Alignments as Compositional Structures

Sarah Berkemer[1], Christian Höner zu Siederdissen[2], and Peter F. Stadler[3]

[1]Max Planck Institute for Mathematics in the Sciences, Leipzig, Germany; and Bioinformatics Group, Department of Computer Science, Universität Leipzig, Germany

[2]Bioinformatics Group, Department of Computer Science, Universität Leipzig, Germany

[3]Bioinformatics Group, Department of Computer Science, Universität Leipzig, Germany; Max Planck Institute for Mathematics in the Sciences, Leipzig, Germany; Department of Theoretical Chemistry, University of Vienna, Austria; Facultad de Ciencias, Universidad Nacional de Colombia, Bogotá, Colombia; Santa Fe Institute, Santa Fe, NM, USA;

Alignments, i.e., position-wise comparisons of two or more strings or ordered lists are of utmost practical importance in computational biology and a host of other fields, including historical linguistics and emerging areas of research in the Digital Humanities. The problem is well-know to be computationally hard as soon as the number of input strings is not bounded. Due to its practical importance, a huge number of heuristics have been devised, which have proved very successful in a wide range of applications. Alignments nevertheless have received hardly any attention as formal, mathematical structures. Here, we focus on the compositional aspects of alignments, which underlie most algorithmic approaches to computing alignments. We also show that the concepts naturally generalize to finite partially ordered sets and partial maps between them that in some sense preserve the partial orders.

1 Introduction

Alignments play an important role in particular in bioinformatics as a means of comparing two or more strings by explicitly identifying correspondences between letters (usually called matches and mismatches) as well as insertions and deletions [11]. The aligned positions are interpreted either as deriving from a common ancestor ("homologous") or to be functionally equivalent. Alignments have also been explored as means of comparing words in natural languages, see e.g. [5, 9, 33, 51], as a convenient way of comparing ranked lists [16], for comparison of text editions [53, 58], and to analyse synteny in the comparison of genomes [21, 55].

The literature on alignments is extensive. However, it its concerned almost exclusively with practical algorithms and applications. The alignment problem for two input strings has an elegant recursive solution for rather general cost models and has served as one of the early paradigmatic examples of dynamic programming [44, 49]. Since these algorithms have only quadratic space and time requirements for simple cost models [19, 44], they are

Sarah Berkemer: bsarah@bioinf.uni-leipzig.de,

Christian Höner zu Siederdissen: choener@bioinf.uni-leipzig.de,

Peter F. Stadler: studla@bioinf.uni-leipzig.de,

Accepted in Compositionality on 9999-99-99. Click on the title to verify.

```
      (a)                 (b)                 (c)                 (d)
A 0000111110000     A 0000111110000     B 000011011----     B 000011011
B 000011011----     C ----100010000     C ----100010000     C 100010000
    s = 4               s = 2               s = -5              s = 5
```

Figure 1: Alignments of three binary sequences `A`, `B`, and `C` with a simple column-wise score of $+1$ for matches, 0 for mismatches, and -1 for gaps. Alignment (c) is transitively implied by (a) and (b), but is it not an optimal pairwise alignment of `B` and `C`.

of key importance in practical applications. The same recursive structure easily generalizes to alignments of more than two sequences [8, 38] even though the cost models need to be more restrictive to guarantee polynomial-time algorithms [31]. The computational effort for these exact solutions to the alignment problem increases exponentially with the number of sequences, hence only implementations for 3-way [20, 32, 34] and 4-way alignments [51] have gained practical importance. A wide variety of multiple sequence alignment problems (for arbitrary numbers of input sequences) have been shown to be NP-hard [6, 14, 28, 30, 56] and MAX SNP-hard [40, 57]. The construction of practical multiple alignment algorithms therefore relies on heuristic approximations. These fall into several classes, see e.g. [3, 12] for reviews.
(1) *Progressive* methods typically compute all pairwise alignments and then use a “guide tree” to determine the order in which these are stepwisely combined into a multiple alignment of all input sequences. The classical example is `ClustalW` [35]. The approach can be extended to starting from exact 3-way [32, 34] or 4-way alignments [51].
(2) *Iterative* methods starting to align small gapless subsequences and then extend and improve the alignment until the score converges. A paradigmatic example is `DIALIGN` [42].
(3) *Consistency*-based alignments and *consensus* methods start from a collection of partial alignments (often exact pairwise alignments) to obtain candidate matches and extract a multiple alignment using agreements between between the input alignments.

Most of the successful multiple alignment algorithm in computational biology combine these paradigms. For example `T-COFFEE` [45] and `ProbCons` [10] use consistency ideas in combination with progressive constructions; `MUSCLE` [13] and `MAFFT` [29] combine progressive alignments with iterative refinements.

A key assumption underlying consistency based methods is transitivity: considering three input sequences x, y, and z, if x_i aligns with y_j and y_j aligns with z_k, then x_i should also align with z_k. While this property holds for the pairwise constituents of a multiple alignment, it is a well known fact that the three score-optimal alignments that can be constructed from three sequences in general violate transitivity, see Fig. 1. `TRANSALIGN` [39] uses transitivity to align input sequences to a target database using an intermediary database of sequences to increase the search space. Here, intermediary sequences show which subsequences of input and target sequence can be transitively aligned. This may result in a few well aligned subsequences that are then extended to one aligned region via a simple scoring function. The same notion of transitivity is also used in `psiblast` [2] to stepwisely increase the set of sequences that are faintly similar to an input sequence.

Practical applications distinguish whether the complete input sequences are to be aligned, or whether a maximally scoring interval is to be considered. In the latter case one allows an additional “unaligned state” for prefixes and/or suffixes of the input. This leads to slight changes in exact algorithms, exemplified by an extra term in the local Smith-Waterman algorithm [50] compared to the global Needleman-Wunsch [44] algorithm. This idea can be generalized to mixed problems in which a user can determine for each of the two ends of each input sequence whether it is to be treated as local or global [48]. For the purpose of the present contribution (partially) local alignments require a slight, trivial extension of the presentation, which we – for the sake of clarity if the presentation – only briefly comment on.

Alignments are usually constructed from strings or other totally ordered inputs, hence the columns of the resulting alignment are usually also treated as a totally ordered set. Consecutive insertions and deletions, however, are not naturally ordered relative to each other:

$$\begin{array}{ll}\texttt{gugugu--acgggcca} & \texttt{guguguac--gggcca}\\ \texttt{gucuguug--gggccc} & \texttt{gucugu--uggggccc}\end{array}\tag{1}$$

are alignments that are equivalent under most plausible scoring models. The idea to consider alignment columns as partial orders was explored systematically in [37] and a series of follow-up publications [22, 36]. Here, (mis)matches are considered as an ordered backbone, with no direct ordering constraints between an insertion and a deletion. The resulting alignments are then represented as directed acyclic graphs (DAGs), more precisely, as the Hasse diagrams of the partial order. The key idea behind the `POA` software [37] is that a sequence of DAGs can be used as an input to a modified version of the Needleman-Wunsch algorithm [44]. Recently this idea has been generalized to the problem of aligning a sequence to a general directed graph [47, 54].

Despite the immense practical importance of alignments, they have received very little attention as mathematical structures in the past. The most comprehensive treatment, at least to our knowledge, is the Technical Report [43], which considers (pairwise) alignments as binary relations between sequence positions that represent matchings and preserve order. We use many of these ideas here. The notion of a composition of pairwise alignments – formalized as composition of partial maps that represent the matching – first appears in [39]. We will return to this point in Section 3. Following our earlier work [46], we will use a language that is closer to graph theory than the presentation of [43].

2 Alignments and Partial Orders

Consider a finite collection X of two or more finite totally ordered sets X_a. It will be convenient in the following to denote an element $i \in X_a$ by (a, i). The following definition rephrases the approach taken e.g. in [43, 52]. It will be generalized below to deal with partial orders instead of total orders.

Definition 1 ([46])**.** *A total alignment of the totally ordered sets X_a is a triple $(X, A, <)$ where (X, A) is a graph and $<$ is a total order relation on the set of connected components $\mathcal{C}(X, A)$ satisfying*[1]

(1) *$Q \in \mathcal{C}(X, A)$ is a complete subgraph of (X, A).*

(2) *If $(a, i) \in Q$ and $(a, j) \in Q$, then $i = j$.*

(4) *If $(a, i), (b, j) \in P$ and $(a, k), (b, l) \in Q$ with $i < k$ then $j < l$.*

(5) *If $(a, i) \in P$, $(a, j) \in Q$ and $i < j$ then $P < Q$.*

The connected components of (X, A) are usually called the alignment columns. Condition (2) ensures that every alignment column contains at most one element of each ordered set X_a. Conversely, every element (a, i) is contained in exactly one connected component, i.e., alignment column. Condition (4) requires that alignment columns do not cross.

[1] There is no condition (3) due to synchronization with the definitions for partial orders defined later.

Condition (5) ensures that the order on the columns is such that the projection of the alignment columns to each individual row exactly recovers the input order. Conditions (4) and (5) in general only specify a partial order as the following result shows:

Lemma 2. *Let (X, A) be the graph of an alignment and denote by $\prec$ the relation on $\mathcal{C}(X, A)$ defined by $P \prec Q$ whenever there is $(a, i) \in P$ and $(a, j) \in Q$ with $i < j$. Then the transitive closure $\ddot{\prec}$ of $\prec$ is a partial order on $\mathcal{C}(X, A)$.*

Proof. Clearly $\ddot{\prec}$ is antisymmetric. If $P \prec Q$, then there there is a sequence of columns $P = Q_0 \ddot{\prec} Q_1 \ddot{\prec} \dots Q_k = Q$. Since the sequence of elements (a, i) belonging to the same X_a is strictly increasing with the column index j for each a along any such sequence of columns, it follows that the transitive closure of $\ddot{\prec}$ is still antisymmetric, and thus a partial order. □

As an immediate consequence, there is also a (not necessarily unique) total order $<_*$ of the alignment columns, obtained as an arbitrary linear extension of $\ddot{\prec}$, which by construction satisfies

$$P <_* Q,\ (a, i) \in P, \text{ and } (a, j) \in Q \quad \text{implies} \quad i < j. \tag{2}$$

Hence, whenever conditions (1), (2), and (4) in Definition 1 are satisfied, there indeed exists a total order on $\mathcal{C}(X, A)$ that satisfies condition (5).

In order to treat (partially) local alignments it is necessary to distinguish aligned and "unaligned" columns. Each unaligned column may contain only a single element – note however, that also regular columns may contain only a single entry from each row. Furthermore, all "unaligned" positions for a prefix and/or a suffix of each input $(X_a, <_a)$ form "unaligned" columns.

In this condition we will consider a more general setting. Instead of totally ordered sets X_a we will consider finite partially ordered sets $(X_a, \prec_a)$.

Definition 3. *An alignment of X is a triple $(X, A, \prec)$ where (X, A) is a graph and $\prec$ is a partial order on the set of connected components $\mathcal{C}(X, A)$ such that*

(A1) *$Q \in \mathcal{C}(X, A)$ is a complete subgraph of (X, A).*

(A2) *If $(a, i) \in Q$ and $(a, j) \in Q$, then $i = j$.*

(A3) *If $(a, i) \in P$, $(a, j) \in Q$ and $(a, i) \prec_a (a, j)$ then $P \prec Q$.*

(A4) *$P \prec Q$, $(a, i) \in P$ and $(a, j) \in Q$ implies $(a, i) \prec_a (a, j)$ or (a, i) and (a, j) are incomparable w.r.t. $\prec_a$.*

Condition (A3) constrains the partial order on the columns to respect the partial order of the rows. Condition (A4) insists that columns also must not cross indirectly.

Condition (A4) obviously implies the following generalization of (4):

(A4*) $(a, i), (b, j) \in P$ and $(a, k), (b, l) \in Q$ and $(a, i) \prec_a (a, k)$ implies $(b, j) \prec_b (b, l)$ or (b, j) and (b, l) are incomparable w.r.t. $\prec_b$.

However, (A4*) is not sufficient to guarantee that the alignment columns form a partially ordered set. A counterexample is shown in Fig. 2. It is therefore necessary to require the existence of the partial order $\prec$ on $\mathcal{C}(X, A)$ in Definition 3.

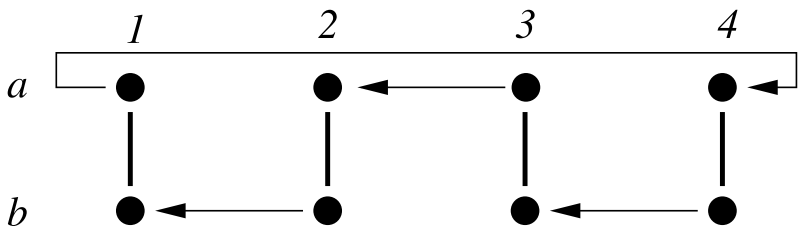

Figure 2: Property (A4*) is not sufficient to ensure the existence of a partial order $\prec$ on $\mathcal{C}(X,A)$. Consider the partial orders $(a,4)\prec_a(a,1)$ and $(a,2)\prec_a(a,3)$ and $(b,1)\prec_b(b,2)$ and $(b,3)\prec_a(b,4)$, with alignment colums $\{(a,i),(b,i)\}$ for $i=1,2,3,4$. Clearly (A2), (A3), and (A4*) holds, but the directed cycle shows that no partial order on the colums exists that is consistent with both partial orders.

In order to model partially local alignments of PO-sets we consider the set $\mathcal{A}$ of aligned columns and a partition of the set of "unaligned columns" into two not necessarily non-empty subsets $\mathcal{P}$ and $\mathcal{S}$ such that for all $U\in\mathcal{P}$, $V\in\mathcal{A}$ and $W\in\mathcal{S}$ it holds that $W\not\prec V$ and $V\not\prec U$, i.e., no "unaligned" suffix column preceeds an aligned column, and no "unaligned" prefix column succeeds an aligned column. "Unaligned" prefix columns belonging to different rows $(X_a,\prec_a)$ are considered mutually incomparable; the same is assumed for "unaligned" suffix columns. With the caveat that "unaligned" columns need to be marked as such, there is no structural difference between local and global alignments.

If all $(X_a,\prec_a)$ are totally ordered then condition (A4) implies the non-crossing condition (4) because (b,j) and (b,l) cannot be incomparable w.r.t. $\prec_b$, and thus the required partial order $\prec$ is obtained as the transitive closure of the relative order of any two columns. Definitions 1 and 3 therefore coincide for totally ordered rows.

The existence of (non-trivial) alignments of any collection of finite partial orders $(X_i,\prec_i)$, $i=1,\dots,N$ is easy to see: each of the partial orders can be linearly extended to a total order $(X_i,<_i)$. Any alignment of these total orders is also an alignment of the underlying partial orders, with a suitable partial order of the columns given by Lemma 2.

It may be interesting to explore alignments satisfying a (much) stronger version of axiom (A4), which stipulates that $(X_a,\prec_a)$ is recovered as projection of (X,A) onto row a, i.e.,

(A5) $P\prec Q$, $(a,i)\in P$ and $(a,j)\in Q$ implies $(a,i)\prec_a(a,j)$.

As argued above, (A4) and (A5) are equivalent if all $(X_a,\prec_a)$ are totally ordered. In general this is not the case, as the example in Fig. 3 shows.

The following simple, technical result is a generalization of Lemma 2, showing that condition (A5) is sufficient to guarantee the existence of a partial order on the columns.

Lemma 4. *Let (X,A) be a graph with connected components $\mathcal{C}(X,A)$ satisfying (A1) and (A2). Let $\prec$ denote the transitive closure of the relation $\dot\prec$ defined by (A3), i.e., $P\dot\prec Q$ whenever $(a,i)\in P$, $(a,j)\in Q$ and $(a,i)\prec_a(a,j)$ then $P\prec Q$. Finally assume that axiom (A5) holds. Then $\prec$ is a partial order on $\mathcal{C}(X,A)$*

Proof. It suffices to show that $\prec$ is antisymmetric. It is clear from the construction that by (A5) we know that $\dot\prec$ is antisymmetric. If $\prec$ is not antisymmetric, then there is a finite sequence of columns P_i, $i=0,\dots,k$ such that $P_0\dot\prec P_1\dot\prec\dots\dot\prec P_k\dot\prec P_0$ such that any two consecutive columns P_i and P_{i+1} have at a pair of entries, say $(a_i,h)\in P_i$ and $(a_i,h')\in P_{i+1}$, in the same row. For the transitive closure this would imply both

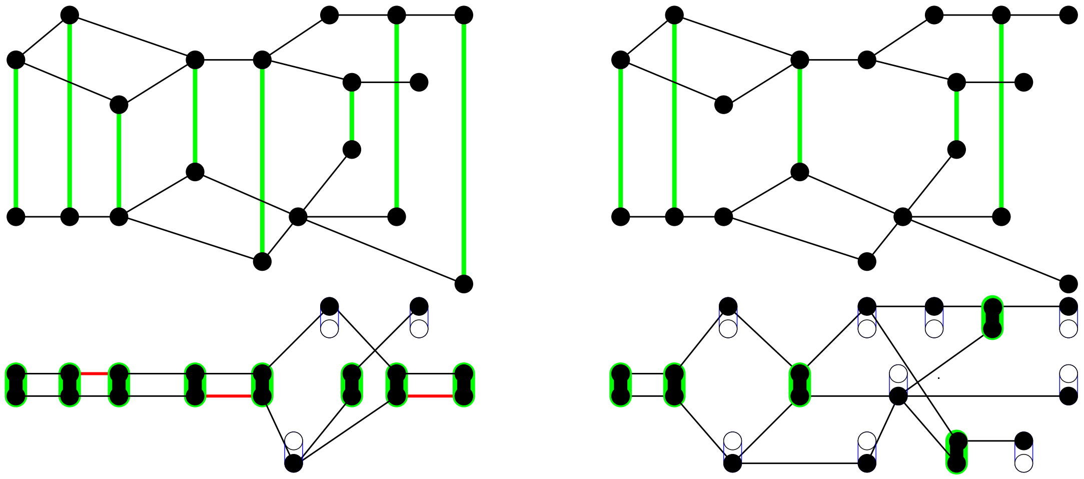

Figure 3: **Top:** Pairwise alignments of partially ordered sets. Thin black edges show the Hasse diagram, to be read from left to right. Alignment edges are shown in green.
Bottom: The induced partial order of the alignment columns with corresponding points vertically aligned. The partial order is again shown as a Hasse diagram, with superflous edges omitted. Both the l.h.s. and the r.h.s. example satisfy (A4), i.e., none of order relations $\prec_1$ and $\prec_2$ is violated in the alignment. The red edges highlight two comparabilities introduced by partial order of the columns that are absent in the input posets. Red edges therefore imply a violation of condition (A5). Hence the l.h.s. alignment violates (A5), while the r.h.s. alignment does not.

$(a_i,h) \prec (a_i,h')$ from $(a_i,h)\dot{\prec}(a_i,h')$ and $(a_i,h') \prec (a_i,h)$ by going around the cycle, contradictiong axiom (A5). □

Condition (A5) implies that the restriction of the partial order $\prec$ on the columns to any subset of columns in which a given set of rows is represented coincides with the induced partial order on the corresponding vertex set in $(X_a,\prec_a)$. Regarding the $(X_a,\prec_a)$ as graphs, the aligned columns form a common induced subgraph. The alignment problem for partially ordered sets under axiom (A5) thus can be seen as a generalized version of a maximum induced subgraph problem. We refer to [7] for a discussion of the relationships of edit distances and maximum common subgraph problems in a more general setting.

The following result generalizes Lemma 1 of [46]:

Lemma 5. *Let $(X,A,\prec)$ be an alignment and let $Y \subseteq X$. Then the induced subgraph $(X,A)[Y]$ with the partial order $\prec$ restricted to the non-empty intersections $Q\cap Y$ for $Q \in \mathcal{C}(X,A)$ is again an alignment. Furthermore, if $(X,A,\prec)$ satisfies* (A5)*, then the restriction to $(X,A)[Y]$ again satisfies* (A5)*.*

Proof. Every induced subgraph of a complete graph is again a complete graph, hence (A1) holds for $(X,A)[Y]$, hence the connected components of $(X,A)[Y]$ are exactly the non-empty intersections of Y with the components Q of (X,A). Condition (A2) remains unchanged by the restriction to Y. Finally, the partial order $\prec$ satisfying (A3) restricted to the non-empty intersections $Q\cap Y$ for $Q \in \mathcal{C}(X,A)$ is a partial order that obviously still satisfies (A4) since the restriction to Y only removes some of the conditions in (A4).

To see that the restriction of $(X,A)[Y]$ again satisfies (A5) it suffices to recall that the partial order in the colums is given by $P\cap Y \prec Q\cap Y$ whenever $P \prec Q$ and both $P\cap Y \neq \emptyset$ and $Q\cap Y\neq\emptyset$. If one of the intersections is empty, axiom (A5) becomes void since the empty set is not a column in $(X,A)[Y]$. On the other hand, if the two restricted columns have entries (a, i) and (a, j) in the same row, then (A5) for $(X, A, \prec)$ ensures $(a, i) \prec_a (a, j)$, i.e., the implication (A5) remains true for the restricted alignment. □

Note that additional partial orders on connected components of the induced subgraph $(X, A)[Y]$ may exist that are not obtained as restrictions of the partial order on $\mathcal{C}(X, A)$. The reason is that omitting parts of the columns may allow a relaxation of their mutual ordering.

Rooted trees can be seen as partially ordered sets, with the natural partial order defined by $x \prec y$ if y lies on the unique path connecting x and the root of the tree. This special case is thus covered in the general framework outlined here. Usually, tree alignments are defined on rooted *oriented trees*, however, where the relative order of siblings is preserved [4, 23, 27], thus imposing additional restrictions on valid alignments. We will return to this point in some generality in the discussion section.

3 Composition of Alignments

The fact that alignments are again totally or partially ordered sets implies that one can also meaningfully define alignments of alignments. More precisely:

Lemma 6. *Let $(X, A, \prec)$ be an alignment and consider a non-trivial partition $\mathfrak{P}$ of the set of objects, i.e., the rows. Denote the site sets of the classes of $\mathfrak{P}$ by X_1, X_2, ..., X_p and consider the sub-alignments $(X, A, \prec)[X_i]$. Then $(X, A, \prec)$ is isomorphic to the (vertex) disjoint union of the $(X, A, \prec)[X_i]$ augmented by extra edges (x', x'') whenever there is a column Q of A with $x' \in Q \cap X_i$ and $x'' \in Q \cap X_j$ for $X_i \neq X_j$.*

Proof. The alignments $(X, A, \prec)[X_i]$ are induced subgraphs of (X, A). Their disjoint union lacks exactly all edges that connect pairs of vertices that are in the same connected component of (X, A) but are not in the same subgraph $(X, A)[X_i]$. Since the partial order on the colums of $(X, A)[X_i]$ is the one inherted from $(X, A, \prec)$, the re-composition of the columns also recovers the original partial order. □

The $(X, A, \prec)[X_i]$ can also be interpreted as partially ordered sets whose *points* are the non-empty restrictions $Q \cap X_i$ of the connected components of (X, A).

Definition 7. *We denote by $(X, A)/\mathfrak{P}$ the quotient graph whose vertices are the columns of the alignments $(X, A)[X_i]$, that is, the non-empty sets $Q \cap X_i$ where Q is a connected component of (X, A). Its edges are the pairs $(Q \cap X_i, Q \cap X_j)$ for which both $Q \cap X_i$ and $Q \cap X_j$ are non-empty.*

The connected components of the graph $(X, A)/\mathfrak{P}$ are therefore of the form $Q' := Q/\mathfrak{P} = \{Q \cap X_i | Q \cap X_i \neq \emptyset\}$. Note that Q' is non-empty since the column Q of (X, A) contains at least one element, which belongs to at least one of the $(X, A)[X_i]$. Thus there is a 1-1 correspondence between the connected components of (X, A) and those of $(X, A)/\mathfrak{P}$. The columns of $(X, A)/\mathfrak{P}$ naturally inherit the partial order $\prec$ of $\mathcal{C}(X, A)$. We write $(X, A, \prec)/\mathfrak{P}$ for the quotient graph with this partial order on its connected components.

Lemma 8. *$(X, A, \prec)/\mathfrak{P}$ is an alignment.*

Proof. Consider the quotient graph $(X, A)/\mathfrak{P}$. By construction, each column Q' is a complete graph and contains at most one node for each class of $\mathfrak{P}$ since it is the quotient of a column of $(X, A, \prec)$ w.r.t. $\mathfrak{P}$. Also by construction, we have $P' \prec Q'$ for the columns of $(X, A)/\mathfrak{P}$ whenever $P \prec Q$ in $(X, A, \prec)$. Since there is a 1-1 correpondence between columns of $(X, A, \prec)$ and $(X, A, \prec)/\mathfrak{P}$, $\prec$ also serves as a partial order on the columns of $(X, A)/\mathfrak{P}$, which is by construction consistent with the partial order on each of the $(X, A)[X_i]$. □

As a consequence, every alignment can be decomposed into an alignment of alignments w.r.t. an arbitrary partition of the rows. The constituent alignments $(X_i, A, \prec_i)$ have at most the same number of columns since "all gap" columns, $Q' = Q \cap X_i = \emptyset$, are removed. By Lemma 8, the decomposition can be used recursively until each constituent is only a single partially ordered set $(X_a, \prec_a)$. Any such recursive composition is naturally represented as a tree $\mathfrak{T}$ whose leaves are the input posets $(X_a, \prec_a)$. Each internal node of $\mathfrak{T}$ corresponds the an alignment of its children, with the root corresponding to $(X, A, \prec)$, the alignment of all the data.

The reverse of this type of decomposition underlies all *progressive alignment* schemes. One starts from a guide tree $\mathfrak{T}$ whose leaves are the $(X_a, \prec_a)$ and for each inner node of $\mathfrak{T}$ constructs an alignment (or a set of alternative alignments) from the (set of) alignments attached to its children. It is important to note that a score-optimal alignment $(X, A, \prec)$ in general is **not** the score-optimal alignment $(X, A, \prec)/\mathfrak{P}$ of score-optimal consitutents $(X_i, A_i, \prec_i)$, or, in other words, if $(X, A, \prec)$ is score-optimal, there is no guarantee that there is any partition of the rows $\mathfrak{P}$ such that all the restrictions $(X, A, \prec)[X_i]$ are score-optimal subalignments. Progressive alignments methods thus can only approximate the solution of the multiple alignment problem. Practical results depend substantially on the choice of the guide tree $\mathfrak{T}$. It is has been suggested early [17], that $\mathfrak{T}$ should closely resemble the evolutionary history of the input sequences. Usually $\mathfrak{T}$ is constructed from distance or similarity measures between all pairs of input sequences – and usually pairwise alignments are employed to obtain these data. A special case of progressive alignment adds a single sequence in each step, instead of also considering alignments of alignments.

4 Blockwise Decompositions

On the other hand, we can also decompose alignments into blocks of columns. More precisely, if $(X, A, \prec)$ is an alignment and $\mathfrak{Q}$ is a partition of the X with classes Y_k such that

(i) If $P \in \mathcal{C}(X, A)$ then $P \subseteq Y_k$ for some class $Y_k \in \mathfrak{Q}$.

(ii) There is a partial order $\lhd$ on $\mathfrak{Q}$ such that for any two distinct classes $Y', Y'' \in \mathfrak{Q}$ such that $Y' \lhd Y''$ whenever there are columns $P \in Y'$ and $Q \in Y''$ with $P \prec Q$.

We call the classes of such a partition *blocks*. By Lemma 5 each block $(X, A, \prec)[Y_k]$ is again an alignment.

Lemma 9. *Given blocks $(X, A, \prec)[Y_k]$ and the partial order $\lhd$, there is an alignment $(X, A, \prec')$, where $\prec'$ is an an extension of $\prec$ defined by $P \prec' Q$ if and only if $P \prec Q$ for $P, Q \in Y$ for some $Y \in \mathfrak{Q}$ and $P \prec' Q$ for $P \in Y'$ and $Q \in Y''$ with $Y' \lhd Y''$.*

Proof. Each alignment block consists of the disjoint union of alignment column(s), thus the disjoint union of complete subgraphs. Given the partial order of alignment columns given by $P \prec Q$, this order is preserverd inside the alignment blocks Y_k as each block is an alignment, too. Given an alignment block Y with $P \prec Q$ for $P, Q \in Y$ for some $Y \in \mathfrak{Q}$, one can decompose this into two blocks Y' and Y'' with at least one column in each block such that $P \in Y'$ and $Q \in Y''$. Based on the decomposition of Y into Y' and Y'' one can restore the order of the alignment blocks such that $Y' \lhd Y''$ based on Y. Thus, one gets the order of $P \prec' Q$ that is present for the alignment columns P and Q as well as for the alignment blocks Y' and Y''. □

In the case of totally ordered inputs, the restriction $X_a \cap Y$ of a block Y to an input X_a is an interval of X_a and the columns in Y form an interval of the columns of $(X, A, <)$. Similarly, one can restrict choice of blocks in such a way that $\lhd$ just "mirrors" the initial partial order, i.e., $Y' \lhd Y''$ if and only if $P \prec Q$ for P in Y' and Q in Y'', in which case $\prec' = \prec$ and the original alignment is recovered by the concatenation of the blocks. In particular, this also guarantees that valid block decompositions can be constructed for alignments satisfying (A5).

Each alignment can thus be recursively decomposed into blocks. This sets the stage for Divide-and-Conquer algorithms such as `DCA` [52], which cuts the sequences to be aligned into subsequences and then concatenates the subalignments so as to optimize a global score. In order to find the best cut-points, the algorithm recurses on differently cut subsequences. Algorithms such as `dialign` [42] work in a conceptually similar manner but use a bottom-up instead of a top-down approach: they first identify blocks with high sequence conservation as "anchors" and recurse to construct alignments for sequences between them.

An extreme case of the block-wise decomposition is to consider the division of an alignment $(X, A, \prec)$ into a single maximal (or minimal) alignment column P, and the rest $(X \setminus P, A', \prec)$ of the alignment. In order for $X \setminus A \lhd P$ to hold, we have to ensure that $p_a \not\prec_a q_a$ for all $p_a \in P$ and $q_a \in X \setminus P$, i.e., the column P must entirely consist of suprema of the respective input posets. Under this condition, we obtain a recursive column-wise decomposition of alignments. As we shall see in the following section, this recursion can also be used constructively.

5 Recursive Construction

Given a PO-set $(Y, \prec)$ we say that $P \subseteq Y$ is a *bottom set* if, for all $p \in P$, every $p' \prec p$ satisfies $p' \in P$. By definition, the empty set, Y itself, as well as the set $\{p' \in Y | p' \preceq y\}$ for each $y \in Y$ are bottom sets. Note, however, that P also may contain points that are incomparable to all other elements of P. Denote by $\sup P$ the set of *suprema* of P, i.e., the points such that there is no $p' \in P$ with $p \prec p'$. Clearly, if P is a bottom set and $p \in \sup P$ then $P \setminus \{p\}$ is again a bottom set. The latter observation suggests that there is a recursive construction for the set of alignments of $(X_1, \prec_1)$ and $(X_2, \prec_2)$.

Denote by $\mathfrak{A}^P_Q$ the set of all pairwise alignments on bottom sets P in X_1 and Q in X_2. An alignment $\mathbb{A} \in \mathfrak{A}^P_Q$ is necessarily of one of three types:

(i) $\mathbb{A} = \mathbb{A}'\binom{p}{q}$ with $\mathbb{A}' \in \mathfrak{A}^{P'}_{Q'}$,

(ii) $\mathbb{A} = \mathbb{A}'\binom{p}{-}$ with $\mathbb{A}' \in \mathfrak{A}^{P'}_{Q}$, or

(iii) $\mathbb{A} = \mathbb{A}'\binom{-}{q}$ with $\mathbb{A}' \in \mathfrak{A}^{P}_{Q'}$,

where $P' := P \setminus \{p\}$ for $p \in \sup P$, $Q' := Q \setminus \{q\}$ for $q \in \sup Q$, and $\mathfrak{A}^{\emptyset}_{\emptyset}$ contains only the empty alignment.

The three cases correspond to a (mis)match, insertion, and deletion. It is important to note that this recursion is in general not unique because the columns extracted from $\mathbb{A}$ in consecutive steps are not necessarily ordered relative to each other whenever $|\sup P| \geq 1$ or $|\sup Q| \geq 1$. It is, however, a proper generalization of the Needleman-Wunsch recursion

[44] for the pairwise alignment of ordered sets (strings): If the $\prec_a$ are total orders, then $\sup P_a$ always contains a single element, and we recover the usual Needleman-Wunsch algorithm. In order to have a proper start and end case for the recursion and thus DP-algorithm, it is convenient to introduce "virtual" source and a sink nodes being connected to all start or end nodes of the poset, respectively.

This idea generalizes to alignments of an arbitrary number of partial orders in the obvious way. Denote by $\mathfrak{A}(P_1, P_2, \dots, P_N)$ the set of all alignments where the P_a are a bottom set of $(X_a, \prec_a)$.

Theorem 10. *Every alignment $\mathbb{A} \in \mathfrak{A}(P_1, P_2, \dots, P_N)$ is of the form $\mathbb{A}'\Xi$ where the alignment column Ξ is a supremum w.r.t the partial order of $\prec$ of alignment columns and $\mathbb{A}' \in \mathfrak{A}(P'_1, P'_2, \dots, P'_N)$. The column Ξ contains in row a either a gap row a, in which case $P'_a = P_a$, or $p_a \in \sup P_a$, in which case $P'_a = P_a \setminus \{p_a\}$, and does not entirely consist of gaps. For every column Υ of $\mathbb{A}'$ we have either $\Upsilon \prec \Xi$ or Υ and Ξ are incomparable.*

Proof. The P'_a are again bottom sets, hence $\mathbb{A}'$ is an alignment. By assumption, there is a partial order on the columns $\prec$ of $\mathbb{A}'$. Since every non-gap entry in Ξ is a $p_a \in \sup P_a$, it follows that this partial order extends to $\mathbb{A}$ if and only if Ξ is a supremum, i.e., it is either incomparable with or larger than any column in $\mathbb{A}'$. Now suppose that the column Ξ contains a $q_a \notin \sup P_a$, i.e., there is a $p_a \in X_a$ with $p_a \succ q_a$. Consider the column Υ containing p_a. Then either no partial order $\prec$ on the columns exists (contradicting that $\mathbb{A}'$ is an alignment), or $\Upsilon \succ \Xi$ (contradicting that Ξ is a supremum for the alignment columns. □

The bottom sets are of course uniquely defined by their suprema. Clearly $\sup P$ is an antichain, i.e., its elements are pairwisely incomparable. Conversely, every antichain U in $(X_a, \prec_a)$ uniquely defines a bottom set $P := \{p \in X_a | p \preceq U\}$. It is obvious therefore that for two bottom sets P and Q it holds that $P = Q$ if and only if $\sup P = \sup Q$. Hence there is a 1-1 correspondence between the antichains of a partial order and their bottom sets. The recursion in the theorem can be written in terms of the antichains of the $(X_a, \prec_a)$.

In order to capture the more restrictive notion of alignments satisfying (A5) the recursion has to be modified in a such a way that for every (mis)match between two rows it can be ensured that all previously formed columns are either comparable in both rows or incomparable in both rows. This is non-trival because this information is not purely local. For ease of discussion, we only consider the case of aligning two posets. There are at least two strategies to maintain this information.

Attempting to construct a similar recursion as in the (A4) case, one could store with each pair $P \in X_1$ and $Q \in X_2$ also all the set $\mathcal{M}$ of all matchings $\binom{p}{q}$ "to the right" of P and Q, i.e., $p \in X_1 \setminus P$ and $q \in X_1 \setminus Q$. Then every allowed matching/column $\binom{p'}{q'}$, $p' \in \sup P$ and $q' \in \sup Q$ must satisfy: for all $\binom{p}{q} \in \mathcal{M}$ holds: either $p' \prec p$ and $q' \prec q$, or both p', p and q', q are incomparable. Every such pair can be appended to $\mathcal{M}$, with corresponding updates $P \to P \setminus \{p'\}$ and $Q \to Q \setminus \{q'\}$. Insertions and deletions of course only require the removal of either p' from P or q' from Q, respectively. Initially, $P = X_1$, $Q = X_2$, and $\mathcal{M} = \emptyset$. Every set of valid partial alignments is characterized by a triple $(P, Q, \mathcal{M})$.

An alternative approach is to store instead for each $p \in P$ and $q \in Q$ also the sets $c_Q(p)$ and $c_P(q)$ that can form matches $\binom{p}{q'}$, $q' \in c_Q(p)$ and $\binom{p'}{q}$, $p \in c_P(q)$, respectively. Initially, we have $P = X_1$, $Q = X_2$, $c_Q(p) = Q$ for all $p \in P$ and $c_P(q) = P$ for all $q \in Q$. Whenever a an alignment is continued with a (mis)match $\binom{p}{q}$, $p \in \sup P$, $q \in \sup Q$, we have to remove all candidates from $c_P(q')$ and $c_Q(p')$ that are inconsistent with $\binom{p}{q}$. That is: if $q' \prec q$, then $c_P(q') \leftarrow \{p' \in c_P(q') | p' \prec p\}$. If q and q' and incomparable, then $c_P(q') \leftarrow \{p' \in c_P(q') | p', p \text{ incomparable}\}$. The $c_Q(p')$ are updated correspondingly. In the case of an insertion $\binom{p}{-}$, we only need to remove p from $f_P(q')$, $q' \in Q$. Similarly, $\binom{-}{q}$ implies that q has to be removed from the $f_Q(p')$ for all $p' \in P$. We suspect that an encoding of alignment sets of the form $(P, f_Q : P \to 2^P; Q, f_P : Q \to 2^P)$ will be efficient if the poset has only small antichains. A more detailed analysis of this kind of recursive construction from the point of view of algorithmic efficiency will be considered elsewhere.

The POA algorithm [37] computes the alignment of two posets satisfying (A5), albeit with the restriction that one of the two inputs is totally ordered. This removes all ambiguities in the totally ordered po-set and implies that, given any match $\binom{u}{v}$ in the alignment, all preceeding matches $\binom{u'}{v'}$ satisfy $v' < v$ in the totally ordered set and thus u' must be a predecessor of u. The alignment thus must follow a single path in the Hasse diagram of the unrestricted input poset.

6 Pairwise Alignments as Relations

Pairwise alignments have a particularly simple structure. In particular, they are bipartite (undirected) graphs, and hence can be regarded equivalently as symmetric binary relations $R \subseteq X_1 \times X_2$. More precisely, we can identify a relation R with an undirected graph with vertex set $X_1 \dot\cup X_2$ and (undirected) edges $\{x_1, x_2\}$ whenever $(x_1, x_2) \in R$. We write this graph as $(X_1 \dot\cup X_2, R)$.

Relations have a natural composition. For $R \subseteq X \times Y$ and $S \subseteq Y \times Z$ is is defined by

$$(x, z) \in S \circ R \quad \text{iff} \quad \exists y \in Y \text{ s.t. } (x, y) \in R \text{ and } (y, z) \in S \tag{3}$$

In the following we will be interested in the following properties of binary relations:

(M) $(x, y) \in R$ and $(x, z) \in R$ implies $y = z$ and $(x, z) \in R$ and $(y, z) \in R$ implies $x = y$ and

(P') There is a partial order $\prec$ on R such that $u \prec_1 x$ or $v \prec_2 y$ implies $(u, v) \prec (x, y)$.

(P) If $(x_1, y_1) \in R$ and $(x_2, y_2) \in R$ then $x_1 \prec x_2$ if and only if $y_1 \prec y_2$.

Lemma 11. *The composition of two binary relations satisfying (M) and (P) is again a binary relation satisfing (M) and (P).*

Proof. Suppose $(x, z) \in R \circ S$. Then there is y such that both $(x, y) \in R$ and $(y, z) \in S$. By (M), there is no other $y' \neq y$ with $(x, y') \in R$ and no $z' \neq z$ such that $(y, z) \in S$, hence in particular there is no $z' \neq z$ such that $(x, z') \in R \circ S$. Analogously, one argues that there is no $x' \neq x$ such that $(x', z) \in R \circ S$. Thus $R \circ S$ again satisfies (M).

Suppose $(x_1, z_1), (x_2, z_2) \in R \circ S$. By (M) there are unique vertices y_1 and y_2 such that $(x_1, y_1), (x_2, y_2) \in R$ and $(y_1, z_1), (y_2, z_2) \in S$, respectively. Now suppose $x_1 \prec_1 x_2$. Then (P) implies $y_1 \prec_2 y_2$, and using (P) again yields $z_1 \prec_3 z_2$. Starting from $z_1 \prec_3 z_2$, the same argument yields $z_1 \prec_1 z_2$. Conversely, suppose $(x_1, z_1), (x_2, z_2) \in R \circ S$ and x_1, x_2 are incomparable. By (M) there are unique vertices y_1 and y_2 with $(x_1, y_1), (x_2, y_2) \in R$ and $(y_1, z_1), (y_2, z_2) \in S$, for which (P) now implies that they are incomparable. Using the same argument again shows that that z_1 and z_2 also must be incomparable. Hence concatenation preserves not only the relative order but also comparability, i.e., $R \circ S$ again satisfies (P). □

It is easy to see that Axiom (P') is in general not preserved under concatenation: Requiring only (P') allows the intermediate vertices y_1 and y_2 to be incomparable. Hence it is possible in this scenario to have $x_1 \prec_1 x_2$, incomparable vertices y_1 and y_2, and $z_2 \prec_3 z_1$ with $(x_1, y_1), (x_2, y_2) \in R$ and $(y_1, z_1), (y_2, z_2) \in S$ while the concatenation violates the (P').

A relation satisfying (M) and (P') can easily be extended to an alignment $(X_1 \cup X_2, R)$ considering each edge (x_1, y_1) and considering all unmatched positions, i.e., every $\{x'\}$ such that there is no $y \in X_2(x', y)$ and every $\{y'\}$ such that there is no $x \in X_1(x, y')$ as alignment columns. The relative order of these columns is inherited from the partial order $(X_1, \prec_1)$ and $(X_2, \prec_2)$.

Lemma 12. *Every pairwise alignment satisfying* (A1)*,* (A2)*,* (A3)*, and* (A4) *can be written as an extension of the a binary relation $R \subseteq X_1 \times X_2$ satisfying* (M) *and* (P')*. Conversely, every binary relation $R \subseteq X_1 \times X_2$ satisfying* (M) *and* (P') *gives rise to an alignment satisfying* (A1)*,* (A2)*,* (A3)*, and* (A4)*.*

Proof. By definition, all edges are incident to one vertex in X_1 and one vertex in X_2, thus the graph is a bipartite matching. Condition (M) is therefore equivalent to (A1) and (A2) for the case of two input posets. Axiom (A3) implies the ordering required by (P') as well as its extension to the in/del columns. (A4) and (P') equivalently guarantee the existence of the partial order on the columns that satisfy (A3). □

Lemma 13. *Every pairwise alignment satisfying* (A5) *corresponds to a binary relation $R \subseteq X_1 \times X_2$ satisfying* (M) *and* (P)*.*

Proof. Axiom (A5) simplifies to (P) in the case of only two inputs. The existence of the required partial order on the set of all columns is guaranteed by Lemma 4. □

This suggests that the more restrictive condition (A5) may be a more natural condition for defining alignments of partially ordered sets. As a down-side, however, it seems that there is no convenient recursive construction of the search space similar to the dynamic programming approaches for sequence alignment. Instead, it seems more natural to treat this class of alignment problems as maximum induced subgraph problems.

Composition of binary relations is a powerful tool to construct multiple alignments. Suppose we are given a set of posets $(X_a, \prec_a)$ and a set $\mathcal{R}$ of pairwise relations satisfying (M) and (P) such that the graph representation of $\mathcal{R}$ is tree, then there is a unique multiple alignment satisfying (A5) obtained as the transitive closure of the graph on X with edges defined by the $R \in \mathcal{R}$. However, not every alignment can be represented in this manner. As a simple counterexample consider the alignment of the three sequences

```
a  A-C        a  A-C                a  A-C
b  -BC        b  -BC    b  -BC      b  -BC
c  AB-                  c  AB-      c A-B-
```

where the composition of any two pairwise alignments gives rise to two different columns for in/del columns of the pairwise components, in the example of two `A` entries. On the other hand the progressive approach, in which sequence `c` is aligned to the pairwise alignment of `a` and `b` yields the example alignment. In fact, Lemma 8 implies that in principle every alignment can be obtained by a progressive alignment scheme.

If $\mathcal{R}$ contains cycles, then there is no guarantee that the transitive closure $\widehat{A}$ of $\bigcup_{R \in \mathcal{R}} R$ is an alignment: In general, both conditions (A1) and (A2) will be violated. So-called *transitive alignment* approaches deliberately accept this at an intermediate stage. Various heuristics can be used to remove superfluous edges from the graph $(X, \widehat{A})$, that is they construct a subgraph (X, A), $A \subseteq \widehat{A}$ that again satisfies all conditions of a valid alignment.

7 Discussion

An interesting idea that follows quite naturally from the discussion above is a general approach towards graph comparison for sets of graphs: it seems natural to generalize the idea of progressive alignments in the following manner: (1) Given two graphs G_1 and G_2 and a common induced subgraph H (strictly speaking together with an embedding of H into G_1 and G_2) the graph defined by identifing the copies of H in G_1 and G_2 can be thought of as pairwise alignment. If G_1 and G_2 have vertex labels $\alpha_i : V(G_i) \to A_i$, $i = 1, 2$ for some alphabets A_i, on labels $G_1 \bullet_H G_2$ with label pairs $(\alpha_1(x), \alpha_2(x))$ for $x \in V(H)$, $(\alpha_1(x), -)$ for $x \in V(G_1 \setminus H)$ and $(-, \alpha_2)$ for $x \in V(G_2 \setminus H)$. Naturally, an optimization criterion such as “maximal common induced subgraph” will be used in practice. Since $G_1 \bullet_H G_2$ is again a (labeled) graph, the procedure can be repeated e.g. along a line of guidetrees. This gives raise to a natural notion of a multiple alignment of graphs G_a with vertex sets $V(G_a)$ and edge sets $E(G_a)$. Let $X = \dot{\bigcup} V(G_a)$ and A be a set of undirected edges on X and let $\mathcal{C}(X, A)$ be the set of connected components of the graph (X, A). Then (X, A, E^*) is a multiple alignment of the graph G_a, where E^* denotes the set of edges on $\mathcal{C}(X, A)$.

(G1) $Q \in \mathcal{C}(X, A)$ is complete subgraph of (X, A).

(G2) If $(a, i) \in Q$ and $(a, j) \in Q$, then $i = j$.

(G3) If $(a, i) \in P$, $(a, j) \in Q$ for some $P, Q \in \mathcal{C}(X, A)$ and $((a, i), (a, j)) \in E(G_a)$ then $(P, Q) \in E^*$

(G5) If $(P, Q) \in E$, $(a, i) \in P$, and $(a, j) \in Q$ then $((a, i), (a, j)) \in E^*$

The graph $(\mathcal{C}(X, A), E^*)$ can be constructed as the quotient graph $(X, A \cup \bigcup_a E(G_a))/\mathcal{C}(X, A)$ obtained by adding in all the edges of G_a and collapsing all columns (connected components) of (X, A) to a single vertex. A plausible generalization of (A4) might be to require

(G4) If $(P, Q) \in E^*$ then there is a row a with $(a, i) \in P$, $(a, j) \in Q$ and $((a, i), (a, j)) \in E(G_a)$,

i.e., (G3) completely determines the edges between alignment columns. In this setting (G4) does not impose additional conditions on the columns. However, if both the input graphs G_a and the alignment graph $(\mathcal{C}(X, A), E^*)$ are restricted to particular graph classes, such constraints appear. The graphs of partially ordered sets, i.e., the transitive acyclic digraphs discussed at length in the previous sections, of course, serve as a non-trivial example.

It is important to note the graph alignment in the sense used here – namely requiring a matching between vertices and notion of structural congruence between the alignment and its consitutent graphs – are more restrictive than some concepts of “graph alignments” discussed in the literature. In particular, we make a sharp distinction here between “graph alignments” and various approaches of comparison by means of graph editing, see e.g. [15] for a recent review.

The example of graph alignments and oriented tree alignments suggests to consider an even broader class of structures: Given a (finite) collection of sets X_a, each endowed with a set of relational structures (or more general set systems), we may ask for collections of partial maps between any pair of them that satisfy (G1) and (G2), i.e., define a partition on $\bigcup_a X_a$ such that each class contains at most one element of each X_a and the relation (or set system) structure is preserved in a sense similar to conditions (G3) and (G5) above. Such constructions are of practical interest e.g. for alignments of oriented trees (or forests) [4, 23, 27], where two distinct partial orders are defined, one capturing the order implied by parent-child relationship and another one representing the relative order of siblings. Alignments of ordered trees preserve both partial orders, since the alignment is defined as an ordered tree on the columns such that each ordered input tree (with vertices X_a) is obtained as a restriction to exactly the columns in which row a does not have a gap entry. Intuitively, this seems to require that (1) a super-object G exists for a pair of objects G_1 and G_2 such that G_1 and G_2 can be obtained as projections and (2) an intersection of the embeddings of G_1 and G_2 into G defines an induced sub-object H common to G_1 and G_2 is well defined. While the super-object corresponds to the alignments, the sub-object takes on the role of matches in the alignment. A similar notion of alignment is used in computational biology for RNA structures, where base pairs need to be preserved in addition the total order of the input sequences [41]. Here, however, only consistency similar in flavor to (A4) is enforced, suggesting that it may be of interest to relax the requirement of *induced* sub-objects.

The recursive formulation of the poset alignments is an extension of the well-known Needleman-Wunsch alignment algorithm. Beyond many implementations of the Needleman-Wunsch algorithm, the implementation based on `ADPfusion` (Algebraic Dynamic Programming with compile-time fusion of grammar and algebra) [24] is designed in a way to be extendable to different scoring functions, problem descriptions, and data structures [25]. Future work thus will include the adaptation of the `ADPfusion` framework written in a functional language (Haskell) to the data structure of posets. Earlier adaptations of the Needleman-Wunsch algorithm to trees, forests and sets already exist [4, 26].

It may also be possible to implement the poset alignment algorithm for the (A5)-notion of alignments in a way similar to the graph alignment algorithm above, i.e., starting from a maximal induced common subgraph that is then extended. Depending on the structure of the posets, this might be more efficient than the recursive DP algorithm where additional information has to be stored and updated in each step.

Finding maximal induced common subgraphs is well known to be a NP-complete problem. Nevertheless, DP algorithms have been devised for restricted settings such as planar graphs [18]. These proved practical for moderate size problems even though their resource requirements still scale exponentially.

For general graphs, there exist algorithms to detect common subgraphs [1]. However, as the problem is NP-complete, the problem can only be solved for small instances of the input structures in a reasonable amount of time. For small graphs such as representations of small chemical molecules, the DP algorithm might be able to solve the maximal common subgraph problem as described in [1]. Here, the DP algorithm is based on the analogous version for trees where the vertex degree has to be bounded in order to find a solution in a reasonable time frame. The algorithm divides the input structures in (overlapping) biconnected components and tries to find the best match between both input graphs preserving the order of the biconnected components of the original graph.

Finally, it seems natural to consider multiple alignments at a more abstract level: In order to properly define them, it seems sufficient that (induced) sub-objects can be used to "glue together" two (and recursively more) objects in such a way that the resulting super-object projects down to the given inputs. It is natural to ask how such structures can be characterized in the language of category theory. Is there an interesting class of categories that admit well-defined alignments objects, and do the resulting alignments themselves from categories with useful properties?

Acknowledgements

This work was supported in part by the German Academic Exchange Service (DAAD), proj.no. 57390771.

References

[1] Tatsuya Akutsu. A polynomial time algorithm for finding a largest common subgraph of almost trees of bounded degree. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 76(9):1488–1493, 1993.

[2] Stephen F Altschul, Thomas L Madden, Alejandro A Schäffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.*, 25:3389–3402, 1997. DOI: 10.1093/nar/25.17.3389.

[3] Shakuntala Baichoo and Christos A. Ouzounis. Computational complexity of algorithms for sequence comparison, short-read assembly and genome alignment. *Biosystems*, 156/157:72–85, 2017. DOI: 10.1016/j.biosystems.2017.03.003.

[4] Sarah J Berkemer, Christian Höner zu Siederdissen, and Peter F Stadler. Algebraic dynamic programming on trees. *Algorithms*, 10:135, 2017. DOI: 10.3390/a10040135.

[5] Tanmoy Bhattacharya, Damian Blasi, William Croft, Michael Cysouw, Daniel Hruschka, Ian Maddieson, Lydia Müller, Nancy Retzlaff, Eric Smith, Peter F. Stadler, George Starostin, and Hyejin Youn. Studying language evolution in the age of big data. *J. Language Evol.*, 3:94–129, 2018. DOI: 10.1093/jole/lzy004.

[6] Paola Bonizzoni and Gianluca Della Vedova. The complexity of multiple sequence alignment with SP-score that is a metric. *Theor. Comp. Sci.*, 259:63–79, 2001. DOI: 10.1016/S0304-3975(99)00324-2.

[7] H. Bunke. On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters*, 18:689–694, 1997. DOI: 10.1016/S0167-8655(97)00060-3.

[8] Humberto Carrillo and David Lipman. The multiple sequence alignment problem in biology. *SIAM J. Appl. Math.*, 48:1073–1082, 1988. DOI: 10.1137/0148063.

[9] Michael Cysouw and Hagen Jung. Cognate identification and alignment using practical orthographies. In *Proceedings of Ninth Meeting of the ACL Special Interest Group in Computational Morphology and Phonology*, pages 109–116. Association for Computational Linguistics, 2007. URL `https://www.aclweb.org/anthology/W/W07/W07-1314.pdf`.

[10] Chuong B Do, Mahathi SP Mahabhashyam, Michael Brudno, and Serafim Batzoglou. ProbCons: Probabilistic consistency-based multiple sequence alignment. *Genome Res.*, 15:330–340, 2005. DOI: 10.1101/gr.2821705.

[11] Richard Durbin, Sean R Eddy, Anders Krogh, and Graeme Mitchison. *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge University Press, Cambridge, UK, 1998.

[12] R C Edgar and S Batzoglou. Multiple sequence alignment. *Curr Opin Struct Biol*, 16:368–373, 2006. DOI: 10.1016/j.sbi.2006.04.004.

[13] Robert C Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.*, 32:1792–1797, 2004. DOI: 10.1093/nar/gkh340.

[14] Isaac Elias. Settling the intractability of multiple alignment. *J. Comp. Biol.*, 13: 1323–1339, 2006. DOI: 10.1089/cmb.2006.13.1323.

[15] Frank Emmert-Streib, Matthias Dehmer, and Yongtang Shi. Fifty years of graph matching, network alignment and network comparison. *Information Sci.*, 346/347: 180–197, 2016. DOI: 10.1016/j.ins.2016.01.074.

[16] Ronald Fagin, Ravi Kumar, and D. Sivakumar. Comparing top-k lists. *SIAM J. Discr. Math.*, 17:134–160, 2003. DOI: 10.1137/S0895480102412856.

[17] Da-Fei Feng and Russell F Doolittle. Progressive sequence alignment as a prerequisite to correct phylogenetic trees. *J. Mol. Evol.*, 25:351–360, 1987. DOI: 10.1007/BF02603120.

[18] Fedor V. Fomin, Ioan Todinca, and Yngve Villanger. Exact algorithm for the maximum induced planar subgraph problem. In Camil Demetrescu and Magnús M. Halldórsson, editors, *Proceedings of the 19th European conference on Algorithms*, volume 6942 of *Lecture Notes Comp. Sci.*, pages 287–298, Berlin, Heidelberg, 2011. Springer-Verlag.

[19] O Gotoh. An improved algorithm for matching biological sequences. *J. Mol. Biol.*, 162:705–708, 1982. DOI: 10.1016/0022-2836(82)90398-9.

[20] O. Gotoh. Alignment of three biological sequences with an efficient traceback procedure. *J. theor. Biol.*, 121:327–337, 1986. DOI: 10.1016/S0022-5193(86)80112-6.

[21] Manfred G. Grabherr, Pamela Russell, Miriah Meyer, Evan Mauceli, Jessica Alföldi, Federica Di Palma, and Kerstin Lindblad-Toh. Genome-wide synteny through highly sensitive sequence alignment: *Satsuma*. *Bioinformatics*, 26:1145–1151, 2010. DOI: 10.1093/bioinformatics/btq102.

[22] Catherine Grasso and Christopher Lee. Combining partial order alignment and progressive multiple sequence alignment increases alignment speed and scalability to very large alignment problems. *Bioinformatics*, 20:1546–1556, 2004. DOI: 10.1093/bioinformatics/bth126.

[23] Michael Höchsmann, Björn Voss, and Robert Giegerich. Pure multiple RNA secondary structure alignments: a progressive profile approach. *IEEE/ACM Trans. Comp. Biol. Bioinf.*, 1:53–62, 2004. DOI: 10.1109/TCBB.2004.11.

[24] Christian Höner zu Siederdissen. Sneaking around concatMap: Efficient combinators for dynamic programming. In *Proceedings of the 17th ACM SIGPLAN international conference on Functional programming*, ICFP '12, pages 215–226, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1054-3. DOI: 10.1145/2364527.2364559. URL `http://www.bioinf.uni-leipzig.de/Software/gADP/`.

[25] Christian Höner zu Siederdissen, Ivo L. Hofacker, and Peter F. Stadler. Product grammars for alignment and folding. *IEEE/ACM Trans. Comp. Biol. Bioinf.*, 12: 507–519, 2015. DOI: 10.1109/TCBB.2014.2326155.

[26] Christian Höner zu Siederdissen, Sonja J. Prohaska, and Peter F. Stadler. Algebraic dynamic programming over general data structures. *BMC Bioinformatics*, 16 Suppl 19:S2, 2015. DOI: 10.1186/1471-2105-16-S19-S2.

[27] Tao Jiang, Lusheng Wang, and Kaizhong Zhang. Alignment of trees – an alternative to tree edit. *Theor. Comp. Sci.*, 143:137–148, 1995. DOI: 10.1016/0304-3975(95)80029-9.

[28] Winfried Just. Computational complexity of multiple sequence alignment with SP-score. *J. Comp. Biol.*, 8:615–623, 2001. DOI: 10.1089/106652701753307511.

[29] Kazutaka Katoh, Kei-ichi Kuma, Hiroyuki Toh, and Takashi Miyata. MAFFT version 5: improvement in accuracy of multiple sequence alignment. *Nucleic Acids Res.*, 33: 511–518, 2005. DOI: 10.1093/nar/gki198.

[30] J. D. Kececioglu. The maximum weight trace problem in multiple sequence alignment. In *Proceedings of the 4th Symposium on Combinatorial Pattern Matching*, volume 684 of *Lecture Notes Comp. Sci.*, pages 106–119, Berlin, 1993. Springer.

[31] John Kececioglu and Dean Starrett. Aligning alignments exactly. In Philip E. Bourne and Dan Gusfield, editors, *Proceedings of the 8th ACM Conference on Research in Computational Molecular Biology (RECOMB)*, pages 85–96, New York, NY, 2004. ACM. DOI: 10.1145/974614.974626.

[32] A. S. Konagurthu, J. Whisstock, and P. J. Stuckey. Progressive multiple alignment using sequence triplet optimization and three-residue exchange costs. *J. Bioinf. and Comp. Biol.*, 2:719–745, 2004. DOI: 10.1142/S0219720004000831.

[33] Grzegorz Kondrak. A new algorithm for the alignment of phonetic sequences. In *Proceedings of NAACL 2000 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 288–295, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. DOI: 10.1.1.19.9698. URL `http://aclweb.org/anthology/A00-2038`.

[34] Matthias Kruspe and Peter F. Stadler. Progressive multiple sequence alignments from triplets. *BMC Bioinformatics*, 8:254, 2007. DOI: 10.1186/1471-2105-8-254.

[35] Mark A Larkin, Gordon Blackshields, N P Brown, R Chenna, Paul A McGettigan, Hamish McWilliam, Franck Valentin, Iain M Wallace, Andreas Wilm, Rodrigo Lopez, J D Thompson, T J Gibson, and D. G. Higgins. Clustal W and Clustal X version 2.0. *Bioinformatics*, 23:2947–2948, 2007. DOI: 10.1093/bioinformatics/btm404.

[36] Christopher Lee. Generating consensus sequences from partial order multiple sequence alignment graphs. *Bioinformatics*, 19:999–1008, 2003. DOI: 10.1093/bioinformatics/btg109.

[37] Christopher Lee, Catherine Grasso, and Mark F. Sharlow. Multiple sequence alignment using partial order graphs. *Bioinformatics*, 18:452–464, 2002. DOI: 10.1093/bioinformatics/18.3.452.

[38] David J Lipman, Stephen F Altschul, and John D Kececioglu. A tool for multiple sequence alignment. *Proc. Natl. Acad. Sci. USA*, 86:4412–4415, 1989. DOI: 10.1073/pnas.86.12.4412.

[39] Ketil Malde and Tomasz Furmanek. Increasing sequence search sensitivity with transitive alignments. *PloS one*, 8:e54422, 2013. DOI: 10.1371/journal.pone.0054422.

[40] Bodo Manthey. Non-approximability of weighted multiple sequence alignment. *Theor. Comp. Sci.*, 296:179–192, 2003. DOI: 10.1007/3-540-44679-6_9.

[41] Mathias Möhl, Sebastian Will, and Rolf Backofen. Lifting prediction to alignment of RNA pseudoknots. *J Comput Biol.*, 17:429–442, 2010. DOI: 10.1089/cmb.2009.0168.

[42] Burkhard Morgenstern. DIALIGN 2: improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics*, 15:211–218, 1999. DOI: 10.1093/bioinformatics/15.3.211.

[43] Burkhard Morgenstern, Jens Stoye, and Andreas W. M. Dress. Consistent equivalence relations: a set-theoretical framework for multiple sequence alignments. Technical report, University of Bielefeld, FSPM, 1999.

[44] Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 48: 443–453, 1970. DOI: 10.1016/0022-2836(70)90057-4.

[45] Cédric Notredame, Desmond G Higgins, and Jaap Heringa. T-coffee: a novel method for fast and accurate multiple sequence alignment. *Journal of molecular biology*, 302: 205–217, 2000. DOI: 10.1006/jmbi.2000.4042.

[46] Wolfgang Otto, Peter F. Stadler, and Sonja J. Prohaska. Phylogenetic footprinting and consistent sets of local aligments. In R. Giancarlo and G. Manzini, editors, *CPM 2011*, volume 6661 of *Lecture Notes in Computer Science*, pages 118–131, Heidelberg, Germany, 2011. Springer-Verlag. DOI: 10.1007/978-3-642-21458-5_12.

[47] Mikko Rautiainen and Tobias Marschall. Aligning sequences to general graphs in $O(V + mE)$ time. Technical report, bioRxiv, 2017.

[48] Nancy Retzlaff and Peter F. Stadler. Partially local multi-way alignments. *Math. Comp. Sci.*, 12:207–234, 2018. DOI: 10.1007/s11786-018-0338-4.

[49] David Sankoff and Joseph Kruskal, editors. *Time Warps, String Edits and Macromolecules: the theory and practice of Sequence Comparison.* Addison-Wesley, London, U.K., 1983.

[50] Temple F Smith and Michael S Waterman. Comparison of biosequences. *Adv. Appl. Math.*, 2:482–489, 1981. DOI: 10.1016/0196-8858(81)90046-4.

[51] Lydia Steiner, Peter F Stadler, and Michael Cysouw. A pipeline for computational historical linguistics. *Language Dynamics & Change*, 1:89–127, 2011. DOI: 10.1163/221058211X570358.

[52] Jens Stoye, Vincent Moulton, and Andreas W M Dress. DCA: an efficient implementation of the divide-and-conquer approach to simultaneous multiple sequence alignment. *Comput. Appl. Biosci.*, 13:625–626, 1997. DOI: 10.1093/bioinformatics/13.6.625.

[53] Jochen Tiepmar and Gerhard Heyer. An overview of canonical text services. *Linguistics Literature Studies*, 5:132–148, 2017. DOI: 10.13189/lls.2017.050209.

[54] Kavya Vaddadi, Naveen Sivadasan, Kshitij Tayal, and Rajgopal Srinivasan. Sequence alignment on directed graphs. Technical report, bioRxiv, 2017.

[55] Cristian A Velandia-Huerto, Sarah J Berkemer, Anne Hoffmann, Nancy Retzlaff, Liliana C Romero Marroquín, Maribel Hernández Rosales, Peter F Stadler, and Clara I Bermúdez-Santana. Orthologs, turn-over, and remolding of tRNAs in primates and fruit flies. *BMC Genomics*, 17:617, 2016. DOI: 10.1186/s12864-016-2927-4.

[56] L Wang and T Jiang. On the complexity of multiple sequence alignment. *J Comput Biol*, 1:337–348, 1994. DOI: 10.1089/cmb.1994.1.337.

[57] H T Wareham. A simplified proof of the NP- and MAX SNP-hardness of multiple sequence tree alignment. *J Comput Biol.*, 2:509–514, 1995. DOI: 10.1089/cmb.1995.2.509.

[58] J G Wolff. Syntax, parsing and production of natural language in a framework of information compression by multiple alignment, unification and search. *J. Universal Comp. Sci.*, 6(8):781–829, 2000. DOI: 10.3217/jucs-006-08-0781.