A min-cut algorithm for the consistency problem in multiple sequence alignment

Eduardo Corel\textsuperscript{1,}*, Florian Pitschi\textsuperscript{2} and Burkhard Morgenstern\textsuperscript{1}

\textsuperscript{1}Georg-August-Universität, Institut für Mikrobiologie und Genetik, Goldschmidtstraße 1, 37077 Göttingen, Germany and \textsuperscript{2}Partner Institute for Computational Biology, CAS-MPG, 320 Yue Yang Rd, 200031 Shanghai, China

ABSTRACT

Motivation: Multiple sequence alignments can be constructed on the basis of pairwise local similarities. This approach is rather flexible and can combine the advantages of global and local alignment methods. The restriction to pairwise alignments as building blocks, however, can lead to misalignments since weak homologies may be missed if only pairs of sequences are compared.

Results: Herein, we propose a graph-theoretical approach to find local multiple sequence similarities. Starting with pairwise alignments produced by DIALIGN, we use a min-cut algorithm to find potential (partial) alignment columns that we use to construct a final multiple alignment. On real and simulated benchmark data, our approach outperforms the standard version of DIALIGN where local pairwise alignments are greedily incorporated into a multiple alignment.

Availability: The prototype is freely available under GNU Public License from E.C.

Contact: ecorel@gwdg.de

Received on November 25, 2009; revised on February 12, 2010; accepted on February 19, 2010

1 INTRODUCTION

Multiple sequence alignment (MSA) is a requisite for almost all aspects of computational sequence analysis, but it is a notoriously difficult task, see Edgar and Batzoglou (2006), Morrison (2006) or Kemena and Notredame (2009) for recent reviews. Traditionally, alignment methods have been characterized as either global or local alignment methods. The restriction to pairwise alignments as building blocks, however, can lead to misalignments since weak homologies may be missed if only pairs of sequences are compared.

Since the optimal multiple alignment problem is NP-hard under any reasonable objective function, virtually all MSA programs are based on heuristic optimization algorithms. Most of these heuristics work by integrating rather simple partial alignments into a final multiple alignment. For global alignment, paradigmatically represented by ClustalW (Thompson et al., 1994), this is usually done by progressive alignment. Here, single sequences and profiles of related sequences are aligned until all input sequences are included in a multiple alignment. In contrast, in the segment-based approach implemented in DIALIGN, local similarities are integrated sequentially under the constraints imposed by a certain consistency criterion. More recently, the progressive approach has also been applied to the segment-based alignment though (Subramanian et al., 2008).

Consistency is an order-theoretic condition that ensures the compatibility of the local similarities with the linear structure of a global alignment (Abdeddaïm and Morgenstern, 2001; Morgenstern et al., 1996). Shortly spoken, a set of (partial) alignments \( A_1, \ldots, A_k \) is called consistent, if an alignment \( A \) of the input sequences exists such that each of the alignments \( A_i \) is represented in \( A \). In the DIALIGN program, the raw material from which a multiple alignment is built is a set of pairwise gap-free local alignments. Such partial alignments are called fragment alignments or, shorter, fragments. Thus, a fragment is represented as a pair of segments of the same length from two of the input sequences. Each fragment is given a weight score based on the probability of its random occurrence, and the optimization task is to find a consistent set of fragments with maximum total weight.

For pairwise alignment, a consistent set of fragments is a chain of fragments, i.e. a set of fragments where for any two fragments one of them is strictly to the left of the other one. An optimal alignment in this approach is, therefore, a chain of fragments with maximum total weight and can be found by dynamic programming, either using standard fragment-chaining algorithms (Gusfield, 1997), or by a more space-efficient algorithm that is implemented in DIALIGN (Morgenstern, 2000, 2002). For multiple alignment, fragments from the respective optimal pairwise alignments are greedily included or discarded according to their consistency with each other.

© The Author 2010. Published by Oxford University Press. All rights reserved. For Permissions, please email: journals.permissions@oxfordjournals.org
A general problem with the algorithmic approach adopted in DIALIGN is that weakly conserved homologies can easily be missed since they may not appear statistically significant in the pairwise alignments that are carried out as the first step of the algorithm. Moreover, even if a weak local similarity is detected and represented by fragments in some of the optimal pairwise alignments, these fragments may be outweighed by spurious random similarities in other pairwise alignments. Therefore, alternative optimization algorithms have been proposed for the segment-based alignment approach, e.g. integer linear programming (Kececioglu et al., 2000; Lenhof et al., 1999). However, these approaches are computationally expensive. Alternatively, it is possible to search for multiple local alignments in a first step, and to use these alignments as anchor points (Morgenstern et al., 2005, 2006) for a subsequent global alignment procedure. Such an approach has been recently proposed by Pitschi (2008). Complementary to this anchored-alignment approach, it is possible for the user to exclude certain regions of the input sequences from being aligned to each other (Dress et al., 2008).

2 APPROACH

In this article, we study a different approach to the segment-based multiple alignment. Like in the original approach, we start with constructing all optimal pairwise alignments in the sense of DIALIGN, i.e. by finding optimal chains of fragments for all sequence pairs. However, instead of inserting these fragments directly into a multiple alignment, we first search for local similarities shared by more than two sequences. Our approach is based on the following observation: typically, local homologies involving more than two sequences correspond to groups of overlapping fragments (Fig. 1). In contrast, spurious random similarities are rarely part of such fragment groups.

In our approach, we therefore search for groups of positions in the sequences that are connected by many DIALIGN fragments. In addition, we require each of these groups to contain at most one position from each input sequence. Such groups are potential (partial) columns of a biologically meaningful multiple alignment. Consequently, the core problem of our approach is to extract partial alignment columns from the fragments produced by DIALIGN. If the positions of the sequences are seen as vertices in a graph and we have an edge between any two positions that are aligned by one of the selected fragments, the core problem of our approach is to extract highly connected subgraphs that contain at most one vertex from each of the input sequences. We use a min-cut, max-flow algorithm to identify such subgraphs.

The partial alignment columns that we find in this way need not be consistent with each other. In a second step, we therefore use a novel algorithm proposed by Pitschi (2008) to obtain a directed acyclic graph (DAG) from our potential alignment columns. This way, we obtain a consistent set of (partial) alignment columns. To align those parts of the sequences that are not yet aligned by the selected columns, we use these columns to define anchor points for DIALIGN in order to find further sequence similarities that are consistent with the partial alignment columns that we already obtained with our graph-theoretical approach.

Test runs on BAliBASE and on simulated local sequence homologies show that our approach is a considerable improvement compared to previous versions of DIALIGN. Interestingly, the numerical scores of our alignments are generally not better than with the standard version of DIALIGN. This indicates, that it is not the greedy optimization algorithm but rather the objective function used in DIALIGN that is to be blamed for the known limitations of this program on weakly but globally related sequence sets.

3 ALGORITHM

We consider a collection \( S \) of \( n \) sequences over a finite alphabet where \( i(t) \) is the length of the \( i \)-th sequence. The site \((i, p)\) is the \( p \)-th position of the \( i \)-th sequence. The site space

\[
S = \{(i, p) \mid 1 \leq i \leq n, 1 \leq p \leq i(t)\}
\]

is endowed with a natural partial ordering \( \leq \) such that \((i, p) \leq (i', p')\) holds if and only if \( i = i' \) and \( p \leq p' \). By \( S_t \) we denote the sites in the \( i \)-th sequence, i.e. the set \((i, p) \mid 1 \leq p \leq i(t)\). In the following, we freely identify the \( i \)-th sequence with the set \( S_i \). An alignment of \( S \) is an equivalence relation \( A \) of \( S \) satisfying the following consistency criterion: the preorder \( \leq_A = (\leq \cup A) \) restricted to any individual sequence \( s \in S \) coincides with \( \leq \). Here, \( R_t \) denotes the transitive closure of a relation \( R \). The equivalence classes of \( A \) correspond to columns of aligned positions of the sequences, and for any two sites \( x, x' \in S \), we have \( x \leq_A x' \) if and only if \( x \) is to the left or in the same column as \( x' \) in \( A \).

We call a subset \( C \subset S \) ambiguous, if there is a sequence \( S \) such that the intersection \( C \cap S \) contains at least two elements \((i, p)\) and \((i, p')\). In this case, we also call the sites \((i, p)\) and \((i, p')\) ambiguous with respect to \( C \). A non-ambiguous subset \( C \subset S \) is

\[\text{1By definition, a preorder is a transitive and reflexive relation on a set } X.\]

\[\text{2Note that the term consistency has been used in a different sense by various authors, such as Goebel (1990); Vinogen and Argos (1991); Nontemad et al. (2000) and ourselves. This has led to some confusion in the literature.}\]
called a partial alignment column. Moreover, we call an equivalence relation \( E \) on \( S \) ambiguous, if it contains an ambiguous equivalence class. A non-ambiguous equivalence relation consists of partial alignment columns only. Consistency of an equivalence relation obviously implies non-ambiguous, but the converse is in general not true.

In our approach to multiple alignment, we start with a set \( \mathcal{F} = \{f_1, \ldots, f_k\} \) of fragments. In the present study, \( \mathcal{F} \) is the set of fragments contained in the respective pairwise alignments produced by DIALIGN, i.e., the union of all optimal chains of fragments from all pairs of input sequences. However, for the algorithm described below, the composition of \( \mathcal{F} \) is not essential, and \( \mathcal{F} \) could be any set of fragments or other (partial) alignments. Thus, our first goal is to extract a set of potential (partial) alignment columns \( C_{ij} \) from these fragments. Formally, we consider the equivalence relation \( E \) induced by \( \mathcal{F} \), and we are looking for a non-ambiguous equivalence relation \( E' \) that is contained in \( E \).

In a graph-theoretical setting, we consider sites as vertices in a graph and pair of sites aligned by our fragments from \( \mathcal{F} \) as edges. According to our above definition, we call this graph non-ambiguous, if each of its connected components contains at most one vertex from each of the input sequences. Our goal is then to remove a minimal number of edges from \( G_{E'} \) to make the resulting graph non-ambiguous.

### 3.1 The incidence graph of a set of fragments

For a set \( \mathcal{F} \) of fragments as above, the incidence graph \( G_{\mathcal{F}} \) of \( \mathcal{F} \) is the undirected graph \( (S, E_{\mathcal{F}}) \) where \( S \) is the set of sites of our input sequences and a pair of sites forms an edge \( \{u, v\} \in E_{\mathcal{F}} \) if there exists a fragment \( f \in \mathcal{F} \) such that \( u \) is aligned to \( v \) by \( f \). The equivalence classes of the equivalence relation induced by the fragment set \( \mathcal{F} \) are the connected components of \( G_{\mathcal{F}} \).

As a rule, one observes that these components are extremely modular. While the graph \( G_{\mathcal{F}} \) as a whole is sparse, it usually consists of highly connected clusters of vertices, i.e., by dense subgraphs. These subgraphs are typically linked to each other by only a small number of edges (Fig. 1). Moreover, one can observe that these dense subgraphs are often non-ambiguous, i.e., each cluster contains at most one vertex per sequence. Thus, the well-connected clusters of vertices in the graph \( G_{\mathcal{F}} \) are potential (partial) alignment columns that we would like to include into a final MSA.

Our goal is therefore to turn the graph \( G_{\mathcal{F}} \) into a non-ambiguous graph by removing some of its edges. Whenever we detect an ambiguous connected component \( C \) in our graph, i.e., a connected component containing two vertices \( u \) and \( v \) from the same sequence, we partition \( C \) into two subsets \( C_1 \) and \( C_2 \) by removing a minimal number of edges such that we have \( u \in C_1 \) and \( v \in C_2 \). To this end, we consider the subgraph \( G \) induced by \( C \) as a flow network and use the max-flow min-cut theorem, that we recall now. We repeat this procedure until no ambiguous connected components are left.

### 3.2 Flow networks

A flow network \( N \) is a directed graph \( G=(V, E) \) with two distinguished vertices \( s \) and \( t \)—the 'source' and the 'sink'—where each edge \( (u, v) \) is assigned a non-negative real number, its 'capacity' \( c(u, v) \). Flow networks are used to model the situation where some material, e.g., some fluid or current, can 'flow' from \( s \) to \( t \) along the edges of the graph \( G \). The capacity of an edge \( (u, v) \) can be thought of as the maximum possible flow that goes directly from \( u \) to \( v \). A cut of a flow network is a partition \( V = S \cup T \) with \( s \in S \) and \( t \in T \). The capacity \( c(S, T) \) of a cut \((S, T) \) is the sum of the capacities of all edges going from \( S \) to \( T \).

A flow in a flow network \( N \) is an assignment of real numbers \( f(u, v) \) to the edges \( (u, v) \) of \( N \) that can be interpreted as the actual movement of material. The flow through any edge is bounded by its capacity and the total incoming flow for any vertex must equal the total outgoing flow, except for the source and the sink. The value \( |f| \) of the flow \( f \) is defined as the total flow leaving the source, which can be shown to equal the total flow entering the sink. Note that for any flow \( f \) in \( N \) and any cut \((S, T) \) of \( N \), we have \( c(S, T) \geq |f| \). The maximum-flow problem is the problem of finding a flow \( f \) with maximum value \( |f| \). According to the max-flow min-cut theorem, a flow \( f \) in \( N \) is maximal if and only if there is a cut \((S', T') \) of the network with \( c(S', T') = |f| \). There are several known polynomial-time algorithms to compute a maximum flow on a flow network, for example, Ford and Fulkerson (1956) or Edmonds and Karp (1972) (see also Cormen et al., 2001, pp. 651–664).

### 3.3 Step one: resolving ambiguities to construct partial alignment columns

Let \( C \) be a connected component in \( G_{\mathcal{F}} \) containing two nodes \( x \neq y \) from the same sequence. We turn \( C \) into a flow network \( N \) by defining two directed edges \( (u, v) \) and \((v, u)\) for any two nodes \( u \) and \( v \) that are connected by an undirected edge in \( G_{\mathcal{F}} \), i.e., are aligned by one of the fragments in \( \mathcal{F} \), and we define the capacity of every edge to be 1. The two 'ambiguous' nodes \( x \) and \( y \) are defined as the source and the sink, respectively, of our flow network.

We then use the Edmonds–Karp algorithm to compute a maximum flow \( f_{\text{max}} \) and we consider the so-called residual network \( N' \), which is obtained from \( N \) by subtracting the maximum flow \( f_{\text{max}} \) from the capacities of the edges, i.e., the capacity \( c'(u, v) \) in \( N' \) is defined as \( c'(u, v) = f_{\text{max}}(u, v) \). Edges \((u, v) \) with capacity \( c'(u, v) < 0 \) are removed. Note that the capacity of each cut in \( N \) is at least \( f_{\text{max}} \) and, according to the Edmonds–Karp theorem, there is a cut \((S, T) \) with capacity \( c(S, T) = |f_{\text{max}}| \). Thus, in the residual network \( N' \), all edges between \( S \) and \( T \) are removed and our ambiguous subgraph \( G \) is split into two connected subgraphs by removing a minimal set of edges from \( E_{\mathcal{F}} \). We apply this algorithm successively to split ambiguous subgraphs of \( G_{\mathcal{F}} \) by removing minimal subsets of edges from \( E_{\mathcal{F}} \) until \( G_{\mathcal{F}} \) is non-ambiguous.

One problem in our approach is that the connected components of our incidence graph \( G_{\mathcal{F}} \) can be very large. We therefore define a threshold \( k \) and apply the above graph algorithm only to those nodes that have degrees \( \geq k \). We start with \( k = \max(\text{deg}(v) \mid v \in S) \) and \( k \) is successively lowered until all nodes of \( G_{\mathcal{F}} \) have been considered.

A high-level description of our algorithm is as follows:

1. **Start:** Let the input be a sequence of aligned columns and a maximum alignment score. Let \( k = \sqrt{|S|} \).
2. **For** each column in the alignment:
   - If the column has a high score, append it to the output.
   - If the column has a low score, remove it from the input.
3. **Repeat** steps 2 until no more columns can be removed.

This algorithm is efficient and can handle large inputs. It is especially useful for aligning long sequences, such as those found in genomics.
there is no partial column we proceed in two steps: (i) successively remove the lowest weighted turn our (possibly) inconsistent set of classes finding a set of partial columns whose succession graph is a DAG. To resolve potential inconsistencies in our set of partial columns, we use an algorithm that was originally introduced by Pitschi (2008).

Input
The input to the algorithm is a set \( S \) of sequences, a set \( E \) of pairs \((x, y)\) of sites, and a set \( C \) of (translated) columns. Each column \( C \) consists of pairs \( (i, p) \) and \( (i', p') \) for sites \( i \) and \( i' \), where \( i, i' \) are aligned positions and \( p, p' \) are alignment columns.

Algorithm 1 Ambiguity resolving algorithm

1. **Input:** \( G_F = (S, E, F) \)
2. **Algorithm:**
   - **Initialization:**
     - \( k = \max \{\text{deg}(y) | y \in S\} \)
     - \( E \leftarrow E_F \)
   - **While loop:**
     - **While condition:** \( k > 0 \)
     - **Update:** \( E \leftarrow \{(u, v) \in E | \text{deg}(u, \text{deg}v) \geq k\} \)
       - **Compute connected components of \( (S, E) \)**
       - **While loop:**
         - **While condition:** there is an ambiguous connected component \( C \) of \( (S, E) \), i.e. with vertices \( x, y \) from same sequence \( S \)
           - **Compute connected components of \( (S, E) \)**
             - **While loop:**
               - **While condition:** there is an ambiguous connected component \( C \) of \( (S, E) \), i.e. with vertices \( x, y \) from same sequence \( S \)
                 - **1. Deline flow network on \( C \) with \( x \) as source and \( y \) as sink**
                 - **2. Apply Edmonds-Karp to find minimal cut \((C_1, C_2)\) of \( C \)**
                 - **3. Remove edges between \( C_1 \) and \( C_2 \) from \( E \)**
               - **end while**
             - **end while**
           - **end while**
         - **end while**
       - **end while**
     - **end while**
   - **Return:** \( (S, E) \) non-ambiguous subgraph of \( G_F \)

3.5 Constructing a final multiple alignment from partial alignment columns

With the algorithms outlined in Sections 3.3 and 3.4, we obtain a consistent set of partial alignment columns, that is an alignment \( A \) in the sense of our above set-theoretical alignment definition. In general, however, it will be possible to further extend this alignment. In the sense of our set-theoretical alignment definition, there may be an alignment \( A' \) that is a proper superset of \( A \). To find a suitable extension \( A' \) of \( A \), we run DIALIGN by using our partial columns as anchor points (Morgenstern et al., 2006). Here, anchor points are considered as ungapped local pairwise alignments, i.e. as fragments in the sense of DIALIGN. In general, the user can specify a set of potential anchor points with user-defined weights from which DIALIGN selects a consistent subset in a greedy way based on their weights. In our case, we can use any set of anchor points representing our selected partial alignment columns, without worrying about their weights, since our set of anchor points is consistent anyway.

To study the influence of the two steps of our approach described in Sections 3.3 and 3.4, respectively, we tested a second version of our method in order to ensure that a consistent set of anchors is used. Therefore, a weight needs to be defined for each of the anchor points, and the proposed anchor points are selected greedily according to these weights. Thus, the resulting alignment depends on how the weights of the anchor points are exactly defined.

To obtain a set of anchor points from a set \( C \) of partial alignment columns, we consider all maximal pairs of segments \((i, p), \ldots, (i, p+k)\) and \((i', p'), \ldots, (i', p'+k)\) such that every pair of sites \((i, p+l)\) and \((i', p'+h)\), \(1 \leq l, h \leq k\), belongs to some partial alignment column in \( C \). Each of these segment pairs (fragments) defines an anchor point. We defined the weights of these anchor points in two different ways, namely \( i \) by their length and \( i \) by using the fragment-weighting function that is used in DIALIGN (Morgenstern, 1999).

3.6 Time complexity

Since the time complexity of the Edmonds–Karp algorithm depends quadratically on the number of nodes of the input graph, the run time of our method strongly depends on the size of the connected components of our incident graphs. In the worst case, there is a single connected component comprising the entire site space \( S \) and each node \((i, p)\) is connected to each sequence \( S_j \neq S_i \) by some edge. Thus, a single connected component has up to \( n + \ell \) nodes and \( n^2 + \ell^2 \) edges where \( n \) is the number of sequences and \( \ell \) their maximum length. The time complexity of the Edmonds–Karp algorithm is \( O(|V|(|E|)^2) \), so the worst-case complexity of our algorithm to find a minimal cut for a single ambiguous connected component is \( O(n^3 + \ell^3) \). In the worst case, each run of Edmonds-Karp splits off a single node, so
the algorithm is run \( n \times t \) times. The worst-case time complexity of our full algorithm is, therefore, \( O(n^6 \cdot t^5) \).

For realistic datasets, the connected components of our incidence graphs are, fortunately, much smaller than in the theoretical worst case, so we could run our method on most datasets in the benchmark databases in reasonable time. An example is reference set RV12 of BAiBASE. RV12 consists of 88 sequence families with an average of 10 sequences per sequence family. The average incidence graph \( G_f \) in RV12 consists of 2877 nodes 10952 edges and 223 connected components and, on average, the Edmonds–Karp algorithm is run 649 times, thereby removing 899 edges. On RV12, the mean CPU time per sequence family is 48 s on an Opteron machine with 2.4 GHz.

As will be discussed in Section 4.1, we had to terminate the program runs on some large sequence families from BAiBASE. To obtain results with our method in reasonable time, we applied a threshold \( T \) and removed all fragments \( f \) with weight scores \( w(f) < T \) from \( F \). An extreme case was sequence family BB30003 from BAiBASE. This sequence set comprises 142 sequences, and the incident graph consists of one single connected component with around \( 1.5 \times 10^5 \) edges. We aborted the program run after 20 h without results. With a threshold of \( T = 4 \), the graph still consists of more than \( 8.3 \times 10^4 \) edges, and we obtained a multiple alignment of these 142 sequences after 13 h.

4 TEST RESULTS

To evaluate the performance of our method, we used three benchmark databases for multiple alignment. For global protein alignment, we used the well-known database BAiBASE 3 developed by Thompson et al. (2005); for local alignment, we used the databases IRMBASE (for protein alignment) and DIRBASE (for DNA alignment) developed by Subramanian et al. (2005, 2008). All reference sequence sets from these three databases contain so-called core blocks for which a correct alignment is known.

The performance of alignment programs can be measured in two different ways: the SPs score measures the proportion of pairs of sites in the core blocks of the reference alignment that are correctly aligned by the method under evaluation. The total columns (TC) score measures the proportion of alignment columns from the core blocks that are correctly aligned. The total column score is a more stringent measure, and it can be applied in a meaningful way only to those benchmark sequences where the core blocks involve all of the input sequences. We used the program alicompare (Notredame et al., 2000) to calculate these scores. Note that both, SP and TC scores, measure the sensitivity of alignment methods. In BAiBASE, we only used the non-truncated long versions of the reference sequence sets.

We evaluated two versions of our software. In a first set of test runs, we only used our min-cut algorithm to resolve ambiguity conflicts in the fragment set \( F \) and constructed sets of partial alignment columns as explained in Section 3.3. Note that these sets of partial alignment columns are not necessarily consistent. Anchor points were extracted from these data as outlined in Section 3.5 and given to DIALIGN. In a second set of test runs, the set of partial alignment columns was further processed as outlined in Section 3.4 to obtain a consistent set of partial alignment columns that were directly given to DIALIGN as anchor points.

4.1 Global alignment benchmark: BAiBASE 3

Tables 1 and 2 summarize the performance of our method on BAiBASE 3. On all six reference sets of BAiBASE, both versions of our approach consistently achieve a considerable improvement over the standard version of DIALIGN. This is true for both, SP and TC scores. For the TC score, however, the improvement is most substantial. It also becomes clear from

<p>| Table 1. Performance of our min-cut method compared with other MSA methods on reference sets RV11 to RV50 in BAiBASE 3 based on an SP evaluation scheme |</p>
<table>
<thead>
<tr>
<th>RV11</th>
<th>RV12</th>
<th>RV20</th>
<th>RV30</th>
<th>RV40</th>
<th>RV50</th>
</tr>
</thead>
<tbody>
<tr>
<td>min-cut, cons. PAC</td>
<td>53.14</td>
<td>88.53</td>
<td>90.51</td>
<td>78.56</td>
<td>87.67</td>
</tr>
<tr>
<td>min-cut, all PAC</td>
<td>50.64</td>
<td>88.96</td>
<td>90.07</td>
<td>78.19</td>
<td>87.21</td>
</tr>
<tr>
<td>DIALIGN-TX</td>
<td>51.52</td>
<td>89.18</td>
<td>87.88</td>
<td>76.18</td>
<td>83.64</td>
</tr>
<tr>
<td>DIALIGN 2.2</td>
<td>50.73</td>
<td>86.66</td>
<td>86.92</td>
<td>74.05</td>
<td>83.31</td>
</tr>
<tr>
<td>CLUSTALW 2.0</td>
<td>49.27</td>
<td>86.89</td>
<td>86.23</td>
<td>70.71</td>
<td>79.65</td>
</tr>
<tr>
<td>MAFFT 6.717</td>
<td>66.19</td>
<td>93.36</td>
<td>92.72</td>
<td>87.08</td>
<td>92.19</td>
</tr>
<tr>
<td>MUSCLE 3.7</td>
<td>57.16</td>
<td>91.54</td>
<td>88.91</td>
<td>81.45</td>
<td>86.49</td>
</tr>
<tr>
<td>PROBCONS 1.12</td>
<td>66.97</td>
<td>91.12</td>
<td>91.68</td>
<td>84.53</td>
<td>90.34</td>
</tr>
<tr>
<td>T-COFFEE 7.81</td>
<td>66.77</td>
<td>94.08</td>
<td>91.62</td>
<td>83.81</td>
<td>89.96</td>
</tr>
</tbody>
</table>

![Table 1](https://academic.oup.com/bioinformatics/article/26/8/1015/206628)

Table 1. Performance of our min-cut method compared with other MSA methods on reference sets RV11 to RV50 in BAiBASE 3 based on an SP evaluation scheme

<p>| Table 2. Performance on BAiBASE 3 based on the TC score |</p>
<table>
<thead>
<tr>
<th>RV11</th>
<th>RV12</th>
<th>RV20</th>
<th>RV30</th>
<th>RV40</th>
<th>RV50</th>
</tr>
</thead>
<tbody>
<tr>
<td>min-cut, cons. PAC</td>
<td>30.51</td>
<td>73.99</td>
<td>35.21</td>
<td>41.01</td>
<td>50.17</td>
</tr>
<tr>
<td>min-cut, all PAC</td>
<td>28.82</td>
<td>74.07</td>
<td>34.07</td>
<td>40.05</td>
<td>47.54</td>
</tr>
<tr>
<td>DIALIGN-TX</td>
<td>26.81</td>
<td>75.69</td>
<td>30.78</td>
<td>38.90</td>
<td>45.17</td>
</tr>
<tr>
<td>DIALIGN 2.2</td>
<td>26.84</td>
<td>70.03</td>
<td>29.71</td>
<td>31.62</td>
<td>44.53</td>
</tr>
<tr>
<td>CLUSTALW 2.0</td>
<td>24.00</td>
<td>72.32</td>
<td>20.44</td>
<td>26.87</td>
<td>40.04</td>
</tr>
<tr>
<td>MAFFT 6.717</td>
<td>44.13</td>
<td>83.83</td>
<td>45.46</td>
<td>58.90</td>
<td>60.56</td>
</tr>
<tr>
<td>MUSCLE 3.7</td>
<td>32.06</td>
<td>80.90</td>
<td>35.30</td>
<td>41.19</td>
<td>45.32</td>
</tr>
<tr>
<td>PROBCONS 1.12</td>
<td>41.96</td>
<td>86.05</td>
<td>41.15</td>
<td>54.73</td>
<td>53.61</td>
</tr>
<tr>
<td>T-COFFEE 7.81</td>
<td>42.65</td>
<td>85.71</td>
<td>39.21</td>
<td>49.99</td>
<td>56.30</td>
</tr>
</tbody>
</table>

Table 2. Performance on BAiBASE 3 based on the TC score

Notation as in Table 1.
Numerical alignment scores in the sense of the we use a graph-theoretical approach to obtain potential (partial) include these fragments directly into a multiple alignment. Instead, with total maximum weight. Unlike DIALIGN, however, we do not (1996), i.e. for each sequence pair, we search for a chain of fragments optimal pairwise alignments in the sense of Morgenstern local pairwise alignments. We first use DIALIGN to construct all (2005), which consistently performed well on global approaches to multiple alignment have two basic components: an objective function assigning quality scores to possible alignments and an optimization procedure for finding high-scoring alignments in the sense of the chosen objective function. Thus, the failure of an alignment program to produce reasonable alignments can have two reasons: the objective function may assign optimal scores to biologically wrong alignments, or the optimization algorithm may fail to find a (near-)optimal alignment. In Table 3, we compared the numerical scores of the improved alignments calculated with our new method to the scores of the standard DIALIGN alignments. In four out of six reference sets in BAliBASE, the DIALIGN alignments have on average higher numerical scores than the biologically superior alignments produced by our new approach. This clearly demonstrates that the objective function currently used in DIALIGN is flawed. Possible reasons for the shortcomings of this objective function are discussed in Subramanian et al. (2005). Thus, it is unlikely that DIALIGN can be further improved by applying more efficient optimization algorithms on the basis of the current objective function. Instead, it seems worthwhile to investigate novel objective functions for the segment-based alignment problem. Probabilistic approaches may provide a way of optimising the weight parameters used by DIALIGN, for example, by using conditional random fields. Also, it seems worthwhile to use (partial) multiple local alignments instead of pairwise fragments as building blocks for MSA and to develop improved objective functions based on such partial multiple alignments.

Table 4. Results on the local benchmark databases DIRMBASE 1 (D1–D4) and DIRMBASE 2 (I1–I4) 

<table>
<thead>
<tr>
<th>Dataset</th>
<th>I1</th>
<th>I2</th>
<th>I3</th>
<th>I4</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
</tr>
</thead>
<tbody>
<tr>
<td>min-cut, cons. PAC</td>
<td>90.7</td>
<td>92.8</td>
<td>92.9</td>
<td>92.5</td>
<td>76.4</td>
<td>75.3</td>
<td>78.5</td>
<td>81.7</td>
</tr>
<tr>
<td>min-cut, all PAC</td>
<td>91.8</td>
<td>89.7</td>
<td>90.8</td>
<td>93.1</td>
<td>75.3</td>
<td>75.2</td>
<td>79.8</td>
<td>81.1</td>
</tr>
<tr>
<td>DIALIGN-TX 1.0.2</td>
<td>89.4</td>
<td>94.9</td>
<td>93.8</td>
<td>93.6</td>
<td>94.4</td>
<td>92.9</td>
<td>95.4</td>
<td>95.7</td>
</tr>
<tr>
<td>DIALIGN 2.2</td>
<td>90.4</td>
<td>93.4</td>
<td>91.8</td>
<td>93.0</td>
<td>92.6</td>
<td>91.1</td>
<td>94.6</td>
<td>94.1</td>
</tr>
<tr>
<td>CLUSTALW 2.0</td>
<td>9.3</td>
<td>12.4</td>
<td>19.6</td>
<td>29.1</td>
<td>10.7</td>
<td>9.8</td>
<td>15.6</td>
<td>22.5</td>
</tr>
<tr>
<td>MAFFT 6.717b</td>
<td>87.7</td>
<td>92.0</td>
<td>89.9</td>
<td>88.3</td>
<td>92.5</td>
<td>83.7</td>
<td>87.3</td>
<td>86.5</td>
</tr>
<tr>
<td>MUSCLE 3.7</td>
<td>30.4</td>
<td>34.5</td>
<td>54.0</td>
<td>57.8</td>
<td>47.3</td>
<td>53.2</td>
<td>56.9</td>
<td>67.7</td>
</tr>
<tr>
<td>PROBCONS 1.12</td>
<td>78.8</td>
<td>85.7</td>
<td>87.1</td>
<td>87.7</td>
<td>29.9</td>
<td>31.3</td>
<td>41.5</td>
<td>52.9</td>
</tr>
<tr>
<td>T-COFFEE 7.81</td>
<td>82.1</td>
<td>89.4</td>
<td>89.6</td>
<td>91.3</td>
<td>8.6</td>
<td>8.8</td>
<td>17.7</td>
<td>32.0</td>
</tr>
</tbody>
</table>

These databases consist of simulated sequences with local homologies in otherwise unrelated sequences. Notation as in Table 1.

Table 3. Numerical alignment scores in the sense of the DIALIGN objective function on BAliBASE for DIALIGN 2.2 and our min-cut approach using consistent partial alignment columns as anchor points for DIALIGN 

<table>
<thead>
<tr>
<th>Method</th>
<th>I1</th>
<th>I2</th>
<th>I3</th>
<th>I4</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIALIGN 2.2</td>
<td>267</td>
<td>3703</td>
<td>113867</td>
<td>70220</td>
<td>27680</td>
<td>28291</td>
<td></td>
<td></td>
</tr>
<tr>
<td>min-cut</td>
<td>255</td>
<td>3758</td>
<td>114174</td>
<td>68555</td>
<td>26099</td>
<td>27855</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.2 Local alignment benchmarks: IRMBASE2 and DIRMBASE1

Since BAliBASE and other standard alignment benchmark data consist almost exclusively of sequence families that are globally related, i.e. with similarity extending over the entire length of the sequences, we additionally used benchmark databases for local protein and DNA alignment, namely IRMBASE and DIRMBASE. Following an approach first proposed by Lastmann and Sonnhammer (2002), these databases consist of simulated protein and DNA sequence families, respectively. Locally conserved sequence motifs created using the ROSE software program (Stoye et al., 1998) are inserted into non-related random sequences.

IRMBASE and DIRMBASE consist of four reference sets each with randomly implanted ROSE motifs. Unlike the core blocks in BAliBASE, these motifs do not necessarily span all sequences in a reference sequence family. Thus, the TC score cannot be defined in a meaningful way and is, therefore, not considered in this study. Table 4 contains the results of our test runs. On the simulated local protein homologies in IRMBASE, our results were comparable to the results of DIALIGN 2.2 and DIALIGN-TX. On the locally related DNA sequences in DIRMBASE our min-cut approach performed worse though. On all locally related benchmark data, however, our method outperformed the global aligners, with the remarkable exception of MAFFT (Katoh et al., 2005), which consistently performed well on global and local sequence data.

5 DISCUSSION

In this article, we introduced a new way of composing MSAs from local pairwise alignments. We first use DIALIGN to construct all optimal pairwise alignments in the sense of Morgenstern et al. (1996), i.e. for each sequence pair, we search for a chain of fragments with total maximum weight. Unlike DIALIGN, however, we do not include these fragments directly into a multiple alignment. Instead, we use a graph-theoretical approach to obtain potential (partial) alignment columns based on the DIALIGN fragments. To align the remainder of the sequences, we use these partial columns as anchor points in DIALIGN. Thus, the major difference between our approach and the original version of DIALIGN is the fact that we use local multiple sequence similarities rather than pairwise similarities as a basis for MSA.

The restriction to local pairwise similarities as building blocks for multiple alignment is a major drawback of DIALIGN. DIALIGN is one of the best methods for local multiple alignment. It also produces good global alignments if sequences are related over their entire length. For sequences with weak global sequence similarity, however, DIALIGN is often outperformed by global alignment methods. With the new approach that we proposed, we focus on local similarities that span more than two sequences; this seems to be a promising way to overcome the current limitations of DIALIGN.
ACKNOWLEDGEMENTS

We thank Andreas Dress and Etienne Birmelé for helpful discussions and Rasmus Steinkamp for help with the computer cluster of the Department of Bioinformatics in Göttingen.

Funding: Deutsche Forschungsgemeinschaft, in English: German Research Foundation (grant MO 1048/6-1 to B.M.); a fellowship of the Max-Planck-Society (to F.P.).

Conflict of Interest: none declared.

REFERENCES


Morgenstern, B. et al. (2005) Multiple sequence alignment with user-defined constraints at GOBICS. Bioinformatics, 21, 1271–1273.


