Partially Local Multi-Way Alignments

Nancy Retzlaff
Peter F. Stadler

SFI WORKING PAPER: 2017-04-010

SANTA FE INSTITUTE

Partially Local Multi-Way Alignments

Nancy Retzlaff[1,2] and Peter F. Stadler[1,2,3,4,5,6,7]

1 MPI Mathematics in the Sciences, Inselstr. 22, Leipzig, Germany
2 Dept. Computer Science, and Interdisciplinary Center for Bioinformatics, Univ. Leipzig, Härtelstr. 16-18, Leipzig, Germany
3 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), University Leipzig, Germany
4 FHI Cell Therapy and Immunology, Perlickstr. 1, Leipzig, Germany
5 Dept. Theoretical Chemistry, Univ. Vienna, Währingerstr. 17, Wien, Austria
6 RTH, Univ. Copenhagen, Grønnegårdsvej 3, Frederiksberg C, Denmark
7 Santa Fe Institute, 1399 Hyde Park Rd., Santa Fe, USA

Abstract

With increasing computational resources exact solutions to 3- and 4-way alignments have become feasible in practise. In the pairwise case there is a clear distinction between local and global alignments. As more sequences considered this distinction, which can in fact be made independently for both ends of each sequence, gives rise to a rich set of partially local alignments. So far these have remained largely unexplored. Here we propose a very general framework for this class of problems and show how exact dynamic programming solutions can be constructed in principled manner.

1998 ACM Subject Classification I.2.8: Dynamic programming, F.2.2: Computations on discrete structures

Keywords and phrases Multiple sequence alignment – Dynamic Programming – Sum-of-pairs score – Grammar

Digital Object Identifier 10.4230/LIPIcs...

1 Introduction

The distinction between global, local, and semi-local *pairwise* alignments is standard material for introductory courses in algorithmic bioinformatics, Figure ??. It is well known that all these problems are solved by slight variations of the same basis dynamic programming algoritms: the Needleman-Wunsch recursion [?] for the global problem and Smith-Waterman algorithm [?] for the local version. The key recursion step compares the scores of extensions of shorter alignments by a (mis)match, insertion, or deletion, $\mathfrak{N}(i,j) := \max\{S_{i-1,j-1} + m(i,j); S_{i-1,j}+\gamma; S_{i,j-1}+gamma\}$. In the local case, $\mathfrak{S}(i,j) := \max\{\mathfrak{N}(i,j); 0\}$ an *unaligned state* with score 0 is added as an extra choice. In addition, the two algorithms differ in the initialization and the entry of the S-matrix that harbours the final result, i.e., the score of the optimal global of local alignment.

Figure 1 Basic types of pairwise alignments. Top: global and local alignments. Below: semi-global alignment and one of several variants of an overhang alignment.

Several Variations on the theme, where some or all end-gaps are not scored, also share the basic structure of the recursion. Semi-global alignments are used in particular in homology search for ncRNAs and ask for a complete match of a small sequences in potentially large database. The algorithm only differs in initalization, setting scores $S_{0,j} = 0$ to allow cost-neutral deletions of the prefixes of the long, second sequence. Correspondingly, the optimal match is found as $\max_k S_{n_1,k}$. Dedicated implementations exists for this task, e.g. `gotohscan` [?]. More general overlap alignments [?] allow free endgaps on all sequences. These play a role e.g. in sequence assembly [?].

With the advances in available computing power dynamic programming algorithms have become feasible beyond pairwise alignments. The basic recursion for the simultaneous alignment of N sequences is straighford generalization of recursion $\mathfrak{N}$ and simply enumerates all $2^N - 1$ possible patterns of gaps in the last column of a alignment ending at position i_p in the p-the sequences [?]. Despite the extra effort, 3-way alignments have at least occasionally found practical applications in computational biology [?, ?, ?, ?]. In computational linguistics, 4-way alignments [?] have been used to align words from related natural languages, an approach that is feasible owing to the short sequence length.

Despite the importance of alignment problems in computational biology, and the need to distinguish global and local versions of the problem, there does not seem to an accessible theory of partially local alignments beyond the pairwise case. Equally surprisingly, alignment problems with complex mixtures of local and global alignments do appear in practice however. For example, we recently asked whether and under which conditions mitochondrial genome rearrangments are accompanied by the duplication or loss of sequence in the vincinity of breakpoint [?]. A natural model consists of a reference (the ancestral state, 1 in Fig. ??) and the two sequences that close to the break point in the derived state (2 and 3 in Fig. ??). Since the latter two continue with sequence not alignable beyond the breakpoint, they are naturally treated as local on one side and global (anchored) at the other. Regions in which both derived fragments overlap are rewarded to increase the sensitivity for the detection of duplicated sequence. As is turns out, the corresponding alignment algorithm has been quite efficient in detecting previously undescribed tandem duplication random loss events in mitochondrial evolution.

In a variety of applications it useful to consider a probabilistic version of alignments. In the pairwise case, the relation between score minimization and a corresponding "partition function version" is well understood [?, ?, ?, ?] and in a more general context explained within the framework of Algebraic Dynamic Programming [?]. More recently probabilistic models were also studied systematically for local pairwise alignments [?]. Exact probabilistic algorithms beyond pairwise alignments seem to have received very little attention so far.

Most 3-way and 4-way alignment algorithms were designed with very specific application in mind and made no attempt to map the world of mixed global and local alignment problems of which [?] covers just a very specific special case. The purpose of this contribution is to fill this gap and develop a concise formal framework for exact DP algorithms of partically local N-way multiple sequence alignments. Wr proceed stepwisely. First we introduce a compact notation for the global alignment problem and argue that (the generalizations of) semi-global and "overhang" alignments are better viewed as global alignments with

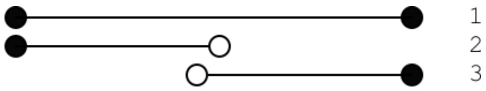

Figure 2 Breakpoint alignment for a reference sequence (1), a prefix (2), and a suffix (3). Sequences (2) and (3) may or may not overlap.

modified scoring functions at the ends of the input sequences. On this basis we then give a complete classification of partially local problems and derive a DP algorithms to solve them. Subsequently we consider the problem from the point of grammars, deriving an unambiguous version that, albeit computationally even more expensive, is suitable for a full probabilitic treatment. We close with some comments on possible future developments.

2 Global and Semi-Global N-Way Alignments

For the moment we consider a fixed (global) alignment of N sequences $X_1, X_2, \dots, X_n$. The alignment in completely determined by specifying, for each column χ, (i) the last position i_p of sequence $p \in \{1, \dots, N\}$ in or the left of column χ, and (ii) a pattern $\pi \in \{0,1\}^N$, $\pi \neq (0,0,\dots,0)$, that specifies, for each sequence p whether column χ contains the i_p-th letter of p or a gap character. Writing $I = (i_1, i_2, \dots, i_N)$, we note that the index vector in the column preceeding χ is $I' = I - \pi$. Observing that at its very left end a global alignment starts with the trivial alignment of empty prefixes, i.e., $I = (0,0,0,\dots,0)$. It is clear, therefore, that knowledge of π for each column is by itself already sufficient to completely determine the alignment.

In the simplest case of linear gap costs, the score S_I of the optimal global alignment satisfies the recursion

$$S_I = \max_{\pi} S_{I-\pi} + \mathrm{score}(\pi, I) \tag{1}$$

where $\sigma(\pi, I)$ is a scoring function that depends on the gaps, i.e., the p for which $\pi_p = 0$, the letters $x^p_{i_p}$ for $p \neq 0$ and possibly also explicitly on the sequence position i_p.

Before we proceed, it is instructive to briefly consider the more complicated case of affine gap costs. Observing that the optimal alignment cost depends on the current as well as the previous alignment column, each of which is characterized by a pattern π and τ of gap and non-gap characters, we can write

$$S^{\pi}_I = \max_{\tau} \left\{ S^{\tau}_{I-\pi} + \mathrm{score}(\tau, \pi | I) \right\} \tag{2}$$

where entry i_p of the multi-index $J = I - \pi$ is $j_p = i_p - 1$ if π_p is not a gap, and $j_p = i_p$ if π_p is a gap. The explicit reference to the pattern τ in the preceeding column is required in Equ. (**??**) to distinguish gap opening ($\pi_p = 0$, $\tau_p = 1$) from gap extension ($\pi_p = \tau_p = 0$). This is the obvious generalization of Gotoh's algorithm [**?**] to more than two sequences. In general, the score contributions take the form $\mathrm{score}(\tau, \pi | I)$ depending on the gap pattern π of the current column, the gap pattern τ of the previous column, and the multi-index I. We formally make the score explicitly dependent on I (rather than the characters that are aligned) to ensure that it also incorporates context-dependent scoring, which has been shown to have large benefits in particular for protein alignments [**?**, **?**].

By far the most widely used scoring model for multiple sequence alignments is the sum-of-pairs score [**?**, **?**] defined as

$$\mathrm{score}(\tau, \pi | I) = \sum_{p>q} \mathrm{score} \left(\begin{array}{cc|c} \tau_p & \pi_p & i_p \\ \tau_q & \pi_q & i_q \end{array} \right) \tag{3}$$

Pairwise score components expressed in terms of π and τ can be interpreted in the following way: A term of the form $\mathrm{score} \left(\begin{smallmatrix} \tau_p & 1 \\ \tau_q & 1 \end{smallmatrix} \middle| \begin{smallmatrix} i_p \\ i_q \end{smallmatrix} \right) = s(X_{p,i_p}, X_{p,i_q})$ corresponds to a (mis)match of the letters X_{p,i_p} at position i_p of sequence p and X_{q,i_q} at position i_q of sequence q. The terms score $\left(\begin{array}{cc|c} 0 & 0 & i_p \\ \tau_q & 1 & i_q \end{array}\right)$ and score $\left(\begin{array}{cc|c} 1 & 0 & i_p \\ 1 & 1 & i_q \end{array}\right)$ evaluate as gap extension and gap open scores, respectively. Since projects of multiple sequence alignments to pairwise alignments may contain gap columns, terms of the form score $\left(\begin{array}{cc|c} \tau_p & 0 & i_p \\ \tau_q & 0 & i_q \end{array}\right) = 0$ also appear.

The end patterns π and τ can be more complex that just the gap patterns in the last and penultimate column. The strict definition of the sum of pairs score requires that columns consisting of two gap are removed before scoring a pairwise subalignment. The following two example show that this cannot be done based only on the gap pattern of the current and the previous column.

$$\begin{array}{ccccc} & a & b & c & d \\ (a) & a' & - & - & - \\ & - & - & - & d' \end{array} \qquad \begin{array}{ccccc} & a & b & c & d \\ (b) & a' & - & - & d' \\ & - & - & - & - \end{array}$$

The naïve generalization of affine gap cost score given above treats $\genfrac{}{}{0pt}{}{-}{d'}$ as gap extension, while it should be scored as opening a new gap. NANCY: what is the issue arising in (b)? These issues are discussed at length in the context of "aligning alignments" and shown to be resolved by characterizing alignment columns by more elaborate "shapes" that describe, for each sequence, the relative position of the immediately preceeding character [**?**]. Another variant that requires more elaborate end patterns is the use of distinct gap types as in the case of piece-wise linear gap cost functions [**?**]. We emphasize that the definition if $I - \pi$ below Equ.(**??**) already accounts for situations with different types of gaps. It can be easily generalized to also the other encodings of end patterns. For our purposes, however, it suffices to require that the patterns τ and π together have sufficient information to determine both the score contribution and the multi-index $J := J(\pi, tau, I)$ of the previous column given the current index I.

3 End Gaps and Overhangs

In pairwise semi-global and overhang alignments it is customary to use the global alignment recursions unaltered. Overhangs on the left hand side are taken care of with the initialization of the DP matrix: one simply sets $Si, 0 = 0$ if the deletion of a prefix of length i from the first sequences is supposed to be score neutral. The right hand end of the alignment is handled differently. Here the "free" deletion is handled by searching for the maximal entry $\max_i S_{i,n_2}$ so determine best position beyond which to delete the suffix from the first sequence. While this is convenient for score maximization, the trick does not carry over to probabilistic frameworks. The problem is that S_{i,n_2} does not correctly score the best alignment but includes gap scores for the deletions from the first sequence also beyond the last character of the 2nd sequence. The easy remedy is explicitly treat the cost-neutral end gap in the scoring function. Conceptually, what is done in the initialization step as well.

For N-way alignment problems this opens a can of worms: in principle one might want to be able to specify for any pair of sequences whether an overhang should be cost-neutral or not. Furthermore, this choice of scoring can be made independently on the left hand end

Figure 3 Scoring of free end gaps. The graphs at both ends denote the unscored deletions. In this example we have $S(i, 0, 0) = S(0, j, 0) = S(0, 0, k) = 0$. The entries $S(i, j, 0)$ coincide with the pairwise score of 1 and 2 and do not incur a gap score for delection of the third sequences.

and the right hand end of the alignment. In the case of a sum-of-pairs scoring model, these choices are conveniently represented by graphs, see Fig. ??. The pairwise scores now depend explicitly on the sequences to which they apply: For instance, we have $s_{13}(X_{1,i_1}, -) = 0$ if $i_3 = 0$ or $i_3 = n_3$ while a regular gap score is used for all other values of i_3.

4 The State of the Alignment

Let us now turn to the concept of locality, that is, the cost-free truncation of prefixes and or suffixes of the input sequence. More precisely, suppose we are asked to align a set Q of sequences in such a way that the a subset $L \subseteq Q$ is may be truncated on the left and a subset $R \subseteq Q$ may be truncated on the right. That is, sequences in L may have unaligned prefixes, while sequences in R may have unaligned suffixes. We call sequences in L and R *left-local* and *right-local*, respectively. The starts of the sequences in $Q \setminus L$, in contrast, are completely contained in the alignment, and the same is true for the ends of the sequences $Q \setminus R$. We call these *left-global* and *right-global*, respectively. A sequence that is both left-global and right-global is *global*.

A global sequence spans all alignment columns. We do allow insertions and deletions at the ends; however these are scored. In contrast, a left-local sequence does not take part in the left end of alignment. Again, the special case that its complete prefix is aligned is allowed. Analogously, a right-local sequence usually does not need to extend all the way to the right end of alignment. An alignment column ξ is thus characterized by a partition of Q into three, possibly empty subsets: A set A of sequences that are “active”, a set D of “done” sequences whose last aligned position lies to the left of ξ, and the set $Q \setminus (A \cup D)$ of of sequences whose first aligned position is located to the right of ξ.

Now consider the set of all possible alignments of prefixes of sequences in Q. Any given alignment is characterized by a state (A, D) and a set of indices $I = \{i_k | k \in A\}$ that denotes the length of prefixes that may still be elongated further. Furthermore, denote by $S_I^{A,D}$ the score of the optimal alignment up to I of type (A, D). Our task is derive a recursion for the $S_I^{A,D}$.

First we observe that if (A, D) remain unchanged, $S_I^{A,D}$ follows the usual recursions of an $|A|$-way sequence alignment. In the simplest, linear gap cost model, this amounts to optimizing over all $2^{|A|} - 1$ possible gap patterns in a column of an $|A|$-way alignment. We symbolically write $\mathfrak{A}_I(S^{A,D})$ for this optimization step.

In addition of continuing an alignment in the same state, we may also change the state by starting the aligned part of a left-local sequence and/or end the aligned part of a right-local alignment. Let (A', D') be the state of a column to the left of ξ, which has state (A, D). The

Figure 4 Example of an 8-way alignment. Black circles indicate global, i.e., non-truncatable sequence ends, open circles denote local, i.e., truncatable ends. The state (A, D) is indicated for a few alignment columns denoted by dashed vertical lines.

Figure 5 Hasse digram of the partial order of states $\prec$ for a 4-way alignment problem with $L = \{1, 2\}$ and $R = \{1, 3\}$. The highlighted path from L to R corresponds to the alignment in the inset.

set of "done" sequences cannot decrease from left to right, thus $D' \subseteq D$. On the other hand, the set of not yet activated sequences cannot increase, i.e., $Q \setminus (A \cup D) \subseteq Q \setminus (A' \cup D')$, i.e., $A' \cup D' \subseteq A \cup D$. This simple observation define a predecessor relation on the set of states:

$$(A', D') \preceq (A, D) \quad\iff\quad A' \cup D' \subseteq A \cup D \text{ and } D' \subseteq D \tag{4}$$

The importance of the partial order $\preceq$ lies in the fact that it defines all allowed state transitions, i.e., if ξ has state (A, D), then the possible states of the previous column are exactly all (A', D') such that $(A', D') \preceq (A, D)$. We write $(A', D') \prec (A, D)$ for $(A', D') \preceq (A, D)$ and $(A', D') \neq (A, D)$. The Hasse diagram of this partial order, Fig. **??**, plays an important role for the solution of the general alignment problems, as we shall see below.

Now consider a state transition that leaves D unchanged. Thus $A' \subset A$, i.e., the sequences in $A \setminus A'$ are activated. It is a technical convenience to initialize the activated dimension with a empty prefix. This corresponds to the score 0 in the usual formulation of the Smith-Waterman algorithm. Thus $S_I^{A,D}$ is at least the better choice of (a) the extension of an alignment with the same state $\mathfrak{A}_I(S^{A,D})$ and (b) the activation of the sequences in $A \setminus A'$ with score $\mathfrak{A}_{I_{|A'}}(S^{A',D})$, where $I_{|A'}$ is the restriction of the index vector to the sequences in A'. It is convenient to activate the $A \setminus A'$ before their first letters are aligned in the following column. We may visualize this in the following way:

Only in the following alignment colums will letters from $A \setminus A'$ be included. The advantage of this choice is that no optimization over an appended column must be computed for all state changes. Instead this is evaluated in the next step as part of the evaluation of the case without state change.

Now let us turn to the case that a subset $Delta \subseteq A'$ is "finished" in the current column, i.e., $D = D' \cup \Delta$ and $A = A' \setminus \Delta$. The score-optimal choice of the index k_h at which the aligned part of sequence $h \in \Delta$ ends is that the maximized the score given the fixed indices I_A. Thus the a transition from state $(A \cup \Delta, D \setminus \Delta)$ to (A, D) contributes

$$\max_{I_{|\Delta}} S_{I \cup I_\Delta}^{A \cup \Delta, D \setminus \Delta} \tag{5}$$

to the alternatives choices for $S_I^{A,D}$.

It remains to consider the case some sequences are not aligned at all, i.e., they transition directly from $Q \setminus (A' \cup D')$ into D. This set is $D \setminus (D' \cup A')$. Since these remain completely unaligned, there is no score contribution.

Let us adopt the convention that for $\Delta = D \cap A' = \emptyset$ we can write $S_I^{A',D'} = \max_{I_{|\Delta}} S_{I \times I_\Delta}^{A',D'}$. The rationale for the this notation is that we take the maximum over an empty set of extensions; but the term $S_I^{A',D'}$ itself still remains in the the collection that we maximize over. Thus we have

$$\max_{(A',D') \prec (A,D)} \begin{cases} S_I^{A',D'} & \text{if } D \cap A' = \emptyset \\ \max_{I_{|\Delta}} S_{I \times I_\Delta}^{A',D'} & \text{for } \Delta = D \cap A' \neq \emptyset \end{cases} = \max_{(A',D') \prec (A,D)} \max_{I_{|\Delta}} S_{I \times I_\Delta}^{A',D'}$$

We summarize the discussion above as:

▶ **Theorem 1.** *The optimal alignment scores satisfy the recursion*

$$S_I^{A,D} = \max\left(\mathfrak{A}_I(S^{A,D}), \max_{(A',D')\prec(A,D)} \max_{I_{|A'\cap D}} S^{A',D'}_{I\cup I_{A'\cap D}}\right) \tag{6}$$

Writing end patterns explicitly, the first term becomes $\mathfrak{A}_I(S^{A,D,\pi})$. This expression maximization over all end patterns τ of the previous column according to equ.(**??**). The remaining terms, which refer to state changes and hence to not produce an alignment column leave the end pattern intact. However, we need to define that a newly activated sequences appear in a well-defined end pattern, which corresponds to an (empty) alignment so that a subsequent gap is correctly scored as gap opening. Similarly, sequence p is completely removed from the pattern π when p is done. The relation between π and new pattern $\tilde{\pi}$ is thus determined completely by (A',D'), (A,D), and π.

The maximization in Thm. **??** is redundant in the following sense: $(A'',D'') \prec (A',D')$ and $(A',D') \prec (A,D)$ also implies $(A'',D'') \prec (A,D)$. Thus (A'',D'') does not need to be maximized over when computing $S_I^{A,D}$, since it is already included in the computation of $S_I^{A',D'}$. Therefore one can restrict the maximization in Thm. **??** to the immediate predecessors $(A',D') \lessdot (A,D)$ in the lattice of partially local alignments.

▶ **Lemma 2.** $(A',D') \lessdot (A,D)$ *if and only if either (i)* $D'=D$, $A' \subsetneq A$, *and* $|A'| = |A|+1$, *or (ii)* $D \setminus D' = A' \setminus A = A' \cap D$ *and* $|A' \setminus A| = |D \setminus D'| = 1$.

Proof. It is easy to check that $(A',D') \prec (A,D)$ is true in both cases and there cannot be another state between (A',D') and (A,D), i.e., both alternatives imply $(A',D') \lessdot (A,D)$. Now suppose that $(A'',D'') \prec (A,D)$ but $(A'',D'') \not\lessdot (A,D)$. If $A'' \cap D$ contains two or more elements, say p and q then $(A \setminus \{p\}, D \cup \{p\}$ is $\prec$-between (A,D) and (A'',D''). Similarly, if $D = D'$ and $A \setminus A'$ contains at least two elements (again called p and q, then $(A' \cup \{p\}, D)$ lies $\prec$-between (A',D') and (A,D). Finally if there $p \in D \setminus (D' \cup A')$, then $(A' \cup \{p\}, D')$ lies $\prec$-between (A',D') and (A,D). Thus, conditions (i) or (ii) together are indeed sufficient. ◀

It follows immediately that (A,D) has most N immediate predecessors.

5 Grammars and Probabilistic Alignments

A convenient formalism in which to discuss complicated DP algorithms in Algebraic Dynamic Programming (ADP) [**?**]. It emphasizes that it is possible for a large class of problems that encompasses also sequence alignments, to strictly separate the construction of the state space, i.e., the recursive structure of the problem, the evaluation of sub-solutions, and the selection of sub-solutions. The key benefit of this approach is that one can move from maximizing the score to computing probabilities of alignment edges by a simple change of the scoring algebra. Another advantage is that within ADP it is not necessary to have a separate implementation of the backtracing steps; a recent extension even provides a generic way to construct outside algorithms and thus posterior probabilities [**?**]. It is worthwhile, therefore to consider partially local MSAs from this point of view.

Combining Thm. **??** and the observation that that it suffices to consider only immediate predecessor among the state changes shows that there are two types recursion: Alignment steps in which the multiindex I changes by adding an alignment column, and steps that only change the state but leave I unchanged except for the expansion of contraction of the index set. In the language of grammars, alignments with a given state (A,D) and a given end pattern correspond to the non-terminals, which we write (A,D,π). The terminals are alignment columns $\mathbf{c}^{\pi\leftarrow\tau}$. Recall that the pair of patterns π and τ uniquely determines the gap pattern in the emitted column. The grammar corresponding to equ.(**??**) can thus be written as

$$\begin{aligned}&(A,D,\pi)\to(A,D,\tau)\mathbf{c}^{\pi\leftarrow\tau}\text{ for all }\tau\\&(A,D,\pi)\to(A',D',\tilde{\pi})\text{ for all }(A',D')\lessdot(A,D)\end{aligned}\tag{7}$$

In addition we need a starting rule of the form $S\to(R,Q\setminus R,\tau)$ for all τ that generates all alignments with the correct status $(R,Q\setminus R)$ at the right end and all corresponding end patterns. The rules of ADP would allow us to convert this into "partitition function version" of the recursion which takes the form

$$Z_I^{A,D,\pi}=\sum_\tau Z_{J(I,\delta,\tau)}^{A,D,\tau}\exp\operatorname{score}(\mathbf{c}^{\pi\leftarrow\tau})+\sum_{(A',D')\lessdot(A,D)}Z_I^{A',D',\tilde{\pi}}\tag{8}$$

where $J(I,\delta,\tau)$ is the multi-index of the previous column given that multi-index I of the current column, and the end patterns of both the current and the previous column.

However, semantically, this is not what one want to compute in a probabilistic setting because the grammar of equ.(**??**) is ambiguous, i.e., it allows multiple distinct parses for the same alignment. First, alignments in which a sequence p that is to be aligned both left and right local are represented multiple times. In fact, the direct transition from not yet active to done may occur in every alignment step, and all these possibilities are accounted for as separate alignments. This contradicts our intuition of when two alignements should be the same. Of course this "overcounting" makes has no effect when the goal is to maximize the score. It does affect the result, however, when the task is compute probabilities over ensembles of alignments.

A second, even worse ambiguity arises from the fact that we allowed ourselved to perform a sequence state changes without emitting any alignment column in between. This amounts to multiple paths in the lattice of Fig. **??**) that lead to the same overall state change. The remedy this problem, we need to restructure the grammar. The basic idea is to ensure that every production emits a column, and makes at most one state change at the same time. Given the order structure of the alignment columns and the states, this ensures unambiguity. This comes at a cost, however. Now we have to allow any combination of state changes in a single step – so as to count it only once. Thus, in general there are exponentially many (in N) state transitions that need to be considered. This yields a grammar of the form

$$(A,D,\pi)\to(A',D',\tau)\mathbf{c}^{\pi\leftarrow\tau}\text{ for all }(A',D')\preceq(A,D)\text{ and all }\tau\tag{9}$$

To ensure that this grammar is unambiguous we need to enforce that a sequence p can be activated or retired only if one of its charcters in emitted in $\mathbf{c}^{\pi\leftarrow\tau}$. This restriction requires a separate treatment of sequences that remain completely unaligned. This can be achieved by deriving these alignments directing from the start symbol. Only the sequences in $L\cap R$ are anchored at both ends and thus cannot be omitted from the alignment. The remaining sequences may also remain empty. Thus the complete set of alignments is obtained as the (disjoint) union of alignments with all non-empty sequences for every set Q' of sequences satisfying $L\cap R\subseteq Q'\subseteq Q$.

6 Block-Local N-Way Alignments

The general treatment of partially local alignments suggests that further variations on the theme may also be of interest. In phylogenetic footprinting the goal is find intervals of common length k such that their gap-less alignment maximizes a score [?]. In the present setting it would be natural to relax the no-gaps condition. Instead, one may ask for intervals $[b_p, e_p]$ for all $p \in Q$ such that the global alignment of the infixes $X_p[b_p, e_p]$ maximizes the score. We call this the *block alignment problem*. This is can be seen as variant of the partial local N-way alignment where (1) all sequences are local at both ends, and (2) the state transitions are restricted to the rather trivial Hasse diagram $(\emptyset, \emptyset) \rightarrow (Q, \emptyset) \rightarrow (\emptyset, Q)$, i.e., all sequences are activated and retired at the same time. Naturally in this setting, end-gap should be costly, i.e., in contrast to the unambiguous grammars discussed in the previous section we would also allow the first and last rows to contain gaps.

7 Concluding Remarks

In this contribution we have outline a general formal framework towards treating partial, complex locality constraints in sequence alignment. Out approach was guided by exact DP algorithms for this class of problems. Of course the resulting algorithms are exponential in the number of sequences; after all they use the same recursive scheme as the well-known DP solution for global N-way MSA. This is not a fundamental shortcoming, however, since (a) the (decision version of the) global N-way sequence alignment problem is well known to be NP-hard [?, ?], and (b) the problem is tractable in practise and of relevance for practical applications for small numbers input sequences.

At present, we have not provide an general implementation. A special case, however, is in practical use for breakpoint determination [?]. In [?] it has been demonstrated that N-way global alignments as well a a variety of complex alignment algorithms can be constructed quite easily with the help of grammar products. In this context a general implementation is Haskell has been provided. A complete implementation of the framework outline in this contribution, however, requires some further developments, in particular an extension of the underlying theory to corresponding products of the scoring algebras and the development of a principled manner to construct the lattice of alignment states.

more ideas for discussion section.

Acknowledgements. NR gratefully acknowledges the hospitality of the Santa Fe Institute, where much of this work was performed.

Appendix

▶ **Lemma 3.** *Equ.(??) defines a partial order.*

Proof. Since $\subseteq$ is reflexive and antisymmetric, so is $\preceq$. Now assume $(A'', D'') \preceq (A', D')$ and $(A', D') \preceq (A, D)$, i.e., $A'' \cup D'' \subseteq A' \cup D' \subseteq A \cup D$ and $D'' \subseteq D' \subseteq D$; thus $(A'', D'') \preceq (A, D)$ and $D'' \subseteq D$, i.e., $(A'', D'') \preceq (A, D)$. Hence $\preceq$ is transitive. ◀