Context Dependent Alignment: A New Method for Comparing Biological Sequences [1]

Anna Gambin, Sławomir Lasota, Radosław Szklarczyk,
Jerzy Tiuryn, Jerzy Tyszkiewicz[2]
Institute of Informatics, Warsaw University.
Banacha 2, 02-097 Warszawa, Poland.
{aniag,sl,radek,tiuryn,jty}@mimuw.edu.pl

February 19, 2004

[1]This work was partially supported by the Polish KBN grant 7 T11F 016 21.
Its abstract version appeared as [GLSTT02].
[2]Corresponding author: phone ++48 22 55 44 430, fax ++48 22 55 44 400.

Abstract

We present a model of contextual alignment of biological sequences. It is an extension of the classical alignment, in which we assume that the cost of a substitution depends on the surrounding symbols. In this model the cost of transforming one sequence into another depends on the order of editing operations. We present efficient algorithms for calculating this cost, as well as reconstructing (the representation of) all the orders of operations which yield this optimal cost. A precise characterization of the sets of linear orders which can emerge this way is given.

Chapter 1

Introduction

A large portion of modern computational biology is concerned with measuring the degree of similarity of biological sequences, the most prominent examples of which are DNA and proteins.

Generally, to get such a model of similarity, one assumes a set of operations, which can change sequences, and a score function, assigning a score to each operation performed on a sequence. Then, each set of operations transforming a biological sequence V into another such sequence W is assigned a score — typically the sum of the scores of individual operations. The operations correspond to evolutionary changes, higher score reflects that the event is more likely to appear. Several values are then of interest, the crucial ones being the maximal possible score of a transformation of V into W and the maximal score of a transformation of a contiguous fragment of V into a fragment of W (maximized over such fragments, too).

The model dominating in the field (the so called alignment model)(cf. [DEKM98, Gus97]), used for DNA and proteins, assumes the operation of substitution of one letter for another, as well as an insertion or a deletion of a sequence of letters. The score of a substitution depends only on the two residues exchanged. It is provided by the so-called substitution tables [DSO78, HH92]. The score for insertions and deletions depends solely on the length of the inserted/deleted subsequence (it is quite often an affine function).

Of course, the above score model is a great oversimplification from the biological point of view. However, it is the most commonly used, because it permits very efficient algorithms to compute the key values, called the maximal global and local alignment scores, respectively. Other models, which

are biologically more realistic, are computationally very hard (or even provably intractable). For example, a very important property of proteins is their fold (i.e., the 3-D shape they assume in the cell), which can decide homology between proteins of quite different sequences. However, despite very intense efforts, predicting the shape of a protein, based on its sequence, is recognized as an extremely difficult problem.

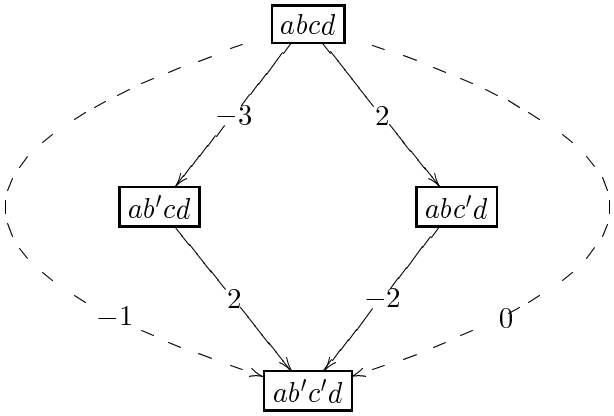
Results. In this paper we offer a model, which extends the classical alignment model, with the intention to bring it a step closer to the biological reality without sacrificing its algorithmic properties. The set of operations is the same as that of the alignment model but the score function of a substitution changes. In our model the score of a substitution *depends on the surrounding letters in the sequence*, too. I.e., the score of substituting b by d in abc can differ from the score of substituting b by d in $a'bc'$. The score for insertions and deletions is inherited from the classical model. We call our model the *contextual alignment model*.

The aim of this paper is to present efficient algorithms for calculating the maximal global and local contextual alignment scores. Indeed, their complexity is (up to a constant factor) the same as that of the algorithms of the classical non-contextual alignment model. Assuming that the sequences V and W are of length m and n, respectively and that the insertion and deletion scores are affine, both the global and the local contextual alignment algorithms work in time $O(|\Sigma|mn)$, where $|\Sigma|$ is the size of the alphabet (4 in the case of DNA, 20 in the case of proteins). In the non-contextual case the complexity is $O(mn)$, independent of the size of the alphabet. The constant in our model is, as one can see above, higher than in the non-contextual case, but is still reasonable.

Another topic of our paper is the order in which operations are performed. As it is easy to see, in the contextual alignment model, the score of a set of operations depends on the order. Indeed, an operation may change the contexts of future operations.

Example 1. Here we see that the relative order of two substitutions applied to the same sequence affects the score, if a contextual scoring function is

used.



Often operations performed at distant fragments of the sequence are independent, in the sense that neither of them changes the context of the other. Independent operations can be performed in any order. Therefore, there are typically many orders, which give the maximal score. Thus, our algorithms find not only an optimal set of operations, but also reconstruct a precise characterization of the set of all possible orders (we call them admissible orders), in which the operations may be performed to yield the maximal score.

Apart from that we present in this paper a series of characterizations of these sets of admissible orders, which can emerge as the result of our algorithms. Analysis of the admissible orders has led us to the construction of efficient algorithms for contextual alignment and justified their correctness.

Summarizing, the main contributions of the paper are:

- Contextual alignment model—a new approach to measuring similarity of biological sequences.
- Efficient algorithms for constructing contextual alignments of maximal score, their scores, and (the representations of) the sets of all admissible linear orders of operations, which give that maximal score.
- Precise characterization of all the sets of admissible linear orders which correspond to complete sets of operations of a maximal score.

The paper is organized as follows. Section 2 introduces the concept of a contextual alignment. The impact of the order of edit operations on the total score is illustrated here. We also introduce in this Section the concept of a

generating poset, which serves as a method to succinctly represent a family of linear orders, each of them yielding an optimal score. Section 2 contains a characterization of generating posets which can come up from the analysis of a contextual alignment. A method of constructing generating posets from a given alignment follows from the results of this Section. In Section 3 we start with a description of an algorithm which finds a representation of all linear orders of edit operations, each of the orders yielding an optimal score for an alignment without gaps (hence the edit operations in this case are substitutions). Then we present an algorithm for a global contextual alignment with gaps and an affine gap penalty function. We justify the correctness of this algorithm, as well as we remark on the possibility of modifying this algorithm for finding local optimal contextual alignments. We conclude this Section with an example of a 'real life' contextual alignment together with a generating poset for this alignment. In Section 4 we discuss some issues related with construction of context-dependent substitution matrices. Section 6 is devoted to discussion of the validation of our model on biological data. We discuss here statistical significance of contextual vs. non-contextual alignment; impact of the number of context groups on the discriminating power of the contextual alignment; a comparison of discriminating power of contextual and non-contextual alignment; and finally the issue of constructing phylogenetic trees with the help of contextual alignment. Appendix contains tables for determining constraints which are generated by an insertion block.

Biological motivation and related work. There are numerous known examples in biology, showing that indeed a context may affect the likelihood of changes in biological sequences. One of them is the elimination of adjacent cytosine-guanine pairs in DNA, caused by biochemical mechanisms of replication. Another one is observed in proteins: substitution of a hydrophobic amino acid by a hydrophilic one in hydrophobic context with much higher probability changes the fold of the protein than an identical substitution in a hydrophilic context. If a protein changes its fold, it may lose its biological activity and thus the underlying mutation is more likely eliminated in the evolution.

We should mention that the contextual alignment we consider is an algorithmic counterpart of work already undertaken in probability theory. Recently several papers have been published [SvH94, vHS98, JKP00], which consider a probabilistic model, in which a biological sequence undergoes random changes due to substitutions, whose probability is context-dependent. This leads to a Markov chain model of quite a complicated structure. The

questions considered in the papers are existence and characterization of the steady-state distribution, estimation of the rate of evolution, as well as estimating the size of the context, which significantly affects the substitution probabilities. E.g., [TG89] estimate the size of the significant context for the DNA evolution in the bacteriophage λ to be 1 or 2 bases (but not 0!). This gives us another argument for considering contextual alignments, as well as for restricting our attention to contexts of size 1.

The paper [WL84] considers contexts for comparing a pair of biological sequences. This is achieved by trying to align without gaps, in various ways, short blocks (the term used there is *aligned fragments*) of characters from each of the two sequences. Scoring of the aligned blocks is given by an external scoring function. Since each pair of blocks receives its own score, the concept of a context is thus present in that approach. This is different understanding of the context than in the present paper — our context is understood as flanking characters which may influence the likelihood of symbol substitution, while in [WL84] the context is understood as having a direct effect on scoring pairs of aligned blocks. It follows that the two approaches, despite of using similar names, have nothing in common.

Chapter 2

Contextual Alignment

Let V and W be strings over an alphabet Σ. A *gap*, denoted $-$, is a symbol assumed not to belong to Σ. Let us fix an alignment $(V^{\#}, W^{\#})$ of these two sequences. Let us recall that an *alignment* $(V^{\#}, W^{\#})$ is a pair of strings over $\Sigma \cup \{-\}$ satisfying the following properties:

1. Erasing gaps in $V^{\#}$ and in $W^{\#}$ yields V and W, respectively.
2. $|V^{\#}| = |W^{\#}| = n$, for some n.
3. If $V_i^{\#}$ is a gap, then $1 < i < n$ and none of the $W_{i-1}^{\#}, W_i^{\#}, W_{i+1}^{\#}$ is a gap.
4. If $W_i^{\#}$ is a gap, then $1 < i < n$ and none of the $V_{i-1}^{\#}, V_i^{\#}, V_{i+1}^{\#}$ is a gap.

It follows from the definition of an alignment that the outermost blocks are always substitutions and that insertions and deletions have to be separated by at least one substitution. Adopting the latter requirement reduces the complexity of the problems considered in this paper. Also, the substitution tables together with gap penalties, used by the biologists are usually constructed in such a way that an insertion followed by a deletion (or vice versa) is more costly than a simple substitution.

An alignment induces three kinds of blocks, each block has its unique *address* and unique *length*:

- A *substitution* has an address i, if $V_i^{\#}, W_i^{\#} \in \Sigma$, i.e. if they are not gaps. Substitutions are one element blocks.

- An *insertion* has an address i, if $V^{\#}_{i-1} \in \Sigma$, $V^{\#}_i = -$. The length of the insertion block is the least $j > 0$ such that $V^{\#}_{i+j} \in \Sigma$.
- A *deletion* has an address i, if $W^{\#}_{i-1} \in \Sigma$, $W^{\#}_i = -$. The length of the deletion block is the least $j > 0$ such that $W^{\#}_{i+j} \in \Sigma$.

Following the above classification of blocks we have three kinds of *operations*, each associated with one block.

- (*Substitutions*) $S_{i,a,b}$, where $1 < i < n$ is an address of a substitution block and $a, b \in \Sigma$. The characters a, b are called *contexts*. There are also two outermost substitutions: S_1 and S_n.
- (*Insertions*) I_i, where i is an address of an insertion block.
- (*Deletions*) D_i, where i is an address of a deletion block.

The insertions, deletions and outermost substitutions are taken without any context.

The *cost* $c(o)$ of each operation o can be read off from the address of the corresponding block and from the alignment $(V^{\#}, W^{\#})$. It also depends on the *gap penalty function* g and on the *contextual substitution tables* $M_{a,b}(\cdot,\cdot)$, where a, b ranges over Σ. The definition follows.

$$c(S_1) = c(S_n) = 0.$$

$$c(S_{i,a,b}) = M_{a,b}(V^{\#}_i, W^{\#}_i), \text{ for } 1 < i < n.$$

$c(I_i) = g(j)$, where j is the length of the insertion block with address i.

$c(D_i) = g(j)$, where j is the length of the deletion block with address i.

A complete set of operations, cso, is any set of operations which correspond to all blocks, one operation for each block. Hence each cso has the same cardinality.

Since the cost of a substitution may depend on the context, it follows that when transforming $V^{\#}$ into $W^{\#}$ the order in which the operations are performed may influence the total cost of the transformation. We first define what it means to perform an operation on a string $X = x_1 \dots x_n \in (\Sigma \cup \{-\})^*$.

- For $1 < i < n$, a substitution $S_{i,a,b}$ is *admissible* for X, if $x_{i-1} = a$ and $x_{i+1} = b$. The substitutions S_1 and S_n are always admissible. The result of performing the substitution (either $S_{i,a,b}$, or S_1, or S_n) on X is $X' = x_1 \dots x_{i-1} W_i^{\#} x_{i+1} \dots x_n$.
- I_i is always admissible for X and the resulting string is $X' = x_1 \dots x_{i-1} W_i^{\#} \dots W_{i+j-1}^{\#} x_{i+j} \dots x_n$.
- D_i is always admissible for X and the resulting string is $X' = x_1 \dots x_{i-1} \underbrace{- \dots -}_{j} x_{i+j} \dots x_n$.

Let $O = \{o_1, \dots, o_k\}$ be a cso. A linear order $o_1 < o_2 < \dots < o_k$ is said to be *admissible*, if starting from V and performing the operations from O in the ascending order yields W without ever performing an inadmissible substitution. More formally, we define a sequence of strings $X_0, \dots, X_k$ such that $X_0 = V$, $X_k = W$ and for every $1 \leq i \leq k$, the operation o_i is admissible for X_{i-1} and yields X_i. A cso is called *admissible* if it has an admissible linear order. In general, an admissible cso may have many admissible linear orders. The aim of this section is to characterize the structure of admissible linear orders on a given admissible cso.

Before this, we give the definition of an *optimal contextual alignment*. Given two strings V, W, we maximize over all alignments $(V^{\#}, W^{\#})$ and over all admissible cso's O the cost

$$c(O) = \sum_{o \in O} c(o).$$

Hence the optimal solution consists not only of an alignment but also of a family of admissible linear orders for this alignment. We will see that in many situations this family of orders can be conveniently represented by *one* principal partial order P, all admissible linear orders being the linear extensions of P.

2.1 Order Constraints for an Insertion Block

We start with an example which illustrates the issues we have to deal with.

Example 2. Consider the following alignment (numbers represent positions used for addresses).

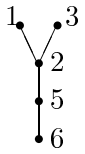
1	2	3	4	5	6
e	a	$-$	$-$	c	t
f	a	c	d	b	u

Numbers in the above alignment represent positions. The upper string $ea--ct$ is a sample $V^{\#}$ and the lower string is a sample $W^{\#}$. The operations transform the upper string into the lower string.

The following set is a cso

$$O_1 = \{S_1, S_{2,e,b}, I_3, S_{5,a,u}, S_6\}.$$

There are exactly two admissible linear orders for O_1: $6 \leq 5 \leq 2 \leq 3 \leq 1$ and $6 \leq 5 \leq 2 \leq 1 \leq 3$. These chains can represented as all linear extensions of the following *principal poset*.



The two admissible linear orders are all (and only) linear extensions of the the above poset. In general the number of linear orders can be bigger and the structure of the principal poset can be more complicated.

Consider now the following cso:

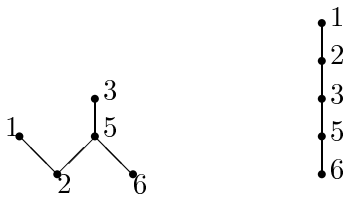
$$O_2 = \{S_1, S_{2,e,b}, I_3, S_{5,d,u}, S_6\}.$$

O_2 imposes the following constraints: 3 must be performed before 5, 5 must be performed before 2 and 2 must be performed before 3. Hence there is no admissible linear order and O_2 is inadmissible.

Finally let us consider the cso:

$$O_3 = \{S_1, S_{2,e,c}, I_3, S_{5,a,u}, S_6\}.$$

The constraints generated by O_3 are: $6 \leq 5 \leq 3$ and $2 \leq 1$, plus the proviso that if 5 was performed before 2, then 3 has to be performed also before 2 (i.e. 2 cannot be between 5 and 3). It is easy to check that in this case there is no single principal poset generating all (and only) admissible linear orders. However, two generating posets can do the job.



Again, the representation works as follows. Every extension to a linear order of any of the above posets is an admissible linear order for O_3 and every admissible linear order is obtained in such a way. In the above representation of the poset it follows that 5 has to be performed after 2 and 3 has to be performed after 3. In particular 3 has to be performed after 2. Even though 3 is depicted in the diagram higher than 1, there is no time relationship between 1 and 3: the diagram does not determine which of the two operations will be performed first. The same remark applies to operations 1 and 5 or 1 and 6.

This example is little degenerated since the second generating poset is already a chain, so there is nothing to extend. In general, though, the generating posets can be more complicated and the number of such posets can be larger than 2.

Now let us consider a general situation of an insertion surrounded by two substitutions.

$$\begin{array}{cccccccc} i & j & & & & & & k \\ a & - & - & \cdot & \cdot & \cdot & - & b \\ a' & c_1 & c_2 & \cdot & \cdot & \cdot & c_m & b' \end{array}$$

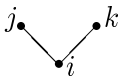
Figure 2.1: A typical triple of blocks: substitution, insertion, substitution.

The second and third strings in the above Figure are assumed to be parts of $V^{\#}$ and $W^{\#}$, respectively. The numbers i, j, k stand for the addresses of the blocks. Clearly we have $j = i + 1$ and $k = i + m + 1$ but for the ease of presentation we choose to work with i, j, k as if they were independent. It is more convenient to examine the mutual constraints which come from choosing the right context for the left substitution *and* the left context for the right substitution.

The substitution i has three possible right contexts: b, b' and c_1. Likewise, the substitution k has three possible left contexts: a, a' and c_m. The case when $m = 0$, i.e. when there is no insertion between i and k will also be covered by our analysis.

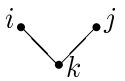
Assume for now that b, b', c_1 are pairwise different and so are a, a', c_m.

Selecting b as the right context for i imposes the constraint that i must be performed before j and k, without further imposing the order between j and k. This constraint is conveniently represented by the poset



All linear extensions of the above poset are admissible for the choice of b as the right context for i.

In a similar way, selecting a as the left context for k generates the constraint



Now, selecting b and a as the right and left contexts for i and k, respectively, generates a contradiction. Hence any cso which contains the above contexts for i and k will be inadmissible.

On the other hand, if c_m is selected as the left context for k, then the resulting constraint is

The above constraint generates three linear orders on $\{i, j, k\}$. Selecting b and c_m as the right and left contexts for i and k, respectively, yields just one chain:

Let us briefly discuss ways of representing families of linear orders on a three element set $\{i, j, k\}$. An explicit way is just to represent a given family by listing all of its elements. Another, more concise way is to represent a family by a finite poset whose all linear extensions form exactly the given family. Such a poset will be called a *principal poset*. Not every family of linear orders can be represented this way. It is easy to show that for a family of linear orders which has a principal poset, the intersection of all linear orders in that family yields this poset. We will use the following notation. $C_{x<y<z}$ stands for the constraint $x < y < z$. $C_{x<y}$ stands for the constraint $x < y$. It corresponds to the poset where the third element is not comparable to the

other two. $\vee_x^{y,z}$ stands for the constraint $x < y, x < z$. $\wedge_{y,z}^x$ stands for the dual constraint $y < x, z < x$. $\perp$ stands for the contradictory constraint, i.e. it generates no linear order. It does not correspond to any poset. Finally $\top$ stands for the constraint which generates all linear orders (on the three element set). It corresponds to the discrete order on the three element set. The constraints are naturally ordered by comparing the sets of linear orders they generate. In fact, under this order the constraints, viewed as sets of linear orders on the three element set, form a Boolean algebra. If $\oplus$ stands for the least upper bound and $\otimes$ stands for the greatest lower bound, then we have the following sample identities

$$C_{i<j} \otimes C_{i<k} = \vee_i^{j,k},$$

$$\vee_i^{j,k} \oplus C_{k<i<j} = C_{i<j}.$$

In Table 2.1 we list the six basic constraints: three constraints for choosing a letter for the right context for i and three for the left context for k.

b	b'	c_1	a	a'	c_m
$\vee_i^{j,k}$	$C_{k<i<j}$	$C_{j<i}$	$\vee_k^{i,j}$	$C_{i<k<j}$	$C_{j<k}$

Table 2.1: Constraints. Part $\{b, b', c_1\}$ is for right contexts for i. Part $\{a, a', c_m\}$ is for left contexts for k. We assume here that b, b', c_1 are pairwise different, and so are a, a', c_m.

If we view constraints as propositions, then $\oplus$ corresponds to disjunction, $\otimes$ corresponds to conjunction, and $\perp$ and $\top$ correspond to the truth values '*false*' and '*true*', respectively. This observation is useful when we want to generate from Table 2.1 the constraints which come from assuming simultaneous contexts: a right context for i and a left context for k. This corresponds to taking a conjunction of the constraints. For example, choosing (b, a) as a pair of contexts yields the constraint $\vee_i^{j,k} \otimes \vee_k^{i,j} = \perp$, while choosing (b, a') yields the constraint $\vee_i^{j,k} \otimes C_{i<k<j} = C_{i<k<j}$. The constraints for all possible pairs of contexts are listed in Chart 1 in the Appendix.

So far, we have assumed that the symbols b, b', c_1 are pairwise different. Likewise for the symbols a, a', c_m. If, for example, the symbols b and b' happen to be equal[1] in Figure 2.1, then the constraint associated with choosing b (which is the same as b') as the right context for i is obtained from Table 2.1

[1] Here we implicitly assume that c_1 is different from b and b'.

by taking disjunction of the constraints which correspond to b and b'. Thus the constraint becomes $\vee_i^{j,k} \oplus C_{k<i<j}$. This constraint cannot be replaced by a constraint which describes a single poset. The above described situation of collapsing symbols b, b' corresponds to taking the partition $\{\{b, b'\}, \{c_1\}\}$ of the set $\{b, b', c_1\}$. Clearly there is a one-to-one correspondence between all possible equalities among the symbols b, b', c_1 and partitions of the set $\{b, b', c_1\}$. There are 5 possible partitions and therefore there are 25 possible tables, one for each pair of partitions on sets $\{b, b', c_1\}$ and $\{a, a', c_m\}$. Table 2.1 is just the table which corresponds to the finest partitions of $\{b, b', c_1\}$ and $\{a, a', c_m\}$.

We are interested in constraints which are generated for pairs of contexts, rather than for single contexts. Combining the methods described in the above two paragraphs yields 25 tables of constraints for pairs of contexts. For example, a constraint for the pair of contexts (b, a), under the assumption that $b = b'$ and $a = a'$ (and assuming that no other equality holds) is the following $(\vee_i^{j,k} \oplus C_{k<i<j}) \otimes (\vee_k^{i,j} \oplus C_{i<k<j}) = \wedge_{k,i}^{j}$. In Table 2.2 we present all possible constraints for the situation when $b = c_1$ and $a' = c_m$. The constraint $\vee_i^{j,k} \oplus \vee_j^{i,k}$ cannot be replaced by a constraint which describes a single poset.

(b, a)	(b, a')	(b', a)	(b', a')
$C_{k<j<i}$	$\vee_i^{j,k} \oplus \vee_j^{i,k}$	$C_{k<i<j}$	$\bot$

Table 2.2: Constraints for the situation $b = c_1$ and $a' = c_m$.

All the 25 tables, each containing all possible pairs of contexts, are listed in the Appendix (Charts 1-25).

The case when substitutions i and k are next to each other, i.e. when $m = 0$, is covered by the above discussion. It suffices to remove from the considerations the contexts c_1 and c_m and remove j from the constraints, i.e. we study then linear orders on the two element set $\{i, k\}$. The resulting table is given below.

b	b'	a	a'
$C_{i<k}$	$C_{k<i}$	$C_{k<i}$	$C_{i<k}$

Table 2.3: Constraints for the case $m = 0$.

Now we are going to characterize the alignments and pairs of contexts for which there is no principal poset. First we introduce some auxiliary notions. Given a pair of partitions: on $\{b, b', c_1\}$ and on $\{a, a', c_m\}$. A pair of contexts (x, y) with $x \in \{b, b', c_1\}$ and $y \in \{a, a', c_m\}$ is said to be *collapsing* if both x and y belong to a block of the partition which has more than one element. For example, for the partition induced by the assumptions of Table 2.2 the pair (b, a') is collapsing, but the pair (b, a) is not.

Consider the triple: substitution, insertion, substitution, as in Figure 2.1. We say that the insertion is *non-principal* if the following four conditions hold.

1. $|\{a, a', c_m\}| \leq 2$.
2. $|\{b, b', c_1\}| \leq 2$.
3. $b \neq b'$, or $a \neq a'$.
4. $b = c_1$, or $a = c_m$.

An insertion which is not non-principal is called *principal*.

Proposition 1. *Consider the triple: substitution, insertion, substitution, as in Figure 2.1. Let (x, y) be a pair of contexts with $x \in \{b, b', c_1\}$ and $y \in \{a, a', c_m\}$, x being the right context for i and y being the left context for k. Let P be a poset which is the intersection of all linear orders on $\{i, j, k\}$ which are admissible for (x, y). Assume there is at least one such linear order. Then every extension of P to a linear order is admissible (i.e. P is the principal poset) iff either (x, y) is not collapsing, or the insertion is principal.*

If (x, y) is collapsing and the insertion is non-principal, then there are two posets P_1, P_2 on $\{i, j, k\}$ such that every linear order extension of any of these posets is admissible for (x, y) and conversely, every admissible linear order is an extension of P_1 or P_2.

Proof. We prove the first part of Proposition 1. The second part follows immediately from inspection of constraint charts given in Appendix. First we prove ($\Rightarrow$). Suppose that (x, y) is collapsing and the insertion is non-principal. It follows from the definition of non-pricipality of an insertion

that we have to consider the following 9 cases.

$$\begin{aligned}
b = c_1, &\qquad a = a' \\
b = c_1, &\qquad a = c_m \\
b = c_1, &\qquad a' = c_m \\
b = c_1, &\qquad a = a' = c_m \\
b = b', &\qquad a = c_m \\
b' = c_1, &\qquad a = c_m \\
b' = c_1, &\qquad a = a' = c_m \\
b = b' = c_1, &\qquad a = c_m \\
b = b' = c_1, &\qquad a' = c_m
\end{aligned}$$

Consider just the first case. The collapsing pair for this case is (b, a) and by inspecting Chart 12 in the Appendix we see that $k < j < i$ and $i < k < j$ are the only admissible linear orders. Now, their intersection is the poset $k < j$ with i being incomparable. Thus the linear order $k < i < j$ is an inadmissible extension of this poset. The other cases are dealt with in a similar way – they are left for the reader.

For the opposite implication we observe, by inspecting the charts in the Appendix, that when the pair of contexts is not collapsing, or when the insertion is non-principal, then the constraint is in the form of one poset. By the construction of constraints it follows that all linear extensions of this poset are admissible and every admissible linear order is obtained in this way. Clearly this poset is the intersection of all its linear extensions. ■

2.2 Order Constraints for a Deletion Block

Consider now a general situation of a deletion block flanked on both sides by a substitution.

$$\begin{array}{cccccccc}
i & j & & & & & & k \\
a & c_1 & c_2 & \cdot & \cdot & \cdot & c_m & b \\
a' & - & - & \cdot & \cdot & \cdot & - & b'
\end{array}$$

Figure 2.2: A typical triple of blocks: substitution, deletion, substitution.

Fortunately we do not have to redo all the considerations from the previous section. There is an easy way of getting all the constraint tables for the

deletion block from those for the insertion block. Define a mapping $\xi : \{a, a', c_1, b, b', c_m\} \to \{a, a', c_1, b, b', c_m\}$ as follows.

$$\begin{array}{lll} \xi(a) = a', & \xi(b) = b', & \xi(c_1) = c_1, \\ \xi(a') = a, & \xi(b') = b, & \xi(c_m) = c_m. \end{array}$$

A *dual* of a poset P is a poset $[P]^{op}$ with the same carrier as P, but with the order relation being reversed, i.e. $x \leq_{[P]^{op}} y$ iff $y \leq_P x$. For example, $[C_{x<y<z}]^{op} = C_{z<y<x}$ and $[\vee_x^{y,z}]^{op} = \wedge_{y,z}^x$. Because $\top$ is represented by the discrete poset, we have $[\top]^{op} = \top$.

It follows that linear extensions of $[P]^{op}$ are the duals of all linear extensions of P. This observation suggests the following extension of taking duals.

$$\begin{aligned} [\bot]^{op} &= \bot, \\ [P_1 \oplus P_2]^{op} &= [P_1]^{op} \oplus [P_2]^{op}, \\ [P_1 \otimes P_2]^{op} &= [P_1]^{op} \otimes [P_2]^{op}. \end{aligned}$$

The constraint tables for the deletion block are obtained as follows. Suppose we want to construct a table T_D of constraints for the situation where the letters in the alignment given by Figure 2.2 satisfy the equalities $x_1 = y_1$, $x_2 = y_2, \ldots$ Then we take the table T_I for insertion block in the alignment given in Figure 2.1 with the equalities $\xi(x_1) = \xi(y_1)$, $\xi(x_2) = \xi(y_2), \ldots$ Now, the constraint in T_D for a pair of contexts (u, v) is the dual of the constraint in T_I for the pair $(\xi(u), \xi(v))$. For example, the table of constraints for a deletion block (as in Fig. 2.2) satisfying $b' = c_1$ and $a = c_m$ is given below (it is obtained directly from Table 2.2).

(b', a')	(b', a)	(b, a')	(b, a)
$C_{i<j<k}$	$\wedge_{j,k}^i \oplus \wedge_{i,k}^j$	$C_{j<i<k}$	$\bot$

Table 2.4: Constraints for the situation $b' = c_1$ and $a = c_m$.

Using the above transformation one carries over all the notions and results from the previous section to the case of a deletion block. For example, the concept of a non-principal deletion is defined in the same way, except that condition 4 is replaced by

4' $b' = c_1$, or $a' = c_m$.

A result analogous to Proposition 1 holds for deletion blocks, after replacing the word 'insertion' by 'deletion'.

2.3 Putting it All Together: Construction of Generating Posets

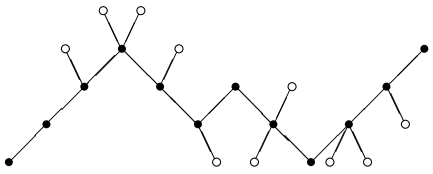
A poset P is called a *hairy zig-zag* if there is an enumeration $P = \{p_1, \ldots, p_n\}$ and a decomposition of P into three disjoint subsets $P = Q \cup H_1 \cup H_2$ such that P is the least poset satisfying

1. $p_1, p_n \in Q$.
2. For every $i < n$, $p_i \in Q$ or $p_{i+1} \in Q$.
3. For every $i < n$, if $p_i \in Q$ and $j > i$ is the least index such that $p_j \in Q$, then p_i and p_j are comparable.
4. For every $p_i \in H_1$, p_i is maximal in P and it is comparable with p_{i-1} or with p_{i+1}.
5. For every $p_i \in H_2$, p_i is minimal in P and it is comparable with p_{i-1} or with p_{i+1}.

P is called a *zig-zag*, if there is a decomposition $P = Q \cup H_1 \cup H_2$ in the above definition with $H_1 = H_2 = \emptyset$. Then conditions (1), (2), (4), (5) hold vacuously and (3) reduces to the property that every two consecutive elements in the above enumeration are comparable.

In the definition of a hairy zig-zag the set Q is a backbone in the form of a zig-zag around which there are attached 'hairs': H_1 oriented upwards, and H_2 oriented downwards.

A typical hairy zig-zag is presented in the figure below. The enumeration for the poset presented below can be read off from the figure by reading the elements from left to right. Discs represent elements of the backbone Q and circles represent the elements which are hairs: $H_1 \cup H_2$.



Let $(V^{\#}, W^{\#})$ be an alignment and let O be a cso. An indel $X \in O$ is said to be *ambiguous* if it is non-principal and (b, c) is collapsing, where $S_{i,a,b} \in O$ and $S_{k,c,d} \in O$ are substitutions surrounding X, from left and right side, respectively. Otherwise X is called *unambiguous*. O is said *unambiguous* if every indel in X is unambiguous.

Finally we define the construction of *poset concatenation*. Given two posets P and Q which have at most one element in common. Their concatenation, $P * Q$, is a poset whose carrier is the union of the two carriers $P \cup Q$ and the partial order is the transitive closure of the union of partial orders $\leq_P \cup \leq_Q$.

Theorem 1. *Given an alignment $(V^{\#}, W^{\#})$. Let O be an admissible complete set of operations and let $m \geq 0$ be the number of ambiguous indels in O. There exist 2^m posets on O: $P_1, \ldots, P_{2^m}$, each being a disjoint union of hairy zig-zags, such that*

1. *For every $i \leq 2^m$, all linear extensions of P_i are admissible for O.*
2. *For every admissible linear order for O, there is an $i \leq 2^m$ such that this order is an extension of P_i.*

In particular, if O is unambiguous, then it has a principal poset and this poset is the intersection of all admissible linear orders on O. The principal poset for O is a disjoint union of hairy zig-zags.

Proof. We identify the operations with their addresses. Hence the elements of the posets we are going to construct are positive integers. Let n be the address of last substitution in $(V^{\#}, W^{\#})$. We show the construction of the posets by induction on the number of substitutions involved in the alignment $(V^{\#}, W^{\#})$. Initially we assume we have one one-element poset $\{1\}$.

Assume we have constructed so far the posets $P_1, \ldots, P_r$ over operations, whose addresses form an initial segment of the set of all addresses for $(V^{\#}, W^{\#})$. Consider the triple of the form $S_{i,a,b}, X_j, S_{k,c,d}$, where $X_j \in O$ is an indel, $S_{i,a,b}, S_{k,c,d} \in O$ are substitutions surrounding X_j (let $i < k$, i.e. the substitution i is to the left of the substitution k) and let k be the smallest address such that $S_{k,c,d}$ has not been considered so far. It follows that X_j has not been considered so far, either.[2] Assume first that $1 < i$ and $k < n$, i.e. neither i nor k are the flanking substitutions of the alignment. Compute a constraint for the pair (b, c) of contexts, according to the rules

[2] We allow the situation when there is no indel between $S_{i,a,b}$ and $S_{k,c,d}$.

described earlier. If X_j is unambiguous, then the constraint is of the form of a poset Q. Otherwise the constraint is of the form $Q_1 \oplus Q_2$, for some posets Q_1, Q_2 (see Proposition 1). In the former case we construct the new posets:

$$P_1 * Q, \ldots, P_r * Q.$$

In the latter case we construct the posets

$$P_1 * Q_1, \ldots, P_r * Q_1, P_1 * Q_2, \ldots, P_r * Q_2.$$

The above construction is slightly different for the case $i = 1$ or $k = n$ (since S_1 and S_n are always admissible). For $i = 1$ we take the constraint for the right context c (use for this Table A.1 in the Appendix). For $k = n$ we take the constraint for the left context b (use Table A.2 in the Appendix).[3]

By the time the construction terminates, we have constructed 2^m posets $P_1, \ldots, P_{2^m}$, where m is the number of ambiguous indels in O. It also immediately follows from the property of the constraints that 1. and 2. of the Theorem hold. It remains to show that each P_i is a disjoint union of hairy zig-zags. As the enumeration in the definition of a hairy zig-zag we take the order on the addresses of the operations as they appear in the alignment. Addresses of the substitutions form the backbone, addresses of the insertions (resp. deletions) form the upward (resp. downward) hairs of the zig-zag. Then it remains to observe that the above obtained posets: Q, Q_1, Q_2 are of a very special form (and there are very few different such forms) and they, as building blocks, maintain the property of being a hairy zig-zag. For example, the constraint $C_{i<k<j}$ is represented as the following building block

i.e., j is going to be an upward hair and i, k are going to be the elements of the backbone zig-zag. When P_i and Q are disjoint, the concatenation $P_i * Q$ produces a disjoint union of posets. ■

Corollary 1. *If $(V^{\#}, W^{\#})$ is an alignment without gaps, then every admissible complete set of operations is unambiguous and it has a principal poset which is a disjoint union of zig-zags.*

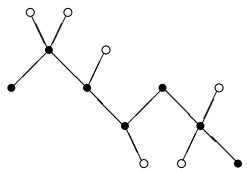
[3]The tables A.1 and A.2 may have to be adjusted, according to the rules described earlier, if there are any collapses of the symbols.

Proof. When there are no indels, then each constraint is of the form of a two element chain. Such constraints generate disjoint unions of pure zig-zags. ■

Theorem 2. *For every hairy zig-zag P there are strings V, W and an alignment $(V^{\#}, W^{\#})$ together with an admissible unambiguous cso O such that the principal poset for O is isomorphic to P.*

Proof. The proof is basically a reverse construction from the proof of Theorem 1. The basic intuition for the present proof is that a given hairy zig-zag we present as a concatenation of building blocks which were mentioned in the proof of Theorem 1. The enumeration of the elements in P suggests an alignment: the number of the operations associated with the blocks in the alignment is equal to the number of the elements in P; elements of the backbone of P correspond to substitutions, upward hairs correspond to insertion blocks, while downward hairs correspond to deletion blocks. For each pair of consecutive substitutions (i.e. consecutive elements of the backbone) we select the context which imposes the constraints for building the block of the poset which is contained between the two elements.

Instead of giving all the details of the proof we illustrate the idea with a sufficiently complicated example. Consider the following hairy zig-zag P.



It has 13 elements, 7 of them (positions: 1,3,5,7,9,11,13) form the backbone, 4 are upward hairs (positions: 2,4,6,12) and 2 are downward hairs (positions: 8,10). This suggests the following alignment.

1	2	3	4	5	6	7	8	9	10	11	12	13
x_1	$-$	x_2	$-$	x_3	$-$	x_4	x_5	x_6	x_7	x_8	$-$	x_9
y_1	y_2	y_3	y_4	y_5	y_6	y_7	$-$	y_8	$-$	y_9	y_{10}	y_{11}

The variables x_i and y_j are metavariables (place holders) for actual symbols of the alphabet. We leave them for the time being unspecified.

Consider the first insertion block (address 2 in the alignment). We have to enforce the constraint $1 < 3 < 2$ for this block. It follows from Chart 1 in

the Appendix that such a constraint is generated by the pair of contexts (x_2, y_1) (this is the pair (b, a') in this chart). In a similar way we obtain the following pairs for the other indels: (y_5, x_2) for insertion 4, (y_7, x_3) for insertion 6, (x_6, y_7) for deletion 8, (y_9, x_6) for deletion 10, and (y_{11}, x_8) for insertion 12.

This choice of the pairs of contexts induces the following substitutions:

$$S_{3,y_1,y_5}, \quad S_{5,x_2,y_7}, \quad S_{7,x_3,x_6}, \quad S_{9,y_7,y_9}, \quad S_{11,x_6,y_{11}}.$$

Thus the complete set of operations for the above alignment, which generates P as the principal poset, is

$$O = \{S_1, I_2, S_{3,y_1,y_5}, I_4, S_{5,x_2,y_7}, I_6, S_{7,x_3,x_6}, D_8, S_{9,y_7,y_9}, D_{10}, S_{11,x_6,y_{11}}, I_{12}, S_{13}\}.$$

Let us notice that if we take two consecutive substitutions, say 3 and 5, then the constraints for choosing the right context for 3 and for choosing the left context of 5 do overlap. For this reason, even thought the substitutions 1 and 13 are always admissible the constraints generated by substitutions 3 and 11 are enough to ensure the right shape of the poset on both of its ends (i.e. the the relative order between the elements 1,2 and 3; as well as between the elements 11,12,13).

In the above reasoning we have been using Chart 1 in the Appendix, i.e. we assumed that the symbols in the corresponding indel blocks are pairwise different. Five symbols are enough to fulfill this requirement. Let the alphabet consists of the following symbols $\{a, b, c, d, e\}$. The following alignment has sufficiently many different symbols

1	2	3	4	5	6	7	8	9	10	11	12	13
a	$-$	b	$-$	a	$-$	b	c	a	c	b	$-$	a
d	c	e	c	d	c	e	$-$	d	$-$	e	c	d

The complete set of operations for the above alignment is given below.

$$O = \{S_1, I_2, S_{3,d,c}, I_4, S_{5,b,e}, I_6, S_{7,a,a}, D_8, S_{9,e,e}, D_{10}, S_{11,a,d}, I_{12}, S_{13}\}.$$

The principal poset generated by the above cso for the above alignment is isomorphic to P. ■

The followng example is a real life one. We present a poset of a local contextual alignment and the corresponding principal poset for two Seryl-tRNA synthetases from *Methanobacterium thermoautotrophicum* (SWISS-PROT accession number O27194) and *Caulobacter crescentus* (TrEMBL accession number Q9A6T4).

As one can see, there is only one hair. It corresponds to the one-element insertion. The poset has 8 connected components (3 of which are isolated points). The maximal chain has 8 elements.

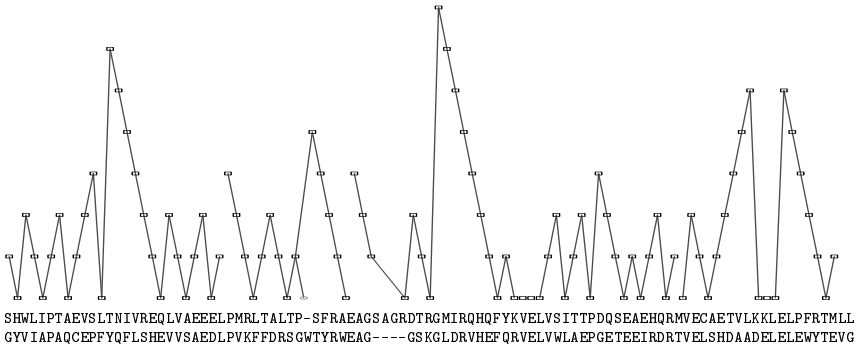


Figure 2.3: Local contextual alignment and the corresponding principal poset for Seryl-tRNA synthetases from *Methanobacterium thermoautotrophicum* and *Caulobacter crescentus*.

Chapter 3

Algorithms for Contextual Alignment

3.1 An algorithm for a gap-free alignment

Below we present a linear time algorithm for contextual alignment without gaps. The input sequences V, W must be of equal length. Note that in the non-contextual case the algorithm is completely obvious. Indeed, if there are no gaps, the alignment is unique and its score is the sum of scores of matches/mismatches at individual locations. In the contextual case, however, it is not only the operations that matter, but also the *order* in which they are performed. Different orders of operations may yield different scores, so our algorithm finds the generating poset of all the optimal admissible chains, as well as computes the score.

We know, that for gap-free alignment, there is always a unique generating poset of all the admissible chains, and that it is a disjoint union of zig-zags (see Corollary 1). Our algorithm is based on dynamic programming and its main idea is to walk along both sequences to be aligned, choosing the optimal one-point extension of the already found poset.

We transform V into W. At step $i+1$, for $i = 2, \ldots, n-1$, we already have two generating posets $P_/$ and $P_\backslash$ of the operations transforming $V_1 \ldots V_i$ into $W_1 \ldots W_i$, and two corresponding scores. They were calculated in step i.

$P_/$ is the generating poset of all the admissible chains which give the optimal score among those in which the substitution of V_i into W_i is performed *after*

the substitution of V_{i-1} into W_{i-1}. Dually, $P_{\backslash}$ is the generating poset of all the admissible chains which give the optimal score among those in which the substitution of V_i into W_i is performed *before* the substitution of V_{i-1} into W_{i-1}. Visually, zig-zag $P_{/}$ ends with /, while $P_{\backslash}$ ends with \. $S_{/}$ and $S_{\backslash}$ are the corresponding optimal scores (recall that the rightmost substitution does not contribute to the score!).

We need to compute the new values for the step $i+1$. We denote them by $P'_{/}$, $S'_{/}$ and $P'_{\backslash}$, $S'_{\backslash}$.

$P'_{/}$ can be obtained either from $P_{/}$ or from $P_{\backslash}$ by extending it by the substitution of V_{i+1} into W_{i+1}, performed *after* the substitution of V_i into W_i. I.e., $P'_{/}$ is chosen from two concatenated posets:

$$P_{/} * C_{i<i+1} \quad \text{and} \quad P_{\backslash} * C_{i<i+1} \tag{3.1}$$

in the following way. The scores corresponding to these two posets are as follows (recall that $M_{a,b}(\cdot,\cdot)$ is a contextual substitution table): $S_{/} + M_{W_{i-1},V_{i+1}}(V_i, W_i)$ and $S_{\backslash} + M_{V_{i-1},V_{i+1}}(V_i, W_i)$, respectively. Then $P'_{/}$ becomes the poset of (3.1) which gives a larger score,[1] and that value becomes $S'_{/}$. $P'_{\backslash}$ and $S'_{\backslash}$ are calculated analogously.

A special case is when $W_i = V_i$ and $W_{i+1} = V_{i+1}$. In this case the operations at addresses i and $i+1$ are not comparable in the principal poset: any order of these two is admissible and gives the same score.

The initialization of the variables is done as follows. When $i = 2$ then $S_{/}$ and $S_{\backslash}$ are set to 0, while $P_{/}$ and $P_{\backslash}$ are set to $C_{1<2}$ and $C_{2<1}$, respectively, unless $V_1 = W_1$, in which case it follows that the operations at addresses 1 and 2 are incomparable in the generating poset.

The algorithm terminates after completing step $n-1$. The output is either $(S_{/}, P_{/})$ or $(S_{\backslash}, P_{\backslash})$, depending on which of the values $S_{/}$ and $S_{\backslash}$ is larger.

3.2 An algorithm for an alignment with gaps

Below we present a *quadratic* (precisely, of $O(|\Sigma|mn)$ time complexity) algorithm for contextual alignment and an affine gap penalty function. The efficiency of the algorithm is based on the observation that for an affine gap penalty function we can compute the score of an indel, by gradually

[1] If posets give the same score, then any one of them is chosen as $P'_{/}$.

extending the indel one symbol at a time. This way, when extending the alignment, it suffices to consider a constant number of cases.

The main idea of the algorithm is to walk along both sequences, choosing the optimal extension of the already found alignment of prefixes. At each step we have the following possible extensions: (i) a single substitution, (ii) an insertion (starting new one or extending the existing one), (iii) a deletion, analogously.

The algorithm uses the dynamic programming approach and works with 7 three-dimensional arrays indexed by the positions from the word V, positions from the word W and the elements of Σ. We use 3 *insertion arrays*, 3 *deletion arrays* and one *substitution array*.

Let us consider a type (ii) extension, i.e., we focus our attention on a single insertion block:

$$\begin{array}{cccccccc} i & j & & & & & & k \\ a & - & - & \cdot & \cdot & \cdot & - & b \\ a' & c_1 & c_2 & \cdot & \cdot & \cdot & c_m & \boldsymbol{x} \\ \alpha & & & & & & \beta & \end{array}$$

For the simplicity of presentation we use, as before, numbers i, j, k as the addresses of the blocks, i.e., the positions in the alignment. The index $\alpha = 1 \ldots n$ is used to number positions in V, and $\beta = 1 \ldots m$ for positions in W. The letter $\boldsymbol{x}$ in the above figure stands for an arbitrary, not yet known letter from Σ, which replaces W_k. Considering all $\boldsymbol{x} \in \Sigma$ is necessary to profit from the affinity of the gap penalty function and to be able extend an insertion gradually in the course of computation. $\boldsymbol{x}$ corresponds to the third dimension in our insertion arrays. Roughly, this third dimension enables us to treat left substitution i and right substitution k separately.

a	a'	c_m
$\vee_k^{i,j}$	$C_{i<k<j}$	$C_{j<k}$

Table 3.1: Constraints for insertion block determined by left contexts for k.

Each insertion array stores at position $(\alpha, \beta, \boldsymbol{x})$ the maximal score of alignment of $V_1 \ldots V_\alpha$ and $W_1 \ldots W_\beta$ which ends with an insertion, under the assumption that $W_{\beta+1} = \boldsymbol{x}$ and that the insertion is immediately followed by a substitution on the right (substitution k in the above figure). We use three insertion arrays, one for each possible selection of the left context for the right substitution k.

- $I_{\vee_k^{i,j}}(\alpha, \beta, \boldsymbol{x})$ stores the maximal score as explained above under the additional condition that a is the left context for the right substitution k (see Table 3.1). This selection results in the constraint $\vee_k^{i,j}$, i.e. the rightmost substitution k precedes insertion j and left substitution i.

- $I_{C_{i<k<j}}(\alpha, \beta, \boldsymbol{x})$ stores the maximal score as explained above under the additional condition that a' is the left context for the right substitution k. This selection results in the constraint $C_{i<k<j}$, i.e. the left substitution is followed by the insertion which is followed by the right substitution.

- $I_{C_{j<k}}(\alpha, \beta, \boldsymbol{x})$ stores the maximal score as explained above under the additional condition that c_m is the left context for the right substitution k. This selection results in the constraint $C_{j<k}$, i.e. the rightmost substitution k is performed after the insertion j.

Note that parameter $\boldsymbol{x}$ together with the subscript of I contain the information which is necessary and sufficient to correctly evaluate an alignment ending with an insertion.

For the dual case of the deletion block we use the following three arrays: $D_{\wedge_{i,j}^k}$, $D_{C_{k<j}}$, $D_{C_{j<k<i}}$. Table 3.2 is obtained from Table 3.1 in the way described in Section 2.2. Together with each entry of tables I and D we store also the selected left context for substitution k. This will be necessary in (3.2) and (3.3) below.

a'	a	c_m
$\wedge_{i,j}^k$	$C_{j<k<i}$	$C_{k<j}$

Table 3.2: Constraints for deletion block determined by left contexts for k.

For the case of the extension by a single substitution we have array $S(\alpha, \beta, \boldsymbol{x})$ It stores the maximal score of an alignment with gaps of words $V_1 \dots V_\alpha$ and $W_1 \dots W_\beta$ which ends with a substitution $V_\alpha \mapsto W_\beta$, whose right context is $\boldsymbol{x}$.

Since our algorithm requires always left and right context for each considered substitution, we need to extend arrays V and W by the new, artificial entries V_0, V_{n+1} and W_0, W_{m+1}. Recall that the leftmost and rightmost substitutions do not contribute to the score of the alignment, they serve only as contexts for the neighboring operations. Hence the choice of the

new entries has a negligible effect on the resulting score — we decided to choose these values in an ad-hoc manner. In the implementation of our algorithm we chose methionine $M = V_0 = W_0$ as the leftmost entry and alanine $A = V_{n+1} = W_{m+1}$ as the rightmost entry.

In the course of computation there are two possibilities to update insertion arrays: (i) starting a new insertion; (ii) extending an existing one. The latter case is simpler: the score of extended alignment is calculated as

$$I_*(\alpha, \beta - 1, \boldsymbol{x}) + GapExt \quad \text{for} \quad * \in \{\vee_k^{i,j}, C_{i<k<j}, C_{j<k}\}$$

When starting a new insertion we have to calculate the score of the substitution which precedes it. To achieve this aim we consider all possible selections of the right context for this substitution. The corresponding scores and constraints are summarized in Table 3.3 below.

right context	constraint	score of extended alignment
b	$\vee_i^{j,k}$	$S(\alpha, \beta - 1, b) + GapOpen$
$\boldsymbol{x}$	$C_{k<i<j}$	$S(\alpha, \beta - 1, \boldsymbol{x}) + GapOpen$
c_1	$C_{j<i}$	$S(i, j' - 1, c_1) + GapOpen$

Table 3.3: Constrains and scores determined by right context for i.

As we are looking for the optimal alignment of prefixes $V_1 \dots V_\alpha$ and $W_1 \dots W_\beta$, we choose the context pair (i.e. the right context for the left substitution and the left context for the right substitution) which does not generate the contradiction and which maximizes the score. The constraints resulting from all possible pairs of contexts for insertion arrays are as follows:

(b, a)	$(\boldsymbol{x}, a)$	(c_1, a)	(b, a')	$(\boldsymbol{x}, a')$	(c_1, a')	(b, c_m)	$(\boldsymbol{x}, c_m)$	(c_1, c_m)
$\bot$	$C_{k<i<j}$	$C_{k<j<i}$	$C_{i<k<j}$	$\bot$	$\bot$	$C_{i<j<k}$	$\bot$	$\vee_j^{i,k}$

E.g. for the left context a there are two admissible pairs $(\boldsymbol{x}, a)$ and (c_1, a) and we should compare the corresponding scores $S(\alpha, \beta - 1, \boldsymbol{x}) + GapOpen$ and $S(\alpha, \beta - 1, c_1) + GapOpen$ with the score of extended insertion $I_{\vee_k^{i,j}}(\alpha, \beta - 1, \boldsymbol{x}) + GapExt$ and choose the maximal value. The update rules for three

insertion arrays are as follows:

$$I_{\vee_k^{i,j}}(\alpha,\beta,\boldsymbol{x}) := \max\left\{\begin{array}{c} I_{\vee_k^{i,j}}(\alpha,\beta-1,\boldsymbol{x}) + GapExt \\ S(\alpha,\beta-1,\boldsymbol{x}) + GapOpen, \\ S(\alpha,\beta-1,c_1) + GapOpen \end{array}\right\}$$

$$I_{C_{i<k<j}}(\alpha,\beta,\boldsymbol{x}) := \max\left\{\begin{array}{c} I_{C_{i<k<j}}(\alpha,\beta-1,\boldsymbol{x}) + GapExt, \\ S(\alpha,\beta-1,b) + GapOpen \end{array}\right\}$$

$$I_{C_{j<k}}(\alpha,\beta,x) := \max\left\{\begin{array}{c} I_{C_{j<k}}(\alpha,\beta-1,\boldsymbol{x}) + GapExt, \\ S(\alpha,\beta-1,b) + GapOpen, \\ S(\alpha,\beta-1,c_1) + GapOpen \end{array}\right\}$$

To calculate a new entry in the substitution table, say $S(\alpha,\beta,\boldsymbol{x})$, we choose the maximal value among the following 8 possibilities:

- 2 cases for the scenario when the substitution $W_\alpha \mapsto W_\beta$ follows another substitution:
  - left substitution precedes the right one: $S(\alpha-1,\beta-1,V_\alpha) + M_{W_{\beta-1},\boldsymbol{x}}(V_\alpha,W_\beta)$
  - right substitution precedes the left one: $S(\alpha-1,\beta-1,W_\beta) + M_{V_{\alpha-1},\boldsymbol{x}}(V_\alpha,W_\beta)$
- 3 cases for the scenario when the substitution follows an insertion.

  $$I_*(\alpha-1,\beta-1,W_\beta) + M_{l_*,\boldsymbol{x}}(V_\alpha,W_\beta) \quad \text{for} \quad * \in \{\vee_k^{i,j}, C_{i<k<j}, C_{j<k}\}, \tag{3.2}$$

  where l_* denotes the corresponding left context in each case. This context is stored together with the array I_*, as mentioned previously.
- 3 cases for the scenario when the substitution follows a deletion.

  $$D_\bullet(\alpha-1,\beta-1,V_\alpha) + M_{l_\bullet,\boldsymbol{x}}(V_\alpha,W_\beta) \quad \text{for} \quad \bullet \in \{\wedge_{i,k}^{j}, C_{j<k<i}, C_{k<j}\}, \tag{3.3}$$

  where $l_\bullet$ denotes the corresponding left context in each case.

The generating posets are reconstructed in the following way. Starting from the maximal position in substitution array, and backtracking in all 7 arrays determines the set of operations performed to achieve the maximal score, and the corresponding alignment. By comparing the contexts we also determine the set of generating posets. Each step of the algorithm, in which the substitution is considered corresponds to a small (two- or three-element)

Algorithm 1 the overview

{ *initialization* }
for $\boldsymbol{x} \in \Sigma$ **do** $S(0,0,\boldsymbol{x}) := 0$;
for $\boldsymbol{x} \in \Sigma$, $\alpha > 0$, $\beta > 0$ **do** $S(\alpha,0,\boldsymbol{x}) := -\infty$; $S(0,\beta,\boldsymbol{x}) := -\infty$;
for $\boldsymbol{x} \in \Sigma$, $\alpha = 0 \ldots n$, $* \in \{\vee_k^{i,j}, C_{i<k<j}, C_{j<k}\}$ **do** $I_*(\alpha,0,\boldsymbol{x}) := -\infty$;
for $\boldsymbol{x} \in \Sigma$, $\beta = 0 \ldots m$, $\bullet \in \{\wedge_{j,k}^{i}, C_{j<k<i}, C_{k<j}\}$ **do** $D_\bullet(0,\beta,\boldsymbol{x}) := -\infty$;
{ *end of initialization* }
for $\alpha = 1 \ldots n$ **do**
  for $\beta = 1 \ldots m$ **do**
    for $\boldsymbol{x} \in \Sigma$ **do**
      calculate the 7 arrays at position $(\alpha, \beta, \boldsymbol{x})$.
    end for
  end for
end for

$$\textbf{Score} := \max \left\{ \begin{array}{c} S(n,m,V_{n+1}), S(n,m,W_{m+1}), \\ I_*(n,m,W_{m+1}), D_\bullet(n,m,V_{n+1}) \end{array} \;\middle|\; \begin{array}{l} * \in \{\vee_k^{i,j}, C_{i<k<j}, C_{j<k}\} \\ \bullet \in \{\wedge_{j,k}^{i}, C_{j<k<i}, C_{k<j}\} \end{array} \right\}$$

poset, and by concatenating these posets in reversed order, we reconstruct the generating poset (potentially with ambiguous indels). Each small poset is determined (tables from the Appendix are used here) by the partition of the set of symbols which serve as the contexts for the corresponding pair of substitutions.

Correctness of the algorithm. It can be proved by induction on α and β that all insertion, deletion and substitution tables are assigned the correct values in the course of the computation at position $(\alpha, \beta, \boldsymbol{x})$, for every $\boldsymbol{x}$. The case of substitution is straightforward, since our algorithm tries all possibilities. The case of insertion (and deletion, dually) needs a little care. As candidates for the best score at position (α, β) we consider:

(i) alignment ending with a singleton insertion;

(ii) the extension of the best alignment at position $(\alpha, \beta - 1)$.

Correctness of this step follows from the following lemma.

Lemma 1. *Assume that the optimal alignment of $V_1 \ldots V_\alpha$ and $W_1 \ldots W_\beta$ ends with an insertion of length $l \geq 2$, and that $W_{\beta+1} = \boldsymbol{x}$, and let the*

left context of the right substitution be c. Then the optimal alignment of $V_1 \dots V_\alpha$ and $W_1 \dots W_{\beta-1}$ ends with an insertion of length $l-1$, under the assumption that W_β equals $\boldsymbol{x}$ and that left context of the right substitution is c.

Proof. Follows immediately from the assumption of the affinity of gap penalty function. ■

Algorithms for local alignment. As with the non-contextual case (see [SW81]), the algorithm for local alignment is in each case a minor modification of the global alignment algorithm. The idea is to add an additional option while filling in the arrays: to give up with the so far constructed alignment and start all over in the middle of the sequences by resetting the score to 0. The local versions of the above algorithms inherit the complexity of their global counterparts.

Chapter 4

Substitution tables

The idea of the approach is to a large extent based on the Henikoff & Henikoff BLOSUM table [HH92].

- The main source of data are blocks of gap-free aligned sequences.
- The method to eliminate the influence of large number of highly similar sequences is by clustering such sequences together and subsequently weighing the contribution of each cluster equally, no matter how many sequences it contains.

However, we have goals substantially extending those faced by the Henikoffs. First of all, we need to make the substitution score dependent of the context. Next, we want to break up the symmetry of the tables.

The former is conceptually not difficult, however, it requires large amounts of data to yield statistically significant results. For the latter, we decided to adopt the convention that: *at each position, the relatively most frequent residue is the primitive one,* being aware, that it is a highly speculative one.

Input data. The input data are blocks of many gap-free aligned protein sequences. In particular, all sequences in one block are of equal length.

Parameters. The following are the parameters of the algorithm, and their choice affects the resulting tables.

- The clustering constant $p \in (0, 1)$.

- The significance threshold ST.

Clustering. In order to compensate for the high influence of many highly similar sequences, we introduce clustering. We cluster together sequences which share more than a fraction of p of residues for the purpose of creating the statistics, exactly as it has been done while creating BLOSUM (op cit. [HH92]). All blocks and all positions are taken into account.

Clusters in a block are connected components of the graph, whose vertices are sequences in the block and whose edge relation $\sim$ is defined as follows:

$$(s \sim t) \iff \frac{|\{i : 1 \leq i \leq |s| \ \&\ s(i) = t(i)\}|}{|s|} \geq p.$$

Subsequently we assume that each block is clustered. The cluster of the sequence s in block B is denoted $[s]_B$.

If B' is a subset of block B, then the cluster of the sequence s in B' is $[s]_{B'} := [s]_B \cap B'$, even though the latter set need not be connected in the graph B' with edge relation $\sim$. We adopt this choice to avoid the large cost of recomputing the clusters over and over again.

Henceforth we assume all blocks to be clustered and all their subsets to inherit the clusters in the way described above.

Identifying contexts. For each block B and each pair of positions $i, i+2$ not exceeding the common length of the sequences in B, and for each choice of amino acids a and b we create the subblock $B^{a,b}_{i,i+2} = \{s \in B : s(i) = a \ \&\ s(i+2) = b\}$.

Subsequently, the subblocks $B^{a,b}_{i,i+2}$ over all B and all i are used to calculate the frequencies and substitution rates in the context a_b.

Frequencies in the given context. For every triple a, b, c of amino acids and each cluster W in each subblock $B^{a,b}_{i,i+2}$ we calculate the frequency of c found between a and b in W, denoted $f^{a,b}_W(c)$ in W, by the following formula:

$$f^{a,b}_W(c) := \frac{|\{s \in W : s(i+1) = c\}|}{|W|}.$$

Now the global frequency $f_{a,b}(c)$ of c being found between a and b is defined by

$$f_{a,b}(c) := \text{average}\left\{ f_W^{a,b}(c) : \begin{array}{c} W \text{ is a cluster of } B_{i,i+2}^{a,b}, \\ B \text{ is a block, } i, i+2 \leq \text{the length of } B \end{array} \right\}.$$

Note that averaging over the *real* frequencies in clusters amounts exactly to imposing that the contribution of each cluster is equal, irrespectively of its cardinality.

The primitive amino acid. Now, for each subblock $B_{i,i+2}^{a,b}$ the amino acid c for which the ratio

$$\frac{f_{B_{i,i+2}^{a,b}}(c)}{f_{a,b}(c)}$$

is the highest, is assumed to be *primitive* at position $i+1$ in $B_{i,i+2}^{a,b}$. This particular c is denoted $c_{B_{i,i+2}^{a,b}}$.

Mutation rates. For each quadruple a, b, c, d of amino acids the observed mutation rate of c into d in the context a_b, denoted $m_{a,b}(c,d)$ is given by the formula

$$m_{a,b}(c,d) := \text{average}\left\{ f_W^{a,b}(d) \cdot f_W^{a,b}(c) : \begin{array}{c} B \text{ is a block,} \\ W \text{ is a cluster of } B_{i,i+2}^{a,b}, \\ i, i+2 \leq \text{the length of } B, \\ c = c_{B_{i,i+2}^{a,b}},\ |W| \geq ST \end{array} \right\}.$$

In this formula, again weighing each cluster equally (but excluding too small clusters, where the frequencies are statistically insignificant), we calculate the frequency of c being replaced by d, in the context a_b.

The expected (under the null hypothesis) mutation rate of c into d in the context a_b, denoted $M_{a,b}(c,d)$ is given by the formula

$$M_{a,b}(c,d) := f_{a,b}(d) \cdot f_{a,b}(c).$$

Note that the null hypothesis is symmetric, since it assumes that the residues are aligned at the same position by a pure accident, and therefore neither of them is primitive (or distinguished in any other sense).

Log-odds. The score of substituting c by d in the context a_b, i.e., the entry in the tables we are creating, is defined by the formula

$$score_{a,b}(c,d) := \log_2 \left(\frac{m_{a,b}(c,d)}{M_{a,b}(c,d)} \right).$$

This method defines the scores as log-odds of the observed and expected mutation rates. For non-contextual gap-free alignment it has been proved by Altschul [Alt91] that this is essentially the only choice, because even tables constructed in another way effectively generate a model of substitution with mutation rates whose log-odds give back the values from the tables.

4.1 The Difficulties

The above algorithm has been implemented by T. Gajewski [Gaj01]. It appears that even for moderate values of ST and p many values in the tables remain undetermined, because the base of blocks used does not provide enough data. For some substitutions in some contexts, even if they happen in the base, they are either removed by the principle of disregarding clusters of cardinality smaller than ST, or the amino acid to be substituted is indeed never chosen as the primitive one.

Being aware of such a risk, we can propose several remedies. Which of them will turn out to give the best results remains to be seen.

4.1.1 Reduced context tables

A context reduction function is any mapping ϕ from the set of amino acids to another set D of reduced contexts. An example could be $D = \{H, P\}$ with

$$\phi(c) = \begin{cases} H & \text{if } c \text{ is hydrophobic,} \\ P & \text{if } c \text{ is polar.} \end{cases}$$

Then, in order to obtain an algorithm for reduced context tables, we modify the following fragments of the main algorithm.

Identifying contexts. For each block B and each pair of positions $i, i+2$ not exceeding the common length of the sequences in B, and for each choice of $a, b \in D$ we create the subblock $B^{a,b}_{i,i+2} = \{s \in B : \phi(s(i)) = a \ \& \ \phi(s(i+2)) = b\}$.

Subsequently, in the whole algorithm a and b range over elements of D, and, in particular, the contexts are pairs of elements of D.

The reduced context tables can be used as such, for reduced context alignment, or as a source of missing values in the general tables. For the latter purpose, one can substitute the undetermined values $score_{a,b}(c,d)$ in the general tables by the values $score_{\phi(a),\phi(b)}(c,d)$ for a suitable context reduction function ϕ.

Another, quite different purpose of reduced context tables is to use them as a limiting case, allowing one to compare the tables produced by our algorithm with their non-contextual inspiration, the BLOSUM tables.

In order to achieve that, one takes a *constant* context reduction function. This amounts to saying that all contexts are the same, i.e., the context does not play any role in the construction of the tables. The outcome tables can be then directly compared with the BLOSUM tables (keeping in mind that BLOSUM is symmetrical, while our tables are not). It turns out that they are indeed quite similar[Gaj01].

4.2 The Tables

Using the presented method we have constructed several dozens of families of contextual substitution matrices. Each set of matrices is the outcome of our procedure for a fixed set of parameters' values. Among all the input parameters the most interesting are: the source of blocks and the context reduction function.

The source of data. As the source data for our procedure we took two different data-bases of biological sequences:

- BLOCKS Database [HGPH00, HHP99]: blocks are multiply aligned (without gaps) segments corresponding to the most highly conserved regions of proteins. The blocks are made automatically by looking for the most highly conserved regions in groups of proteins documented in the Prosite Database.

- COGs Database [TKL97]: Clusters of Orthologous Groups of proteins (COGs) were created by aligning protein sequences encoded in 44 complete genomes, representing 30 major phylogenetic lineages. Each COG consists of individual proteins or groups of paralogs from at least 3 lineages and thus corresponds to an ancient conserved domain. The alignments found in COGs may contain gaps.

  The database of COGs does not contain itself blocks of sequences which can be directly used by our procedure, which requires gap-free alignments of multiple sequences. Therefore we extracted gap-free blocks from the multiple alignments cutting off the maximal gap-free fragments of the alignments.

The comparison of matrices built from these two datasets shows that, despite of quite different origins of molecular sequences (mammalian protein sequences in the case of BLOCKS vs. whole genomic sequences of several microorganisms in the case of COGs) the resulting matrices are not significantly different. The explanation for this phenomenon is very rigorous clustering which is unavoidable when identifying contexts.

Partitions of the set of amino acids. The appropriate choice of context reduction function is crucial for the applications, particularly because the quantity of available data is too small for creating full-context tables. It seems reasonable to consider the partition of the set of amino acids which reflects their chemical properties and which is suitable for studying the molecular evolution.

We have made some preliminary experiments using three different context reduction functions:

- Full context, i.e. we consider 400 different pairs of contexts. The resulting matrices have a number of non-determined entries, which should be filled with reduced context values. In our experiments these values are taken from corresponding tables for 6 groups of context. The number of non-determined entries depends on: the clustering constant, the significance threshold and the context reduction function. For full context tables this number is between 70384 and 87787, while for 6 context group table it is between 12 and 302.
- Contexts with 6 groups. These groups are based on accepted point mutation data [DSO78]. The molecular sizes and shapes are very similar within each group.

- HP contexts. We consider the partition of amino acids into the polar ones $\{C, D, E, G, H, K, N, Q, R, S, T, Y\}$ and the hydrophobic ones $\{A, F, I, L, M, P, V, W\}$.

GROUP NAME	AMINO ACID RESIDUE
Small Aliphatic	Alanine, Proline, Glycine
Acid amide	Glutamine, Aspargine, Glutamic Acid, Aspartic Acid
Hydroxyl & Sulfhydryl	Serine,Threonine, Cysteine
Aliphatic	Valine, Isoleucine, Methionine, Leucine
Basic	Lysine, Arginine, Histidine
Aromatic	Phenylalanine, Tyrosine, Tryptophan

4.3 Evaluation of the Tables

4.4 Statistical Properties of the Tables

Context sensitivity. To see how much the scores are indeed context-sensitive, we took the contextual substitution matrix in the case of 6 context groups. For each substitution we calculated the minimal and maximal value of the substitution score over all possible contexts, the mean and the standard deviation. If the substitution rates (and thus also scores) were indeed context-independent, all those values should be almost equal, except the standard deviation, which should be around 0. The substitutions which are strongly context-dependent are listed below.

substitution	mean	standard deviation	min	max
His → Pro	-1.487	6.703	-12.498	0.276
Val → Asp	-4.521	4.794	-11.918	-2.577
Asp → Met	-3.21	4.335	-9.762	-0.524
Gly → Met	-3.427	3.988	-13.401	-1.393
Pro → Met	-1.86	3.646	-12.339	-0.349
His → Asp	-0.108	3.603	-10.915	0.946
Ala → Asp	-3.606	3.357	-10.643	-1.2
Arg → Cys	-2.426	3.053	-10.9	0.106
Glu → Cys	-3.351	3.044	-11.488	-0.325
Thr → Cys	-1.602	2.974	-10.658	0.403
Cys → Lys	-0.857	2.954	-10.085	0.535
Ile → Asn	-3.174	2.933	-11.675	-1.024
Tyr → Asp	-2.088	2.807	-10.506	-0.136
Leu → Cys	-3.053	2.618	-9.492	-0.44
Met → His	-0.555	2.438	-7.35	1.347

Asymmetry in the tables. The asymmetry in matrices implies that the score of an optimal alignment depends on the order of compared sequences. I.e., the score of transforming sequence W into sequence V can differ from the score of the reverse transformation. In some cases this difference can be significant. In general the degree of asymmetry observed does not depend crucially on the number of context groups, which is illustrated below. This rules out the possibility that the change of the shape of the empirical curves with increasing number of context groups, are due to the context influence rather than growing asymmetry of the tables.

The relative entropy. The relative entropy is defined as a weighted average of all substitution scores, where weights are the observed frequencies of substitutions [Alt91]. In out setting we define the entropy as follows:

$$H = \sum_{a,b} \sum_{c,d} m_{a,b}(c,d) score_{a,b}(s,d) = \sum_{a,b} \sum_{c,d} m_{a,b}(c,d) \log_2 \left(\frac{m_{a,b}(c,d)}{M_{a,b}(c,d)} \right).$$

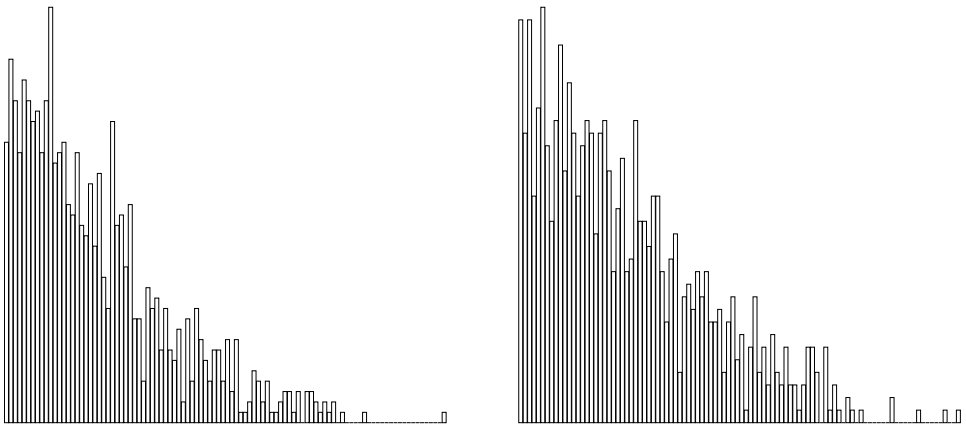
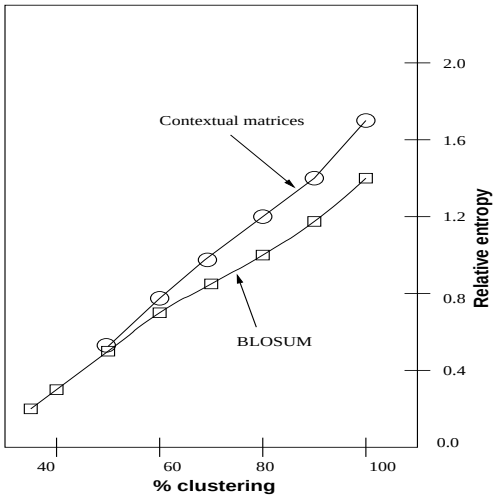


Figure 4.1: The distribution of differences between the score of an alignment versus the score of the reverse-direction alignment. The histograms of differences are shown for all pairs of sequences from $COG0013$ (Alantyl-tRNA synthetase). The left one is for contextual tables (6 groups of context) and the right one is for one-context (i.e., non-contextual) tables. The maximal difference is 169, rather low if compared with the maximal score 2051.



In information theoretic terms, H is the relative entropy of the target and background distributions and intuitively it measures the average information available per position to distinguish the alignment from chance. The figure above illustrates the relationship between p, the parameter of clustering, and relative entropy for contextual tables compared with the results for the

BLOSUM family.

Chapter 5

Statistical significance of contextual alignment

The outcome of the alignment algorithm is a raw score which is a function of the chosen similarity matrix and gap penalty. These raw scores depend upon the length of the original sequences. Therefore, the measure of the quality of the alignment is needed to ensure that a high raw score is obtained due to the homology and not due only to aligning long sequences. The best method used to estimate the statistical significance of the alignment is to approximate the probability distribution of scores from aligning random sequences. Then the raw score can be expressed as the likelihood of achieving such a score or higher by aligning random sequences of the same lengths.

5.0.1 Global alignments.

Unfortunately, under even the simplest non-contextual models and scoring systems, very little is known about the probability distribution of optimal global alignment scores. The standard approach here is to express the score of interest in terms of standard deviations from the mean (Z-value), or to perform Monte Carlo simulations that provide rough distributional results.

The analogous approach can be applied in the case of contextual global alignment. Particularly, it would be of interest to adopt the method from [WB01], which empirically derive the distribution of Z-values of scores, to the case of the contextual setting.

5.0.2 Local alignments.

In the case of local non-contextual alignments the statistics for the scores are well understood. The precise mathematical results are given for ungapped alignments [KA90, DKZ94]. If the presence of gaps is allowed analytical derivations exist only for some special cases of scoring systems [AW94, MT99]. However, by extensive simulation experiments the applicability of the method for the ungapped case for the analysis of alignments with gaps has been verified [Mot92, WV94, WV94a, VW95].

In this section we recall the main results concerning the statistical significance of local gap-less non-contextual alignment and we show how to generalize them to work in the contextual setting. In particular, we prove that the probability distribution of properly normed contextual scores follows approximately the Gumbel Extreme Value Distribution (EVD) [Gum58].

The outline of this section is as follows. First, we formulate the results for the non-contextual case. Then we give the analogous statement when two aligned sequences are generated by two independent Markov chains. We show that the contextual alignment model can be expressed in this way.

As a consequence the statistical significance of contextual alignment can be estimated using the method analogous to that of [VW95].

A standard extreme value distributed random variable G is defined by

$$P(G < t) = \exp(-\exp(-t)).$$

Shifting and rescaling yields a two-parameter family of variables

$$H = \theta G + \xi,$$

where ξ is called the *location* and θ the *scale* of H.

The following theorem states that the maximal gap-less local alignment score of two independent sequences with independent and identically distributed letters asymptotically follows the Gumbel distribution.

Theorem 3 ([DKZ94]). *Let $X_1 \dots X_n$ and $Y_1 \dots Y_n$ be two independent i.i.d. sequences with values in some finite alphabet Σ distributed according to μ_1 and μ_2, respectively. Let $S : \Sigma \times \Sigma \to \mathcal{R}$ be a score function, and let θ be a positive solution of $E_{\mu_1 \otimes \mu_2}[\exp(S\theta)] = 1$, i.e.:*

$$\sum_{(a,b) \in \Sigma \times \Sigma} \exp(\theta S(a,b)) \mu(a) \mu(b) = 1.$$

Let H_n denotes the optimal ungapped alignment score, i.e.:

$$H_n = \max_{i,j,L} \sum_{k=0}^{L-1} S(X_{i+k}, Y_{j+k}).$$

There exists a constant K such that

$$\lim_{n\to\infty} P(H_n \leq \frac{2\log(n)}{\theta} + t_n) \cdot \exp(K \exp(-\theta t_n)) = 1$$

for any bounded sequence t_n such that $t_n + \frac{2\log(n)}{\theta}$ is a possible value of the score.

In the case of i.i.d. letters the Gumbel approximation is a consequence of approximating the number of matching regions scoring above some threshold by a Poisson distribution. It is assumed that the expected score per letter is negative. Hence positive scoring regions are rare events. These events are clearly dependent, thus the classical Poisson approximation that deals with counts of independent rare events cannot be applied. To deal with the dependencies the Chen-Stein method is used. It allows one to estimate the quality of approximation: the Chen-Stein theorem says that the variation distance[1] between the count variable W, which counts the number of high scoring regions and the Poisson variable $Z(\lambda)$ is comparable to the discrepancy of their second moments, i.e. $E[W^2] - E[Z(\lambda)^2]$.

The analogous statement for asymptotic behavior of the score when the two aligned sequences are modeled by Markov chains has been proven recently in [Han03]. The proof of Theorem 4 below generally follows the same methodology. Gumbel approximation is a consequence of an underlying Poisson process approximation of the positions in the score matrix exceeding a high threshold. However, this Poisson approximation requires several assumptions about the structure of the considered Markov chains and the score function.

Consider two independent stationary irreducible and aperiodic Markov chains $(X_k)_{k\geq 1}$ and $(Y_k)_{k\geq 1}$ on finite state-space $\mathcal{E}$ defined by transition probabilities matrices P and Q, respectively, with invariant probability distributions π_P and π_Q. Assume that, these Markov chains generate two sequences to be aligned. We are going to analyze the behaviour of the stochastic process called Markov additive process (MAP). MAP corresponds to a random

[1] *Variation distance* between two probability distributions ν, μ on $\mathcal{X}$ is defined by $\|\nu - \mu\| = \frac{1}{2}\sum_{x\in\mathcal{X}} |\nu(x) - \mu(x)| = \max_{A\subseteq\mathcal{X}} |\nu(A) - \mu(A)|$.

walk, whose increments are controlled by a Markov chain. For a given score function $S : \mathcal{E} \times \mathcal{E} \to R$ define a random walk $(S_n)_{n\geq 1}$ by

$$S_n = \sum_{k=1}^{n} S(X_k, Y_k).$$

The increments of this walk are controlled by two-dimensional Markov chain $(X_k, Y_k)_{k\geq 0}$ with the state space $\mathcal{E} \times \mathcal{E}$ and transition probabilities $P \otimes Q$, where $\otimes$ is a direct product (Kronecker product) of two matrices.

For $\theta \in R$ define a $\mathcal{E} \times \mathcal{E}$ matrix $\Phi(\theta)$ as follows:

$$\Phi(\theta)_{(x_0,y_0),(x_1,y_1)} = \exp(\theta S(x_1, y_1))P_{x_0,x_1}Q_{y_0,y_1}.$$

Entries of this matrix correspond to moment generating functions for increment distribution. The change of state from (x_0, y_0) to (x_1, y_1) contributes $S(x_1, y_1)$ to a random walk S_n. The matrix $\Phi(\theta)$ is clearly a positive matrix with real eigenvalues. Denote by $\phi(\theta)$ its spectral radius (i.e., its largest eigenvalue, called also the Perron-Frobenius eigenvalue). Let $\theta^* > 0$ be the unique solution of

$$\phi(\theta) = 1 \tag{5.1}$$

It is proved in [Han03] that the existence and uniqueness of θ^* is guaranteed due to the two-dimensional negative drift condition:

(i)
$$\sum_{x,y\in\mathcal{E}} S(x, y)\pi_P(x)\pi_Q(y) < 0$$

Define similarly the $\mathcal{E}^3 \times \mathcal{E}^3$ matrices $\Phi_i(\theta)$ for $i = 1, 2$ by

$$\Phi_1(\theta)_{(x_0,y_0,z_0)(x_1,y_1,z_1)} = \exp(\theta S(x_1, y_1) + \theta S(x_1, z_1))P_{x_0,x_1}Q_{y_0,y_1}Q_{z_0,z_1}$$
$$\Phi_2(\theta)_{(x_0,y_0,z_0)(x_1,y_1,z_1)} = \exp(\theta S(x_1, z_1) + \theta S(y_1, z_1))P_{x_0,x_1}P_{y_0,y_1}Q_{z_0,z_1}$$

Let $\phi_i(\theta)$ be the Perron-Frobenious eigenvalues for $\Phi_i(\theta)$ for $i = 1, 2$.

We require that Markov chains P and Q fulfill the following regularity condition:

(ii)
$$\phi_1\left(\frac{3}{4}\theta^*\right) < 1 \quad \text{and} \quad \phi_2\left(\frac{3}{4}\theta^*\right) < 1,$$
where θ^* is the solution of (5.1).

The score function S has to satisfy the following non-degeneracy conditions w.r.t $P \otimes Q$:

(iii) there exist cycles (x_1, x_2) w.r.t. P and (y_1, y_2) w.r.t. Q satisfying
$$S(x_1, y_1) + S(x_2, y_2) \neq S(x_1, y_2) + S(x_2, y_1).$$

(iv) there exist $n > 0$ and two cycles $(x_1, \ldots, x_n)$ and $(y_1, \ldots, y_n)$ such that
$$\Sigma_{k=1}^{n} S(x_k, y_k) > 0.$$

The following theorem is an analogue of Theorem 3.

Theorem 4 ([Han03]). *Let $X_1 \ldots X_n \ldots$ and $Y_1 \ldots Y_n \ldots$ be two independent, stationary, irreducible and aperiodic Markov chains with finite state space $\mathcal{E}$, transition probabilities P and Q and invariant measures π_P and π_Q, respectively. Let $S : \mathcal{E} \times \mathcal{E} \to \mathcal{R}$ be a score function satisfying (i), (iii) and (iv). Assume that the condition (ii) is fulfilled for θ^* defined as in (5.1). Let H_n denote the optimal ungapped alignment score of $X_1 \ldots X_n$ with $Y_1 \ldots Y_n$. Then there exists a constant K^* such that*
$$\lim_{n\to\infty} P(H_n \leq \frac{2\log(n)}{\theta^*} + t_n) \cdot \exp(K^* \exp(-\theta^* t_n)) = 1$$
for any bounded sequence t_n such that $t_n + \frac{2\log(n)}{\theta^*}$ is a possible value of the score.

The explicit formula for the constant K^* can be found in [Han03].

Markov chain alignment and contextual model. We apply Theorem 4 to the statistics for ungapped contextual alignment. To this aim we define Markov chains to be aligned and a contextual scoring function.

¿From Corollary 1 we know that for an ungapped contextual alignment every admissible set of operations is unambiguous and the principal poset

is a disjoint union of zig-zags. The optimal alignment is maximized for all admissible cso's. As we need to evaluate the score function, the cso should be fixed. To this aim we can define the random variable equal to the maximum local score $H_{n|z}$ given a fixed cso z and then to ask for the distribution of the random variable

$$H_n^{opt} = \max_{\text{all admissible cso's } z} H_{n|z}.$$

If all $H_{n|z}$ were independent and had EVD distribution it would follow from the so called max-stability property of EVD family that the new maximum was also EVD distributed [Gum58]. Unfortunately in our case these variables are not independent.

To overcome this difficulty we work with averaged contexts, i.e., we penalize the substitution also considering two surrounding letters but we do not care about the relative order of operations. Instead, we consider all the possible contexts (there are at most 4 of them if indels are not allowed) for the substitution and take the average of their contextual scores.

This approach differs from the real contextual alignment. Even in the standard non-contextual score, e.g. given by a BLOSUM matrix, each entry can be viewed as an average over all 400 possible pairs of context.

However, extensive simulations shows that the difference of distributions of alignment scores between the true contextual alignment and the 'averaged' contextual model is marginal (data not published). The latter model has been indeed used in [GO03] for contextual multiple alignment.

Let $\mathcal{E} = \Sigma^3$ be the state space of the Markov chain $\mathcal{M}$. The transition probabilities matrix is defined according to the distribution on the set of amino acids (say $\mu : \Sigma \to [0, 1]$), i.e.

$$P_{(abc)(bcd)} = \mu(d) \quad \forall a, b, c, d \in \Sigma.$$

The Markov chain $\mathcal{M}$ is clearly irreducible, aperiodic and stationary. We consider two sequences generated independently by two Markov chains with transition probability matrix P. Define the averaged contextual score function S.

$$S : \mathcal{E} \times \mathcal{E} \to R$$
$$S((abc),(def)) = \frac{1}{4}\left(score_{a,c}(b,e) + score_{a,f}(b,e) + \right.$$
$$\left. + score_{d,c}(b,e) + score_{d,f}(b,e)\right)$$

where $score_{a,c}(b,e)$ is the score of substituting b by e with left context a and right context c. To apply Theorem 4 we need to verify assumptions (i)-(iv) for the Markov chain $\mathcal{M}$ and the score function S.

Recall that our contextual scores are defined as log-odds of the observed and expected mutation rates. Hence the negative drift condition (i) is fulfilled. Conditions (iii) and (iv) are easy to check for any given contextual substitution table. The most non-intuitive condition is (ii). Its verification is also difficult because it requires the spectral analysis of huge matrices. With standard computational linear algebra packages we have managed to verify (ii) only in reduced context models. Nevertheless we strongly believe that it remains true for the full-context situation, too.

For example consider the simple situation when two sequences are generated as i.i.d. from the distribution μ. We know that in this case Theorem 3 holds but our aim is to verify regularity condition (ii). As a score function S we take BLOSUM62 and assume μ to be the standard distribution of amino acids in protein sequences. The spectral radius of $\Phi(\theta)$ can be calculated explicitly:

$$\phi(\theta) = \Sigma_{i,j\in\mathcal{E}} \exp(\theta S(i,j))\mu_i\mu_j.$$

The equation $\phi(\theta) = 1$ can be numerically solved, yielding $\theta^* = 0.002127144711$. The spectral radius of $\mathcal{E}^3$ matrices $\Phi_i(\theta)$, for $i = 1, 2$ can be calculated in the same way:

$$\phi_1(\theta) = \Sigma_{i,j,k\in\mathcal{E}} \exp(\theta(S(i,j) + S(i,k)))\mu_i\mu_j\mu_k$$
$$\phi_2(\theta) = \Sigma_{i,j,k\in\mathcal{E}} \exp(\theta(S(i,k) + S(j,k)))\mu_i\mu_j\mu_k$$

We can verify condition (ii) now:

$$\phi_1\left(\frac{3}{4}\theta^*\right) = 0.9999967752 < 1$$
$$\phi_2\left(\frac{3}{4}\theta^*\right) = 0.9999953183 < 1$$

It is conjectured in [Han03] that condition (ii) is too strong in the case of symmetric score function.

Conjecture 1. *If two independent Markov chains to be aligned are identical (i.e. $P = Q$) the condition (ii) is obsolete for Theorem 4 to hold, provided the symmetry of the score function S (i.e. $S(x, y) = S(y, x)$ for any $x, y \in \mathcal{E}$).*

This conjecture, if proven, would imply the EVD approximation for all 'averaged' contextual alignment scenarios, in which the used tables are symmetric.

Chapter 6

Experimental Results

We have performed elementary validation of the contextual alignment model. We have found some interesting phenomena, suggesting that further work in this direction is worthwhile.

The experiments reported here are based purely on score values. We decided to assess mainly the influence of the context on the score value. For this we have used four substitution tables: $\mathrm{CONT}_1 62$, $\mathrm{CONT}_2 62$, $\mathrm{CONT}_6 62$ and $\mathrm{CONT}_{20} 62$ (but, as explained above, about one third of the entries had to be taken from $\mathrm{CONT}_6 62$). The first table assumed all amino acids to be identical, hence the table is indeed non-contextual. The experiment consisted in considering a group of homologous proteins and for each pair of proteins from the group, comparing the Smith-Waterman optimal score under the non-contextual table $\mathrm{CONT}_1 62$ against the optimal score of the contextual alignement under one of the contextual tables: $\mathrm{CONT}_2 62$, $\mathrm{CONT}_6 62$ and $\mathrm{CONT}_{20} 62$.

We decided to use the database of Clusters of Orthologous Groups of proteins [TNG01] (see also the NIH COG page `http://www.ncbi.nlm.nih.gov/COG`). It consists currently of 3307 COGs including 74059 proteins from 43 genomes of bacteria, archaea and the yeast *Saccharomyces cerevisiae*. COG database represents an attempt of a phylogenetic classification of the proteins encoded in complete genomes. Each COG includes proteins that are thought to be orthologous, i.e. connected by vertical evolutionary descent.

We restrict our attention to the list of 84 COGs, which contain at least one protein from each genome. ¿From this list 27 COGs (which include as few paralogs as possible) are selected to our analysis. They can be parti-

tioned into 2 groups; the first one consists of 12 COGs, which represents a wide spectrum of functional categories, and the second consists of 15 *tRNA synthetases* families.

This experiment had the following goals.

1. Estimation of the statistical significance of contextual alignment vs. non-contextual one.
2. Observation of the impact the number of context groups has on the discriminating power of contextual approach.
3. Verification of the accuracy of contextual scores w.r.t. the classification of proteins inside COGs and evolutionary relationships between proteins.

6.1 Statistical significance of contextual vs. non-contextual alignment

To estimate the statistical significance of our alignments we calculated Z-value [CAGRHSC99] in the case of global alignment and we adopted the method of Vingron and Waterman [VW95] of calculating P-value for local alignments. The sequences from each COG are pairwise aligned globally and locally and the statistical significance (Z-value and P-value, respectively) are computed.

The majority of aligned pairs of proteins gives better statistical significance measures for contextual approach. A typical outcome is listed in Figure 6.1: + indicates that the calculated P-value was 2 orders of magnitude smaller for contextual alignment when compared with the non-contextual one. Figure 6.1 contains the results for locally aligned proteins from COG0030.

6.2 Impact of the number of context groups on the discriminating power of the contextual alignment

The results of such a comparison of local alignments for all proteins of COG0089 (Ribosomal proteins—large subunit A, see the NIH COG page `http://www.ncbi.nlm.nih.gov/COG`) are presented in Figure 6.2. Observe

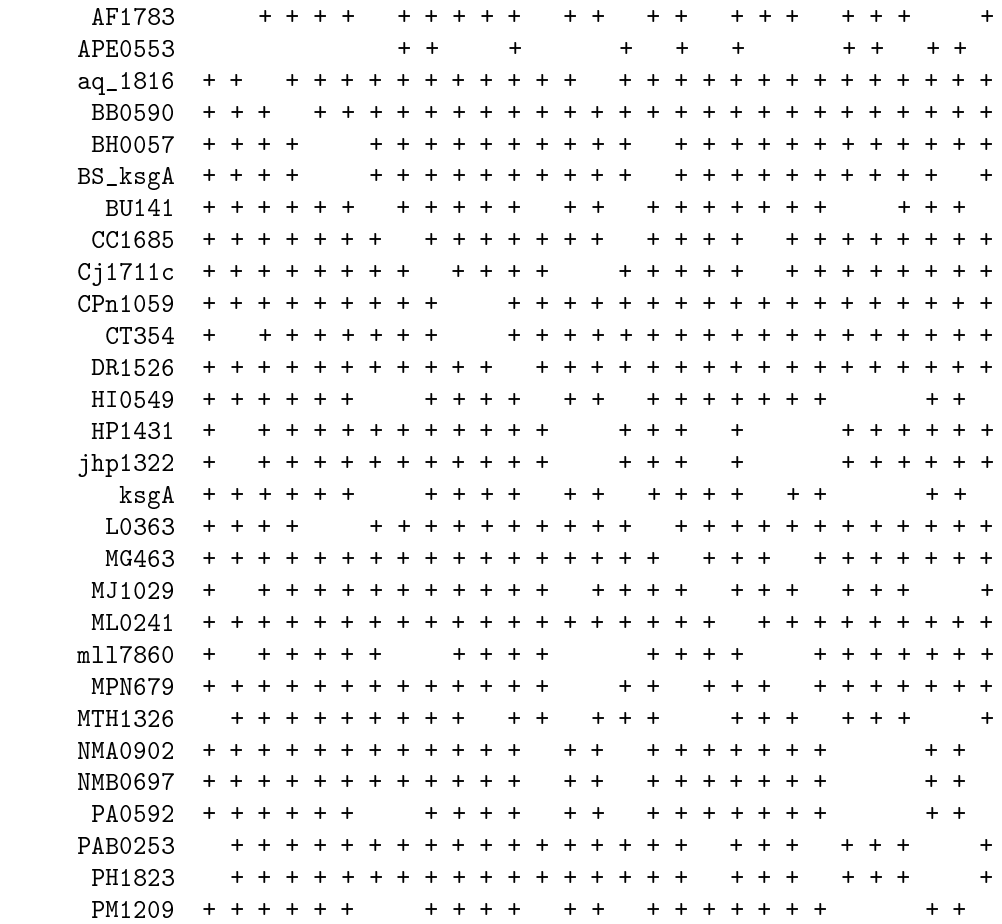
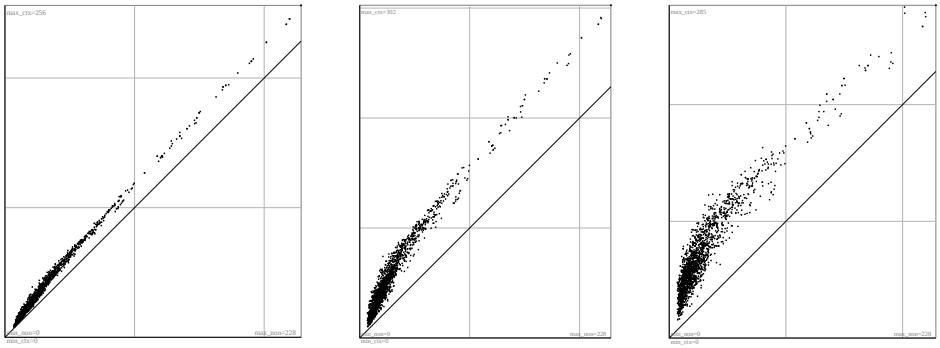


Figure 6.1: Comparing statistical significance of contextual vs. non-contextual alignment for COG0030. + indicates that the P-value for the corresponging pair of proteins is at least two orders of magnitude smaller for contextual local alignment than for the non-contextual one. Names of the proteins are listed in the first column.



(a) (b) (c)

Figure 6.2: Non-contextual vs. contextual local alignment. The results are for COG0089 (Ribosomal proteins – large subunit A). Horizontal and vertical axis are scores. Neighbor grid points are 100 score units apart. A point (x, y) is in plot (a) if for a certain pair p, q of proteins from COG0089 the optimal Smith-Waterman score of the local alignment of p with q is x, under $\mathrm{CONT}_1 62$ substitution table; while the optimal score of the local contextual alignment of p with q is y, under $\mathrm{CONT}_2 62$. The plots (b) and (c) are obtained in a similar way, except $\mathrm{CONT}_6 62$ ($\mathrm{CONT}_{20} 62$, respectively) was used instead of $\mathrm{CONT}_2 62$.

that the distinguishing power increases with the number of context groups. It is particularly visible in the area of low scores (close to the bottom left corner of the plot), where the vertical cut through the area occupied by the alignments' scores is particularly large. Note that the difference between the plots (b) and (c) is clearly visible, even though the tables used to align sequences have one third of entries common.

6.3 Discriminating power of contextual alignment vs. non-contextual alignment

An interesting phenomenon is observed for 4 tRNA synthetase families, namely for COGs 0172, 0441, 0442, and 0495. See Figure 6.3. On the plot for all pairwise local comparisons of sequences from each of these 4 COGs, two separated groups of points are clearly visible. This indicates that the contextual algorithm subdivides the alignments into subsets, while the non-contextual alignment score alone does not distinguish these groups, i.e.

projecting the plot on the x-axis yields a continuous spectrum of score values, while projecting it on the y-axis yields two (or even three for COG0441) components.

A detailed analysis leads to an interesting observation that in each case, the subdivision of scores gives rise to a partition of proteins into two subsets, say A and B. The group with lower contextual scores contains the scores of pairs from different subsets (i.e. one from A vs. one from B), while the group of higher contextual scores contains scores of pairs from the same subset (i.e. both from A, or both from B). It always turns out that one of the subsets is much smaller than the other: for COG0172 it has just two elements (MJ1077, MTH1122); for COG0442 we have 16 proteins; for COG0495 we have 9 proteins. Moreover, in COG0441 one can distinguish even 3 groups. In that case, the group with the smallest scores corresponds to alignments of protein APE0117 with all others. A promising fact is that our partitions are in agreement with known groups of evolutionarily close proteins inside COGs. This can be verified by comparison with the existing phylogenetic trees. This readily confirms the accuracy of the contextual approach.

Plots of Figure 6.3 should be contrasted with plots obtained for unrelated sequences. Figure 6.4 presents the results. Plot in Figure 6.4(a) contains a comparison of contextual vs. non-contextual local alignment for 1176 pairs of distantly related homologs (less than 25% sequence identity). This data was kindly given to us by Maricel Kann [Ka02]. The plot suggests that there is more discriminative power for detecting remote homology when contextual alignment is used instead of the ordinary non-contextual one. Figure 6.4(b) contains a comparison for 1176 unrelated pairs of proteins. They were obtained from pairs of protein sequences of Figure 6.4(a) by choosing a random order of these pairs $(S_1, T_1), \ldots, (S_{1176}, T_{1176})$. The i-th new pair (U, V) was obtained by taking $U = S_i$ and $V = T_{i+2}$ (the index $i+2$ was taken modulo 1176). This plot clearly indicates that in this case both methods shrink the spectrum of possible score values to those near zero. Figure 6.4(c) contains a comparison for completely random sequences which preserve only the amino acid frequency of the sequences used in Figure 6.4(a).

We conclude this section with an example of a comparison for the global alignment. Quite another shape is found on the plot of the non-contextual score (horizontal axis) vs. contextual score (vertical axis) for all pairwise *global* comparisons of sequences from COG0575 (CDP diglyceride synthetase). This time the contextual asymmetric table $\mathrm{CONT}_6 62$ was used,

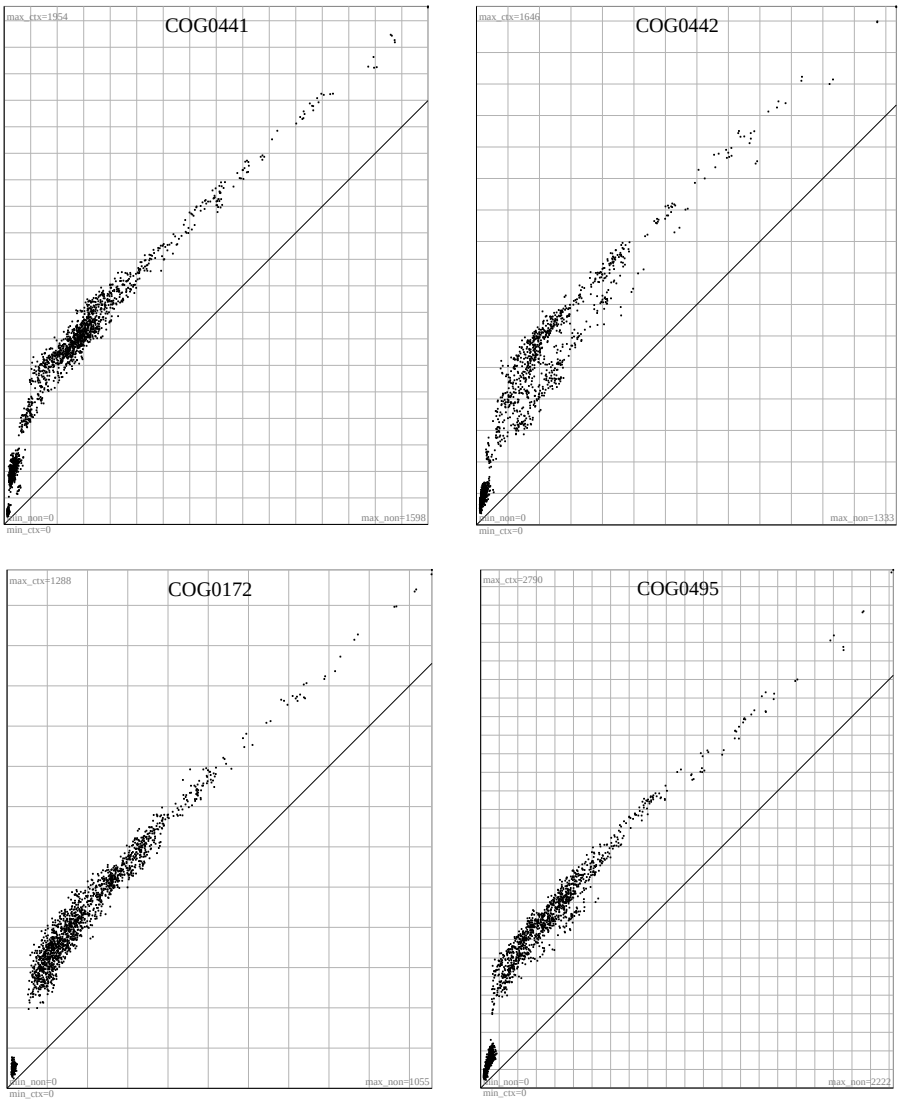
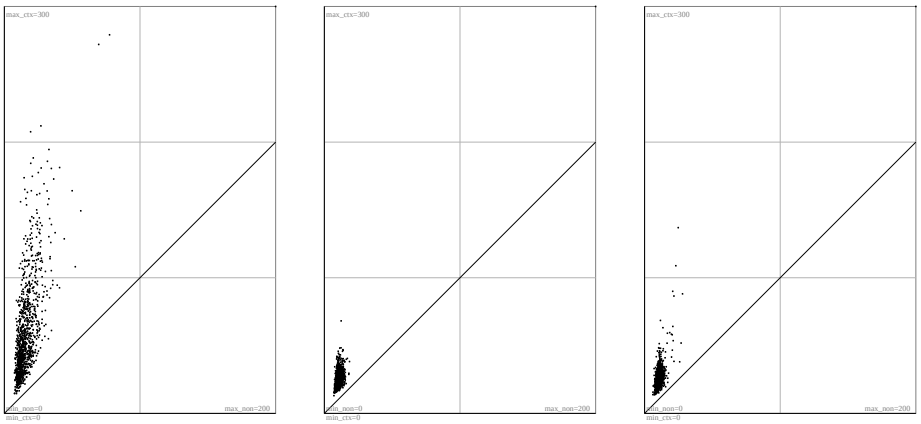


Figure 6.3: Comparison of local alignment for COGs 0172 (Seryl-tRNA synthetase), 0441 (Threonyl-tRNA synthetase) , 0442 (Prolyl-tRNA synthetase) and 0495 (Leucyl-tRNA synthetase). For the contextual alignment we used the table $\mathrm{CONT}_{20}62$. Explaination of the plot is that same as in Figure 6.2.



(a) (b) (c)

Figure 6.4: Comparing unrelated sequences. Plot (a) presents the results of comparing non-contextual (x-axis) with contextual (y-axis) local alignment for 1176 pairs of remote homologs (less than 25% of sequence identity). Plot (b) represents a comparison of 1176 pairs of protein sequences which are unrelated. Plot (c) shows such a comparison for 1176 pairs of random sequences, each obtained by a random permutation of amino acids in the pairs of proteins of plot (a). Description of the plots in this Figure is the same as in Figure 6.2.

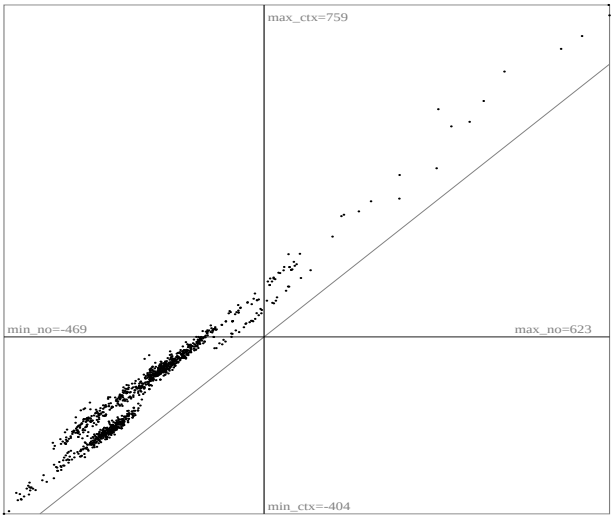


Figure 6.5: Comparison of global alignment for COG0575 (CDP diglyceride synthetase). Horizontal and vertical axis are scores. Neighbor grid points are 100 score units apart. A point (x, y) is in the plot if for a certain pair p, q of proteins from COG0575 the optimal Smith-Waterman score of the global alignment of p with q is x, under $\mathrm{CONT}_1 62$ substitution table; while the optimal score of the global contextual alignment of p with q is y, under $\mathrm{CONT}_6 62$.

see Figure 6.5. Two stripes are clearly visible, indicating that the contextual algorithm subdivides the alignments into two subsets, while the non-contextual alignment score alone does not distinguish these two groups.

6.4 Phylogenetic trees

Comparison of contextual with non-contextual alignment for building phylogenetic trees has been done in Gambin and Slonimski [GS02]. The pairwise contextual alignment scores were used to phylogenetic reconstruction of evolution of several protein families from COGs. For the same families phylogenetic trees were also reconstructed, based on the standard non-contextual approach. The comparative analysis shows that the use of contextual model results in a more consistent set of trees. The difference, although small, is with no exception in favor of the contextual model. The consistency of a

family of trees is measured by several consensus and agreement methods, as well as by the inter-tree distance approach.

For the experiment two sets of COGs were used. One set, consisting of 12 gene families of different tRNA synthetases, constituted of strongly functionally related families of genes. The other set, consisting of 8 COGs, includes functionally more diverse groups of genes (DNA polymerases, ribosomal proteins, CDP-diglyceride synthetases). These two sets were selected from the list of 85 COGs, in which all organisms are present. For each of the gene family four phylogenetic trees were constructed, depending on the choice of the method of aligning (contextual vs. non-contextual), as well as depending on the method of deriving evolutionary distance from the alignment score (P-value statistics of Vingron and Waterman [VW95] vs. the approach by Linial *et al.* [LLTY97]).

Under different methods of measuring dissimilarity of pairs of trees (e.g. *Nearest Neighbor Interchange Distance*, or *Maximal Agreement Subtree*) it turns out that the set of trees obtained by the contextual alignment method forms a more compact set (i.e. the mutual distances are smaller) than the set obtained by the standard non-contextual alignment method. Moreover this phenomenon did not depend on choice of the method of deriving the evolutionary distance.

Chapter 7

Conclusions and open problems

We would like to remark that there is no need to restrict the context to consist of one letter to the left and one letter to the right, only. Indeed, our algorithms extend without much difficulty to the case with contexts consisting of k letters on each side, for a fixed k. The algorithms are still of $O(f(|\Sigma|, k) \cdot mn)$ complexity. However, the factor $f(|\Sigma|, k)$ grows extremely fast with k, which makes algorithms even for $k = 2$ impractically slow. An interesting observation is that the local sequence around each amino acid can be used to approximate the "3 D context" i.e. the local environment in secondary and tertiary structure of the protein. For this aim the context consisting only of one pair of letters is sufficient. For example, in β-sheets only amino acids at positions $i \pm 2$ contact the ith one [SS79]. It would be interesting to adopt our methodology to deal with this situation.

An additional problem, mentioned already above, is that already for $k = 1$ there is hardly enough biological data available to construct the substitution tables. For $k = 2$ the situation becomes completely hopeless. Therefore, based on purely pragmatic reasons, we decided to limit our presentation to the case $k = 1$.

Another possible extension is to permit the scores of insertions and deletions to depend on the context, too. It is again well motivated from the biological point of view. In the case of DNA it is known, that transposons insert themselves in the places identified by a specific, transposon dependent sequence of base pairs. Thus, on the level of DNA, transposon caused inser-

tions are context dependent (cf. [Str95]). To conform to this more general situation, the overall idea of the algorithms need not be changed, but the details become more messy. We decided not to present it here because we can't imagine determining a plausible contextual indel penalty — given that even the non-contextual one is not inferred from the data, but rather chosen *ad hoc*, based on experience.

Bibliography

[Alt91] S. F. Altschul. Amino acid substitution matrices from an information theoretic perspective. *Journal of Molecular Biology*, 219:555–565, 1991.

[AW94] Arratia, R., Waterman, M.S. (1994) 'A phase transition for the score in matching random sequences allowing deletions' *Ann. Appl. Prob.* **4**:200-225.

[CAGRHSC99] J.P. Comet, J.C. Aude, E. Glemet, J.L. Henaut, P.P. Slonimski, and J.J. Codani. Significance of Z-value statistics of Smith-Waterman scores for protein alignments. *Comput. Chemistry*, 23:317–331, 1999.

[DKZ94] Dembo, A., Karlin, S., Zeitouni, O. (1994) 'Limit distribution of maximal non-aligned two-sequence segmental score', *Ann. Probability* **22**(4), 2022-2039

[DEKM98] R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison. *Biological sequence analysis. Probabilistic models of proteins and nucleid acids*. Cambridge University Press, 1998.

[DSO78] M.O. Dayhoff, R.M. Schwartz, and B.C. Orcutt. A model of evolutionary change in proteins. In M.O. Dayhoff, editor, *Atlas of Protein Sequence and Structure*, volume 5, suplement 3, pages 345–352. National Biomedical Research Foundation, 1978.

[Gaj01] , T. Gajewski, Tablice substytucyjne dla modelu przyrwnania kontestowego. Warsaw University, 2001,*M.S. Thesis (in Polish)*.

[GLSTT02] A. Gambin, S. Lasota, R. Szklarczyk, J. Tiuryn, J. Tyszkiewicz. Contextual alignment of biological sequences. *Proceedings of the European Conference on Computational Biology (ECCB 2002)*, ECCB (Supplement of *Bioinformatics*): s116-s127. It appeared also in: *Currents in Computational Molecular Biology. Posters from RECOMB 2002*: 59–60.

[GO03] Gambin,A., Otto, R. 'Contextual Multiple Alignment (Context helps in aligning orphan genes)' (2003) submitted.

[GS02] A. Gambin and P.S. Slonimski. Contextual protein alignment in phylogenetics. *Submitted for publication*. 2002.

[GT02] A. Gambin and J. Tyszkiewicz. Substitution Matrices for Contextual Alignment. Proc. Journees Ouvertes Biologie Informatique Mathematique JOBIM 2002, Saint Malo, France. It appeared also in: *Currents in Computational Molecular Biology. Posters from RECOMB 2002*: 61–62.

[Gum58] [9] Gumbel, E. J. (1958) "Statistics of extremes." Columbia University Press, New York, NY.

[Gus97] D. Gusfield. *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge University Press, 1997.

[Han03] Niels Richard Hansen, 'Local Alignment of Markov chains' PhD thesis, Department of Applied Mathematics and Statistics, University of Copenhagen.

[vHS98] Arndt von Haeseler and Michael Schöniger. Evolution of DNA or amino acid sequences with dependent sites. *Journal of Computational Biology*, 5:149–163, 1998.

[HH92] S. Henikoff and J.G. Henikoff. Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. USA*, 89:10915–10919, 1992.

[HGPH00] Henikoff, J.G. and Greene, E.A. and Pietrokovski, S., Henikoff, S. Increased coverage of protein families with the blocks database servers, *Nucl. Acids Res.*,**vol.28**: 228–230, 2000.

[HHP99] Henikoff, S. and Henikoff, J.G., Pietrokovski, S. Blocks+: A non-redundant database of protein alignment blocks dervied from multiple compilations, *Bioinformatics*, **vol. 15(6)**:471–479, 1999.

[Ioe97] T.R. Ioerger. The context-dependence of amino acid properties. In *Intelligent Systems in Molecular Biology*, pages 157–166. AAAI Press, 1997.

[JKP00] Jens Ledet Jensen and Anne-Mette Krabbe Pedersen. Probabilistic models of DNA sequence evolution with context dependent rates of substitution. *Advences in Applied Probability*, 32:499–517, 2000.

[Ka02] Maricel Kann, *private communication*, 2002.

[KA90] Karlin, S., Altschul, S.F. (1990) 'Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes.' Proc.Natl. Acad. Sci. USA **87**:2264-2268.

[LLTY97] M. Linial, N. Linial, N. Tishby, and G. Yona. Global self organization of all known protein sequences reveals inherent biological signatures. *Journal of Molecular Biology*, 268:539–556, 1997.

[Mot92] Mott, R. (1992) 'Maximum-likelihood estimation of the statistical distribution of Smith-Waterman local sequence similarity scores', *Bull. Math. Biol.* **54**:59-75.

[MT99] Mott, R., Tribe, R. (1999) 'Approximate Statistics of Gapped Alignment' *J. Comp. Biol.*, **6**:91-112.

[SS79] G. Schulz and R. Schirmer. *Principles of Protein Structure*. Springer Verlag, New York, 1979.

[SvH94] Michael Schöniger and Arndt von Haeseler. A stochastic model for the evolution of autocorrelated DNA sequences. *Molecular Phylogenetics and Evolution*, 3:240–247, 1994.

[SW81] T.F. Smith and M.S. Waterman. The identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[Str95] L. Stryer. *Biochemistry*. W.H. Freeman and Company, 1995.

[TNG01] R.L. Tatusov, D.A. Natale, I.V. Garkavtsev, T.A. Tatusova, U.T. Shankavaram, B.S. Rao, B. Kiryutin, M.Y. Galperin, N.D. Fedorova, and E.V. Koonin. The COG database: new developments in phylogenetic classification of proteins from complete genomes. *Nucleic Acid Research*, 29(1):22–28, 2001.

[TKL97] Tatusov, RL and Koonin, EV, Lipman, DJ. Genomic perspective on protein families, *Science*,**vol. 278**: 631–637, 1997.

[TG89] S. Tavaré and B.W. Giddings. Some statistical aspects of the primary structure of nucleotide sequences. In M.S. Waterman, editor, *Mathematical methods for DNA sequences*, pages 117–132. CRC Press, Boca Raton, FL, 1989.

[VW95] M. Vingron and M.S. Waterman. Statistical significance of local alignments with gaps. In: *Bioinformatics: from nucleid acids and proteins to cell methabolism*, 75–84, 1995.

[WV94] Waterman, M.S., Vingron, M. (1994) 'Rapid and accurate estimates of statistical significance for sequence database searches.' Proc. Natl. Acad. Sci. USA **91**:4625-4628.

[WV94a] Waterman, M.S., Vingron, M. (1994) Sequence comparison significance and Poisson approximation. Stat. Sci. **9**:367-381.

[WB01] Webber, C., Geoffrey J. Barton (2001) Estimation of P-values for global alignments of protein sequences, *Bioinformatics* **17**: 1158-1167.

[WL84] W.J. Wilbur and D.J. Lipman. The context dependent comparison of biological sequences. *SIAM J. Appl. Math.*, 44: 557–567, 1984.

Appendix A

Appendix: Constraint Tables for an Insertion Block

For the tables in this Appendix we use the following notation (same as in Figure 2.1).

$$\begin{array}{cccccccc} i & j & & & & & & k \\ a & - & - & \cdot & \cdot & \cdot & - & b \\ a' & c_1 & c_2 & \cdot & \cdot & \cdot & c_m & b' \end{array}$$

First we give for right contexts for i and separately for left contexts for k.

b	b'	c_1
$\vee_i^{j,k}$	$C_{k<i<j}$	$C_{j<i}$

Table A.1: Constraints for right contexts for i.

a	a'	c_m
$\vee_k^{i,j}$	$C_{i<k<j}$	$C_{j<k}$

Table A.2: Constraints for left contexts for k.

Next we give the constraint tables for all possible relationships between the characters in the alignment (all equalities are explicitly mentioned for each table).

(b,a)	(b,a')	(b,c_m)	(b',a)	(b',a')	(b',c_m)	(c_1,a)	(c_1,a')	(c_1,c_m)
$\bot$	$C_{i<k<j}$	$C_{i<j<k}$	$C_{k<i<j}$	$\bot$	$\bot$	$C_{k<j<i}$	$\bot$	$\vee_j^{i,k}$

Chart 1: Constraints for all possible pairs of contexts (no equalities assumed).

(b,a)	(b,a')	(b,c_m)	(b',a)	(b',a')	(b',c_m)
$C_{k<j<i}$	$C_{i<k<j}$	$C_{j<k}$	$C_{k<i<j}$	$\bot$	$\bot$

Chart 2: Constraints for the situation $b = c_1$.

(b,a)	(b,a')	(b,c_m)	(b',a)	(b',a')	(b',c_m)
$\bot$	$C_{i<k<j}$	$C_{i<j<k}$	$\vee_k^{i,j}$	$\bot$	$\vee_j^{i,k}$

Chart 3: Constraints for the situation $b' = c_1$.

(b,a)	(b,a')	(b,c_m)	(c_1,a)	(c_1,a')	(c_1,c_m)
$C_{k<i<j}$	$C_{i<k<j}$	$C_{i<j<k}$	$C_{k<j<i}$	$\bot$	$\vee_j^{i,k}$

Chart 4: Constraints for the situation $b = b'$.

(b,a)	(b,a')	(b,c_m)
$\vee_k^{i,j}$	$C_{i<k<j}$	$C_{j<k}$

Chart 5: Constraints for the situation $b = b' = c_1$.

(b,a)	(b,a')	(b',a)	(b',a')	(c_1,a)	(c_1,a')
$C_{i<j<k}$	$C_{i<k<j}$	$C_{k<i<j}$	$\bot$	$C_{j<i}$	$\bot$

Chart 6: Constraints for the situation $a = c_m$.

(b,a)	(b,a')	(b',a)	(b',a')	(c_1,a)	(c_1,a')
$\bot$	$\vee_i^{k,j}$	$C_{k<i<j}$	$\bot$	$C_{k<j<i}$	$\vee_j^{i,k}$

Chart 7: Constraints for the situation $a' = c_m$.

(b,a)	(b,c_m)	(b',a)	(b',c_m)	(c_1,a)	(c_1,c_m)
$C_{i<k<j}$	$C_{i<j<k}$	$C_{k<i<j}$	$\bot$	$C_{k<j<i}$	$\vee_j^{i,k}$

Chart 8: Constraints for the situation $a = a'$.

(b,a)	(b',a)	(c_1,a)
$\vee_i^{k,j}$	$C_{k<i<j}$	$C_{j<i}$

Chart 9: Constraints for the situation $a = a' = c_m$.

(b,a)	(b,a')	(b',a)	(b',a')
$C_{i<j<k} \oplus C_{j<i}$	$C_{i<k<j}$	$C_{k<i<j}$	$\bot$

Chart 10: Constraints for the situation $b = c_1$ and $a = c_m$.

(b,a)	(b,a')	(b',a)	(b',a')
$C_{k<j<i}$	$\vee_i^{j,k} \oplus \vee_j^{i,k}$	$C_{k<i<j}$	$\bot$

Chart 11: Constraints for the situation $b = c_1$ and $a' = c_m$.

(b,a)	(b,c_m)	(b',a)	(b',c_m)
$C_{k<j<i} \oplus C_{i<k<j}$	$C_{j<k}$	$C_{k<i<j}$	$\bot$

Chart 12: Constraints for the situation $b = c_1$ and $a = a'$.

(b,a)	(b',a)
$\vee_i^{j,k} \oplus C_{j<i}$	$C_{k<i<j}$

Chart 13: Constraints for the situation $b = c_1$ and $a = a' = c_m$.

(b,a)	(b,a')	(b',a)	(b',a')
$C_{i<j<k}$	$C_{i<k<j}$	$\vee_k^{i,j} \oplus \vee_j^{i,k}$	$\bot$

Chart 14: Constraints for the situation $b' = c_1$ and $a = c_m$.

(b,a)	(b,a')	(b',a)	(b',a')
$\bot$	$\vee_i^{j,k}$	$\vee_k^{i,j}$	$\vee_j^{i,k}$

Chart 15: Constraints for the situation $b' = c_1$ and $a' = c_m$.

(b,a)	(b,c_m)	(b',a)	(b',c_m)
$C_{i<k<j}$	$C_{i<j<k}$	$\vee_k^{i,j}$	$\vee_j^{i,k}$

Chart 16: Constraints for the situation $b' = c_1$ and $a = a'$.

(b,a)	(b',a)
$\vee_i^{j,k}$	$\vee_j^{i,k} \oplus \vee_k^{i,j}$

Chart 17: Constraints for the situation $b' = c_1$ and $a = a' = c_m$.

(b,a)	(b,a')	(c_1,a)	(c_1,a')
$C_{k<i<j} \oplus C_{i<j<k}$	$C_{i<k<j}$	$C_{j<i}$	$\bot$

Chart 18: Constraints for the situation $b = b'$ and $a = c_m$.

(b,a)	(b,a')	(c_1,a)	(c_1,a')
$C_{k<i<j}$	$\vee_i^{j,k}$	$C_{k<j<i}$	$\vee_j^{i,k}$

Chart 19: Constraints for the situation $b = b'$ and $a' = c_m$.

(b,a)	(b,c_m)	(c_1,a)	(c_1,c_m)
$\wedge_{k,i}^{j}$	$C_{i<j<k}$	$C_{k<j<i}$	$\vee_j^{i,k}$

Chart 20: Constraints for the situation $b = b'$ and $a = a'$.

(b,a)	(c_1,a)
$C_{i<j}$	$C_{j<i}$

Chart 21: Constraints for the situation $b = b'$ and $a = a' = c_m$.

(b, a)	(b, a')
$\vee_k^{i,j} \oplus C_{j<k}$	$C_{i<k<j}$

Chart 22: Constraints for the situation $b = b' = c_1$ and $a = c_m$.

(b, a)	(b, a')
$\vee_k^{i,j}$	$\vee_i^{j,k} \oplus \vee_j^{i,k}$

Chart 23: Constraints for the situation $b = b' = c_1$ and $a' = c_m$.

(b, a)	(b, c_m)
$C_{k<j}$	$C_{j<k}$

Chart 24: Constraints for the situation $b = b' = c_1$ and $a = a'$.

(b, a)
$\top$

Chart 25: Constraint for the situation $b = b' = c_1$ and $a = a' = c_m$.