Department of Computing

Guided Studies: COMP6821 Bioinformatics
PhD candidate: Yip Chi Kin
Chief Supervisor: Prof. Keith C.C.Chan
Presentation Date: 21-12-2006 (2:30 pm Room PQ815)
Multiple Sequence Composition Alignment

• Studied Papers

[B03] Composition Alignment
Gary Benson. (2003)
Algorithms in Bioinformatics.
Third International Workshop, WABI 2003, pp. 447-461

[SMS03] Divide-and-conquer multiple alignment with segment-based constraints
Michael Sammeth, Burkhard Morgenstern and Jens Stoye. (2003)
Bioinformatics. Vol.19 Suppl. 2, 2003, pages ii189-ii195.

[M99] DIALIGN 2: improvement of the segment-to-segment approach to multiple sequence alignment
Burkhard Morgenstern. (1999)
Bioinformatics. Vol.15 No.3, 1999, pages 211-218.

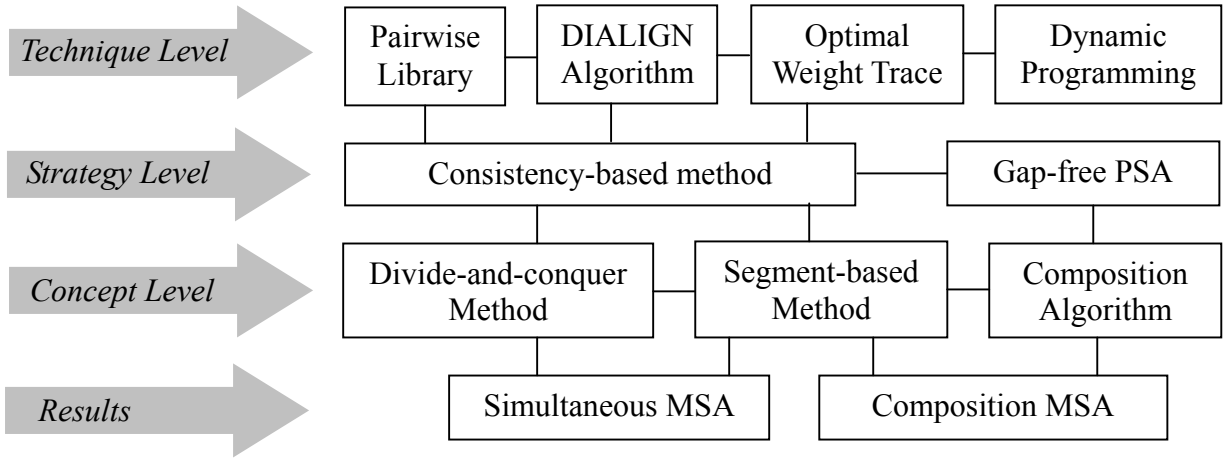
[S98] Multiple sequence alignment with the divide-and-conquer method
Jens Stoye. (1998)
Gene, 211 (1998) GC45-GC56.

• Report Contents:

Multiple Sequence Composition Alignment

1. Introduction

Multiple sequence alignment (MSA) is aimed at reproducing these homologous relationships among individual residues in a set of gene or protein sequences. Mutations (substitutions, deletions, or insertions) that have occurred during the evolutionary process make the inter-residue homologous relationships obscure and sometimes barely detectable. Hence, it is vital to obtain a reliable MSA from a set of remotely related sequences. On the other hand, the sequence data to be analyzed are accumulating at an increasing rate, and therefore, fast and reliable MSA methods are earnestly desired. However, dynamic programming-based algorithms can only handle a small number of sequences due to time and space complexities these algorithms have. Hence, divide-and-conquer method and Segment-based Alignment would be applied in MSA. The main strategies and techniques are using in this report as shown below. They are close link together.



The main purpose of this report is to introduce a new algorithm “Composition Alignment” to apply in MSA by using Divide-and-conquer method. Meta-code is applied to combine those algorithms, in order to find the simultaneous MSA. I tried to evaluate algorithms by examples and concentrate most of our attention on the real applications. All the theories proofs are referred to [B03], [SMS03], [M99], and [S98].

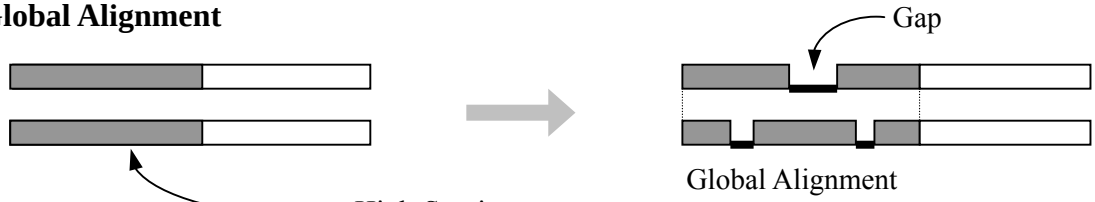
2. Terminology

- **Pairwise Sequence Alignment (PSA)**

ACGTCTAG	5 mismatches	ACGTCTAG	2 mismatches	ACGTCTAG	0 mismatches
ACTCTAG–	1 gap	–ACTCTAG	1 gap	AC–TCTAG	1 gap
••	2 matches	•••••	5 matches	•• •••••	7 matches

Alignment Score = Matches + Mismatches + Gaps

- **Global Alignment**



- **Local Alignment**

- **Semi-global Alignment**

A lower global score but is much more biologically meaningful by using DP.

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

Mismatch

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

An optimal global alignment matching

- **Fragment**

Ungapped local alignments between two sequences are called ***fragments*** or ***diagonals***.

```
        1  2  3  4  5  6  7  8  9
S1:     T  C  T  A  G  G  T  A  A
S2:     G  A  A  T  A  G  G  T  C
```

Fragment, f_{12} = ([1,3], [2,4], 5)

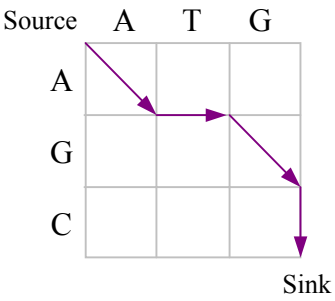
- **Multiple Alignment**

e.g. A multiple alignment of the sequences AAGATA, ATCGATG, and CTCGG is

```
S1:  A  A  –  G  A  T  A
S2:  A  T  C  G  A  T  G
S3:  C  T  C  G  –  –  G
```

- **Edit Graph of Dynamic Programming**

  The longest path between vertices (source) and (sink) in the edit graph.



- **Consistency**

  The notion of 'consistency' implied in the DIALIGN algorithm [M99] means that two ungapped segment pairs (diagonals or fragments)

- **Composition Difference** $\Sigma = \{0,1\}$ x = 010111010001000 y = 010001101110111

  CD(x,y) = (3,-3) where CD is Composition Difference

- **Composition Match** (cm) X = AACGTCTTTGAGCTC Y = AGCCTGACTGCCTA

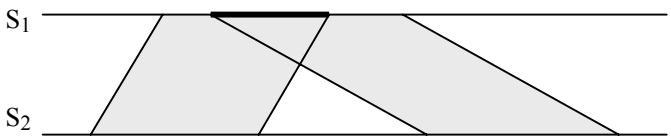
  X[2,4] = ACG Y[6,8] = GAC

- **Composition Alignment**

  | Single character match
  ⟷ Composition matches
  Total length of CM = 8

```
A A C G T C T T T G A G C T C
|   | ⟷     |   ⟷⟷⟷⟷⟷
A G C C T G A C T - G C C T A
```

- **Overlap match**



- **Pairwise Library**

  A pairwise library is a collection of the global pairwise alignment of the input sequence. For a data set containing N sequences, there should be N(N–1)/2 pairwise alignments in the pairwise library.

- **T-COFFEE** **T**ree-based **C**onsistency-based **O**bjective **F**unction **F**or alignm**E**nt **E**valuation

- **DIALIGN** **DI**agonal **ALIGN**ment

3. Dynamic Programming (Global Alignment & Local Alignment)

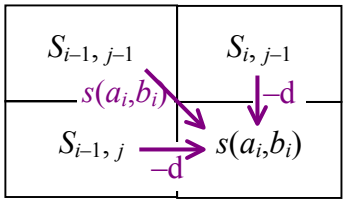
- Dynamic Programming (DP)

A dynamic programming algorithm typically consists of a forward and backward iteration algorithm. The forward iteration algorithm evaluates the data required by the backward iteration algorithm to determine the optimal decisions. The optimal pairwise alignment algorithms presented follow the paradigm of dynamic programming which has been introduced by Bellman [B57]. The optimal global alignment has been developed by Needleman & Wunsch [NW70] and the optimal local alignment by Smith & Waterman [SW81].

Here is an example of *edit graph* of dynamic programming. The idea is to build up an optimal alignment of small sub-sequences. We need to construct a matrix S indexed by i and j, one index for each sequence, where the values $S_{i,j}$ is the score of the best alignment between the initial segments of each sequence. We build $S_{i,j}$ recursively, and begin by initialising $S_{0,0} = 0$, then proceed to fill the matrix from top left to bottom right. There are three possible ways that the best score $S_{i,j}$ of an alignment could be obtained. The best score up to (i,j) will be the largest of these three options.

Therefore, $S_{i,j}$ equation shown as below. The equation is applied repeatedly to fill in the matrix of $S_{i,j}$ values, calculating the value in the bottom right-hand corner of each square of four cells from one of the other three values as in the following figure.

$$S_{i,j} = \max \begin{cases} S_{i-1,\,j-1} + s(a_i,b_i) \\ S_{i-1,\,j} - \mathrm{d} \\ S_{i,\,j-1} - \mathrm{d} \end{cases}$$



As we fill in the $S_{i,j}$ values, we also keep a pointer in each cell back to the cell from which its was derived, as shown in the example of the dynamic programming matrix in the following paragraph of global alignment and local alignment.

- Global Alignment

Global alignment is the common means to measure the degree of overall similarity between two sequences. Here is a computational example to find the maximum score and alignment for aligning **A** = GCATC with **B** = CTTCT shown as below:

For scoring, take if $a_i = b_j$ then $s(a_i,b_j) = +1$,
if $a_i \neq b_j$ then $s(a_i,b_j) = -1$,
and $s(a_i,b_{j-1}) = s(a_{i-1},b_j) = -2$.

		j	0	1	2	3	4	5
i			–	C	T	T	C	T
0	–		0	–2	–4	–6	–8	–10
1	G		–2					
2	C		–4	•			•	
3	A		–6					
4	T		–8		•	•		•
5	C		–10	•			•	

Step 1: Write down the alignment matrix using **B** along the top and **A** in a column at the side.

Step 2: Fill in the first row and first column by using $S_{i,0} = -2i$, $S_{0,j} = -2j$.

Step 3: Fill in all matrix elements using the scoring rules of

$$S_{i,j} = \max \begin{cases} S_{i-1,\,j-1} + s(a_i,b_i) \\ S_{i-1,\,j} - 2 \\ S_{i,\,j-1} - 2 \end{cases}$$

and keeping track of the path into each element that was employed. Sometimes two paths will yield equal scores.

		–	C	T	T	C	T
–		**0**	–2	–4	–6	–8	–10
G		**–2**	–1	–3	–5	–7	–9
C		–4	**–1**	–2	–4	–4	–6
A		–6	–3	**–2**	–3	–5	–5
T		–8	–5	–2	**–1**	–3	–4
C		–10	–7	–4	–3	**0**	**–2**

Step 4: Follow arrows corresponding to the path for the highest-scoring alignment.

The ***Global Alignment*** result is

A: G C A T C –
B: – C T T C T

- Local Alignment

Sequences often exhibit poor global similarity, but may contain similar subsequences that are preserved during evolution. The problem of finding highly related subsequences of two sequences is accomplished by local alignment. Here is a computational example to determine the best local alignment and the maximum score for aligning **A** = ACCTAAG and **B** = GGGCTCAATCA shown as below.

For scoring, take if $a_i = b_j$ then $s(a_i,b_j) = +2$,
if $a_i \neq b_j$ then $s(a_i,b_j) = -1$,
and $s(a_i,b_{j-1}) = s(a_{i-1},b_j) = -2$.

Step 1: Write down the alignment matrix using **B** along the top and **A** in a column at the side.

Step 2: Fill in the first row and first column by using $S_{i,0} = S_{0,j} = 0$.

Step 3: Then fill in all matrix elements using the scoring rules $S_{i,j} = \max \begin{cases} S_{i-1,\,j-1} + s(a_i,b_i) \\ S_{i-1,\,j} - 2 \\ S_{i,\,j-1} - 2 \\ 0 \end{cases}$,

keeping track of the path into each element. For clarity, we have included below only the arrows around the higher-scoring path. Observe some cells corresponding to scoring rule would have yielded –1 were them not for the requirement that all elements be non-negative, as indicated in above rule.

	j	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*
i		–	T	T	T	A	C	A	G	G	C	A	G
0	–	0	0	0	0	0	0	0	0	0	0	0	0
1	G	0	0	0	0	0	0	0	**2**	**2**	0	0	**2**
2	A	0	0	0	0	**2**	0	**2**	0	1	1	**2**	0
3	A	0	0	0	0	**2**	1	**2**	1	0	0	**3**	1
4	C	0	0	0	0	0	**4**	2	1	0	2	1	2
5	G	0	0	0	0	0	2	3	**4**	**3**	2	1	**3**
6	G	0	0	0	0	0	0	1	**5**	**6**	4	2	**3**
7	T	0	**2**	**2**	**2**	0	0	0	3	4	5	3	1

Step 4: The local alignment ends at cell(6,8) (shaded cell), which contains the maximum alignment score = 6. Read out the alignment, starting at cell(6,8) and working backward along the directions of entry into each cell until an element containing zero is encountered.

The resulting ***Local Alignment*** result is enclosed in the box below.

A:			G	A	A	C	–	G	G	T		
B:	T	T	T	A	C	A	G	G	C	A	G	

4. Multiple Sequence Alignment (MSA)

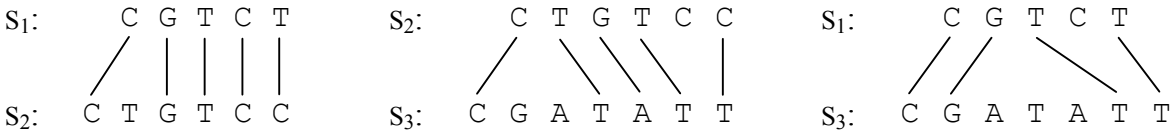
- Consistency-based method

Multiple sequence alignment (MSA), in which three or more sequences must be aligned, is useful in finding conserved regulatory patterns in nucleotide sequences and for identifying structural and functional domains in protein families. Unfortunately, MSA is much more challenging than single pairwise alignment (PSA) shown as before. The papers [SMS03] & [M99] are using *Consistency-based* method to solve the MSA problems. The concept of consistency among a set of pairwise alignments was introduced by Gotoh [G90]. Here is an example of local MSAs plausibly embedded in an optimal global MSA.

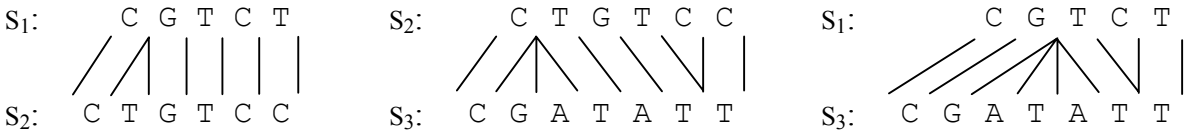
Step 1: Three PSAs between sequences S_1, S_2, S_3 are $P_{12}(S_1, S_2)$, $P_{23}(S_2, S_3)$, and $P_{13}(S_1, S_3)$.

S_1: C – G T C T	S_2 C – – T G T C C	S_1 C G – – – T C T
S_2: C T G T C C	S_3: C G A T A T – T	S_3: C G A T A T – T

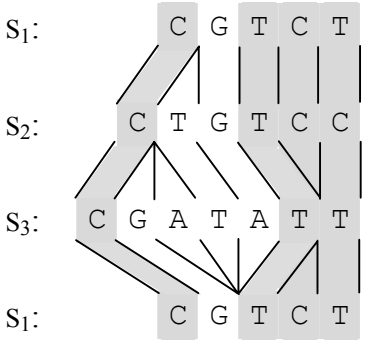
Step 2: Trace formulation --- Each PSA is represented by an undirected bipartite graph where a vertex corresponds to a residue and an edge, corresponds to matched pair.



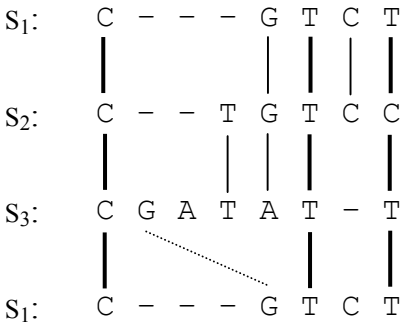
Step 3: Latter formulation --- Take care of gapped alignments that each vertex corresponds to a residue. Every vertex belongs to one or more edges.



Step 4: Formalization of consistently aligned regions
--- Combined S_1, S_2, and S_3 from *Step 3*.



Step 5: The results of MSA



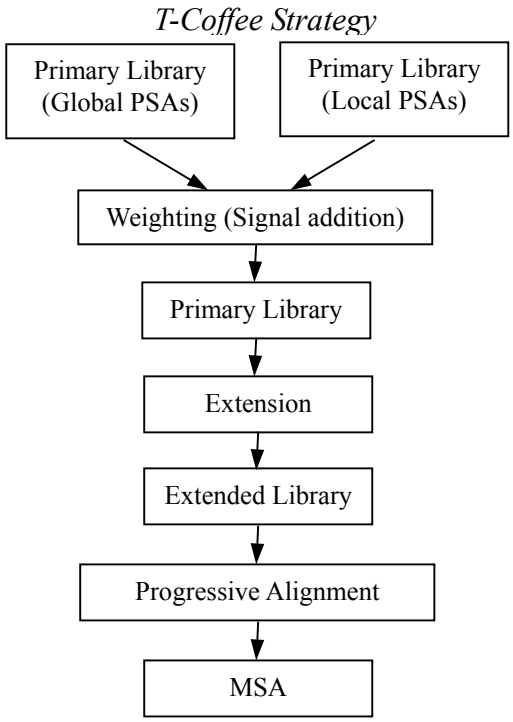
In the MSA, edges realized by solid bars (consistent in thick bars), and unrealised by dotted bars.

- Maximum Weight Trace

In the formulation processes, they need to find the maximum weight of MSA that is closest to all PSAs in the set. The input to the maximum weight trace problem is set of edges, that are obtained from a set of pairwise optimal sequence alignments in a special case. The algorithm finds an MSA in which the maximum number of input edges is realized, or in generally, maximum sum of weights associated with the edges. Kececioglu [K93] proposed a branch-and-bound algorithm somewhat similar to the MSA problem described above.

- T-Coffee Algorithm

T-Coffee algorithm [NHH00] is based on a more practical use of consistency information derived from a set of PSAs. All PSAs obtained with different global and local PSA methods for each pair of sequences. The primary library consists of residue pairs with associated weights. When the same residue pair appears in the set of PSAs more than once, the associated weights are summed up to yield the primary weight. The primary library is nearly equivalent to the input edges used in the special case of the special case of maximum weight trace. The edges in a primary library occupy only a sparse subset of all the edges to be examined by a complete MSA procedure. T-Coffee extends the library by adding residue pairs that are indirectly matched. Using total weight (given to a residue pair) in place of an ordinary score matrix in the DP-based pairwise alignment of single or pre-aligned groups of sequences without imposing any gap penalties. Otherwise, T-Coffee adopts the typical progressive alignment strategy. The distance matrix and the guide tree are constructed in the same manner as those of ClustalW [THG94]. The major advantage of T-Coffee over ClustalW and other ordinary progressive methods is that information about alignments between all pairs of sequences is



condensed in the weights, which are utilized even at the very beginning of the progressive. Another advantage of T-Coffee is that several different sources of alignment information, which obtained from global and local PSAs, are mixed to compute a residue-pair weight.

- DIALIGN Algorithm

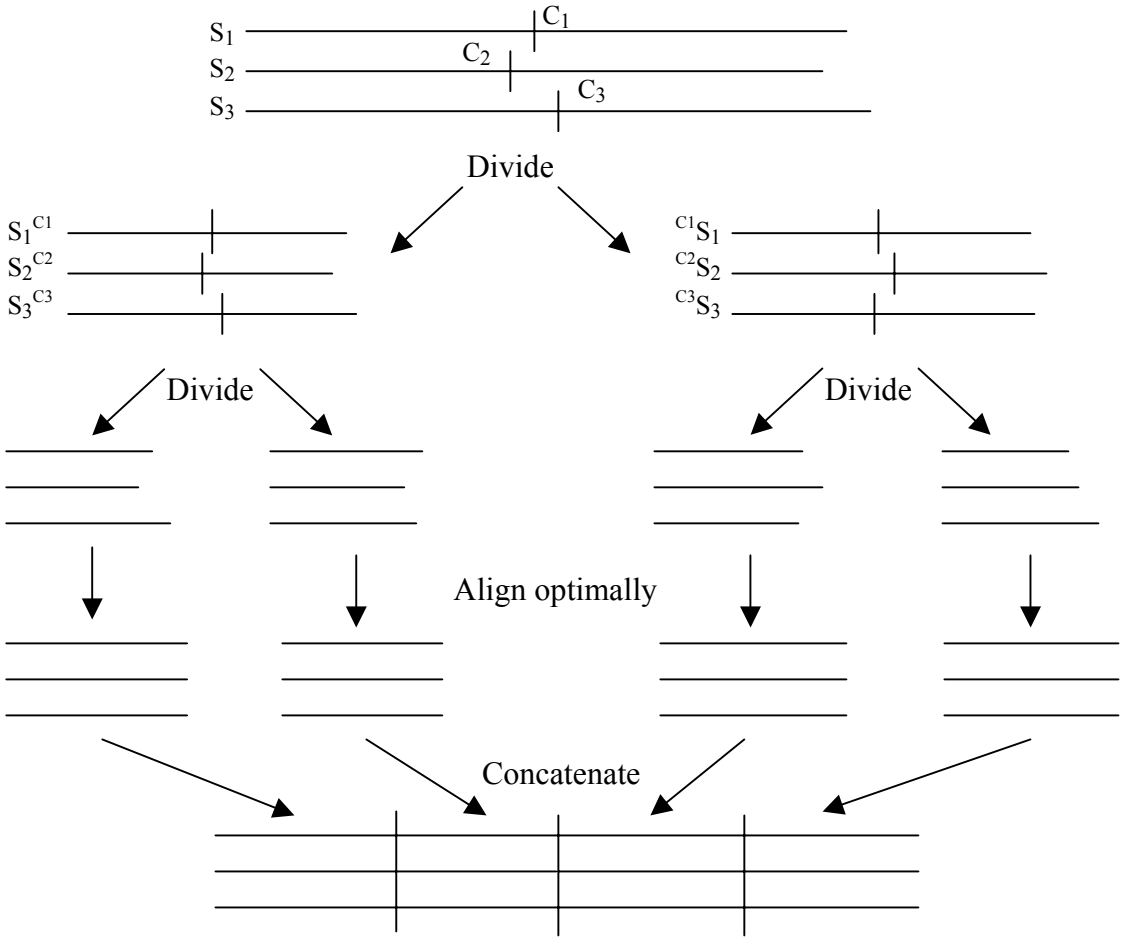
The idea of the DIALIGN algorithm is somewhat related to that of the maximum weight trace problem, although units used in an alignment process are fragments with positive weights (significant fragments) rather than individual matched pairs. The objective function is the sum of weights of the fragments that are involved in the final alignment, where no penalty is imposed on the gaps. The residues that are not included in these fragments remain unaligned. Hence, DIALIGN tends to produce global alignments with a decrease in similarity of sequences under comparison. DIALIGN are uses most standard procedures which is DP algorithm, and adopts a greedy strategy for MSA.

The most crucial step of the DIALIGN algorithm is the evaluation of the weight for a fragment. All combinations of input sequences are aligned to yield the initial list of significant fragments. The weight for each fragment is recalculated in a similar fashion to that used in the construction of the extended library in the T-Coffee program, except that the supplementary weights are derived from indirectly aligned segment pairs. The fragments in the initial list are examined in the order of their values, and moved to the second list as long as they are compatible with all the fragments already present in the second list. After all the fragments are examined, the process is repeated again from the first member remaining in the initial list until no member in the initial list is compatible with those in the second list. Hence, DIALIGN is most effective for detecting local alignment among distantly related sequences or sequences composed of several domains. The local nature is favorable for searching exons or regulatory elements in genomics sequences. On the other hand, DIALIGN is too expensive to align many (more than 100) sequences, because the theoretical computational complexity, which is spent for recalculating weight values.

5. Divide-and-conquer & Segment-based MSA

- Divide-and-conquer multiple sequence alignment (DCA)

Given a family $S = (S_1, \ldots, S_k)$ of sequences, and cut each sequence at a suitable position near to its midpoint, such as sequence S_1 at position C_1, S_2 at position C_2, and so on. Therefore, two new families of shorter sequences are shown as below, and take advantage of this fact by first dividing at S_1 its midpoint $C_1 = \lceil |S_1|/2 \rceil$, then searching for compatible positions in the other sequence.



The optimal concatenating alignment of the resulting prefix and suffix sequences (slicing position C_1, C_2 of sequence S_1, S_2) by additional cost $C_{S1,S2}[C_1,C_2] = W$ opt (prefix) + W opt (suffix) – W opt (total), where W opt denotes an appropriate measure of fit of the three optimal alignments in following example. The sum of pairwise additional costs $C(C_1,C_2,C_3) = \alpha_{12}C_{S1,S2}[C_1,C_2] + \alpha_{23}C_{S2,S3}[C_2,C_3] + \alpha_{13}C_{S1,S3}[C_1,C_3]$, where α_{12}, α_{12}, α_{12} are appropriately chosen weight factors reflecting, for example, phylogenetic relationships.

e.g. $C_{S1,S2}[2,2] = W\,\text{opt}\,[\text{CT},\text{GT}] + W\,\text{opt}\,[\text{ATAC},\text{ATC}] - W\,\text{opt}\,[\text{CTATAC},\text{GTATC}] = 1 + 2 - 3 = 0$

$C_{S1,S2}[4,3] = W\,\text{opt}\,[\text{CTAT},\text{GTA}] + W\,\text{opt}\,[\text{AC},\text{TC}] - W\,\text{opt}\,[\text{CTATAC},\text{GTATC}] = 3 + 1 - 3 = 1$

DP distance matrices (+2 for unit cost and penalty)

		C	T	A	T	A	C
	0	2	4	6	8	10	12
G	2	1	3	5	7	9	11
T	4	3	1	3	5	7	9
A	6	5	3	1	3	5	7
T	8	7	5	3	1	3	5
C	10	8	7	5	3	2	3

	C	T	A	T	A	C
G	3	4	3	4	6	8	10
T	4	2	3	2	4	6	8
A	6	4	2	2	2	4	6
T	8	6	4	2	1	2	4
C	10	8	6	4	2	0	2
	12	10	8	6	4	2	0

Additional-cost matrix

	C	T	A	T	A	C
	0	3	4	7	11	15	19
G	3	0	3	4	8	12	16
T	7	4	0	2	4	8	12
A	11	8	4	0	1	4	8
T	15	12	8	4	0	0	4
C	19	15	12	8	4	1	0

The optimal alignment is marked in gray color or circled, its additional-cost are equal to zero.

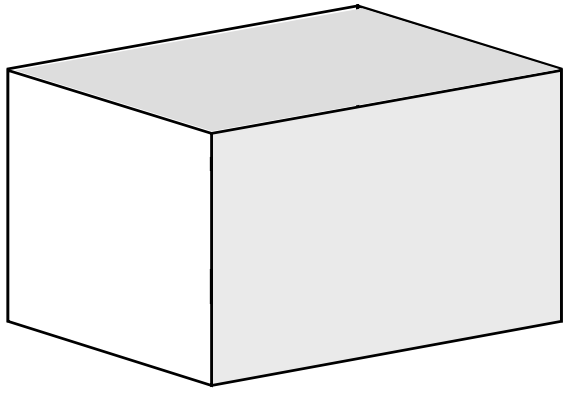
The DCA in threshold length L as following:

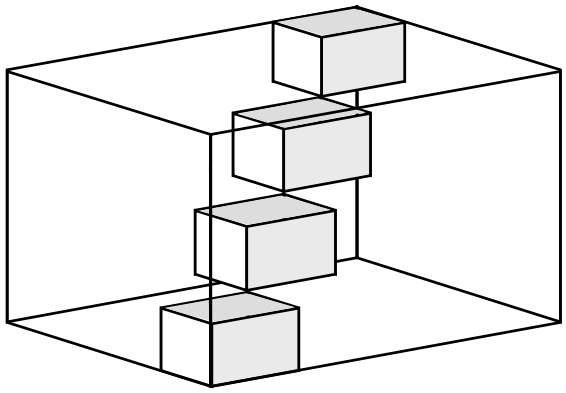
$$\text{DCA}([S_1, S_2, S_3], L) = \text{MSA}([S_1, S_2, S_3]), \text{ if } \min\{|S_1|, |S_2|, |S_3|\} \le L >>$$
$$= \text{DCA}([S_1(\le C_1), S_2(\le C_1), S_3(\le C_1)], L) ++ \text{DCA}([S_1(>C_1), S_2(>C_1), S_3(>C_1)], L)$$

where $C_1 = \lceil |S_1|/2 \rceil$, $(C_2, C_3) \in \{0, \ldots, |S_2|\} \times \{0, \ldots, |S_3|\}$, such that $C(C_1, C_2, C_3)$ is minimal.

- Reducing *search space* and *computation time*

The following figures are shown space size of alignment problems in dynamic programming. The size of three sequences alignment problem is proportional to the volume of the box. After divided sequences in several small sub-sequences, the search space boxes will chain along the main diagonal of the large box, and the search space is the sum of the small boxes.





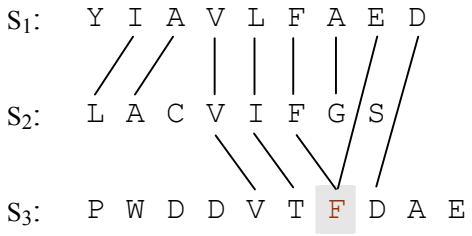
Obviously, the computing time of the optimal alignments can be delimited by choosing L appropriately small. Although for three sequences is remarkable improvement compared to the cubic space and time complexity of standard DP, computing time still grows exponentially with the

number of sequences. Therefore, find the suitable cut positions for the resulting total alignment is closed to an optimal alignment. Any cut of position C_1 of sequence S_1, there always exist in position $(C_2,\ldots,C_k)$, then the good result of cut position in the weighted sum over all pairwise additional-cost matrix entries by minimizing the value of $C(C_1; C_2, \ldots, C_k) = \sum_{1\le p<q\le k} \alpha_{pq} C_{Sp,Sq}(C_p,C_q)$.

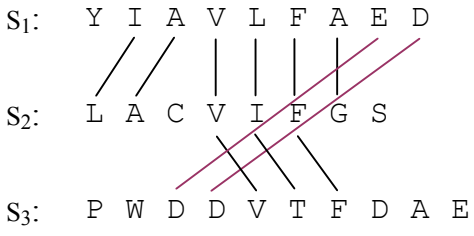
- Segment-based method

The idea of the DIALIGN algorithm is based on segment-to-segment comparison instead of the commonly used residue-to-residue comparison. And the algorithms are composed from gap-free pairs of segments of equal length. Such segment pairs are referred to as diagonals. A pairwise as well as a multiple alignment is considered to be representing by a collection of each diagonals meeting a certain consistency criterion.

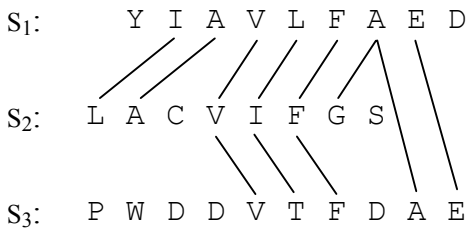
Non-Consistent ('F' is assigned simultaneously)



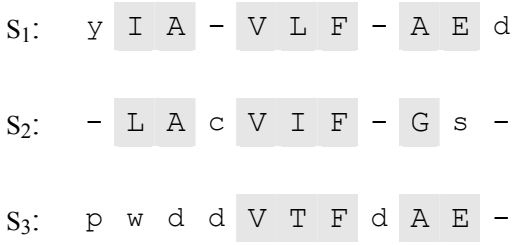
Non-Consistent ('Cross over residues)



Consistent collection of diagonals



Consistent with gaps (Residues in lower-case letters)



From the above segment comparisons are successfully employed by alignment strategies based on comparing gap-free segments of the sequences rather than on comparing single residues. A collection of diagonals is called consistent if there exists an alignment such that all segment pairs are matched. Diagonals may overlap if different pairs of sequences are involved. However, diagonals involving the same pair of sequences are not allowed to have any overlap.

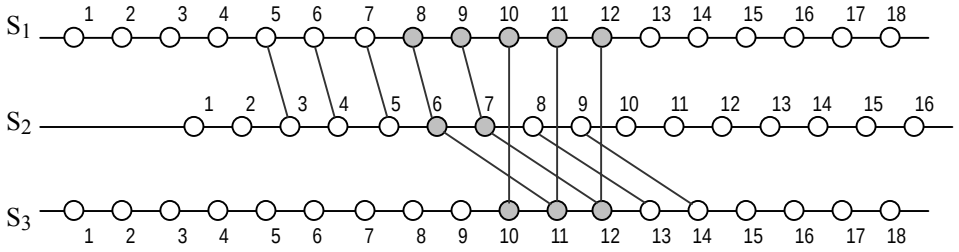
- Transitivity Frontiers and consistency bounds

An alignment A of a sequence family $S = (s_1, \ldots, s_k)$ imposes for $x \in X$ where position $x = [p,i]$ and every sequence $s_i \in S$ an upper bound $Pred_F(x,p)$ and lower bound $Succ_F(x,p)$, such that a site $y = [p,i] \in s_i$ is alignable with x without leading to inconsistencies with A if and only if $Pred_F(x,p) \geq p \geq Succ_F(x,p)$ holds. $y \leftarrow_F x$ means that y is to the left-hand side or in the same column as x in every alignment A_F that realizes set of fragments F. Formally, defined as

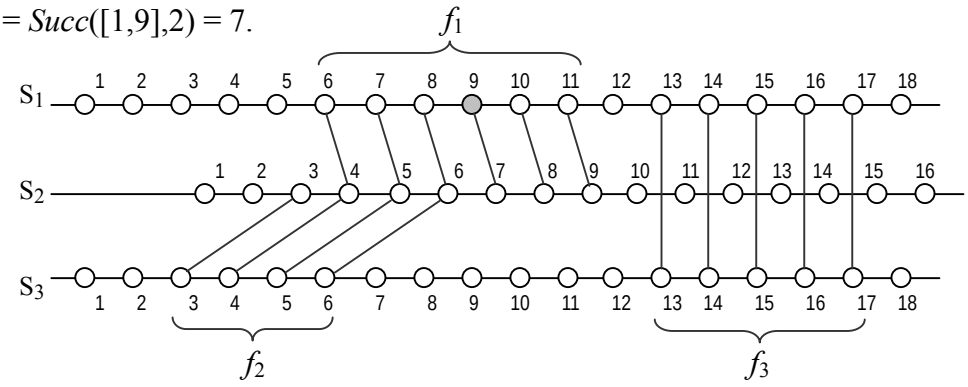
$$Pred_F(x,p) = \max\{ j : [p,j] \leftarrow_F x \} \qquad \text{and} \qquad Succ_F(x,p) = \min\{ j : x \leftarrow_F [p,j] \}$$

Example: $f_{12} = ([1,5],[2,3],5)$, $f_{23} = ([2,6],[3,11],4)$, and $f_{31} = ([1,10],[3,10],3)$.

Contradicting alignment of the positions marked in grey in a global multiple alignment.



For Transitivity Frontiers of position [1,9], $f_1 = ([1,6],[2,4],6)$, $f_2 = ([2,3],[3,3],4)$, $f_3 = ([1,13],[3,13],5)$.
For s_3 $Pred([1,9],3) = 6$ and $Succ([1,9],3) = 13$.
For s_2 $Pred([1,9],2) = Succ([1,9],2) = 7$.



- Weight functions for diagonals

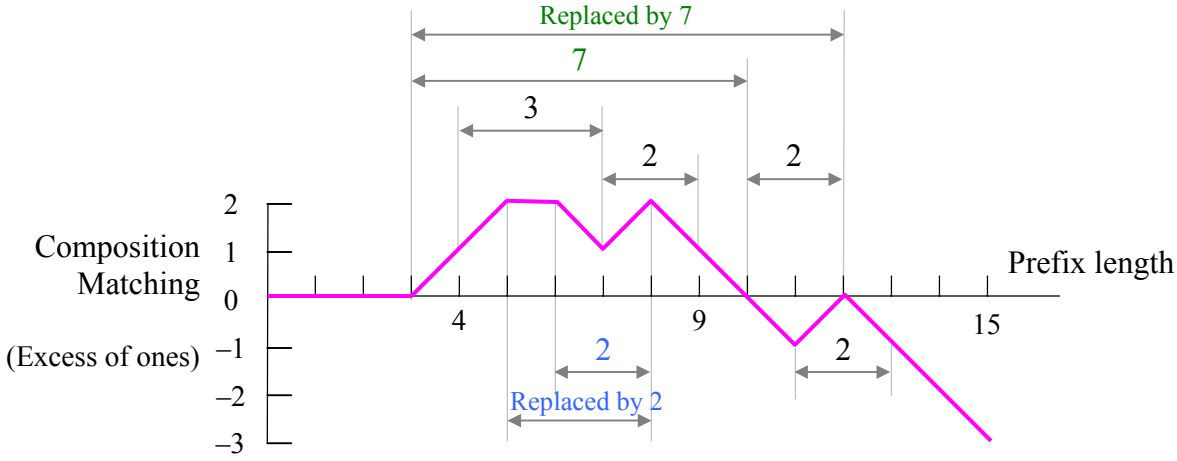
DIALIGN assigns a weight to every possible diagonal called *weight function*, which the score of an alignment is defined to be the sum of the weights of the incorporated diagonals. Given a diagonal D of length l_D, denoted by s_D the sum of the individual similarity values of residue pairs with this diagonal. $P(l_D, s_D)$ denoted the probability that a random diagonal of the same length l_D has at least the same sum s_D of similarity values. The weight $w(D)$ of the diagonal D is defined to be $w(D) = -\log P(l_D, s_D)$ provided this value exceeds a certain user-defined threshold.

6. Composition Alignment

- Composition matching

Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Sequence S_1:	0	1	0	1	1	1	0	1	0	0	0	1	0	0	0
Sequence S_2:	0	1	0	0	0	1	1	0	1	1	1	0	1	1	1
Mismatching				+	+		–	+	–	–	–	+	–	–	–
Prefix length	0	0	0	1	2	2	1	2	1	0	-1	0	-1	-2	-3

The graph of shortest composition match length



Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Longest CM length	1	1	1	–	–	1	3	3	2	7	–	9	2	–	–
Shortest CM length	1	1	1	–	–	1	3	2	2	7	–	2	2	–	–

- Composition Difference of S_1 and S_2

S_1:	0	1	0	1	1	1	0	1	0	0	0	1	0	0	0
S_2:	0	1	0	0	0	1	1	0	1	1	1	0	1	1	1
Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CD:	(0,0)	(0,0)	(0,0)	(–1,1)			(–1,1)		(–1,1)	(0,0)	(1,–1)	(0,0)	(1,–1)		

- Composition Matching of S_1 and S_2

For CD = (0,0), sub-string pairs of CM are (S_1[4,10], S_2[4,10]) and (S_1[11,12], S_2[11,12]).
For CD = (–1,1), CM are (S_1[5,7], S_2[5,7]) and (S_1[8,9], S_2[8,9]), or (S_1[5,9], S_2[5,9]).
For CD = (1,–1), CM is (S_1[12,13], S_2[12,13]).

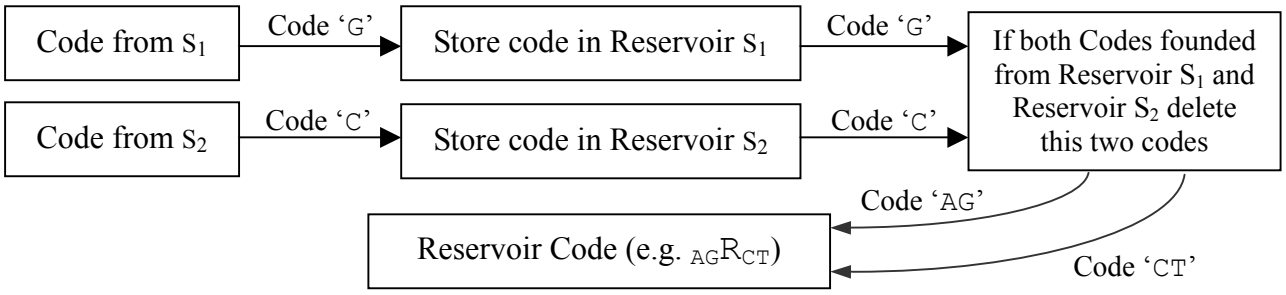
Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Sequence S_1:	0	1	0	1	1	1	0	1	0	0	0	1	0	0	0
	|	|	|	←								→			
Sequence S_2:	0	1	0	0	0	1	1	0	1	1	1	0	1	1	1

The better CM is CD = (0,0) because it is the longest length of composition matching.

7. Meta-code Composition Alignment

- Reservoir Codes

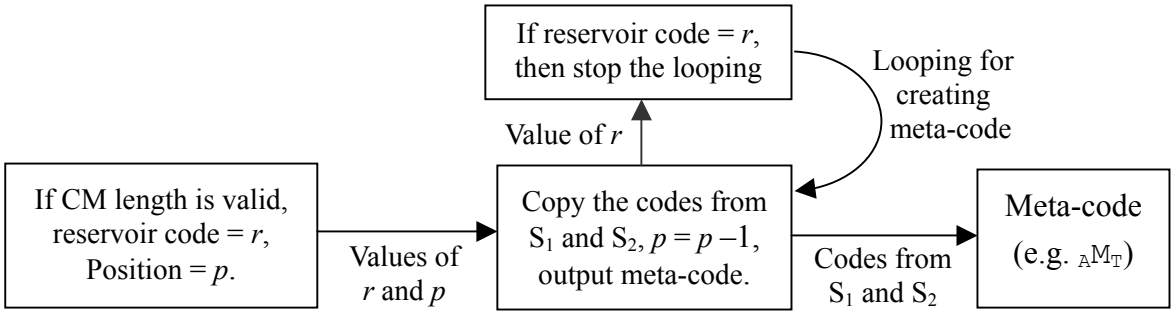
There are two reservoirs to store the input code from S_1 and S_2 for creating reservoir codes in position *p*. Reservoir Codes ($_{AG}R_{CT}$) means that $_{AG}$ is in reservoir S_1 which contains '`A`' & '`G`' code, and $_{CT}$ is in reservoir S_2 which contains '`C`' & '`T`' code. $R_{\varnothing}$ means reservoirs are empty, and length of CM is the numbers of codes in the reservoirs.



The purpose of reservoir code is used for checking the composition matching of prefix length is valid or not. Reservoir codes should be equal in composition matching. The ordering of codes will not be considered inside its sequence, such as $_{AG}R_{CT} = {}_{GA}R_{TC}$. The block diagram example is operated at *position #7*, other details see following working example in next page.

- Meta-code

Meta-code defined as Code about code. Meta-code ($_AM_T$) means that the operational code '`T`' instead of the original code '`A`'. The iterative method of meta-code generation is shown below.



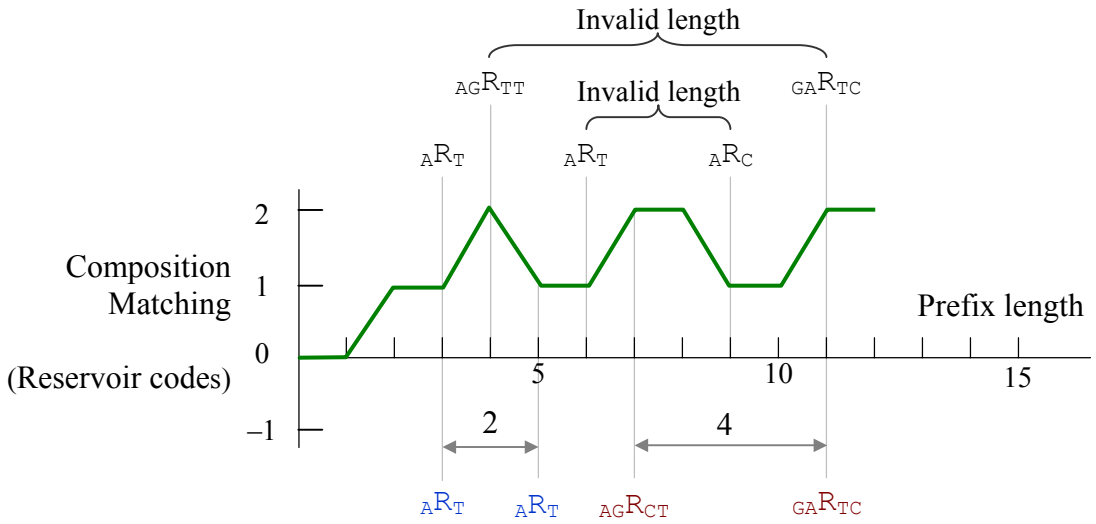
The iteration loop of producing meta-codes in sequence S = (S_1, … , S_k), excluded S_1. This is the prior sequence of family classification. But it could be changed in another generation of family classification with other set of sequences.

Example: Find the composition alignment of S_1 and S_2.

Sequence S_1:	T	A	C	G	T	C	G	T	C	G	A	C
Sequence S_2:	T	T	C	T	G	C	C	C	G	A	T	C

Composition Matching of S_1 and S_2 in prefix length

S_1:	T	A	C	G	T	C	G	T	C	G	A	C
S_2:	T	T	C	T	G	C	C	C	G	A	T	C
Position	1	2	3	4	5	6	7	8	9	10	11	12
CM in length	0	1	1	2	1	1	2	2	1	1	2	2
Reservoir Codes	$R_\emptyset$	${}_AR_T$	${}_AR_T$	${}_{AG}R_{TT}$	${}_AR_T$	${}_AR_T$	${}_{AG}R_{CT}$	${}_{AG}R_{CC}$	${}_AR_C$	${}_GR_C$	${}_{GA}R_{TC}$	${}_{GA}R_{TC}$



For CM length = 1, ${}_AR_T$ (at position 3) = ${}_AR_T$ (at position 5), CM is (S_1[4,5], S_2[4,5]).

For CM length = 2, ${}_{AG}R_{CT}$ (at position 7) = ${}_{GA}R_{TC}$ (at position 11), CM is (S_1[8,11], S_2[8,11]).

In CM length = 1, (S_1[7,9], S_2[7,9]) does not be a composition matching because of ${}_AR_T \neq {}_AR_C$.

Composition matching of sequence S_1 and S_2

Sequence S_1:	T	A	C	G	T	C	G	T	C	G	A	C
	|		|	⟷		|		⟷			|
Sequence S_2:	T	T	C	T	G	C	C	C	G	A	T	C

- Composition MSA

Sequence S_1:	T	A	C	G	T	C	G	T	C	G	A	C
Sequence S_2:	T	T	C	T	G	C	C	C	G	A	T	C
Meta-code transform	T	T	C	${}_TM_G$	${}_GM_T$	C	C	${}_CM_T$	${}_GM_C$	${}_AM_G$	${}_TM_A$	C
New sequence S_2:	T	T	C	G	T	C	C	T	C	G	A	C

According to S_1 and new S_2, we could put the meta-code sequence in the PSA library. The other processes would be implemented by new sequence, in order to get MSA. But the resulting of MSA should be reverse back to the original sequence S_2.

8. Further Applications (Family Classification)

- Fixed-segment composition alignment

In some case, we could apply fixed length L_S of a segment for composition simultaneous MSA. Here is an example of bank's family classification (L_S = 5, weekly behaviour). L_S can be defined by data miner. This depends on what the mining results is desired. Why composition alignment is used? Normally, the financial behaviour would be repeated in a week and the sequence ordering in a week is not main issue for planning a branch bank.

Code catalogue

A = Currency / Cards
B = Stock / Structured P.
C = Unit Trusts / Bonds
D = Insurance / Finance
E = Mortgages / Loans

Notations:

'C' in Bank #1 sequence means that customers buy/sell UTs (highest amount) in that day.

$^{1}t_3$ where ^{1} means Week #1, t_3 means Day #3 (Wednesday).

HongKong Banks' Service Timelines

Time Granularities	$^{1}t_1$	$^{1}t_2$	$^{1}t_3$	$^{1}t_4$	$^{1}t_5$	$^{2}t_1$	$^{2}t_2$	$^{2}t_3$	$^{2}t_4$	$^{2}t_5$	$^{3}t_1$	$^{3}t_2$	…
Branch bank #1	A	C	B	C	E	B	A	A	E	B	C	A	…
Branch bank #2	B	E	B	A	A	A	C	E	D	B	E	A	…
Branch bank #3	A	C	A	A	B	C	E	E	D	B	E	E	…
⋮													

Week #1 Week #2

The global alignment of sequences is same as the local alignment, because all the codes are grouped into fixed size. Therefore, simple PSA could be used and computing problems easier to solve as well. If the resulting of MSA is obtained as following:

Branch bank #1	C	C	B	C	A	D
Branch bank #2	A	A	B	A	D	C
Branch bank #3	A	B	A	A	B	B

Notes: ☐ Composition Alignment

In composition alignment the results is frequent counting.

From above results, three Banks are in the same class. They have same financial behavior of customers. But Bank #2 and Bank #3 are very close to the same familiar class, and they presented that many customers needs currency, cards, stocks, and structured products services in a week, mortgages or loans are neglected.

9. Conclusions

DIALIGN's "segment-to-segment" score extends to multiple dimensions by using the same *p*-value score as in the two-sequence case on ***ungapped*** pairwise matches, adding weight to a pairwise match if it is also matched in other sequences. The method relies on there being enough significant ungapped matches to string together a correct alignment, which will be true only when the sequences are closely related. DCA and DIALIGN algorithms are combined with "segment-based constraints" strategy. The MSA of sequences sufficiently diverged that limited regions remain ungapped, making it impossible for a segmentation strategy alignment using ungapped segments and scoring function to find the every types of sequence alignment, such as deterministic sequences.

Composition alignment could be applied to alignment of promoters in bioinformatics. Although the algorithm considers gapped and ungapped pairwise of equal length local alignment, the overlap matching would be happened in MSA, and it would create a lot of matching problems as well, especially in gapped pairwise and iterative procedure in MSA.

The meta-code approach is new introduced for composition alignment, which can be applied simultaneous temporal mining for classification in other fields, not only in bioinformatics. This method should be implement in ***fixed-segment*** composition simultaneous MSA, in order to eliminate problems during MSA, such as overlap matching. It is because the algorithm difficult to overcome overlapping matching in global sequence alignment. A further research direction is to generalize the input simultaneous sequence in the meta-code Gap-free MSA.

References

[B57] R.E. Bellman: *Dynamic Programming.* Princeton University Press, Princeton, NJ, 1957.

[G90] O. Gotoh: *Consistency of optimal sequence alignments.* Bulletin of Mathematical Biology, 162:705-708, 1990.

[K93] J.D. Kececioglu: *The maximum weight trace problem in multiple sequence alignment.* Lecture Notes Computer Science, Vol.684:106-119, 1993.

[NHH00] C. Notredame, D. Higgins, J. Heringa: *T-Coffee: A novel method for multiple sequence alignments*. Journal of Molecular Biology, Vol.302, pp205-217, 2000.

[NW70] S.B. Needleman and C.D. Wunsch: *A general method applicable to the search for similarities in the amino acid sequence of two proteins.* Journal of Molecular Biology, 48:443-453, 1970.

[SW81] T.F. Smith and M.S. Waterman: *The identification of common molecular subsequences.* Journal of Molecular Biology, 147:195-197, 1981.

[THG94] J.D. Thompson, D.G. Higgins, and T.J. Gilson: *CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice.* Nucleic Acids Res., 22:4673-4680, 1994.

[B03]
[M99]
[S98]
[SMS03]
} Details in page one