Computational Design of New and Recombinant Selenoproteins

Rolf Backofen and Anke Busch

Friedrich-Schiller-Universitaet Jena
Institute of Computer Science, Chair for Bioinformatics
Ernst-Abbe-Platz 1-2, D-07743 Jena, Germany
{backofen,busch}@inf.uni-jena.de

Abstract. Selenoproteins contain the 21th amino acid Selenocysteine, which is encoded by the STOP-codon UGA. For its insertion it requires a specific mRNA sequence downstream the UGA-codon that forms a hairpin-like structure (called Sec insertion sequence (SECIS)). Selenoproteins have gained much interest recently since they are very important for human health.
In contrast, very little is known about selenoproteins. For example, there is only one solved crystal structure available. One reason for this is that one is not able to produce enough amount of selenoproteins by using recombinant expression in a system like *E.coli*. The reason is that the insertion mechanisms are different between *E.coli* and eukaryotes. Thus, one has to redesign the human/mammalian selenoprotein for the expression in *E.coli*. In this paper, we introduce an polynomial-time algorithm for solving the computational problem involved in this design, and we present results for known selenoproteins.
Keywords: Selenocysteine, SECIS, Protein Design

1 Introduction

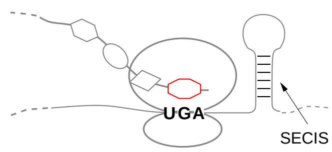

Fig. 1. Translation of mRNA requires a SECIS-element in case of selenocysteine.

Selenocysteine (Sec) is a rare amino acid, which was discovered as the 21st amino acid [7] about a decade ago. Proteins containing selenocysteine are called *selenoproteins*. Selenocysteine is encoded by the UGA-codon, which is usually a STOP-codon. It has been shown [7] that, in the case of selenocysteine, termination of translation is inhibited in the presence of a specific mRNA sequence in the 3'-region after the UGA-codon that forms a hairpin-like structure (called **Sec i**nsertion **s**equence (SECIS), Figure 1).

Selenoproteins have gained much interest recently since selenoproteins are of fundamental importance to human health. They are an essential component of several major metabolic pathways, including such important ones as the antioxidant defence systems, the thyroid hormone metabolism, and the immune function (for overviews see e.g. [6]). Just recently, the complete mammalian selenoproteome was determined [11]. For this reason, there is an enormous interest in the catalytic properties of selenoproteins, especially since selenocysteine is more reactive than its counterpart cysteine. Results of the research have shown that a selenocysteine containing protein has greatly enhanced enzymatic activites compared to the cysteine homologues [8]. The general way of investigating properties of proteins is to provide 1.) large amounts of pure proteins, and 2.) generating variants by site directed mutagenesis. In both cases, one has to use recombinant protein expression systems with *E.coli* being the simplest one to handle. However, there are differences between the mechanisms for inserting selenocysteine in eukaryotes and bacteria. In eukaryotes, the SECIS-element is located in the 3' UTR of the mRNA with a distance from the UGA-codon that varies from 500 to 5300 nucleotides [13]. In bacteria, the situation is quite different. We consider the case of *E.coli*, where the mechanism of selenocysteine insertion is well understood and all corresponding factors are identified [15]. The SECIS-element is located immediately downstream the UGA-codon [17], which implies that the SECIS-element is in the coding part of the protein. A displacement of the SECIS-element by more than one codon or a displacement not preserving the reading-frame results in a drastic reduction of selenocysteine insertion efficiency [9].

Therefore, for selenoproteins, recombinant expression is complicated and fails very often [16]. To our knowledge, there is only one solved structure of a selenoprotein: glutathione peroxidase (PDB 1GP1). Besides, Sandalova et al. [14] determined the three-dimensional structure of a Cys-mutant of the rat selenoprotein thioredoxin reductase (TrxR) which is expected to fold similar to the Sec-variant. Furthermore, there are only two cases of successive heterologous expression [8,1], which both required a careful, hand-crafted design of the nucleotide sequence. Thus, we have the following implications. First, an eukaroytic selenoprotein cannot directly be expressed in the *E.coli* system. Second, the expression of an eukaroytic selenoprotein requires the design of an appropriate SECIS-element directly after the UGA-position. Third, this design always changes the protein sequence. Therefore, one has to make a compromise between changes in the protein sequence and the efficiency of selenocysteine insertion (i.e. the quality of the SECIS-element).

The corresponding bioinformatic problem is to search for similar proteins under sequential and structural constraints imposed on the mRNA by the SECIS-elements. A first positive solution to this problem was given in [2]. This algorithm considers a fixed SECIS-element without any insertions/deletions. In [3], it was shown that the problem becomes NP-hard if more complicated structures (i.e. pseudoknots) are considered and an approximation algorithm was given for this case.

In this paper, we consider the extension of [2] where we allow insertions and deletions in the amino acid sequence. The reason is that, albeit the SECIS-element is fixed in the mRNA, it is possible to place it on different parts of the amino acid sequence via insertions and deletions. Furthermore, we can gain information about possible insertions/deletions from well known alignments of the corresponding protein sequence (e.g. from seed alignments of the

Pfam-database [5]) and from additional biological knowledge that restricts insertions/deletions to certain positions. As we will explain in the next section, in this problem insertions and deletions are more complicated than in the usual alignment problems, since an insertion or deletion changes the mapping between the mRNA and the protein sequence. In addition, we introduce a second optimization criteria: optional bonds. The reason for this extension is that in the SECIS-element, only some part of the structure is fixed. Other elements (like the lower part of the hairpin stem) are not really required, albeit they improve the quality of the SECIS-elements. This is captured by the concept of optional bonds.

2 The Computational Problem

As input, we have the consensus sequence of the SECIS-element $S = S_1 ... S_{3n}$ and the original amino acid sequence $A = A_1 ... A_n$ from the 3'-region of the position where we wish to insert selenocysteine. The process of inserting a SECIS-element poses the problem of finding an appropriate mRNA sequence $N = N_1 ... N_{3n}$. This mRNA sequence must contain a SECIS-element (sequence and structure) at the right position. This implies that we do not have to measure only the similarity on the mRNA sequence but also to guarantee that the SECIS structure can be formed. Thus, the corresponding nucleotides in N must be complementary bases. In addition, we also need to require that the encoded amino acid sequence $A' = A'_1 ... A'_n$ has maximum similarity with A. Thus, we have the following picture for the similarity without insertions and deletions:

$$\begin{array}{rcccccc}
A = & A_1 & \dots & A_i & \dots & A_n \\
 & \sim & & \sim & & \sim \\
A' = & A'_1 & \dots & A'_i & \dots & A'_n \\
N = & \overbrace{N_1 N_2 N_3} & \dots & \overbrace{N_{3i-2} N_{3i-1} N_{3i}} & \dots & \overbrace{N_{3n-2} N_{3n-1} N_{3n}} \\
 & \sim\ \sim\ \sim & & \sim\ \ \sim\ \ \sim & & \sim\ \ \sim\ \ \sim \\
S = & S_1 S_2 S_3 & \dots & S_{3i-2} S_{3i-1} S_{3i} & \dots & S_{3n-2} S_{3n-1} S_{3n}
\end{array}$$

$N = N_1 ... N_{3n}$ is the mRNA sequence to be searched for. $\sim$ indicates the similarity on both the amino acid ($A \sim A'$) and the nucleotide level ($N \sim S$). This similarity will be measured by functions $F_{A_i}^{S_{3i-2} S_{3i-1} S_{3i}}(N_{3i-2} N_{3i-1} N_{3i})$. These functions also consider conditions on the nucleotide and the amino acid level: Some positions of the SECIS-consensus sequence S as well as some positions of the amino acid sequence A are highly conserved and ensure the biological function of the SECIS-element and the protein. They must not be changed. Therefore, we restrict our problem by nucleotide and amino acid conditions which penalize changes at conserved positions. Of course, a problem arises if the nucleotide and amino acid conditions are contradictory. In this paper we show how such contratictions can be solved. We will consider two extensions to the original problem:

1.) Insertions and Deletions. In the original problem there is a direct mapping from nucleotide positions to amino acid positions, where the codon $N_{3i-2} N_{3i-1} N_{3i}$ on the nucleotide level correspond to the i-th position on the amino acid level. This changes if one considers insertions and deletions additionally. Since the SECIS-element is relatively fixed, we consider only insertions and deletions on the amino acid level.

Now consider fixed $N_{3i-2}N_{3i-1}N_{3i}$ corresponding to the i-th amino acid. Suppose that we have no insertions and deletions so far. An *insertion* at this position means that we insert a new amino acid A'_i, which does not have a counterpart in A. This implies that 1.) only the similarity between $N_{3i-2}N_{3i-1}N_{3i}$ and the corresponding part of the SECIS-consensus $S_{3i-2}S_{3i-1}S_{3i}$ is measured, 2.) a gap cost for the insertion is added, and 3.) the mapping from the nucleotide sequence to the original amino acid sequence is changed. Therefore, we have to compare $N_{3j-2}N_{3j-1}N_{3j}$ with the $j-1$-th amino acid (for $j > i$). Furthermore, this means that we can have at most one *insertion* per codon position in the SECIS-element.

In comparison, a *deletion* implies that we compare the codon $N_{3i-2}N_{3i-1}N_{3i}$ with A_{i+1} (thus skipping A_i), and there could be several deletions per SECIS-codon site. Since only few insertions and deletions are allowed, and since we are restricted by one insertion per SECIS-codon site by the nature of the problem, we confine ourselves to one deletion per SECIS-codon site.

By the given restriction, it is easier to skip the alignment notation and to represent insertions and deletions directly by a vector t with $t_i \in \{-1, 0, 1\}$ that shows which operation is applied on the i-th SECIS-codon site. Here, -1 indicates a deletion, $+1$ an insertion and 0 a substition. Furthermore, these values determine the offset that has to be added in order to find the mapping between A'_i and the corresponding position in the original amino acid sequence. An example is given in Figure 2a. Here, we have $t_1 = 0$, which implies that we have a substitution and compare A'_1 with A_1. For position $i = 2$, we have a deletion indicated by $t_2 = -1$. Hence, we have to compare A'_2 with A_3. Thus, position j of the original amino acid has to be compared with the amino acid A'_i in the newly generated amino acid sequence. In this case j can be calculated by $j = i - \sum_{k \leq i} t_k$. A further consequence of the restriction is that each deletion must be followed by a substitution. Figure 2a shows a correct deletion followed by a substitution at SECIS-position 2 whereas Figure 2b exemplifies the incorrect case of two successive deletions at SECIS-positions 2 and 3.[1]

Therefore, we have to consider a modified similarity function, where the changes in the index are taken into account. Thus, we introduce $f_i(L_i, a_i, t_i)$, where

- L_i represents the *codon* corresponding to the nucleotides $N_{3i-2}N_{3i-1}N_{3i}$
- t_i as defined above
- a_i is the difference between the number of insertions and the number of deletions up to position i, i.e. $a_i = \sum_{j=1}^{i} t_j$, which reflects the relative displacement of the old and the new amino acid sequence to each other.

Three different cases of the new similarity function have to be taken into account:

[1] To allow for successive deletions (as indicated in Figure 2b), one must allow two or more deletions per SECIS-codon site within the *basic case* (see equation (3)).

a)	A :	A_1	A_2	A_3	A_4	–	–	A_5
	A' :	A'_1	–	A'_2	A'_3	A'_4	A'_5	A'_6
	N :	$N_1N_2N_3$	– – –	$N_4N_5N_6$	$N_7N_8N_9$	$N_{10}N_{11}N_{12}$	$N_{13}N_{14}N_{15}$	$N_{16}N_{17}N_{18}$
	S :	$S_1S_2S_3$	– – –	$S_4S_5S_6$	$S_7S_8S_9$	$S_{10}S_{11}S_{12}$	$S_{13}S_{14}S_{15}$	$S_{16}S_{17}S_{18}$
	t :	0	-1		0	+1	+1	0
	SECpos	1	2		3	4	5	6
b)	A :	A_1	A_2	A_3	A_4	–	–	A_5
	A' :	A'_1	–	–	A'_2	A'_3	A'_4	A'_5
	N :	$N_1N_2N_3$	– – –	– – –	$N_4N_5N_6$	$N_7N_8N_9$	$N_{10}N_{11}N_{12}$	$N_{13}N_{14}N_{15}$
	S :	$S_1S_2S_3$	$S_4S_5S_6$	– – –	$S_7S_8S_9$	$S_{10}S_{11}S_{12}$	$S_{13}S_{14}S_{15}$	$S_{16}S_{17}S_{18}$
	t :	0	-1	-1		+1	+1	0
	SECpos	1	2	3		4	5	6

Fig. 2. Allowed and disallowed deletion patterns. a) Possible solution, each deletion must be followed by a substitution, b) Not allowed solution, a deletion not followed by a substitution

$$\begin{aligned}
f_i(L_i, a_i, 0) &= F^{S_{3i-2}S_{3i-1}S_{3i}}_{A_{i-a_i}}(N_{3i-2}N_{3i-1}N_{3i}) \\
f_i(L_i, a_i, +1) &= IP + F^{S_{3i-2}S_{3i-1}S_{3i}}(N_{3i-2}N_{3i-1}N_{3i}) \\
f_i(L_i, a_i, -1) &= \begin{cases} f_i(L_i, a_i, 0) + DP & \text{if no amino acid cond. at } i-(a_i-t_i) \\ -\infty & \text{otherwise} \end{cases}
\end{aligned}$$

If an insertion is done at position i, the similarity is calculated by a combination of an *insertion-penalty IP* and the similarity on the nucleotide level. Then there is no amino acid restriction because inserting each amino acid should be allowed. If the amino acid at position i is deleted, one has to distinguish between two cases: (a) if there is no amino acid condition at position $i - a_i - 1$ (original amino acid position to be deleted), the similarity is calculated the normal way with an additional *deletion-penalty DP*, and (b) if an amino acid condition exists at position $i - a_i - 1$, deleting this amino acid is not allowed and the similarity function has to be -∞.

2.) Optional Bonds. For the second extension, we now declare some bonds as being optional. In contrast to the mandatory bonds, these optional ones are not fixed. It would be of advantage if they form but they are not necessary to ensure the function of the SECIS-element.

Due to these two improvements (allowing insertions/deletions and declaring some optional bonds) there are two different and possibly competing values to be maximized: the similarity and the number of realized optional bonds.

3 The Formal Description of the Problem

An undirected graph is said to be *outer-planar* if, when drawn, it satisfies the following conditions: 1.) all the vertices lie on a line, 2.) all the edges lie on one side of the line, and 3.) two edges that intersect in the drawing do so only at their end points. The solution to the problems we address in this paper are a vector whose values are from $\{-1, 0, 1\}$ representing deletions/insertions and a string whose letters are from $\Sigma_{\text{RNA}} = \{A, C, G, U\}$. The elements of this set are referred to as *nucleotides*, an element of $(\Sigma_{\text{RNA}})^3$ is called *codon*. We allow standard Watson-Crick bonds (A-U, C-G) and in some cases, we consider G and U as complements to each other, too. The complement set of a variable is denoted by the superscript C.

Input: Table 1 and Figure 3 show the input to our problem. We have an egde-labeled graph $G = (V, E, Lab)$ on $3n$ vertices with $V = \{v_1, ..., v_{3n}\}$. For every edge $\{v_k, v_l\} \in E$, the label $Lab(v_k, v_l)$ is taken from the set $\{-2, -1, 1, 2, 3, 4\}$. Note that the labeling of the egdes is chosen arbitrarily. We decided to label prohibited bonds with negative values and favoured bonds with positive ones. $Lab(v_k, v_l)$ determines the bonding between nucleotides at positions k and l in the mRNA according to Table 1. The only label that requires a bit of explanation is $Lab(v_k, v_l) = -2$ (optionally prohibited bond). This implies that it would be advantageous if there was no bond. Prohibited bonds are necessary since the SECIS-element needs some bulged nucleotides. In the following we assume that $\{v_k, v_l\} \in E(G)$ and $Lab(v_k, v_l) = -1$ imply that v_k and v_l are not part of a bond. Figure 3 shows a small example.

$Lab(v_k, v_l)$	Condition	Meaning
+1	$N_k \in N_l^C$	bond
+2	$N_k \in \Sigma_{\text{RNA}}$	optional bond
+3	$N_k \in N_l^{C(G-U)}$	G-U bond
+4	$N_k \in \Sigma_{\text{RNA}}$	opt. G-U bond
-1	$N_k \not\in N_l^C$	prohibited bond
-2	$N_k \in \Sigma_{\text{RNA}}$	opt. prohib. bond

Table 1. Edge labels

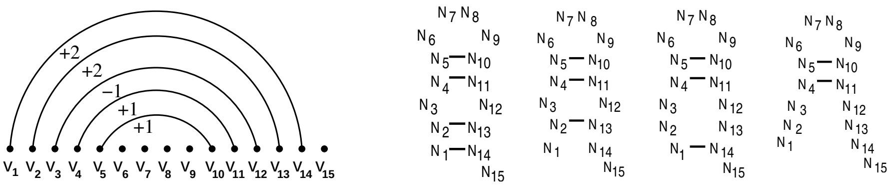

Fig. 3. Graph example and possible structures

The number of all possible optional bonds, optional G-U bonds and optionally prohibited bonds is denoted as $maxOp$. Furthermore n functions $f_1, ..., f_n$ and conditions on the nucleotide and amino acid level are also part of the input. The set of functions f_i is named f and the set of conditions is denoted as C. f_i is associated with $\{v_{3i-2}, v_{3i-1}, v_{3i}\}$, $1 \leq i \leq n$, see above.

Output and Kinds of Bonds: Find vectors $t = (t_1, ..., t_n) \in \{-1, 0, 1\}^n$ and $N = (N_1, .., N_{3n}) \in \{A, C, G, U\}^{3n}$, so that N_k is assigned to v_k, t represents the succession of insertions and deletions and

1. all egdes $\{v_k, v_l\} \in E(G)$ are labeled according to Table 1
2. depending on the optimization criteria, $\sum_{i=1}^{n} f_i(N_{3i-2}N_{3i-1}N_{3i}, a_i, t_i) = \sum_{i=1}^{n} f_i(L_i, a_i, t_i)$ (referred to as the *similarity* of the assignment) is maximized (a) over all assignments or (b) among all assignments having the maximal number of realized optional bonds.

We denote the problems by $MRSODI_S(G, f, C)$ (2a) and $MRSODI_B(G, f, C)$ (2b). MRSODI cuts *mRNA Structure Optimization allowing Deletions and Insertions* short. In the following we refer to condition 1 as the *complementary condition.*

Additional Notations: G is referred to as the *structure graph* on the mRNA level. Accordingly, we need such a graph on the amino acid level as well. Given an mRNA structure graph G, we define the *implied graph* G^{impl} as a graph on the vertices $V(G^{impl}) = \{u_1, ..., u_n\}$ with

$$E(G^{impl}) = \left\{ \{u_i, u_j\} \middle| \begin{array}{l} \exists r \in \{3i-2, 3i-1, 3i\} : \\ \exists s \in \{3j-2, 3j-1, 3j\} : \{v_r, v_s\} \in E(G) \end{array} \right\}.$$

Note that in the implied graph every node has at most degree 3 if the nodes of the input graph G have at most degree 1. (This holds in our problem since we have at most one prohibited bond per node.) Hence, up to 3 edges can emerge from a node in the implied graph. We follow the convention that, independent of the subscript or superscript, u denotes a vertex from $V(G^{impl})$ and v denotes a vertex from $V(G)$. Given $J \subseteq \{1...n\}$, U_J denotes $\{u_j | j \in J\}$. $E(G^{impl})|_J$ is the set of egdes of the induced subgraph of G^{impl} on U_J and $E(G)||_J$ denotes the edge set of the induced subgraph of G on $\{v_{3j-2}, v_{3j-1}, v_{3j} | j \in J\}$.

A codon sequence $L_1...L_n$ is said to *satisfy E(G)* iff the corresponding nucleotides $\{N_{3i-2}N_{3i-1}N_{3i}\}_{i=1}^n$ satisfies the complementary conditions for G. L_i and L_j are said to be *valid for* $\{u_i, u_j\}$ *w.r.t. E(G)* if the corresponding nucleotides $N_{3i-2}N_{3i-1}N_{3i}$ and $N_{3j-2}N_{3j-1}N_{3j}$ satisfy the complementary conditions imposed by $E(G)$.

4 Algorithm for SECIS-like Structures Allowing Insertions, Deletions and Optional Bonds

We present a polynomial time recursive algorithm that solves $MRSODI_S$ and $MRSODI_B$ when G^{impl} is outer-planar *and* every node in G has at most degree one. The hairpin shape of the SECIS-element is captured by the outer-planarity of G^{impl}. Our algorithm is based on a recurrence relation that we introduce in this section.

We fix $u_1, ..., u_n$ as the ordering of the vertices from left to right on a line in an outer-planar embedding of G^{impl}. Let f_i be the function associated with u_i. Having fixed an embedding of the graph on a line, we do not distinguish between a vertex and its index. That is, the interval $[i...i+k]$ denotes the set of vertices $\{u_i, ..., u_{i+k}\}$.

We now define the notion of compatibility between codons L_i and L_j assigned to vertices u_i and u_j, respectively. The compatibility with respect to the complementary conditions in $E(G)$ is denoted by $\equiv_{E(G)}$. Thus, for $1 \le i, j \le n$, we define $(i, L_i) \equiv_{E(G)} (j, L_j)$ by

$$(i, L_i) \equiv_{E(G)} (j, L_j) = \begin{cases} \text{true} & \text{if } \{u_i, u_j\} \notin E(G^{impl}) \\ \text{true} & \text{if } \{u_i, u_j\} \in E(G^{impl}) \text{ and} \\ & \quad L_i \text{ and } L_j \text{ are } \textit{valid} \text{ for } \{u_i, u_j\} \text{ w.r.t. } E(G) \\ \text{false} & \text{otherwise} \end{cases}$$

Some additional variables have to be defined: The number of realized optional bonds, optional G-U bonds and optionally prohibited bonds in $E(G)||_{[i...i+k]}$ (in short: $ob(L_i...L_{i+k})$) is named m, where $m \leq maxOp$. The difference of insertions and deletions up to position i (at all pos. $\leq i$) is denoted as l ('**l**eft'), whereas s ('in**s**ide') corresponds to the difference of insertions and deletions at and between positions $i+1$ and $i+k$ ($i+1 \leq$ pos. $\leq i+k$). Furthermore, s^I is the number of insertions at and between positions $i+1$ and $i+k$, accordingly s^D is the number of deletions. As mentioned above only one insertion or deletion per position is allowed.

Therefore, it is necessary that $|l| \leq i$ and $|s| \leq k$. If these two inequations are fulfilled, (l,s) are said to be *VALID* for $\{i,k\}$. Let $SPLIT_k^s$ be a set of tupels (s^I, s^D) which fulfil

$$s^I - s^D = s, \qquad s^I + s^D \leq k, \quad \text{and} \quad 0 \leq s^I, s^D. \tag{1}$$

Let $V(a,b,L)$ be the set of vectors with length L having a coordinates with value $+1$, b with value -1 and $L-(a+b)$ coordinates being 0. It is necessary that $a+b \leq L$.

We now define the central function which will be implemented by dynamic programming. $w_{i+k}^i(L_i, L_{i+k}, m, l, s)$ gives the maximal similarity of the designed sequence to the original one between positions i and $i+k$ where L_i is at position i and L_{i+k} at position $i+k$ of the new sequence. The number of realized optional and optionally prohibited bonds in the interval $[i...i+k]$ is m. l and s are defined as given above. l is used to find the relative position of the original sequence to the new one resulting from possible insertions and deletions before and at i which is important to the similarity function.[2] Formally this means that $w_{i+k}^i(L_i, L_{i+k}, m, l, s)$ equals

$$\max_{\substack{L_{i+1}...L_{i+k-1} \\ t \in V(s^I, s^D, k) \\ (s^I, s^D) \in SPLIT_k^s}} \left\{ \sum_{i<j\leq i+k} f_j(L_j, l + \sum_{a=1}^{j-i} t_a, t_{j-i}) \,\middle|\, \begin{array}{l} L_i...L_{i+k} \text{ satisfies} \\ E(G)||_{[i...i+k]} \text{ and} \\ ob(L_i...L_{i+k}) = m \end{array} \right\} \tag{2}$$

If the set over which the maxima are taken is empty, the value of the function is considered as $-\infty$. This function forms the central part of our algorithm because at first w treats both objective functions separately (maximizing the similarity and maximizing the number of realized optional bonds and optionally prohibited bonds) and one can combine them afterwards, as follows, to get the value of interest:

1. $MRSODI_S$: Maximize the similarity overall and among all assignment having this maximal similarity, choose the one with the highest number of realized optional and optionally prohibited bonds. The value of interest is $\max_{(L_1,L_n,m)} \max_{|l|\leq 1, |s| \leq n-1} \{w_n^1(L_1, L_n, m, l, s) + f_1(L_1, l, l)\}$.
2. $MRSODI_B$: Maximize the number of realized optional bonds and optionally prohibited bonds first. Therefore, the value of interest is

[2] **Notice:** In contrast to position $i+k$, the similarity at position i is not included here.

$\max_{(L_1,L_n)} \max_{|l|\leq 1, |s| \leq n-1} \{w_n^1(L_1, L_n, m_{max}, l, s) + f_1(L_1, l, l)\}$, where $m_{max} \leq maxOp$ is the maximal value which can be adopted by m without violating any conditions.

The *basic case* for w is the following: ($m = 0, ..., maxOp$, $|l| \leq i$, $|s| \leq 1$)

$$w_{i+1}^i(L_i, L_{i+1}, m, l, s) = \begin{cases} f_{i+1}(L_{i+1}, l+s, s) & \text{if } (i, L_i) \equiv_{E(G)} (i+1, L_{i+1}), \\ & \text{and } (l,s) \text{ are VALID for } \{i, 1\}, \\ & \text{and } ob(L_i L_{i+1}) = m \\ -\infty & \text{otherwise} \end{cases} \tag{3}$$

It shows that at most one insertion or deletion per position is allowed. This is an algorithmic reason for inequations $|l| \leq i$ and $|s| \leq k$ mentioned above. Nevertheless, it is not possible to prevent two or more insertions one after another straightforward. The insertion-penalty has to be chosen appropriately.

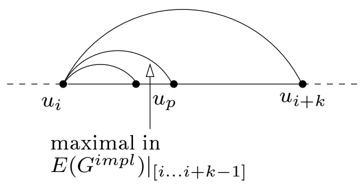

Fig. 4. Definition of $next(i, i+k)$

We solve the problem on an interval $[i...i+k]$ by splitting it into two parts and solving the resulting subproblems. If there is no edge between i and any vertex in $[i...i+k-1]$, the interval is split into $[i...i+1]$ and $[i+1...i+k]$. Otherwise, we choose the farthest vertex p in $[i...i+k-1]$ which is adjacent to i. The edge $\{i, p\}$ is called the *maximal* edge in $E(G^{impl})|_{[i...i+k-1]}$. Then, the interval is split into $[i...p]$ and $[p...i+k]$ (see Figure 4). Hence, we define $next(i, i+k)$ for $k \geq 2$ by

$$next(i, i+k) = \begin{cases} i+r & \text{if } \{i, i+r\} \text{ is maximal in } E(G^{impl})|_{[i...i+k-1]} \\ i+1 & \text{otherwise.} \end{cases}$$

Theorem 1 (Recurrence). *Let (G, f, C) be an instance of $MRSODI_S$ or $MRSODI_B$. For $k \geq 2$, $1 \leq i \leq n-k$, (l, s) are VALID for $\{i, k\}$, $0 \leq m \leq maxOp$, let $p = next(i, i+k)$. Then $w_{i+k}^i(L_i, L_{i+k}, m, l, s)$ equals*

$$\begin{cases} -\infty & \text{if } (i, L_i) \not\equiv_{E(G)} (i+k, L_{i+k}) \\ \max\limits_{\substack{L_p \\ m_1 + m_2 = m_{L_{i+k}}^{L_i} \\ m_1, m_2 \geq 0}} \; \max\limits_{\substack{s_1 + s_2 = s; \\ |s_1| \leq p - i; \\ |s_2| \leq i+k-p}} \begin{pmatrix} w_p^i(L_i, L_p, m_1, l, s_1) + \\ w_{i+k}^p(L_p, L_{i+k}, m_2, l+s_1, s_2) \end{pmatrix} & \\ & \text{if } (i, L_i) \equiv_{E(G)} (i+k, L_{i+k}) \end{cases} \tag{4}$$

where $m_{L_{i+k}}^{L_i} = m - opt_{L_{i+k}}^{L_i}$ and $opt_{L_{i+k}}^{L_i}$ is the number of realized optional bonds only between codons L_i and L_{i+k}.

A proof of the Theorem is given in the appendix B.

Remark 1. The similarity at the left boundary of the interval $[i...i+k]$ is not included in $w_{i+k}^i(L_i, L_{i+k}, m, l, s)$. It suffices to keep the codon on this position to combine the subproblems properly.

4.1 Complexity of the Algorithm

We present the properties of the algorithm at level $[i...i+k]$ of the recursion. Recall that our goal is to compute $w_{i+k}^{i}(L_i, L_{i+k}, m, l, s)$ for all choices of L_i, L_{i+k}, m, l, s, corresponding assignments of codons and insertions and deletions. Therefore, $O(n^2)$ (to be exact: $64^2 * maxOp * (2i+1) * (2k+1)$) different cases have to be considered. Let $p = next(i, i+k)$. Since each vertex in G^{impl} has, at most, degree 3, p can be found in not more than 3 steps. If $w_p^i(L_i, L_p, m_1, l, s_1)$ and $w_{i+k}^{p}(L_p, L_{i+k}, m_{L_{i+k}}^{L_i} - m_1, l + s_1, s - s_1)$ are known for all choices of L_p, m_1 and s_1, we can compute $w_{i+k}^{i}(L_i, L_{i+k}, m, l, s)$. This computation takes at most linear time because only s_1 depends on n. Therefore, a subproblem can be solved in $O(n^3)$.

Now we have to check how many subproblems will be generated. Note that the subproblems are determined by the result of $next(.,.)$. Since G^{impl} is outerplanar, the application of $next(.,.)$ to some specific subproblem will always return a new position (a position that has not been considered in any other subproblem). Hence, we get only $n-2$ subproblems which can be split again and take $O(n^3)$ time each. (Notice that the basic-case-subproblems need only $O(n)$ time.) Thus, we can compute our goal in $O(n^4)$. Analogous to a fixed parameter tractability, this can be reduced to $O(n^2)$ by restricting the number of insertions and deletions to a constant number.

4.2 Results

We have implemented the algorithm in C++ and used it to search for mammalian selenoproteins in which a typical SECIS-element of *E.coli* (*FdhF*) can be designed directly after the UGA-position with only a few changes within the amino acid sequence. According to [10,12] we chose SECIS-consensus sequences and structures as shown in Figure 5.

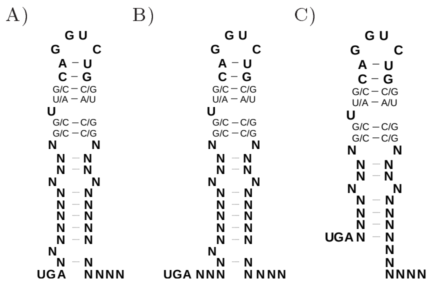

Fig. 5. SECIS-consensus sequence and structure: A) standard, B) an additional codon between UGA and the actual SECIS-element, C) standard, lacking the first codon

The mammalian methionine sulfoxide reductase B (MsrB) has already been studied in [4]. We gained an even better sequence with a higher similarity to the original amino acid sequence (BLOSUM62-score = 43) than the one described in [4] which yields an BLOSUM62-score of 35. Our results are shown in Figure 6. Using our program, we have found modifications of several human selenoproteins that have high similarity to both the original amino acid sequence and the SECIS-element. Based on these modifications the coding mRNA is able to form the required hairpin structure. For more results see appendix A.

A)	mouse MsrB	… U I F S S S L K F V P K G K E …	
	new_RecSec	… U I F S S L P G L V P K G K E …	Sim = **43**
	RecSec [4]	… U I F S T V A G L H P K G K E …	Sim = 35
	new_mRNA_seq	… UGAAUUUUCUCUUCGCUACCAGGUCUGGUGCCAAAAGGAAAAGAA	
	new_mRNA_struc	… .....(((((.((.((.((((....)))))).)).)))))).....	opt_bonds = 9
B)	mouse MsrB	… U I F S S S L K - F V P K G K E …	
	new_RecSec	… U I F S S S L P G L V P K G K E …	Sim = 57 + DP
	new_mRNA_seq	… UGAAUAUUUUCCUCUUCGCUACCAGGUCUGGUGCCAAAAGGAAAAGAA	
	new_mRNA_struc	… ......(.(((((.((.((.((((....)))))).)).)))))))....	opt_bonds = 10
C)	mouse MsrB	… U I F S S S L K F V P K G K E …	
	new_RecSec	… U I F S L P - G L V P K G K E …	Sim = 41 + IP
	new_mRNA_seq	… UGAAUCUUUUCGCUACCAGGUCUGGUGCCAAAAGGUAAAGAA	
	new_mRNA_struc	… ...(((((((.((.((((....)))))).)))))))......	opt_bonds = 7

Fig. 6. Results for MsrB, designing (A) the standard SECIS-element shown in Fig. 5A, (B) the SECIS-element of Fig. 5B, (C) the standard SECIS-element missing the first codon ($IP = -10$, $DP = -5$, changed positions are underlined) where RecSec is the sequence designed in [4], new_RecSec is the one designed by our algorithm

References

1. E. S. Arner, H. Sarioglu, F. Lottspeich, A. Holmgren, and A. Bock. High-level expression in escherichia coli of selenocysteine-containing rat thioredoxin reductase utilizing gene fusions with engineered bacterial-type SECIS elements and co-expression with the selA, selB and selC genes. *Journal of Molecular Biology*, 292(5):1003–16, 1999.
2. Rolf Backofen, N. S. Narayanaswamy, and Firas Swidan. Protein similarity search under mRNA structural constraints: application to targeted selenocysteine insertion. *In Silico Biology*, 2(3):275–90, 2002.
3. Rolf Backofen, N.S. Narayanaswamy, and Firas Swidan. On the complexity of protein similarity search under mrna structure constraints. In H. Alt and A. Ferreira, editors, *Proc. of 19th International Symposium on Theoretical Aspects of Computer Science (STACS2002),*, volume 2285 of *Lecture Notes in Computer Science*, pages 274–286, Berlin, 2002. Springer Verlag.
4. Shoshana Bar-Noy and Jackob Moskovitz. Mouse methionine sulfoxide reductase b: effect of selenocysteine incorporation on its activity and expression of the seleno-containing enzyme in bacterial and mammalian cells. *Biochem Biophys Res Commun*, 297(4):956–61, 2002.
5. A. Bateman, E. Birney, L. Cerruti, R. Durbin, L. Etwiller, S. R. Eddy, S. Griffiths-Jones, K. L. Howe, M. Marshall, and E. L. Sonnhammer. The Pfam protein families database. *Nucleic Acids Res*, 30(1):276–80, 2002.
6. K. M. Brown and J. R. Arthur. Selenium, selenoproteins and human health: a review. *Public Health Nutr*, 4(2B):593–9, 2001.
7. A. Böck, K. Forchhammer, J. Heider, and C. Baron. Selenoprotein synthesis: an expansion of the genetic code. *Trends Biochem Sci*, 16(12):463–467, 1991.
8. Stephane Hazebrouck, Luc Camoin, Zehava Faltin, Arthur Donny Strosberg, and Yuval Eshdat. Substituting selenocysteine for catalytic cysteine 41 enhances enzymatic activity of plant phospholipid hydroperoxide glutathione peroxidase expressed in *escherichia coli*. *Journal of Biological Chemistry*, 275(37):28715–28721, 2000.
9. J. Heider, C. Baron, and A. Böck. Coding from a distance: dissection of the mRNA determinants required for the incorporation of selenocysteine into protein. *EMBO J*, 11(10):3759–66, 1992.

10. A. Hüttenhofer and A. Böck. RNA structures involved in selenoprotein synthesis. In R. W. Simons and M. Grunberg-Manago, editors, *RNA Structure and Function*, pages 603–639. Cold Spring Habor Laboratory Press, Cold Spring Habor, 1998.
11. Gregory V. Kryukov, Sergi Castellano, Sergey V. Novoselov, Alexey V. Lobanov, Omid Zehtab, Roderic Guigo, and Vadim N. Gladyshev. Characterization of mammalian selenoproteomes. *Science*, 300(5624):1439–43, 2003.
12. Z. Liu, M. Reches, I. Groisman, and H. Engelberg-Kulka. The nature of the minimal 'selenocysteine insertion sequence' (SECIS) in Escherichia coli. *Nucleic Acids Research*, 26(4):896–902, 1998.
13. Susan C. Low and Marla J. Berry. Knowing when not to stop: selenocysteine incorporation in eukaryotes. *Trends in Biochemical Sciences*, 21(6):203–208, 1996.
14. T. Sandalova, L. Zhong, Y. Lindqvist, A. Holmgren, and G. Schneider. Three-dimensional structure of a mammalian thioredoxin reductase: implications for mechanism and evolution of a selenocysteine-dependent enzyme. *Proc. Natl. Acad. Sci. USA*, 98(17):9533–8, 2001.
15. G. Sawers, J. Heider, E. Zehelein, and A. Böck. Expression and operon structure of the sel genes of escherichia coli and identification of a third selenium-containing formate dehydrogenase isoenzyme. *J Bacteriol.*, 173(16):4983–93, 1991.
16. R. M. Tujebajeva, J. W. Harney, and M. J. Berry. Selenoprotein P expression, purification, and immunochemical characterization. *Journal of Biological Chemistry*, 275(9):6288–94, 2000.
17. F. Zinoni, J. Heider, and A. Böck. Features of the formate dehydrogenase mRNA necessary for decoding of the UGA codon as selenocysteine. *Proc. Natl. Acad. Sci. USA*, 87(12):4660–4, 1990.

A Results

A)	P49895	... P S F M F K - F D Q F K R L ...
	P49895'	... P S F M V P G L D Q F K R L ...
	Q9NZV5	... G S G R T L R E T V L E S S P ...
	Q9NZV5'	... G - G R T L P G L V L E S S P ...
	$P49908_{59}$	... Y L C I I E A S K L E D L R V ...
	$P49908_{59}$'	... Y L C I V - A G L L E D L R V ...
B)	Q9NZV5	... G S G R T L R E T V L E S S P ...
	Q9NZV5'	... G S G R T L P G L V L E S S P ...
	$P49908_{59}$	... Y L C I I E A S K L E D L R V ...
	$P49908_{59}$'	... Y L C I I V A G L L E D L R V ...
	$P49908_{318}$	... Q C K E N L P S L C S C Q G L ...
	$P49908_{318}$'	... Q C K E N L P G L V A C Q G L ...
C)	Q92813	... P P F T S Q L P A F R K L ...
	Q92813'	... P P F L A G L Q A F R K L ...
	P07203	... G T T V R D Y T Q M N E L ...
	P07203'	... G T T V A G L L Q M N E L ...
	P22352	... G L T G Q Y I E L N A L Q ...
	P22352'	... G L T L P G L E L N A L Q ...
	Q9NZJ3	... G Y R R V - F E E Y M R V ...
	Q9NZJ3'	... G Y R L P G L E E Y M R V ...
	P59797	... S Y S L R Y I L L K K S L ...
	P59797'	... S Y S L A G L L L K K S L ...
	O15532	... G Y K S K Y L Q L K K K L ...
	O15532'	... G Y K L A G L Q L K K K L ...
	Q99611	... G C K V P Q E A L L K L L ...
	Q99611'	... G C K V P G L V L L K L L ...

Fig. 7. Results for human selenoproteins, designing (A) the standard SECIS-element shown in Figure 5A, (B) the SECIS-element of Figure 5B, (C) the SECIS-element missing the first codon when $IP = -10$ and $DP = -5$ (changed positions are underlined)

B Proof of the Recurrence (Theorem 1)

Proof. In the first case, when $(i, L_i) \not\equiv_{E(G)} (i+k, L_{i+k})$ the result is true as we follow the convention that the maximum over an empty set is $-\infty$. Thus, we just have to consider the second case where $(i, L_i) \equiv_{E(G)} (i+k, L_{i+k})$. Let $k \geq 2$ and the other conditions of the theorem be fulfilled. We wish to evaluate,

$$\max_{\substack{L_p \\ m_1+m_2=m^{L_i}_{L_{i+k}} \\ m_1, m_2 \geq 0}} \max_{\substack{s_1+s_2=s \\ |s_1| \leq p-i \\ |s_2| \leq i+k-p}} \begin{pmatrix} w^i_p(L_i, L_p, m_1, l, s_1) \\ \qquad +w^p_{i+k}(L_p, L_{i+k}, m_2, l+s_1, s_2) \end{pmatrix}. \tag{5}$$

where $w^i_p(L_i, L_p, m_1, l, s_1)$ and $w^p_{i+k}(L_p, L_{i+k}, m_2, l+s_1, s_2)$ are defined as given in equation (2).

Now we have to show that equation (5) equals

$$\max_{\substack{L_{i+1}...L_{i+k-1} \\ t \in V(s^I, s^D, k) \\ (s^I, s^D) \in SPLIT^s_k}} \left\{ \sum_{i<j\leq i+k} f_j(L_j, l+\sum_{a=1}^{j-i} t_a, t_{j-i}) \middle| \begin{array}{l} L_i...L_{i+k} \text{ satisfies} \\ E(G)||_{[i...i+k]} \text{ and} \\ ob(L_i...L_{i+k}) = m \end{array} \right\} \tag{6}$$

There are no edges between vertices in the interval $[i+1...p-1]$ and vertices in $[p+1...i+k-1]$ because we have assumed as input an outer-planar drawing of G^{impl}. Firstly, we have to prove the compatibility of the codons to the graph in correspondence with the number of realized optional bonds, optional G-U bonds and optionally prohibited bonds and secondly, we have to show that all possible vectors $\in \{-1,0,1\}^n$ (representing insertions and deletions) on all possible assignments of s^I_1, s^I_2, s^D_1 and s^D_2 are considered in both quantities.

Let $L'_{i+1}...L'_{p-1}$ be some codon sequence such that $L_iL'_{i+1}...L'_{p-1}L_p$ satisfies $E(G)||_{[i...p]}$ and $ob(L_i...L_p) = m_1$, and $L''_{p+1}...L''_{i+k-1}$ be a codon sequence such that $L_pL''_{p+1}...L''_{i+k-1}L_{i+k}$ satisfies $E(G)||_{[p...i+k]}$ and $ob(L_p...L_{i+k}) = m_2$. Now we have to show that $L_iL'_{i+1}...L'_{p-1}L_pL''_{p+1}...L''_{i+k-1}L_{i+k}$ satisfies $E(G)||_{[i...i+k]}$ and $ob(L_i...L_{i+k}) = m$. L_i and L_{i+k} are compatible since we have assumed $(i, L_i) \equiv_{E(G)} (i+k, L_{i+k})$. The number of realized optional bonds, optional G-U bonds and optionally prohibited bonds only between L_i and L_{i+k} is $opt^{L_i}_{L_{i+k}}$ (fixed for fixed L_i and L_{i+k}). The compatibility within the regions $[i...p]$ and $[p...i+k]$ holds by assumption. The compatibility of vertex pair (u_r, u_s) with $r \in [i...p-1]$ and $s \in [p+1...i+k]$ holds since (i,p) is a maximal edge and there are no egdes in $E(G^{impl})$ (and corresponding egdes and optional edges in $E(G)||_{[i...i+k]}$) between r and s. Therefore, $ob(L_i...L_{i+k})$ can be identified by adding $ob(L_i...L_p)$, $ob(L_p...L_{i+k})$ and $opt^{L_i}_{L_{i+k}}$:

$$\begin{aligned} ob(L_i...L_{i+k}) &= ob(L_i...L_p) + ob(L_p...L_{i+k}) + opt^{L_i}_{L_{i+k}} \\ &= m_1 + m_2 + opt^{L_i}_{L_{i+k}} \\ &= m^{L_i}_{L_{i+k}} + opt^{L_i}_{L_{i+k}} = m. \end{aligned}$$

The maximum over all possible values of m_1 (and $m_2 = m - opt^{L_i}_{L_{i+k}} - m_1$) ensures that every assignment of realized optional bonds etc. is considered.

Now we have to show, that all possible vectors $t \in V = V(s^I, s^D, k)$ for all possible s^I and s^D are included in the combination of all possible vectors $t^1 \in V^1 = V^1(s_1^I, s_1^D, p-i)$ and $t^2 \in V^2 = V^2(s_2^I, s_2^D, i+k-p)$ on all possible assignments of s_1^I, s_2^I, s_1^D and s_2^D and vice versa, i.e. that

$$\bigcup_{\substack{(s^I, s^D)\\ \in SPLIT_k^s}} V = \bigcup_{\substack{s_1+s_2=s\\ |s_1| \leq p-i\\ |s_2| \leq i+k-p}} \left(\bigcup_{\substack{(s_1^I, s_1^D)\\ \in SPLIT_{p-i}^{s_1}}} V^1 \times \bigcup_{\substack{(s_2^I, s_2^D)\\ \in SPLIT_{i+k-p}^{s_2}}} V^2 \right)$$

The left set (let us name it L, where the right set is denoted as R) includes all vectors with valid numbers of -1 (s^D) and $+1$ (s^I). We have to show, that all vectors included in L are also included in R and vice versa.

Case $L \supseteq R$: Let $s_1 + s_2 = s$ be a valid splitting of s. Let $t^1 = t_1^1 t_2^1 ... t_{p-i}^1$ be a vector $\in V^1(s_1^I, s_1^D, p-i)$ so that $(s_1^I, s_1^D) \in SPLIT_{p-i}^{s_1}$, and $t^2 = t_1^2 t_2^2 ... t_{i+k-p}^2$ be a vector $\in V^2(s_2^I, s_2^D, i+k-p)$ so that $(s_2^I, s_2^D) \in SPLIT_{i+k-p}^{s_2}$. We have to show that $t^1 t^2 = t_1^1 t_2^1 ... t_{p-i}^1 t_1^2 t_2^2 ... t_{i+k-p}^2$ is a vector $\in \bigcup_{\substack{(s^I, s^D)\\ \in SPLIT_k^s}} V(s^I, s^D, k)$.

The vectors t^1 and t^2 have the following properties

$$\begin{array}{lrlr}
t^1: & s_1^I & \text{many} & +1 \\
 & s_1^D & \text{many} & -1 \\
 & (p-i) - (s_1^I + s_1^D) & \text{many} & 0
\end{array}
\qquad
\begin{array}{lrlr}
t^2: & s_2^I & \text{many} & +1 \\
 & s_2^D & \text{many} & -1 \\
 & (i+k-p) - (s_2^I + s_2^D) & \text{many} & 0
\end{array}$$

Thus, it is true that $t^1 t^2$ consists of

$$\begin{array}{rlr}
s_1^I + s_2^I & \text{many} & +1 \\
s_1^D + s_2^D & \text{many} & -1 \\
k - (s_1^I + s_2^I) - (s_1^D + s_2^D) & \text{many} & 0
\end{array}$$

It follows from this that $t^1 t^2 \in V(s_1^I + s_2^I, s_1^D + s_2^D, k) = V(s^I, s^D, k)$ with $(s_1^I + s_2^I, s_1^D + s_2^D) = (s^I, s^D) \in SPLIT_k^s$ since:

- $(s_1^I + s_2^I) - (s_1^D + s_2^D) = (s_1^I - s_1^D) + (s_2^I - s_2^D) = s_1 + s_2 = s$
- $(s_1^I + s_2^I) + (s_1^D + s_2^D) = \underbrace{(s_1^I + s_1^D)}_{\leq p-i} + \underbrace{(s_2^I + s_2^D)}_{\leq i+k-p} \leq k$
- $s_1^I, s_2^I \geq 0 \Rightarrow s^I \geq 0$ and $s_1^D, s_2^D \geq 0 \Rightarrow s^D \geq 0$

Up to now we have shown that $L \supseteq R$.

Case $L \subseteq R$: We still have to prove, that the relation holds in the other direction as well (that $L \subseteq R$).

Let $t = t_1 t_2 ... t_k$ be a vector $\in V(s^I, s^D, k)$ so that $(s^I, s^D) \in SPLIT_k^s$. Then t can be split into $t^1 = t_1 t_2 ... t_{p-i}$ and $t^2 = t_{p-i+1} t_{p-i+2} ... t_k$. The number of insertions (+1) and deletions (-1) in t^1 is $s_1^I \geq 0$ and $s_1^D \geq 0$ and in t^2 it is $s_2^I \geq 0$ and $s_2^D \geq 0$. Obviously, it is true that $s_1^I + s_2^I = s^I$ and $s_1^D + s_2^D = s^D$. Furthermore, it is true that $t^1 \in V^1(s_1^I, s_1^D, p-i)$ with $(s_1^I, s_1^D) \in SPLIT_{p-i}^{s_1 = s_1^I - s_1^D}$ since $s_1^I + s_1^D \leq p-i$ because t^1 has only $p-i$ positions. Analogously, $t^2 \in V^2(s_2^I, s_2^D, i+k-p)$ with $(s_2^I, s_2^D) \in SPLIT_{i+k-p}^{s_2 = s_2^I - s_2^D}$ since $s_2^I + s_2^D \leq i+k-p$. It can also be confirmed that the splitting of t results in vectors t^1 and t^2 which have valid values s_1 and s_2 concerning to the right side (R) since

- $s_1 + s_2 = (s_1^I - s_1^D) + (s_2^I - s_2^D) = (s_1^I + s_2^I) - (s_1^D + s_2^D) = s^I - s^D = s$
- $|s_1| = |s_1^I - s_1^D| \leq p - i$ because $0 \leq s_1^I, s_1^D \leq p - i$ since t^1 has only $p - i$ positions
- $|s_2| = |s_2^I - s_2^D| \leq i + k - p$ because $0 \leq s_2^I, s_2^D \leq i + k - p$ since t^2 has only $i + k - p$ positions

Now we have also shown that $L \subseteq R$. From this and $L \supseteq R$ it follows that $L = R$. Therefore, each possible vector t of insertions and deletions is considered.

The last point to be mentioned is the correct calculation of the similarity. The numbers of insertions and deletions in the interval $[i+1...p]$ result in a further relative displacement (besides l) of the original to the new amino acid sequence. This displacement is the only effect the intervals $[i+1...p]$ and $[p+1...i+k]$ have on each other, which has to be considered during the calculation of the similarity within $[p+1...i+k]$. This is achieved through the similarity function $f_x = f_x(L_x, \sum_{j \leq x} t_j, t_x)$ taking into account all insertions and deletions made before x. Since the similarity function is additive, the similarity can be calculated separately in both intervals.

Therefore, the two quantities above are equal because the maximum for the problem in the region $[i...i+k]$ for fixed L_i, L_p, L_{i+k}, m_1, l and s_1 can be obtained by finding $w_p^i(L_i, L_p, m_1, l, s_1)$ (maximum in the region $[i...p]$) and $w_{i+k}^p(L_p, L_{i+k}, m_{L_{i+k}}^{L_i}, l + s_1, s - s_1)$ (maximum in the region $[p...i+k]$). To find the maximum in $[i...i+k]$ with only L_i and L_{i+k} fixed, we have to maximize over all possible assignments of L_p, m_1 and s_1.

□