Diagonal Consistency Problem Resolution in DIALIGN Algorithm

Ibrahim Chegrane, Athmane Sighier, Chahrazed Ighilaza and Aicha Boutorh
USTHB, Laboratory of Artificial Intelligence (LRIA), Algiers, Algeria

Keywords: Bioinformatics, Multiple Alignment, DIALIGN, Diagonals Consistency.

Abstract: DIALIGN is a well known Algorithm for pairwise as well as multiple alignment of nucleic acid and protein sequences. It combines local and global alignment features. In this paper we present a new method to better solve the problem of diagonals consistency in DIALIGN algorithm using graph theory modeling. and we describe a new implementation of the method from the extraction of diagonals to the final alignment process. We show the power of our proposed aproach by comparing it with DIALIGN 2.2 using benchmarks from "BAliBASE" and "SMART" databases.

1 INTRODUCTION

The alignment problem has been widely considered as being solved for pairwise alignments (Needleman and Wunsch, 1970; Smith and Waterman, 1981). Most efforts focused on improving the algorithm to find optimal or reasonably good suboptimal multiple alignments. There are many methods and algorithms that provide solutions to this problem. (Thompson et al., 1994; Morgenstren et al., 1996; Notredame et al., 2000; Edgar, 2004; Derrien et al., 2005).

Among the alignment methods, we mention DIALIGN method (Morgenstren et al., 1996; Morgenstern et al., 1998a). It combines local and global alignment features. The alignments are composed by aligning local pairwise similarities. DIALIGN alignment comprises a collection of diagonals meeting a certain consistency criterion. It try to select a consistent set of diagonals with a maximal sum of weights. The main difference between DIALIGN and other traditional alignment approaches is the underlying scoring scheme or the objective function. Gaps are not penalized.

In this paper we propose a new method in order to solve the problem of diagonals consistency in DIALIGN algorithm. Our improvement comes into the phase of the construction of all consistent diagonals, and then the way they are going to be aligned. We describe a re-implementation of the DIALIGN algorithm from the extraction of diagonals to the final alignment process.

The paper is organized as follow: section 2 covers the concept of diagonals consistency in DIALIGN algorithm. In section 3 we present related work, in section 4 we explain the method to solve the problem of diagonals inconsistency and we give the details of our proposed method of alignment .In section 5 we present the final steps in alignment using our new approach. Section 6 covers the analysis and comparison, and finally we give the conclusion in section 7.

2 DIAGONAL CONSISTENCY IN DIALIGN

One crucial concept described in DIALIGN is the diagonals consistency. How to decide whether or not a diagonal is consistent with the diagonals already incorporated into the alignment. A collection of diagonals is called consistent if there is no double conflicting or crossover assignment of residues (Morgenstren et al., 1996). Also diagonals involving the same pair of sequences are not allowed to have any overlap. see figure 1.

A diagonal which crosses or overlaps with the diagonal that has the greatest weight is an inconsistent diagonal, because it does not enable a correct alignment for this diagonal of the greatest weight.

Each time we add a diagonal to the alignment we check if it is consistent with the diagonals already incorporated. A collection of diagonals is called consistent if there is no conflicting double or crossover assignment of residues (Morgenstren et al., 1996). Diagonals that are not consistent with the growing set of consistent diagonals will be rejected.

It can be shown that the problem of finding an op-

Chegrane I., Sighier A., Ighilaza C. and Boutorh A..
Diagonal Consistency Problem Resolution in DIALIGN Algorithm.
DOI: 10.5220/0005284802250231
In *Proceedings of the International Conference on Bioinformatics Models, Methods and Algorithms* (BIOINFORMATICS-2015), pages 225-231
ISBN: 978-989-758-070-3

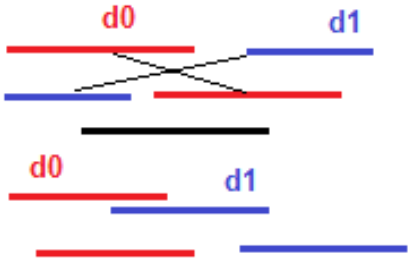

Figure 1: inconstancy diagonal, d_1 crossover/overlap withe d_0.

timal consistent set A of fragments is NP-complete (Subramanian et al., 2008).

3 RELATED WORK

Morgestern et al. proposed a new alignment method with a new scheme score relaying on segment comparison (Morgenstren et al., 1996; Morgenstern et al., 1998b). The alignments are composed of gap-free pairs of sequence segments of equal length. Segment pairs are also called diagonals because they appear as diagonals in comparison matrix. The implementation of this method is called DIALIGN 1.

Abdeddaiim (Abdeddam, 1997), proposed a method similar to (Morgenstren et al., 1996) that enters the phase of incorporating diagonal in greedy manner.

In the DIALIGN 1 there is a general problem with the weight function, it needs to specify certain minimum length of diagonals and a positive threshold. Morgenstern et al. introduced a new weight function for diagonals to remove the tow parameters in DIALIGN 1 (Morgenstern et al., 1998a). A simple modification of the weight function used by the original version of the DIALIGN alignment program turns out to both, global and local alignment problems without the need to specify a threshold parameter. The new implementation is called DIALIGN 2 (Morgenstern et al., 1998a; Morgenstern, 1999).

In (Kim and Lee, 2005) they provide an improved procedure of calculation of the probabilities for DIALIGN2, which is more accurately and eliminates the need to compute them by random experiments.

In (Subramanian et al., 2005) They noted that in the original DIALIGN approach, an inconsistent fragment f is completely discarded in the greedy procedure, even if just a few residue pairs are inconsistent with the current alignment. In that case they remove only those inconsistent residue pairs from f and give the remaining sub-fragments a second chance in the greedy selection process, and they proposed several heuristics to improve the segment-based alignment approach. The implementation of this improvement is called DIALIGN-T.

In paper (Subramanian et al., 2008), they present DIALIGN-TX, a substantial improvement of DIALIGN-T (Subramanian et al., 2005) that combines their previous greedy algorithm with a progressive alignment approach.

4 NEW METHOD FOR DIAGONALS CONSISTENCY

The diagonals consistency in the DIALIGN algorithm is a crucial concept and difficult problem. Unlike previous approaches, which show that a diagonal is consistent, our approach does the opposite, it proves that a new diagonal is inconsistent with the the diagonals already tested and accepted as consistent.

Let S be the set of all biological sequences, and $S = \{s_1, s_2, ..., s_l\}$ where l is the total number of all sequences, $s_k \in S$ where $k \in [1..l]$. Let D be the set of all diagonals, $D = \{d_1, d_2, ..., d_n\}$ where n is the total number of all initial diagonals, $d_j \in D^{'}$ where $j = [1..m]$ and m is the total number of consistent diagonals only. Each diagonal has a pair of fragment (subsequence), the first segment is designated by x, and the other by y, so $d_i = (x_i, y_i)$ where $i \in [1..n]$.

We use graph theory modeling in order to represent diagonals position and solve the problem of consistency.

In our work we consider the beginning of sequences in diagonal x_i or y_i as the vertices of the graph and the edges correspond to the relationship of the beginning of a subsequence of diagonal $d_i = (x_i, y_i)$ with the beginning of another subsequence $(x_j or y_j)$ of diagonal d_j located in the same sequence, where the starting position of the subsequence d_i is before that of d_j, also the relationship of the beginning of the subsequence x_i in a diagonal d_i to the other subsequence y_i in the same diagonal (or vice versa).

The graphe $G = (V, E)$ is defined as follows:

$V = \{x_{11}, x_{21}, ..., x_{n1}\} \cup \{y_{11}, y_{21}, ..., y_{n1}\}$ where x_{i1} and y_{i1} :the beginning of the 1st and 2nd subsequence of the diagonal d_i , and $i \in [0..n]$

$E = E_1 \cup E_2 \cup E_3 \cup E_4 \cup E_5$
$E_1 = \{(x_i, y_i), (y_i, x_i) \mid i \in [0..n]\}$
$E_2 = \{(x_i, y_j) \mid i, j \in [0..n]$ and $\exists k$ where $x_i, y_j \in s_k$ and y_j the first subsequence that appears after x_i and s_k the biological sequence number k $\}$

$E_3 = \{(x_i, x_i) \mid i, j \in [0..n]$ and $\exists k \mid x_i, x_j \in s_k$ and x_j the first subsequence that appears after x_i $\}$

$E_4 = \{(y_i, y_i) \mid i, j \in [0..n]$ and $\exists k; \mid y_i, y_j \in s_k$ and y_j the first subsequence that appears after y_i $\}$

$E_5 = \{(y_i, x_j) \mid i, j \in [0..n]$ and $\exists k \mid y_i, x_j \in s_k$ et x_j the first subsequence that appears after y_i $\}$

Example: we have 3 diagonals (6 subsequences)

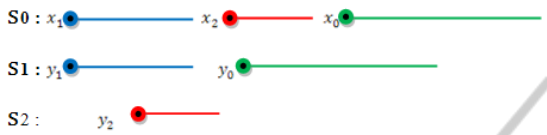

Figure 2: 3 diagonals and their fragments.

So the graph will be G (6 ,9) with 6 vertices and 9 edges, see figure 3:

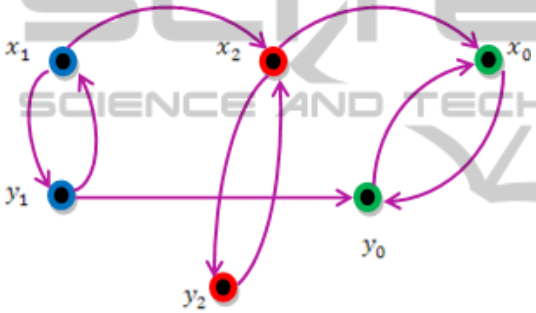

Figure 3: G (6 ,9) consistency graph.

4.1 Diagonal Inconsistency with a Simple Path

A diagonal d_i is called inconsistent with all the diagonals in D', if and only if there is a simple path [1] which connects x_{i1} and y_{i1} passing through the subsequences of the consistent diagonals in D'. $(x_{i1} \longrightarrow y_{i1}$ or $y_{i1} \longleftarrow x_{i1})$. in an other word, to prove that d_i is inconsistent diagonal withe D' we have to find a simple λ path between x_{i1} and y_{i1} where $\lambda = x_0 u_1 x_1 ... x_{k-1} u_k x_k$ where $(x_0 = x_{i1}$ and $x_k = y_{i1})$ or $(x_0 = y_{i1}$ and $x_k = x_{i1})$.

Let us illustrate with an example, we consider the diagonal d_red with the red color in the figure 4, we check the inconstancy of this diagonal by finding a simple path that pass through subsequences of consistence diagonals in D'.

[1]A path in a directed graph $G = (V, E)$. is an alternating sequence of vertices and edges :$\lambda = x_0 e_1 x_1 ... x_{k-1} e_k x_k$ where $i \in [1..k]$, and the vertex x_i is the initial end of the edge u_k and the vertex x_{i+1} is its terminal end. λ is a path from x_0 to x_k of length k. in simple path all it's vertices are distinct from one another.(Bondy and Murty, 1976)

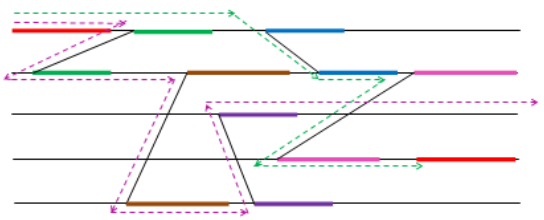

Figure 4: A simple path between the two subsequences of the diagonal with red color.

In the figure 4 we find a simple path that goes through other consistent diagonals fragments (the green path), so the red diagonal is considered inconsistent, then it is removed. when we try to find a path, we can have a lot of possibilities. in this figure 4, we can see that we have 2 paths, the green one lead to the solution and the other does not. We call this method the inconsistency path.

4.2 The Simple Path Search Method

We consider our graph as a rooted tree. The root is the vertex represented by the first fragment of diagonal d_i, it can either x_i or y_i. the goal is to find the other node in the graph. The simple path search uses the "depth-first search" algorithm, see (Cormen et al., 2001).

The structure of our tree differs from the usual structure i.e two different branches may converge on the same vertex see figure 5. The path passes only once by a given vertex.

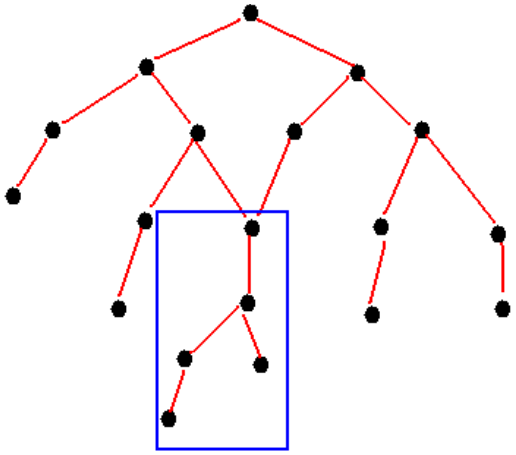

Figure 5: The tree search for a simple path. The path framed will be passed by only once.

4.3 Inconsistency with the Found Simple Path

In the phase when seeking a simple path to check the consistency of a diagonal d_i. If there is a path, all

diagonals in D belong to this path in the specified direction will be deleted. see figure 6.

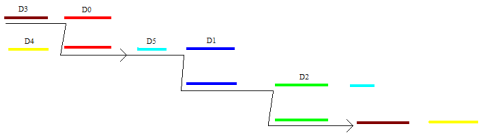

Figure 6: Diagonal belong to a simple path.

The simple path found to check the inconsistency of the diagonal D_3, will be used to eliminate the diagonal D_4 and D_5 that belong to this path and follow the right direction. With this technique we avoid to check some diagonals with the search of the simple path.

4.4 Simple Inconsistency

The first method inconsistency path (see 4.1) is sufficient alone to eliminate all inconsistent diagonals, but the implementation of this approach requires the use of recursion and the search can take several paths before finding the solution. For these reasons, we chose to use a second method (single inconsistency) that eliminates the inconsistencies between the two diagonals located in the same sequences, this saves a considerable execution time.

The algorithm takes a consistent diagonal d_i ($d_i \in D^{'}$), and compares it with the diagonal d_j where $d_j \in D$), if their sub-sequences are in the same sequence, and d_j cross or overlap with the d_i, then d_j is inconsistent so it is deleted directly from D.

Example : let us consider two biological sequences $S = \{s_0, s_1\}$, and let $d_0 = (x_0, y_0)$ a consistent diagonal, and 3 diagonals are not checked yet $D = \{d_1(x_1, y_1), d_2(x_2, y_2), d_3(x_3, y_3)\}$, all their sub-sequences are in the same sequence. See figure 7.

Figure 7: Simple inconsistency.

In this figure we clearly see the diagonal d_1, d_2 crosses d_0 and d_3 overlaps with d_0, so all these diagonals d_1, d_2, d_3 are inconsistent and they will be deleted.

4.5 Vertical Path

A path is called vertical only if it passes by the vertex (sub-sequences) of the same diagonal d_i. And if we have more than one diagonal, in this case their fragments must share the same vertex (it means their sub-sequences are in the same sequence and have the same beginning).

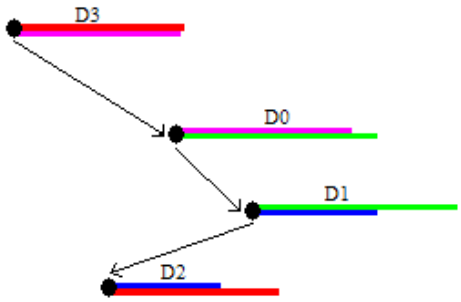

Figure 8: A simple vertical path.

According to the inconsistency path 4.1, the diagonal D_3 (with the red color) is inconsistent diagonal because we have a simple path between its two sub-sequences, it will be deleted, even if it does not cause a problem for alignment. D_3 will be aligned by transitivity when D_0, D_1 and D_2 will be aligned.

But, to remove the inconsistent diagonals with D_3 using the simple inconsistency method 4.4, we must keep it, and add it to the consistent diagonals $D^{'}$.

4.6 The Approach Process

Step I: Initializing the set of consistent diagonals $D^{'}$:

- We add the first diagonal d_0 to $D^{'}$. The first diagonal d_0 has the greatest weigh in the set D, so it is consistent automatically.
- We use the simple inconsistency method(see section 4.4) to remove all diagonals from D that are simply inconsistent with d_0.
- We add the top diagonal in D (let us call it d_1) to the consistent set $D^{'}$. We are sure that the diagonal d_1 is consistent thanks to the simple inconsistency method, hence d_1 is consistent with d_0, now $D^{'} = \{d_0, d_1\}$.
- We call the simple inconsistency method to remove all diagonals from D that are simply inconsistent with d_1.

Step II: Treat all remaining inconsistent diagonals in D:

- Find a consistent diagonal :
  apply the "path inconsistency" (see 4.1) that will eliminate all inconsistent diagonals in D with all diagonals in $D^{'}$, one by one, until it finds a consistent diagonal with diagonals already accepted, or it reaches the end.
  In this method, when we find a path between the two sub-sequences of a diagonal D, we will delete

it. All diagonals that are in this simple path will be deleted using the method "inconsistent with path found", see subsection 4.3.

When we check a diagonal and we don't find a simple path, it means that this diagonal is consistent, and hence we add it to the set D'.

- Apply the simple inconsistency method:
  For each consistent diagonal d_i obtained, we apply the simple inconsistency method to remove all diagonals from D that are simply inconsistent with it.
- Repeat these two steps until we treat all diagonals in D, at the end D' will contain only consistent diagonals, and D will be empty.

5 CONSISTENT DIAGONALS ALIGNEMENT

Alignment is performed on the remaining diagonals, each subsequence is placed in front of the other subsequence of the same diagonal. For that, we must sort all consistent diagonals for the final alignment, we sort them according to their position in the biological sequences.

- Indexing of subsequences diagonals according to their sequences number where they are located.
- Sort the diagonals, the diagonal that its two subsequences are before all the other subsequences, come first.
- Each sub-sequence is placed in front of the other sub-sequences of the same diagonal.
- Insert gaps at the end.

6 ANALYSIS AND COMPARISON

In our analysis, we will base on the computation time only because we did not change the score formula defined in the article (Morgenstren et al., 1996; Morgenstern et al., 1998b; Morgenstern et al., 1998a; Morgenstren et al., 1996).

Our implementation is modular, we use C++ language and Qt framework. The tests were done on an Intel 3.0 Gigahertz core 2 duo e8400 processor running Windows 7. We only used a single core. The source code and our software is available at https://github.com/chegrane/DiaWay for the first implementation, and the second one https://github.com/chegrane/NewDiAWay. It is licensed under the GNU Lesser General Public License (LGPL).

We make an analysis of the test results with the implementation of our approach. A comparison is made with the algorithm "DIALIGN". DNA sequences benchmarks are from "BAliBASE" and "SMART" databases [2].

The execution time varies according to the number of sequences to be aligned, their lengths, and the length of diagonal permitted.

The test results shown before arriving to the set of consistent diagonals, we are forced to remove almost all the initials diagonals, and saw that the number of diagonals is very important, hence the processing requires a lot of time.

Let us consider a set of biological sequences S with length $|S|$, the average sequences length will be noted as $\bar{S}$. The $|D|$ represents the length of all diagonals set D, and $|D'|$ the length of consistent diagonals only D'.

The following table 1 illustrates the relationship between the number of consistent diagonals and the total number of diagonals.

Table 1: Total number of diagonals, and the remaining consistent diagonals.

$\|S\|$	$\bar{S}$	$\|D\|$	$\|D'\|$
3	139	43757	156
5	202	101261	499
8	170	243845	661
10	101	131515	1474

The number of the consistent diagonals is less than 1% of all initial diagonals.

The execution time varies according to the number of all initial diagonals, and also according to the number of consistent diagonals.

In the figure 9 we show the execution time in millisecond for diagonal minimum length permitted 7.

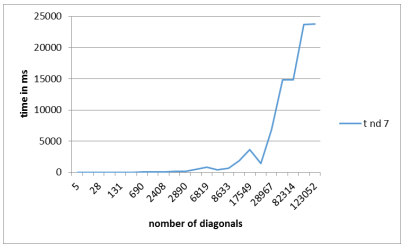

Figure 9: Execution time in MS for diagonal minimum length = 7.

The graph show that the execution time increases in correlation with the number of diagonals. A much smaller number of diagonals imply a significantly lower execution time.

[2] http://dna.cs.byu.edu/mdsas/download.shtml

When we have a biological sequences with high degree of similarity, the number of all diagonals and also consistent diagonals will be huge, and this influences and require a higher execution time, because, after some iteration to delete an inconsistent diagonals, the set of consistent diagonals becomes large, so the graph becomes very large too, and this makes the treatment methods as "Inconsistent path" slower. View the number of paths that must be tested before finding the right one.

6.1 Comparison with DIALIGN 2.2

We made two implementations of our approach, the first one called DiaWay (Diagonal Way), in this implementation when we extract a diagonal we extract also all its sub diagonals, if d_i is inconsistent we delete it completely, even if just a few residue of this diagonal are inconsistent with the current alignment. And their sub diagonals will have a chance to be aligned.

Our second implementation called "NewDiaWay", in this implementation we do not extract all sub diagonals, if we have some inconsistent residues from a diagonal we can directly cut the part consistent (the consistent residues), and consider it as a new diagonal.

We compared our two implementations with DIALIGN 2.2 (Morgenstern, 1999), the code is available here http://bibiserv2.cebitec.uni-bielefeld.de/dialign/.

The execution time depends on the number of biological sequences, and their average length, see 2.

Table 2: Execution time for DIALIGN 2.2, DiaWay (DW), and NewDiaWay (NDW).

$\|S\|$	$\bar{S}$	NDW	DW	DIALIGN 2.2
3	530	63	421	1312
4	250	31	125	766
5	200	47	125	*
6	119	32	1203	1750
7	195	125	422	1640
8	294	437	1453	4062
9	325	1047	6828	6297
10	476	3204	14782	16094
11	190	875	3656	3281

The * in the table 2 means that the application crashes and does not work for this example.

- We note that: the time to align 4 and 5 sequences is small compared to the time to align 3 sequences, because their length is much smaller than 3 sequences.
- Note that the method DiaWay (DW) has a better execution time than DIALIGN 2.2 for a small set of sequences, and relatively small average length, but for a large number of sequences its execution time is very bad because DiaWay (DW) extracts for each diagonal all its sub diagonals, hence the set of diagonals becomes huge, and that implies a very important execution time.
- The time execution for our implementation "New DiaWay" (NDW) is much smaller than the other 2 methods, DIALIGN and DiaWay (DW).

In the next test (see table 3), we will take just 2 biological sequences and we change every time the length in order to see the influence of sequences length on th execution time.

Table 3: Execution time in millisecond according to sequences length.

$\|S\|$	$\bar{S}$	NDW	DW	DIALIGN 2.2
2	2000	203	57640	6796
2	3000	437	172359	14843
2	4032	812	392703	25718
2	5000	1235	710984	38828
2	7499	2828	2223297	81281

In this table we can see that: the execution time of our NDW method is much smaller than DIALIGN 2.2.

7 CONCLUSION

In this paper we have presented an efficient and robust algorithm for dealing with diagonals consistency concept in DAILIGN method. This improvement comes into the phase of the construction of all consistent diagonals, and then how they are going to be aligned. We used graph theory modelling in order to represent diagonals position and solve the problem of consistency. Unlike previous approaches, which show that a diagonal is consistent, our approach does the opposite, it proves that a diagonal is inconsistent with diagonals already tested and accepted as consistent. We made two implementations of our approach DiaWay and NewDiaWay, and the results show that our implementations have a very good execution time comparing to DIALIGN 2.2.

REFERENCES

Abdeddam, S. (1997). On incremental computation of transitive closure and greedy alignment. In Apostolico, A. and Hein, J., editors, *Combinatorial Pattern Matching*, volume 1264 of *Lecture Notes in Computer Science*, pages 167–179. Springer Berlin Heidelberg.

Bondy, J. and Murty, U. (1976). *Graph Theory with Applications*. Macmillan.

Cormen, T., Leiserson, C., Rivest, R., and Stein, C. (2001). Depth-first search. In *Introduction To Algorithms*, chapter 22, pages 540–549. MIT Press.

Derrien, V., Richer, J.-M., and Hao, J.-K. (2005). Plasma, un nouvel algorithme progressif pour l'alignement multiple de séquences. In Solnon, C., editor, *Premières Journées Francophones de Programmation par Contraintes*, pages 39–48, Lens. CRIL - CNRS FRE 2499, Université d'Artois. http://www710.univ-lyon1.fr/c̃solnon.

Edgar, R. C. (2004). Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792–1797.

Kim, C. and Lee, B. (2005). Improved probability calculation for dialign2.

Morgenstern, B. (1999). Dialign 2: improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics*, 15(3):211–218.

Morgenstern, B., Atchley, W., Hahn, K., and Dress, A. (1998a). Segment-based scores for pairwise and multiple sequence alignments. *Ismb*.

Morgenstern, B., Frech, K., Dress, A., and Werner, T. (1998b). Dialign: finding local similarities by multiple sequence alignment. *Bioinformatics*, 14(3):290–294.

Morgenstren, B., Dress, A., and Werner, T. (1996). Multiple DNA and protein sequence alignment based on segment-to-segment comparison. 93(October):12098–12103.

Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443 – 453.

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

Smith, T. and Waterman, M. (1981). Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195 – 197.

Subramanian, A. R., Kaufmann, M., and Morgenstern, B. (2008). DIALIGN-TX: greedy and progressive approaches for segment-based multiple sequence alignment. *Algorithms for molecular biology : AMB*, 3:6.

Subramanian, A. R., Weyer-Menkhoff, J., Kaufmann, M., and Morgenstern, B. (2005). DIALIGN-T: an improved algorithm for segment-based multiple sequence alignment. *BMC bioinformatics*, 6:66.

Thompson, J. D., Higgins, D. G., and Gibson, T. J. (1994). 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.