cdot k^2)$

Space and Time Complexity of Sum-of-Pairs Multiple Alignment

Space Complexity

The space complexity is the size of the k-dimensional edit graph:
$O(n_1 n_2 \dots n_k) = O(n^k)$ if n is the maximum sequence length.
We can save one dimension if we do not need traceback: $O(n^{k-1})$.

Time Complexity

- $O(n^k)$ nodes to compute
- For each node: minimization/maximization over $O(2^k)$ predecessors
- For each predecessor: Compute edge cost/score: $O(k^2)$ (sum-of-pairs)

Total (for linear gap costs): $O((2n)^k \cdot k^2)$

NP Hardness

Optimal Multiple Alignment with both weighted and unweighted sum-of-pairs score (or distance) is an NP-hard optimization problem in k (Wang and Jiang, 1994).

Approximation Algorithms

c-approximation, $c \geq 1$

For $c \geq 1$, an algorithm for a cost or distance minimization problem is a **c-approximation** if its output solution has cost at most c times the optimal solution:

$$output \leq c \cdot opt_{\text{min}}$$

For a maximization problem, an algorithm is called a c-approximation if its output solution has score at least $\frac{1}{c}$ times the optimal solution:

$$output \geq opt_{\text{max}}/c$$

Approximation Algorithms

c-approximation, $c \geq 1$

For $c \geq 1$, an algorithm for a cost or distance minimization problem is a ***c*-approximation** if its output solution has cost at most c times the optimal solution:

$$output \leq c \cdot opt_{\text{min}}$$

For a maximization problem, an algorithm is called a c-approximation if its output solution has score at least $\frac{1}{c}$ times the optimal solution:

$$output \geq opt_{\text{max}}/c$$

Opinion

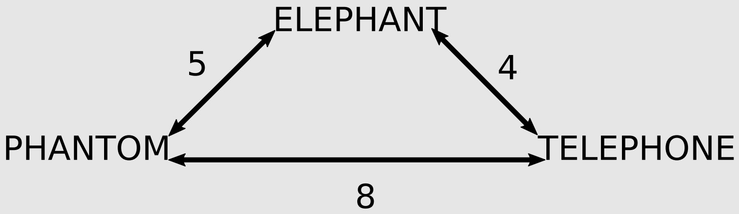
Theoretical computer scientists like approximation algorithms. I don't:
Even a 2-approximation algorithm is bad in practice (in the worst case).
Good: $(1 + \epsilon)$ approximation algorithm for every ϵ, polynomial time.

The Center Star Approximation

Algorithm

The **center star algorithm** (Gusfield, 1991, 1993) is a 2-approximation for the sum-of-pairs **distance** multiple alignment problem if the underlying weighted edit distance satisfies the **triangle inequality**.

Example



The Center Star Approximation

Overall distance

For each sequence s_p, $1 \leq p \leq k$, its total distance d_p to the other sequences is the sum of the pairwise optimal alignment costs:

$$d_p = \sum_{1 \leq q \leq k} d(s_p, s_q)$$

Idea

let s_c be the sequence that minimizes this overall distance, called **center sequence**.
A multiple alignment A_c is constructed from all pairwise optimal alignments where the center sequence is involved, i.e., all the optimal alignments of s_c and the other s_p, $p \neq c$, are combined into one multiple alignment A_c.

Center Star Theorem

Theorem

The Center Star algorithm is a 2-approximation for the optimal sum-of-pairs multiple alignment: $D_{\mathrm{SP}}(A_c) \leq 2 \cdot D_{\mathrm{SP}}(A^*)$.

Center Star Theorem

Theorem

The Center Star algorithm is a 2-approximation for the optimal sum-of-pairs multiple alignment: $D_{\mathrm{SP}}(A_c) \leq 2 \cdot D_{\mathrm{SP}}(A^*)$.

Proof.

Write D^c_{ij} for the cost of the induced pairwise alignment $\pi_{\{i,j\}}(A^c)$, and D^*_{ij} for A^*.
Write $d(s_i, s_j)$ for the (optimal) pairwise distance between s_i and s_j.

Lemma: $D^c_{ij} \leq d(s_i, s_c) + d(s_c, s_j)$ for all i, j (proof follows).

Center Star Theorem

Theorem

The Center Star algorithm is a 2-approximation for the optimal sum-of-pairs multiple alignment: $D_{\mathrm{SP}}(A_c) \leq 2 \cdot D_{\mathrm{SP}}(A^*)$.

Proof.

Write D^c_{ij} for the cost of the induced pairwise alignment $\pi_{\{i,j\}}(A^c)$, and D^*_{ij} for A^*. Write $d(s_i, s_j)$ for the (optimal) pairwise distance between s_i and s_j.

Lemma: $D^c_{ij} \leq d(s_i, s_c) + d(s_c, s_j)$ for all i, j (proof follows).

Then, $2D_{\mathrm{SP}}(A_c) = \sum_{i \neq j} D^c_{ij} \leq \sum_{i \neq j} [d(s_i, s_c) + d(s_c, s_j)] = 2(k-1) \cdot \sum_{j \neq c} d(s_c, s_j)$.
Also, $2D_{\mathrm{SP}}(A^*) = \sum_{i \neq j} D^*_{ij} \geq \sum_{i \neq j} d(s_i, s_j) = \sum_i \sum_{j \neq i} d(s_i, s_j) \geq k \cdot \sum_{j \neq c} d(s_c, s_j)$ by the choice of c.

Center Star Theorem

Theorem

The Center Star algorithm is a 2-approximation for the optimal sum-of-pairs multiple alignment: $D_{\mathrm{SP}}(A_c) \leq 2 \cdot D_{\mathrm{SP}}(A^*)$.

Proof.

Write D^c_{ij} for the cost of the induced pairwise alignment $\pi_{\{i,j\}}(A^c)$, and D^*_{ij} for A^*.
Write $d(s_i, s_j)$ for the (optimal) pairwise distance between s_i and s_j.

Lemma: $D^c_{ij} \leq d(s_i, s_c) + d(s_c, s_j)$ for all i, j (proof follows).

Then, $2D_{\mathrm{SP}}(A_c) = \sum_{i \neq j} D^c_{ij} \leq \sum_{i \neq j} [d(s_i, s_c) + d(s_c, s_j)] = 2(k-1) \cdot \sum_{j \neq c} d(s_c, s_j)$.
Also, $2D_{\mathrm{SP}}(A^*) = \sum_{i \neq j} D^*_{ij} \geq \sum_{i \neq j} d(s_i, s_j) = \sum_i \sum_{j \neq i} d(s_i, s_j) \geq k \cdot \sum_{j \neq c} d(s_c, s_j)$
by the choice of c.

It follows that $D_{\mathrm{SP}}(A_c)/D_{\mathrm{SP}}(A^*) \leq 2(k-1)/k \leq 2$ for any k. $\square$

Proof of Lemma

Lemma

$D_{ij}^{c} \leq d(s_i, s_c) + d(s_c, s_j)$ for all i, j

Proof of Lemma

Lemma

$D^c_{ij} \leq d(s_i, s_c) + d(s_c, s_j)$ for all i, j

Proof.

Because the cost function satisfies the triangle inequality, we have for all i, j:

$$D^c_{ij} \leq D^c_{ic} + D^c_{cj}.$$

Because the induced alignments involving the center sequence are by construction optimal, we have $D^c_{ic} = d(s_i, s_c)$ and $D^c_{cj} = d(s_c, s_j)$. The lemma follows. $\square$

Time and Space Complexity

Time complexity

- Phase 1: Compute $\binom{k}{2}$ pairwise alignments; pick center: $O(k^2 n^2)$
- Phase 2: Combine $k-1$ alignments into one multiple alignment: $O(k^2 n)$
- Overall running time: $O(k^2 n^2)$

Time and Space Complexity

Time complexity

- Phase 1: Compute $\binom{k}{2}$ pairwise alignments; pick center: $O(k^2 n^2)$
- Phase 2: Combine $k-1$ alignments into one multiple alignment: $O(k^2 n)$
- Overall running time: $O(k^2 n^2)$

Space complexity

- $\mathcal{O}(n+k)$ for computing and storing k computed values of d_p
- $\mathcal{O}(k^2 n)$ to store k pairwise alignments and intermediate/final multiple alignments (size of multiple alignment: up to $O(nk)$ columns, typically only $O(n)$)
- Overall space complexity: $\mathcal{O}(k^2 n)$

Summary

Multiple Sequence Alignment (MSA)

- Sum-of-pairs objective function
- Universal DP algorithm: exponential space and time in k; NP-hard
- The Center Star 2-approximation: $O(n^2k^2)$ time, $O(k^2n)$ space

Possible Exam Questions

- Define a global multiple sequence alignment (MSA).
- What is the advantage of an MSA compared to a pairwise alignment?
- Define the sum-of-pairs objective for MSA.
- Do you know an algorithm to find the optimal MSA (wrt this objective)?
- What is its time and space complexity?
- Is there an exact algorithm with running time polynomial in the number of sequences k?
- What is an approximation algorithm?
- Explain the Center Star algorithm and its assumptions.
- What is its running time?
- Can you sketch the ideas for proving it to be a 2-approximation?