Compositional Properties of Alignments

Sarah J. Berkemer, Christian Höner zu Siederdissen and Peter F. Stadler

Abstract. 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-known 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. As a consequence of this discussion we observe that alignments of even more general structure, in particular graphs, are essentially characterized by the fact that the restriction of alignment to a row must coincide the the corresponding input graphs. Pairwise alignments of graphs are therefore determined completely by common induced subgraphs. In this setting alignments of alignments are well-defined, and alignments can be decomposed recursively into subalignments. This provides a general framework within which different classes of alignment algorithms can be explored for objects very different from sequences and other totally ordered data structures.

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 [13]. 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. [36, 10, 56, 6], as a convenient way of comparing ranked lists [19], for comparison of text editions [63, 58], and to analyse synteny in the comparison of genomes [24, 60].

Alignment problems are usually phrased as optimization problems. Most commonly a scoring model is defined for pairs of sequences and generalized to multiple alignments as sums over certain pairwise alignments that are obtained as projections. The pairwise scoring is usually specified either in terms of matches or in terms of edit operations (insertions, deletions, or substitutions). In this contribution, however, we will almost completely disregard the scoring of alignments and instead focus on the structure of (multiple) alignments as combinatorial objects. Our aim here is not to construct concrete alignment algorithms but the systematic generalization of alignments from string to more general discrete objects.

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:

```
gugugu--acgggcca        guguguac--gggcca
gucuguug--gggccc        gucugu--uggggccc        (1)
```

are alignments that are equivalent under most plausible scoring models. The idea to consider alignment columns as partial orders was explored systematically in [40] and a series of follow-up publications [39, 25]. 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 [40] is that a sequence of DAGs can be used as an input to a modified version of the Needleman-Wunsch algorithm [48]. Recently this idea has been generalized to the problem of aligning a sequence to a general directed graph [51, 59].

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 [47], which considers (pairwise) alignments as binary relations between sequence positions that represent matchings and preserve order. Here, we will make use of many of these ideas and show how they can be extended to a notion of alignments on partially ordered sets. We shall see that such a generalization still supports the recursive construction that underlies the exact dynamic programming algorithms employed to compute score-optimal alignments in the totally ordered case. Following our earlier work [50], we will use a language that is closer to graph theory than the presentation of [47, 46].

The notion of a composition of pairwise alignments – formalized as composition of partial maps that represent the matching – first appears in [42], see also Section 5. In the next two sections, we first review the sequence alignment problem and introduce a formal framework that separates the structure of multiple alignments from their scoring. In the following sections, we explore the consequence of relaxing some of the axioms to cover partial orders in general. Then we explore the compositional properties. Our main concerns are to ensure that alignments of alignments are well-defined as a foundation for progressive alignment procedures, and that decompositions into blocks exist that can form the basis of divide-and-conquer approaches to aligning partially ordered sets. Following a brief discussion of the view of alignments are compositions of pairwise matching relations, we further generalize the formalism to include first ordered trees, then directed and undirected graphs, and finally essentially arbitrary finite spaces that admit well-behaved subspace constructions. We shall conclude that alignments are alternatively specified in terms by common induced subgraphs (or the corresponding common induced subspaces in full generality).

2. A Very Brief Review of Sequence Alignments

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 [48, 54]. Since these algorithms have only quadratic space and time requirements for simple cost models [48, 22], they are of key importance in practical applications. The same recursive structure easily generalizes to alignments of more than two sequences [9, 41] even though the cost models need to be more restrictive to guarantee polynomial-time algorithms [34]. The computational effort for these exact solutions to the alignment problem increases exponentially with the number of sequences, hence only implementations for 3-way [23, 35, 37] and 4-way alignments [56] 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 [33, 61, 7, 31, 17] and MAX SNP-hard [62, 43]. The construction of practical multiple alignment algorithms therefore relies on heuristic approximations. These fall into several classes, see e.g. [14, 3] 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` [38]. The approach can be extended to starting from exact 3-way [35, 37] or 4-way alignments [56].

(2) *Iterative* methods starting to align small gapless subsequences and then extend and improve the alignment iteratively until the score converges. The iterative approach often used as a refinement step in combination with different other basics methods.

(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 the input alignments.

```
      (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 scoring model considering additive contributions of columns. In this example, we use a 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.

A paradigmatic example for the combination of consistency-based alignments and the iterative approach is DIALIGN [45] using additionally local motifs as anchors.

Most of the successful multiple alignment algorithms in computational biology combine these paradigms. For example T-COFFEE [49] and ProbCons [11] use consistency ideas in combination with progressive constructions; MUSCLE [15] and MAFFT [32] 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 [42] 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 [55] compared to the global Needleman-Wunsch [48] 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 [53].

3. Formal Definitions of Sequence Alignments

Suppose we are given a set S of $|S| \geq 1$ sequences with not necessarily equal length. For $s \in S$ we write s_i for the i-th position in s, and $|s|$ denotes the length of s, i.e., the number of positions. The most common representation of an alignment is as a rectangular matrix whose rows are indexed by the sequences and whose columns are index by integers $i \in [1, L]$, where L is the number of alignment columns. Each sequence is then associated with a strictly monotonically increasing function $\alpha_s : [1, |s|] \to [1, L]$ such that for each $i \in [1, |s|]$, $\alpha_s(i)$ is the index of the column containing s_i. The alignment matrix contains a gap symbol in row s and column k whenever $\alpha_s^{-1}(k) = \emptyset$, otherwise, the matrix element is $s_{\alpha_s^{-1}(k)}$. Consider the following simple example:

$$\begin{array}{ll} \texttt{a} & \texttt{0000111110000} \\ \texttt{b} & \texttt{000011011----} \\ \texttt{c} & \texttt{----100010000} \end{array} \tag{2}$$

We have $\alpha_A(i) = i$ for $1 \le i \le 13$, $\alpha_B(i) = i$ for $1 \le i \le 9$, and $\alpha_C(i) = i + 4$ for $1 \le i \le 9$. For the 10th column of the example we have $\alpha_a^{-1}(10) = 10$, $\alpha_b^{-1}(10) = \emptyset$, $\alpha_c^{-1}(10) = 6$; hence the entries in the 10th column are $a_{\alpha_a^{-1}(10)} = a_{10} = \texttt{0}$, $-$ because $\alpha_b^{-1}(10) = \emptyset$, and $c_{\alpha_c^{-1}(10)} = c_6 = \texttt{0}$. It will be convenient in the discussion below to also consider single sequences as (trivial) alignments, using the identity on $[1, |s|]$.

The actual values of the sequence elements, i.e., the s_i are of course important to determine the scoring. For our purposes, however, they are irrelevant, since we will only be interested in the structure of the alignments. It therefore suffices to consider the *sequence positions* $X_s := [1, |s|]$ for each input sequence and their arrangement in the alignment columns. This information is completely contained in the functions α_s. We can therefore "forget" about almost all the details about the sequence s except its length, which by construction satisfied $|X_s| = |s|$. From here on, we can therefore treat s simply as an index used solely to enumerate the elements of S. We will use the symbol $\bullet$ to indicate that a particular cell in the alignment matrix is occupied, while $-$ indicates gaps. The $\bullet$ eventually will become vertices in a graph representation.

For our purposes the set of sequence positions X_s is simple a finite ordered set. To emphasize this fact, and to make generalizations below more transparent, we write $(X_s, <_s)$ to explicitly expose the order relation on X_s. For a given set of sequences, furthermore, we will need the set of all sequence positions defined as the disjoint union $X := \biguplus_{s \in S} X_s$ of all sequence positions. The structure of an alignment with L columns is completely determined by the function $\omega : [1, L] \to \prod_{s \in S}(X_s \uplus \{-\})$ such that $\omega(k) = (\omega_s(k) | s \in S)$, $\omega_s(k) = -$ iff $\alpha_s^{-1}(k) = \emptyset$ and $\omega_s(k) = j \in X_s$ iff $\alpha_s(j) = k$. Thus ω plays the role of a (slightly modified) inverse of α: If $\alpha_s^{-1}(k) \neq \emptyset$, then $\omega_s(k) = \alpha_s^{-1}(k)$, while $\omega_s(k) = -$ if $\alpha_s^{-1}(k) = \emptyset$. It is customary, furthermore, to exclude alignment columns that consist entirely of gap symbols.

Definition 1. *An alignment on $X = \biguplus_{s \in S} X_s$ defined by ω is* proper *if there is no k such that $\omega_s(k) = -$ for all $s \in S$.*

Given ω, we construct a graph with vertex set $X = \biguplus_{s \in S} X_s$ and edge set A such that $xy \in A$ if there is $k \in [1, L]$ and distinct sequences s and t such that $\omega_s(k) = x$ and $\omega_t(k) = y$. In other words, positions x and y are joined by an edge if and only if x and y appear in the same column of the alignment. We call this graph the *alignment graph.*

Lemma 2. *Consider an alignment on $X := \dot{\bigcup}_{s\in S} X_s$ determined by ω. Suppose $x \in X_r$, $y \in X_s$, and $z \in X_t$ and both xy and xz are edges in the alignment graph. Then*

(i) *r, s, and t are pairwise distinct.*
(ii) *yz is also an edge in the alignment graph.*

Proof. Property (i) following immediately from the requirement that α_s is strictly monotonically increasing, i.e., any two positions of the same sequence are mapped to distinct alignment columns. Property (ii) follows directly from the definition. If xy and xz are edges, then x, y, and z are located in the same alignment column and thus yz is an edge of the alignment graph. □

The alignment graph therefore is the disjoint union of complete graphs such that every connected component (which is a clique) contains at most one element of each of the input sequences X_s. Every clique thus corresponds to an alignment column. We write $\mathcal{C}(X, A)$ for the set of alignment columns, which for convenience we identify with their vertex sets. More precisely, $Q \in \mathcal{C}(X, A)$ is an alignment column if and only if there is $k \in [1, L]$ such that $x \in Q$ if and only if $x = \omega_s(k) \neq -$ for some $s \in S$. In particular, for each $s \in S$ we have either $Q \cap X_s = \emptyset$ or $Q \cap X_s = \{\omega_s(k)\}$.

The alignment graph is consistent with the input orders $<_s$ on X_s, $s \in S$ in the following sense:

Lemma 3. *Let Q' and Q'' be two distinct connected components of the alignment graph with vertex set X determined by ω and suppose there is $s, t \in S$ such that $x_s \in Q' \cap X_s$, $y_s \in Q'' \cap X_s$, $x_t \in Q' \cap X_t$, and $y_t \in Q'' \cap X_t$. Then $x_s <_s y_s$ if and only if $x_t <_t y_t$.*

Proof. By Lemma 2, two vertices x_s and x_t are in the same connected component Q if and only if they are in the same column, i.e., if $\alpha(x_s) = \alpha(x_t) =: k'$. Analogously, $\alpha(y_s) = \alpha(y_t) =: k''$. By monotonicity of α_s and α_t, we therefore have $x_s <_s y_s$ if and only if $k' < k''$, which in turn is true if and only if $x_t <_t y_t$. □

In particular, we may conclude:

Observation 1. *Consider an alignment on X determined by ω. Then there exists an order on the alignment columns such that $Q' <_A Q''$ implies $x <_s y$ whenever $x \in Q' \cap X_s$ and $y \in Q'' \cap X_s$.*

Proof. The by construction, the alignment columns are ordered. Lemma 3 implies that this order is consistent with the order $<_s$ of each X_s. □

In the following it will be convenient to write each element of X as a pair that explicitly specifies the input sequence from which it derives. That is, we write $(a, i) \in X$ for $i \in X_a$ and $a \in S$.

The simple observations of this section suggest to *define* an alignment by means of an alignment graph with a suitable order of the columns. The following definition rephrases the approach taken e.g. in [57, 47, 46] in a form that will be most convenient for further generalizations:

Definition 4 (Total Alignment [50]**).** *A* total alignment *of a finite collection of finite totally ordered sets* $(X_s, <_s)$*,* $s \in S$*, is a triple* $(X, A, <)$ *where* $X := \dot{\bigcup}_{s\in S} X_s$*,* (X, A) *is an undirected, loop-free graph with vertex set* X *with* $\mathcal{C}(X, A)$ *being the set of its connected components, and* $<$ *is a total order relation on* $\mathcal{C}(X, A)$ *such that the following conditions are satisfied.*[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* $i <_a k$ *then* $j <_b l$*.*
(5) *If* $(a, i) \in P$*,* $(a, j) \in Q$ *and* $(a, i) <_a (a, j)$ *then* $P < Q$*.*

As above, the connected components of the alignment graph (X, A) play the role of 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. Condition (5) ensures that the restriction of order on the columns to each row recovers the order $(X_a, <_a)$. A bit more formally, we may phrase this as follows:

Observation 2. *Let* $(X, A, <)$ *be an alignment,* $P, Q \in \mathcal{C}(X, A)$*,* $P \cap X_a = \{(a, i)\}$*, and* $Q \cap X_a = \{(a, j)\}$*. Then* $P < Q$ *if and only if* $(a, i) <_a (a, j)$*.*

A well known observation in the theory of alignments is that Conditions (4) and (5) in general only specify a partial order but not a total order of the alignment columns:

Lemma 5. *Let* (X, A) *be an alignment graph and denote by* $\prec$ *the relation defined for all* $P, Q \in \mathcal{C}(X, A)$ *by* $P \dot{\prec} Q$ *whenever there is an* $a \in S$ *such that* $(a, i) \in P$*,* $(a, j) \in Q$ *and* $i < j$*. Then the transitive closure* $\prec$ *of* $\dot{\prec}$ *is a partial order on* $\mathcal{C}(X, A)$*.*

Proof. By construction, $\dot{\prec}$ is antisymmetric. By definition $P \prec Q$ if and only if there is a sequence of columns $P = Q_0 \dot{\prec} Q_1 \dot{\prec} \dots \dot{\prec} Q_k = Q$. 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 $\prec$ is still antisymmetric. Thus $\prec$ is 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 $\prec$, which by construction satisfies

$$P < Q,\ (a, i) \in P, \text{ and } (a, j) \in Q \quad \text{implies} \quad i < j. \tag{3}$$

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

We summarize this reasoning in

Theorem 6. *Let (X, A) be an alignment graph for $X = \dot{\bigcup}_{s\in S} X_s$ and conditions (1), (2), and (4) of Definition 4 are satisfied. Then there exists a total order $<$ on $\mathcal{C}(X, A)$ satisfying condition (5), i.e., such that $(X, A, <)$ is a total alignment.*

Theorem 6 provides the justification for considering alignment graphs with ordered columns instead of the matrix representation defined by ω. Obviously $(X, A, <)$, or more precisely the order $<$ of the alignment columns completely defines α, ω, and L *provided* we require that there are no alignment columns consisting entirely of gap symbols.

Before we proceed, a few remarks are in order: In this setting the actual data associated with the sequence element (a, i), whether it is simply the i-th letter of input sequence a or an extensive entry at position i of the list a, is treated as a label that influences only the scoring but not the structure of the alignment. This separation between the underlying (index) structure and the data associated with them is also used in algebraic dynamic programming approaches to alignments [29, 5], where the structure of the recursions depends only on the possible alignments $(X, A, <)$ for a given set X, while the scoring depends on the labeling of X. 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, i.e., every unaligned position is considered as an insertion relative to all other elements of X. Whether a position is aligned or unaligned affects only the scoring, hence at the level of alignment graphs we do not need to concern ourselves with a distinction of local, partially local, and global alignments.

4. Alignments of Partially Ordered Sets

Since the alignment of totally ordered sets in general only specifies a partial order of columns but not a total order, it seems natural to ask whether the concept of alignments and alignment graphs can be extended to partial orders instead of total orders and inputs. From here, one therefore considers a collection of finite partial orders $(X_a, \prec_a)$, $a \in S$, $|S| \geq 1$. As a generalization of Def. 4 we consider

Definition 7 (PO Alignment). *A partial order (PO) 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 $(a, i) \in P$, $(a, j) \in Q$ for some $P, Q \in \mathcal{C}(X, A)$ 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. $\prec_a$.*

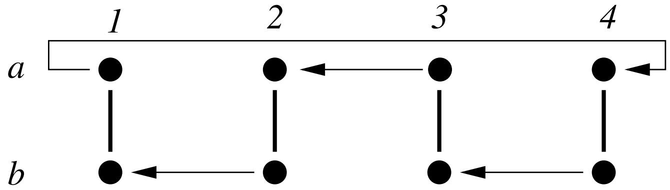

FIGURE 2. Property (A4*) is not sufficient to ensure the existence of a partial order $\prec$ on $\mathcal{C}(X,A)$. Consider the partial orders $(a,4) \prec_a (a,1)$ and $(a,2) \prec_a (a,3)$ and $(b,1) \prec_b (b,2)$ and $(b,3) \prec_a (b,4)$, with alignment columns $\{(a,i),(b,i)\}$ for $i = 1,2,3,4$.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

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.

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 4 and 7 therefore coincide for totally ordered rows.

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. $\prec_b$ $\forall P,Q \in \mathcal{C}(X,A)$.

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 the alignment columns $\mathcal{C}(X,A)$ as an extra condition in Definition 7.

The existence of (non-trivial) alignments of any collection of finite partial orders $(X_s, \prec_s)$, $s \in S$, is easy to see: each of the partial orders can be linearly extended to a total order $(X_s, <_s)$. 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 5.

Before we proceed we briefly remark that at the level of our discussion we do not need to concern ourselves with the distinction of global and local alignments. In order to model a partially local alignment of posets 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 $W \not\preceq V$ and $V \not\preceq U$, i.e., no "unaligned" suffix column precedes 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 again no structural difference between local and global alignments.

The projection of $(X, A, \prec)$ onto a row $a \in S$ is obtained as the set $\pi_a(X) := \{(a,i) \in X_a | \exists Q \in \mathcal{C}(X,A) : (a,i) \in Q\}$ endowed with the partial order $\prec_a^\pi$ such that $(a,i) \prec_a^\pi (a,j)$ whenever there are columns $P, Q \in \mathcal{C}(X,A)$ with $P \prec Q$. A potential shortcoming of Def. 7 is that is does not guarantee that $(\pi_a(X), \prec_a^\pi) = (X_a, \prec_a)$. It is therefore of interested to to consider a (much) stronger version of axiom (A4):

(A5) $P \prec Q$, $(a,i) \in P$ and $(a,j) \in Q$ implies $(a,i) \prec_a (a,j)$; $\forall P, Q \in \mathcal{C}(X,A)$.

First we note that (A5) implies (A4). The definition of the projection of $(X, A, \prec)$ to a row $a \in S$ then immediately implies

Observation 3. *Suppose a $(X, A, \prec)$ satisfies (A1), (A2). Then (A5) is equivalent to $(\pi_a(X), \prec_a^\pi) = (X_a, \prec_a)$.*

Observation 2 furthermore implies that (A4) and (A5) are equivalent if all $(X_a, \prec_a)$ are totally ordered. In general this is not the case, however, as the example in Fig. 3 shows.

The following simple, technical result is a generalization of Lemma 5, showing that condition (A5) is sufficient to guarantee the existence of a partial order on the columns.

Lemma 8. *Let (X, A) be a graph with connected components $\mathcal{C}(X,A)$ satisfying* (A1) *and* (A2)*. Let $\dot{\prec}$ denote the transitive closure of the relation $\prec$ 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 $P \prec Q$; $\forall P, Q \in \mathcal{C}(X,A)$. Finally assume that axiom* (A5) *holds. Then $\prec$ is a partial order on $\mathcal{C}(X,A)$*

Proof. It suffices to show that $\dot{\prec}$ is antisymmetric. It is clear from the construction that by (A5) we know that $\prec$ is antisymmetric. If $\dot{\prec}$ 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 $(a_i, h) \prec (a_i, h')$ from $(a_i, h) \dot{\prec} (a_i, h')$ and $(a_i, h') \prec (a_i, h)$ by going around the cycle, contradicting axiom (A5). □

Finite partial orders $(X_a, \prec_a)$ are equivalent to finite directed transitive acyclic graphs. The projection property of Observation 3, can be expressed in graph-theoretical terms in the following manner:

Observation 4. *Let $(X, A, \prec)$ be an alignment of partial orders $(X_a, \prec_a)$, $S' \subseteq S$ a subset of columns, and $\mathcal{Q}' \subseteq \mathcal{C}(X,A)$ such that $X_a \cap Q \neq \emptyset$ for all $a \in S'$ and $Q \in \mathcal{Q}'$. Then the graph with vertex set $\mathcal{Q}'$ and directed edges whenever $P \prec Q$ is an induced subgraph of (the graph representation of) $(X_a, \prec_a)$.*

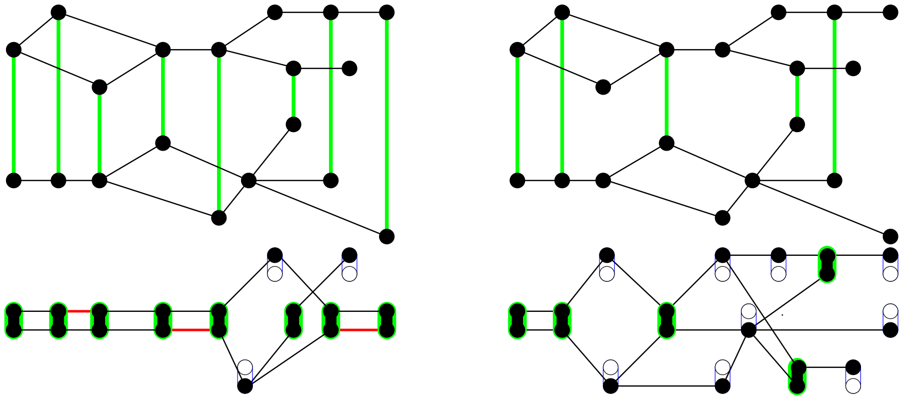

Figure 3. **Top:** Pairwise alignments of partially ordered sets. Thin black edges show the Hasse diagram, to be read from left to right. Alignment edges are shown in green.
Bottom: The induced partial order of the alignment columns with corresponding points vertically aligned. The partial order is again shown as a Hasse diagram, with superfluous 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.

Thus the set of alignment columns $\mathcal{Q}'$ defines an *induced common subgraph* of the transitive acyclic graphs $(X_a, \prec_a)$ in $a \in S'$. This is of course also true for pairwise alignments. In the pairwise case, none of the columns $Q \in \mathcal{C}(X, A) \setminus \mathcal{Q}'$ describe a (mis)match, i.e. they contain only insertions and deletions, while all $Q \in \mathcal{Q}'$ describe (mis)matches. A score-optimal alignment of two partial orders therefore corresponds to a maximal induced common subgraph of two transitive acyclic graphs. In both specifications of the problem, the scoring function will of course depend on the labels. We refer to [8] for a discussion of the relationships of edit distances and maximum common subgraph problems in a more general setting.

5. Composition of Alignments

In order to study the composition of alignments it seems natural to first consider the properties of parts of given alignments. The most natural starting point is to consider restrictions induced by considering subsets of the input sequences. The following result, which generalizes Lemma 1 of [50], provides a convenient starting point.

Lemma 9. *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 $Q \cap Y$ $\forall Q \in \mathcal{C}(X, A)$ 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 column 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 [30, 26, 5], thus imposing additional restrictions on valid alignments. We will return to this point in some generality in the discussion section.

The fact that alignments are again totally or partially ordered sets implies that one can also meaningfully define alignments of alignments. As before, we start from a collection of finite partial orders $(X_a, \prec_a)$, $a \in S$. Let $\mathfrak{P}$ be a non-trivial partition of the rows, i.e., of S, whose classes we will write as S_α indexed by α. We write $X_\alpha := \biguplus_{a \in S_\alpha} X_a$. By construction, $X_\alpha \cap X_\beta = 0$ for $\alpha \neq \beta$, i.e., the site sets of the row classes are disjoint. The row partition $\mathfrak{P}$ thus implies a partition of X.

Lemma 10. *Let $(X, A, \prec)$ be an alignment of the $(X_a, \prec_a)$, $a \in S$, $\mathfrak{P}$ be a non-trivial partition of S, $X_\alpha := \biguplus_{a \in S_\alpha} X_a$ the site set of the row calls α and $(X, A, \prec)[X_\alpha]$ the corresponding sub-alignment of $(X, A, \prec)$. Then $(X, A, \prec)$ is isomorphic to the (vertex) disjoint union of the $(X, A, \prec)[X_\alpha]$ for all row classes α, augmented by*

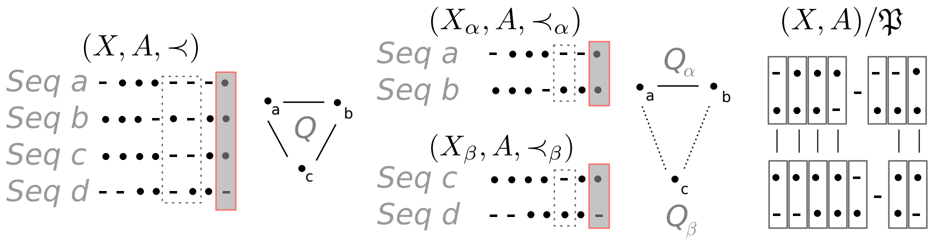

FIGURE 4. Example of an alignment (left) being composed out of sub-alignments (middle) corresponding to the partition of the rows in the two classes $S_\alpha = \{a, b\}$ and $S_\beta = \{c, d\}$. Columns marked by dashed lines show how the creation of sub-alignments removes gap columns. Column $Q \in \mathcal{C}(X, A)$ (marked in grey) is highlighted as an example. On the right, the two sub-alignments with the corresponding restrictions of Q are shown: $Q_\alpha \in \mathcal{C}(X, A)[X_\alpha]$ and $Q_\beta \in \mathcal{C}(X, A)[X_\beta]$ are connected components and complete subgraphs of the sub-alignment graphs and can be composed to Q by applying the disjoint union and adding extra edges between all elements in Q that are in distinct sub-alignments thus Q_α and Q_β (dashed lines). Indices at nodes in the graph refer to the sequence the node is coming from. The alignment $(X, A)/\mathfrak{P}$ on the right is the alignment of the sub-alignments (X_α, A) and (X_β, A). Thus the nodes in the alignment graph are columns of the sub-alignments. Alignment edges show matched columns. The unmatched columns correspond to the columns marked by dashed lines in the alignments on the left and middle.

extra edges (x', x'') whenever there is a column $Q \in \mathcal{C}(X, A)$ with $x' \in Q \cap X_\alpha$ and $x'' \in Q \cap X_\beta$ for classes $\alpha \neq \beta$.

Proof. By Lemma 9, the alignments $(X, A, \prec)[X_\alpha]$ subalignments of $(X, A, \prec)$ and thus $(X, A)[X_\alpha]$ is an induced subgraph of (X, A). Their disjoint union therefore lacks exactly all edges that connect pairs of vertices that are in the same connected component of (X, A) but do not below to the same class of rows α. Since the partial order on the columns of $(X, A)[X_i]$ is the one inherited from $(X, A, \prec)$, the re-composition of the columns also recovers the original partial order. □

A corresponding example is shown in Fig. 4 where the alignment $(X, A, \prec)$ is composed out of sub-alignments $(X, A, \prec)[X_\alpha]$ and $(X, A, \prec)[X_\beta]$ with Q, Q_α, and Q_β as examples for connected components of the alignment graphs and their composition by disjoint union and extra edges (dashed lines) between elements of distinct sub-alignments.

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

Definition 11. *We denote by $(X, A)/\mathfrak{P}$ the quotient graph whose vertices are the columns of the induced sub-graphs $(X, A)[X_\alpha]$, that is, the non-empty sets $Q \cap X_i$ where Q is a connected component of (X, A). Its edges are the pairs $(Q \cap X_\alpha, Q \cap X_\beta)$ for which both $Q \cap X_\alpha$ and $Q \cap X_\alpha$ are non-empty.*

The connected components of the graph $(X, A)/\mathfrak{P}$ are therefore of the form $Q' := Q/\mathfrak{P} = \{Q \cap X_\alpha | Q \cap X_\alpha \neq \emptyset\}$. Note that Q' is non-empty since the column Q of (X, A) contains at least one element, which belongs to $(X, A)[X_\alpha]$ for at least one of the classes α of $\mathfrak{P}$. Thus there is a 1-1 correspondence between the connected components of (X, A) and those of $(X, A)/\mathfrak{P}$. The columns of $(X, A)/\mathfrak{P}$ therefore 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 12. *$(X, A, \prec)/\mathfrak{P}$ 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 correspondence 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 $(X, A)[X_\alpha]$ for each of the row classes α. □

Theorem 13. *Let $(X, A, \prec)$ be an alignment and let $\mathfrak{P}$ be an arbitrary row partition. Then $(X, A, \prec)$ is isomorphic to the alignment $(X, A, \prec)/\mathfrak{P}$ of its restrictions $(X, A, \prec)[X_\alpha]$ to the row classes α of $\mathfrak{P}$.*

Proof. Since $(X, A, \prec)/\mathfrak{P}$ is well defined by Lemma 12, Lemma 10 shows that expanding the classes points of $(X, A, \prec)/\mathfrak{P}$ into corresponding sets Q_α building the union of those the belong the a column of $(X, A, \prec)/\mathfrak{P}$ exactly recovers the columns of $(X, A, \prec)$ and their partial order. □

We note that the constituent alignments $(X_\alpha, A_\alpha, \prec_\alpha) := (X, A, \prec)[X_\alpha]$ have at most the same number of columns since "all gap" columns, $Q' = Q \cap X_i = \emptyset$, are removed. The decomposition of Theorem 13 can be applied recursively until each constituent alignment is one of the input posets $(X_a, \prec_a)$, $a \in S$. Any such recursive composition is naturally represented as a rooted tree $\mathfrak{T}$. The leaves of $\mathfrak{T}$ are the input posets $(X_a, \prec_a)$, while the root represents $(X, A, \prec)$. Each internal node of $\mathfrak{T}$ corresponds the an alignment of its children. In particular, one can choose $\mathfrak{T}$ to be any binary tree.

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 constituents $(X_\alpha, A_\alpha, \prec_\alpha)$, or, in other words, if $(X, A, \prec)$ is score-optimal, there is no guarantee that there is any nontrivial partition of the rows $\mathfrak{P}$ such that all the restrictions $(X, A, \prec)[X_i]$ are score-optimal subalignments. Progressive alignment methods thus cannot guarantee an exact solution of the multiple alignment problem. Results in practical applications depend substantially on the choice of the guide tree $\mathfrak{T}$. It is has been suggested early [20], 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 larger sub-alignments.

6. Blockwise Decompositions

On the other hand, we can also decompose alignments into blocks of columns. More precisely, we consider an alignment $(X, A, \prec)$ and a partition $\mathfrak{Q} := \{Y_1, \dots, Y_q\}$ satisfying the following properties:

(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 $Y_k, Y_l \in \mathfrak{Q}$ such that $Y_k \lhd Y_l$ whenever there are columns $P \in Y_k$ and $Q \in Y_l$ with $P \prec Q$.

We call the classes of such a partition *blocks*. By Lemma 9 each block $(X, A, \prec)[Y_k]$, $Y_k \in \mathfrak{Q}$ is again an alignment.

Theorem 14. *Given blocks $(X, A, \prec)[Y_k]$ with $Y_k \in \mathfrak{Q}$, and the partial order $\lhd$ on the blocks, there is an alignment $(X, A, \prec_\lhd)$, where $\prec_\lhd$ is an an extension of $\prec$ defined by $P \prec_\lhd Q$ if and only if $P \prec Q$ for $P, Q \in Y$ for some $Y \in \mathfrak{Q}$ and $P \prec_\lhd Q$ for $P \in Y_k$ and $Q \in Y_l$ with $Y_k \lhd Y_l$ and $k, l \in (1, q)$.*

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 preserved inside the alignment blocks $Y_k \in \mathfrak{Q}$ 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_k and Y_l with at least one column in each block such that $P \in Y_k$ and $Q \in Y_l$. Based on the decomposition of Y into Y_k and Y_L one can restore the order of the alignment blocks such that $Y_k \lhd Y_l$ based on Y. Thus, one gets the order of $P \prec_\lhd Q$ that is present for the alignment columns P and Q as well as for the alignment blocks Y_k and Y_l. □

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 the choice of blocks in such a way that $\lhd$ just "mirrors" the initial partial order, i.e., $Y_k \lhd Y_l$ if and only if $P \prec Q$ for

P in Y_k and Q in Y_l, in which case $\prec_{\lhd} = \prec$ 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` [57], 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` [45] 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.

7. Recursive Construction

Given a poset $(Y, \prec)$ 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.

For simplicity of exposition, we first consider the pairwise case, i.e., the set of alignments of two finite posets $(X_1, \prec_1)$ 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 [48] 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 15. *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 $(X_a, \prec_a)$. Note that the recursion of Thm. 15 can be transformed into an exact dynamic programming algorithm for alignment of posets, provided the scoring function is the sum of column-wise contributions.

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-trivial 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 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 [40] 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 poset and implies that, given any match $\binom{u}{v}$ in the alignment, all preceding 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.

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) [27] is designed in a way to be extendable to different scoring functions, problem descriptions, and data structures [28]. 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 [5, 29].

8. 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 defined by

$$(x,z) \in S \circ R \quad \text{iff} \quad \exists y \in Y \text{ s.t. } (x,y) \in R \text{ and } (y,z) \in S \tag{4}$$

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 $x = y$.

(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 16. *The composition of two binary relations satisfying (M) and (P) is again a binary relation satisfying (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 17. *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). □

Theorem 18. *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 8. □

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 12 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.

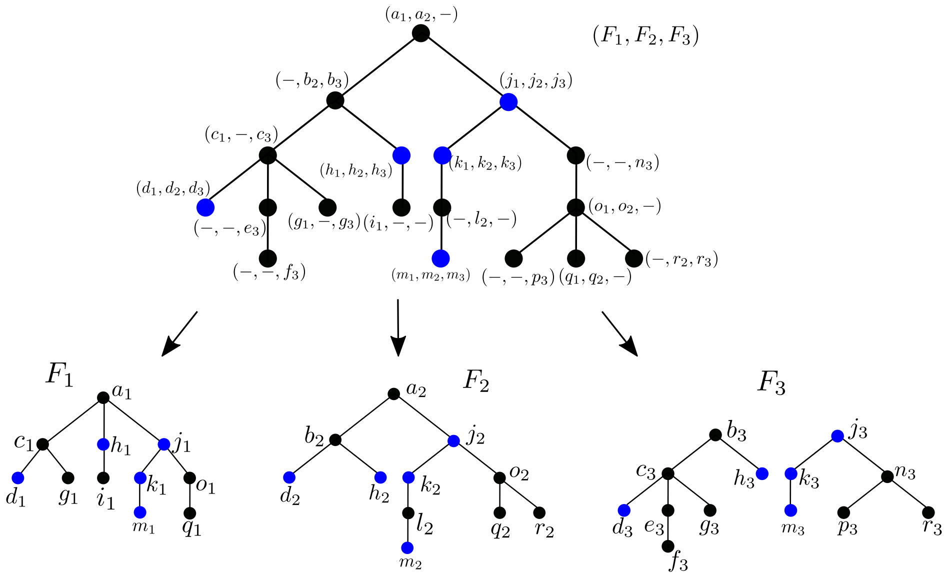

FIGURE 5. Example of a forest alignment of three forests (bottom). The resulting forest (top) is the superstructure combining all of the input trees. The nodes labels correspond to alignment columns and blue nodes indicate matches such that they exist in all the input trees. Original trees can be recovered from the supertree by only taking nodes without gap symbol in the corresponding alignment column. A node with gap symbol is then removed and its edges contracted such that its children will be the its parents children afterwards. This can be seen in F_1 where node b does not exist and nodes c_1 and h_1 become children of the root a_1.

9. Tree Alignments

A rooted tree with vertex set V is uniquely defined by two mutually exclusive partial order relations: the *ancestor order* $\prec$ defined such that $x \preceq y$ whenever y is located on the path from x to the root, and the *sibling order* $\lhd$ defined in terms of the ordering of the children of each vertex: For two vertices x and y that are incomparable w.r.t. $\prec$, let w be their last common ancestor and u and v be the distinct children of w such that $x \preceq u$ and $y \prec v$. Then $x \lhd y$ if and only if $u \lhd v$. By construction, two vertices are either identical or comparable w.r.t. either the ancestor or the sibling order. The observation extends to ordered forests, where the sibling order is extended such that vertices from any two constituent subtrees are are always comparable w.r.t. the sibling order.

Consider a forest T with vertex set V and define T_v with vertex set $V\setminus\{v\}$ as follows: (1) if v is the root of a subtree, delete v and replace the tree $T(v)$ rooted at v by trees rooted at the children of v in sibling order; (2) if v is not the root of a subtree, contract the edge from the parent of v to v. That is, the children of v become children of the parent of v. It is not hard to check that both the ancestor and sibling orders for T_v is simply the restriction of $\prec$ and $\lhd$ to $V \setminus \{v\}$.

A forest alignment is defined as a forest T such that each vertex v is labeled by an alignment column Q_v. The constituent tree T_s, $s \in S$ is obtained from T by first simplifying the label on T to $Q_v \cap X_s$ at each vertex v; then all v with $Q_v \cap X_s = \emptyset$ are removed by deletion or contraction of their parent edge as outlined above [30, 26, 5]. Thus T_s has the vertex set $V' := \{v \in V | Q_v \cap X_s \neq \emptyset\}$ and both its ancestor and sibling orders are the restriction of $\prec$ and $\lhd$ to V'. Tree or forest alignments thus fit seamlessly into the mathematical formal for partial order alignments. We simply have to require that the alignment graph (X, A) satisfies (A1) and (A2) and that properties (A3) and (A5) hold w.r.t. *both* partial orders $\prec$ and $\lhd$. This observation suggest how alignments satisfying an analog of (A5) can be defined in a meaningful way for a much broader class of discrete structures.

A notion of alignment similar to tree/forest alignments is used in computational biology for RNA structures, where base pairs need to be preserved in addition the total order of the input sequences [44]. Here, however, only consistency similar in flavor to (A4) is enforced, suggesting that it may also be of interest to relax the requirement that restriction to the columns Q for which $Q \cap X_a \neq \emptyset$ exactly recovers the input tree $(X_a, \prec_a, \lhd_a)$.

10. Alignments of Graphs

In Section 4 we have seen that alignments of partially ordered sets can alternatively be viewed as alignments of graphs from a very restricted class, namely transitive acyclic digraphs. This begs the question whether the construction can be generalized to arbitrary (di)graphs. In this section we consider an input set of digraphs G_a, $a \in S$, with with vertex sets $V(G_a) = X_a$ and edge sets $E(G_a)$, resp. As before, we write $X = \biguplus V(G_a)$, introduce a set of alignment edges A, and denote by $\mathcal{C}(X, A)$ be the set of connected components of the (undirected) graph (X, A).

Definition 19. *A triple (X, A, E^*), where A is a set of unordered pairs on X and E^* is a relation on $\mathcal{C}(X, A)$, is a multiple alignment of the graphs G_a, $a \in S$, where A if the following conditions are satisfied:*

(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 $(P, Q) \in E^*$*
(G4) *If $(P, Q) \in E^*$ then there is a row a with $(a, i) \in P$, $(a, j) \in Q$ and $((a, i), (a, j)) \in E(G_a)$,*
(G5) *If $(P, Q) \in E^*$, $(a, i) \in P$, and $(a, j) \in Q$ then $((a, i), (a, j)) \in E(G_a)$.*

Condition (G4) is redundant and is included here only to emphasize the similarity to the constructions in the previous sections. It may also be interesting to consider graph alignments that satisfy only (G4) but not (G5).

Lemma 20. *$(\mathcal{C}(X,A),E^*) \simeq (X,\bigcup_{a\in S} E(G_a))/\mathcal{C}(X,A)$.*

Proof. The vertex set $X/\mathcal{C}(X,A)$ has a single representative for each column $Q \in \mathcal{C}(X,A)$. By axioms (G3) and (G5), there is an edge $(P,Q) \in E^*$ if and only there $(a,i) \in P$ and $(a,j) \in Q$ with $((a,i),(a,j)) \in E(G_a)$ for some $a \in S$. The edge set on the r.h.s., amounts to identical condition. □

Thus $(\mathcal{C}(X,A),E^*)$ is obtained from $(X,\bigcup_{a\in S} E(G_a))$ by identifying the vertices within each alignment column. In particular, therefore, the set $\mathcal{Q}'$ of columns Q such that $Q \cap X_a \neq \emptyset$ for all a in a given subset $S' \subseteq S$ forms an induced subgraph $(\mathcal{C}(X,A),E^*)$ that is present in each G_a. Observation 5 thus remains true for graphs in general:

Observation 5. *Let (X,A,E^*) be an alignment of graphs (X_a,E_a), $S' \subseteq S$ a subset of columns, and $\mathcal{Q}' \subseteq \mathcal{C}(X,A)$ such that $X_a \cap Q \neq \emptyset$ for all $a \in S'$ and $Q \in \mathcal{Q}'$. Then the graph with vertex set $\mathcal{Q}'$ and edge set E^* is an induced subgraph of (the graph representation of) (X_a,E_a).*

We note in passing that alignments of ordered and partially ordered sets assuming axiom (A5) are special cases of the graph alignments satisfying (G5), since total and partial orders are isomorphic to transitive acyclic digraphs. One easily checks that (G3) and (G5) indeed reduce to the corresponding statements for the (partial) orders.

Again this is in particular true for pairwise alignments. 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 identifying the copies of H in G_1 and G_2 is pairwise alignment $G_1 \bullet_H G_2$ of the input graphs. Naturally, an optimization criterion will be used in practice. The problem of aligning graphs therefore coincides with the maximum common induced subgraph problem. Finding maximal induced common subgraphs (MCIS) is well known to be a NP-complete problem and closely related to the maximal common edge subgraph problem (MCES), together often referred to as the maximal common subgraph problem (MCS) [16, 12]. However, several approaches exist to find exact or approximate solutions for connected (cMCS) or disconnected (dMCS) common subgraphs using different algorithmic strategies such as backtracking algorithms, dynamic programming, or clique-finding.

It is very easy to check that Lemmas 9, 10 and 12 – and thus also Thm. 13 – remain true for the graph alignments of Definition 19. Indeed, the alignment of two graphs is again a graph. Its vertices, corresponding to the columns of the alignment, are labeled by the content of the columns. Therefore, we can build alignments of alignments for graphs. In particular, furthermore, progressive alignments of graphs are well-defined. Given a guide tree $\mathcal{T}$, at each inner node of $\mathcal{T}$ the maximum

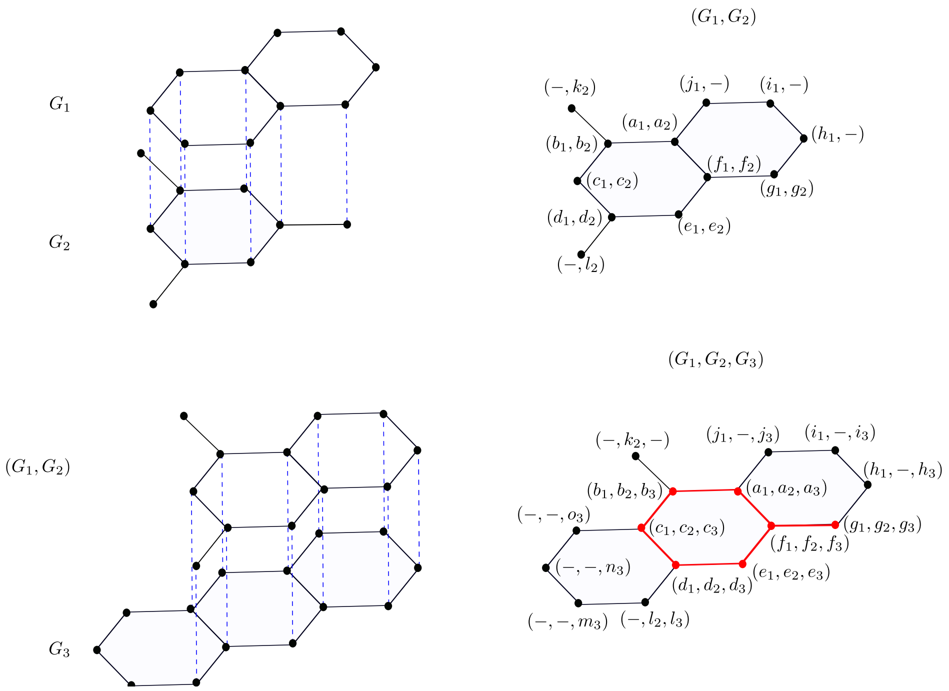

FIGURE 6. Example for (progressive) graph alignment of G_1 and G_2 (top) with aligned graph structure on the r.h.s and alignment of (G_1, G_2) with G_3 and aligned graph structure again on the r.h.s. Dashed blue lines show matches between nodes of the input graphs. Labels at nodes correspond to alignment columns, indices refer to input graphs G_1, G_2 or G_3. The red subgraph is the maximal common induced subgraph of all three input graphs.

common induced subgraph of the graphs at its child-nodes is computed, and the graphs are "glued together" at the common vertices.

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 constituent 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. [18] for a recent review.

11. Alignments for General Structures

So far, we have considered alignments for sequences (strings), partially ordered sets, rooted ordered trees, and graphs. How far can we generalize the idea of alignments, and what are minimal conditions for well-defined alignments? Let us start from a finite space $(X, \mathscr{S})$ with some structure $\mathscr{S}$. We are not really interested in the particular properties of $\mathscr{S}$. Examples for $\mathscr{S}$ might be systems of not necessarily binary relations, topologies, proximities, etc. As a minimum requirement we ask that $(X, \mathscr{S})$ admits well-defined subspaces, that is, if $Y \subseteq X$, then there exists a unique subspace $(Y, \mathscr{S}_Y) =: (X, \mathscr{S})[Y]$. Furthermore we require that

$$(X, \mathscr{S})[Z] = ((X, \mathscr{S})[Y])[Z] \tag{5}$$

holds for all $Z \subseteq Y \subseteq X$, i.e., that induces subspaces that can be formed stepwisely in a consistent manner. This property is satisfied for the examples we have considered so far: strings and totally ordered sets in general, partial orders, as well as directed and undirected graphs. It also holds for ternary relations such as betweenness, as well as topologies, proximities, and similar constructions.

Now suppose we are given input spaces $(X_a, \mathscr{S}_a)$ for all $a \in S$. As in the previous sections, we set $X := \dot{\bigcup}_{a \in S} X_a$, we introduce a set A of edges connecting the vertices in X and write $\mathcal{C}(X, A)$ for the set of connected components of the graph (X, A). Furthermore, we define

$$\mathcal{C}_a := \{Q \in \mathcal{C}(X, A) | Q \cap X_a \neq \emptyset\}.$$

Endowing $\mathcal{C}(X, A)$ with some structure $\mathscr{S}$ consider the subspace $(X, \mathscr{S})[\mathcal{C}_a]$ obtained from $(X, \mathscr{S})$ to the connected components (columns) of (X, A) in which X_a is represented. As in the previous sections we assume

(X1) $(X, A)[Q]$ is a complete graph for all $Q \in \mathcal{C}(X, A)$, and
(X2) $|X_a \cap Q| \leq 1$ for all $a \in S$ and $Q \in \mathcal{C}(X, A)$.

Assumption (X1) implies that there is a 1-1 correspondence between the columns of $Q \in \mathcal{C}_a$ and the elements $q \in X_a$ define by $Q \cap X_a = \{q\}$. Denote the corresponding map by $\pi_a : \mathcal{C}_a \to X_a$. The condition that "projecting" $(\mathcal{C}(X, A), \mathscr{S})$ down the constituent rows $a \in S$ recovers the input spaces can then be expressed as

(X3) $(\mathcal{C}(X, A), \mathscr{S})[\mathcal{C}_a] \simeq (X_a, \mathscr{S}_a)$ with π_a being an isomorphism.

This construction provides a well-defined notion of an alignment in a very general setting. Again, the restriction of the alignment to a set $\mathcal{C}'$ of columns that are represented in X_a for all $a \in S'$, i.e., $(\mathcal{C}(X, A), \mathscr{S})[\mathcal{C}']$ is a common subspace of the $(X_a, \mathscr{S}_a)$ with $a \in S'$. This corresponds the poset alignments satisfying (A5).

Properties (X1), (X2), and (X3) are sufficient to ensure that key properties of totally ordered alignments still hold in this much more general setting. Repeating the simple arguments leading to Lemmas 9, 10 and 12 above, we observe:

(i) The restriction $(X, A, \mathscr{S})[Y]$ to $Y \subseteq X$ is an alignment for the restricted input spaces $(X_a, \mathscr{S}_a)[X_a \cap Y]$.

(ii) If $\mathfrak{P}$ is a partition of X into groups of rows, the quotient $(X, A, \mathscr{S})/\mathfrak{P}$ is an alignment of alignments: The rows of $(X, A, \mathscr{S})/\mathfrak{P}$ are of the form $(\mathfrak{C}(X, A), \mathfrak{S})[\mathcal{C}']$, where $\mathcal{C}' := \{C \in \mathcal{C}(X, A) | C \cap X_a \neq \emptyset, a \in S'\}$ where $S' \subseteq S$ determines a class of the row-wise partition $\mathfrak{P}$. That is, every row of $(X, A, \mathscr{S})/\mathfrak{P}$ is (isomorphic to) a subspace of $(\mathfrak{C}(X, A), \mathfrak{S})$.

(iii) For a given class of $\mathfrak{P}$ determined by the row indices, we observe that by construction the restriction of $(X, A, \mathscr{S})[Y]$ to $Y := \dot\bigcup_{a \in S'} X_a$ is isomorphic to $(\mathfrak{C}(X, A), \mathfrak{S})[\mathcal{C}']$. By assumption, $(\mathcal{C}(X, A), \mathscr{S})[\mathcal{C}_a] = ((\mathcal{C}(X, A), \mathscr{S})[\mathcal{C}'])[\mathcal{C}_a]$ for all $a \in S'$. Therefore we can construct $(X, A, \mathscr{S})$ as the alignment $(X, A, \mathscr{S})/\mathfrak{P}$ of the alignments $(X, A, \mathscr{S})[Y]$ of the rows in each class of the partition $\mathfrak{P}$.

We conclude therefore, that alignments defined by (X1), (X2), and (X3) can be decomposed recursively into alignments of alignments on all spaces with subspaces satisfying equ. (5). In particular, these properties are sufficient to guarantee that progressive alignments are well defined.

A natural question that arises at this abstract level is whether for any collection $(X_a, \mathscr{S}_a)$, $a \in S$, there exists an alignment. To answer this question we consider trivial alignments for which $A = \emptyset$. Then every alignment column contains an element from exactly one of the X_a. Thus there is a 1-1 correspondence between $\mathcal{C}(X, \emptyset)$ and X, ensuring that $(X, \emptyset, \mathcal{S})$ and $(X, \mathcal{S})$ are isomorphic. By (X3), $(X, \emptyset, \mathcal{S})$ is an alignment of the $(X_a, \mathscr{S}_a)$ whenever $(X, \mathscr{S}^*)[X_a] \simeq (X_a, \mathscr{S})$ for all $a \in S$. The existence of such a "disjoint union" $(X, \mathscr{S}^*)$ is thus a sufficient condition for the existence of alignments. All the examples discussed in this section allow such "disjoint unions" and hence support alignments of arbitrary input data.

12. Concluding Remarks

In this contribution we have analyzed the compositional properties of sequence alignments and explored the generalization to much more general structures. We find that meaningful concepts of alignments are not restricted to ordered sets as inputs, but can be extended to very general relational or topological structures that need not bear any resemblance with order relations. The key property of the generalized alignments considered here is that the restriction of the alignment to a row recovers the input row. While this property is a simple consequence for the familiar sequence alignments, it becomes an important defining property of alignments in general. It suffices under very mild conditions of the structure of input spaces to ensure that alignments of alignments and recursive, row-wise decompositions of alignments are well-defined. We have observed, furthermore, that some well-studied examples of alignment problems, such as tree alignment and the alignment of totally ordered sets to a poset seamlessly fit into the framework developed here.

In this setting, alignments are defined on common subspaces. In the case of graph alignments, alignment columns corresponding to (mis)matches form common induced subgraphs. The pairwise alignment problem for two input graphs G_1 and G_2 therefore boils down to the problem of finding a maximum common induced subgraph (MICS). The MICS is a well-known NP-hard problem, which can be reduced to clique finding [4]. Nevertheless it is of substantial practical importance, in particular in chemoinformatics, since molecules are conveniently represented as graphs. A variety of practically applicable algorithms are therefore available [52, 16, 12]. In addition to clique-finding, dynamic programming algorithms have been explored in particular for restricted classes of graphs [1, 21]. In the setting of graph alignments, it may be interesting not only to score the matches, i.e., the common induced subgraph, but also the insertions and deletions, possibly requiring modified algorithmic approaches. We will explore aspects of scoring graph alignments and the computation of pairwise and multiple alignments of graph in future work.

In the context of poset alignments we also explored notions of alignments that require less stringent conditions than the exact recovery of the structure of each input row: it also seems to be of interest to require only that the restriction to a row is an extension of the input order. In the case of graphs, a similarly relaxed condition would only require that the input is a subgraph of the restriction. RNA structures may be considered as totally ordered sets that in addition carry a graph structure defined by the base pairs. Structure annotated alignments, then, have to recover the sequence order upon restriction to the input order, while the restriction of consensus base pairing on the alignment columns only needs to be a subgraph of the input base pairings. It will certainly be interesting to study such relaxed requirements on structure preservation more systematically in future work.

The fact that (multiple) alignments can be defined for very general structures, in essence for finite spaces with reasonably well-behaved notions of subspaces, suggests that alignments may be of interest as mathematical objects also for infinite spaces. Can the idea of alignments be captured 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 form 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*, 76:1488–1493, 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.

[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.

[4] H G Barrow and R M Burstall. Subgraph isomorphism, matching relational structures and maximal cliques. *Information Processing Letters*, 4:83–84, 1976.

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

[6] 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.

[7] 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.

[8] H. Bunke. On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters*, 18:689–694, 1997.

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

[10] 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.

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

[12] Edmund Duesbury, John Holliday, and Peter Willett. Comparison of maximum common subgraph isomorphism algorithms for the alignment of 2D chemical structures. *ChemMedChem*, 13:588–598, 2018.

[13] 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.

[14] R C Edgar and S Batzoglou. Multiple sequence alignment. *Curr Opin Struct Biol*, 16:368–373, 2006.

[15] Robert C Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res.*, 32:1792–1797, 2004.

[16] Hans-Christian Ehrlich and Matthias Rarey. Maximum common subgraph isomorphism algorithms and their applications in molecular science: a review. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 1:68–79, 2011.

[17] Isaac Elias. Settling the intractability of multiple alignment. *J. Comp. Biol.*, 13:1323–1339, 2006.

[18] 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.

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

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

[21] 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.

[22] O Gotoh. An improved algorithm for matching biological sequences. *J. Mol. Biol.*, 162:705–708, 1982.

[23] O. Gotoh. Alignment of three biological sequences with an efficient traceback procedure. *J. theor. Biol.*, 121:327–337, 1986.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] Tao Jiang, Lusheng Wang, and Kaizhong Zhang. Alignment of trees – an alternative to tree edit. *Theor. Comp. Sci.*, 143:137–148, 1995.

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

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] Matthias Kruspe and Peter F. Stadler. Progressive multiple sequence alignments from triplets. *BMC Bioinformatics*, 8:254, 2007.

[38] 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.

[39] Christopher Lee. Generating consensus sequences from partial order multiple sequence alignment graphs. *Bioinformatics*, 19:999–1008, 2003.

[40] Christopher Lee, Catherine Grasso, and Mark F. Sharlow. Multiple sequence alignment using partial order graphs. *Bioinformatics*, 18:452–464, 2002.

[41] 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.

[42] Ketil Malde and Tomasz Furmanek. Increasing sequence search sensitivity with transitive alignments. *PloS one*, 8:e54422, 2013.

[43] Bodo Manthey. Non-approximability of weighted multiple sequence alignment. *Theor. Comp. Sci.*, 296:179–192, 2003.

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

[45] Burkhard Morgenstern. DIALIGN 2: improvement of the segment-to-segment approach to multiple sequence alignment. *Bioinformatics*, 15:211–218, 1999.

[46] Burkhard Morgenstern, Andreas Dress, and Thomas Werner. Multiple DNA and protein sequence alignment based on segment-to-segment comparison. *Proc. Natl. Acad. Sci. USA*, 93:12098–12103, 1996.

[47] 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.

[48] 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.

[49] 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.

[50] 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.

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

[52] John Raymond and Peter Willett. Maximum common subgraph isomorphism algorithms for the matching of chemical structures. *J. Computer-Aided Mol. Design*, 16:521–533, 2002.

[53] Nancy Retzlaff and Peter F. Stadler. Partially local multi-way alignments. *Math. Comp. Sci.*, 12:207–234, 2018.

[54] 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.

[55] Temple F Smith and Michael S Waterman. Comparison of biosequences. *Adv. Appl. Math.*, 2:482–489, 1981.

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

[57] 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.

[58] Jochen Tiepmar and Gerhard Heyer. An overview of canonical text services. *Linguistics Literature Studies*, 5:132–148, 2017.

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

[60] 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.

[61] L Wang and T Jiang. On the complexity of multiple sequence alignment. *J Comput Biol*, 1:337–348, 1994.

[62] 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.

[63] 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.

Sarah J. Berkemer
MPI Mathematics in the Sciences
Inselstraße 22, D-04103 Leipzig, Germany

Bioinformatics Group
Department of Computer Science, and Interdisciplinary Center for Bioinformatics
Leipzig University, Härtelstraße 16-18, D-04107 Leipzig, Germany
e-mail: `bsarah@bioinf.uni-leipzig.de`

Christian Höner zu Siederdissen
Bioinformatics Group
Department of Computer Science, and Interdisciplinary Center for Bioinformatics
Leipzig University, Härtelstraße 16-18, D-04107 Leipzig, Germany
e-mail: `choener@bioinf.uni-leipzig.de`

Peter F. Stadler
MPI Mathematics in the Sciences
Inselstraße 22, D-04103 Leipzig, Germany

Bioinformatics Group
Department of Computer Science and Interdisciplinary Center for Bioinformatics,
Leipzig University, Härtelstraße 16-18, D-04107 Leipzig, Germany

Competence Center for Scalable Data Services and Solutions Dresden/Leipzig
German Centre for Integrative Biodiversity Research (iDiv) Halle-Jena-Leipzig
Centre for Biotechnology and Biomedicine, and Leipzig Research Center for Civilization Diseases (LIFE)
Leipzig University, Germany

Facultad de Ciencias
Universidad Nacional de Colombia, Bogotá, Colombia

Institute for Theoretical Chemistry
University of Vienna, Währingerstraße 17, A-1090 Wien, Austria

Center for non-coding RNA in Technology and Health
Copenhagen University, Grønnegårdsvej 3, Frederiksberg C, Denmark

Santa Fe Institute
1399 Hyde Park Rd., Santa Fe, NM 87501 USA
e-mail: `studla@bioinf.uni-leipzig.de`