Universität Bielefeld

Technische Fakultät
Abteilung Informationstechnik
Forschungsberichte

Sequencing from compomers in the presence of false negative peaks

Sebastian Böcker

Report 2003-07

Impressum: Herausgeber:
Robert Giegerich, Ralf Hofestädt, Franz Kummert,
Peter Ladkin, Ralf Möller, Helge Ritter,
Gerhard Sagerer, Jens Stoye, Ipke Wachsmuth

Technische Fakultät der Universität Bielefeld,
Abteilung Informationstechnik, Postfach 10 01 31,
33501 Bielefeld, Germany

ISSN 0946-7831

SEQUENCING FROM COMPOMERS IN THE PRESENCE OF FALSE NEGATIVE PEAKS

SEBASTIAN BÖCKER

Abstract. One of the main endeavors in today's Life Science remains the efficient sequencing of long DNA molecules. Today, most de-novo sequencing of DNA is still performed using electrophoresis-based Sanger Sequencing, based on the Sanger concept of 1977. Methods using mass spectrometry to acquire the Sanger Sequencing data are limited by short sequencing lengths of 15–25 nt. Recently, we proposed a new method for DNA sequencing using base-specific cleavage and mass spectrometry, that appears to be a promising alternative to classical DNA sequencing approaches. This leads to the combinatorial problem of Sequencing From Compomers (SFC) and, finally, to the graph-theoretical problem of finding a walk in a subgraph of the de Bruijn graph. Simulations indicate that this method might be capable of sequencing DNA molecules with 200+ nt.

But the way the Sequencing From Compomer Problem is formulated, it does not take into account the problem of *false negative peaks* that is common for real-world data: Even though an *in silico* simulation predicts a peak to be present in a mass spectrum, it is absent from the measured mass spectrum. We may evade this problem by choosing a very sensitive peak detection algorithm, minimizing the number of false negative peaks. Still, a single false negative peak is usually sufficient to prohibit reconstruction of the correct DNA sequence by SFC.

Here, we show how to extend SFC as well as sequencing graphs to deal with false negative peaks. In addition, we present a branch-and-bound algorithm to find all sequences that agree with the sample mass spectra with the exception of a certain number of false negative peaks. Simulation results indicate that even in the presence of several false negative peaks, the presented method might be capable of sequencing DNA molecules of length 200 nt.

Contact: Sebastian Boecker
AG Genominformatik
Technische Fakultaet
Universitaet Bielefeld
PF 100 131
33501 Bielefeld
Germany
boecker@CeBiTec.uni-bielefeld.de

Date: September 12, 2003.

Sebastian Böcker is currently supported by "Deutsche Forschungsgemeinschaft" (BO 1910/1-1) within the Computer Science Action Program.

1. Introduction

Today, most de-novo sequencing of DNA without any *a priori* information regarding the sample sequence under examination, is still performed based on the Sanger concept from 1977, see (Sanger et al., 1977). Typically, gel or capillary electrophoresis is used to acquire the sample data. Many other methods were proposed during the last decades (Maxam and Gilbert, 1977; Bains and Smith, 1988; Lysov et al., 1988; Jett et al., 1989; Köster et al., 1996; Ronaghi et al., 1998; França et al., 2002), but none was able to compete with Sanger Sequencing regarding sequencing length, cost, and reliability. In particular, the sequencing length of 500–1000 bases is an order of magnitude higher than for most other de-novo sequencing methods.

In (Böcker, 2003b) we propose a new approach to DNA de-novo sequencing not based on the Sanger concept, using MALDI-TOF mass spectrometry to acquire the experimental data. It has the potential advantages of fast data acquisition and reliability, among others. Furthermore, we introduce the Sequencing From Compomers (SFC) Problem as an abstraction of the resulting data analysis issues. Simulations indicate that this method might be capable of sequencing DNA molecules with 200+ nt, so sequencing lengths have the same order of magnitude as for Sanger Sequencing.

But a shortcoming of the Sequencing From Compomers Problem is its inability to cope with false negative peaks in the mass spectra: A *false negative peak* (or *missing peak*) is a peak that an *in silico* simulation predicts to be present in a mass spectrum — assuming "error-free" biochemistry and mass spectrometry — but that cannot be detected in the measured mass spectrum. We can possibly evade this problem by choosing a very sensitive peak detection algorithm minimizing the number of false negative peaks, while simultaneously leading to the detection of many false positive peaks: a *false positive peak* (or *additional peak*) is a peak detected in the measured mass spectrum, that was not predicted by an *in silico* simulation. Simulation results in (Böcker, 2003a) indicate that even a large portion of false positive peaks will generally not interfere with reconstruction of the correct sequence. Still, a *single* false negative peak is usually sufficient to prohibit reconstruction of the correct DNA sequence using SFC.

In this paper, we extend the Sequencing From Compomer Problem — and, in particular, the graph theoretical tool introduced in (Böcker, 2003b) to solve it — to deal with false negative peaks in the sample mass spectrum. For that, we introduce the Weighted Sequencing from Compomers (WSC) Problem and weighted sequencing graphs, and show how the latter can be used to solve WSC. We have applied our method to simulated mass spectra generated from random as well as biological sequences, and simulation results indicate high chances of successful reconstruction even in the presence of false negative peaks.

2. Experimental setup and data acquisition

Suppose we are given an amplified, single stranded target DNA molecule (or *sample DNA*) of length 100–500 nt.[1] We cleave the sample sequence with a base-specific chemical or biochemical cleavage reaction: Such reactions cleave at exactly those positions where a specific base can be found. Several methods to achieve base-specific cleavage, such as RNAse A (Hartmer et al., 2003), have been described in literature (Rodi et al., 2002; von Wintzingerode et al., 2002). We modify the cleavage reaction by offering a mixture of cleavable versus non-cleavable "cut bases," such that not all cut bases but only a certain percentage will be cleaved. The resulting mixture contains in principle all fragments that can be obtained from the sample DNA by removing two cut bases, cf. Fig. 1 for an example. We call such cleavage reactions *partial*.

MALDI TOF mass spectrometry (MS for short) is then applied to the products of the cleavage reaction, resulting in a sample spectrum that correlates mass and signal intensity of sample particles (Karas and Hillenkamp, 1988). The sample spectrum is analyzed to extract

[1]We will talk about sample DNA even though a cleavage reaction might force us to transcribe the sample to RNA.

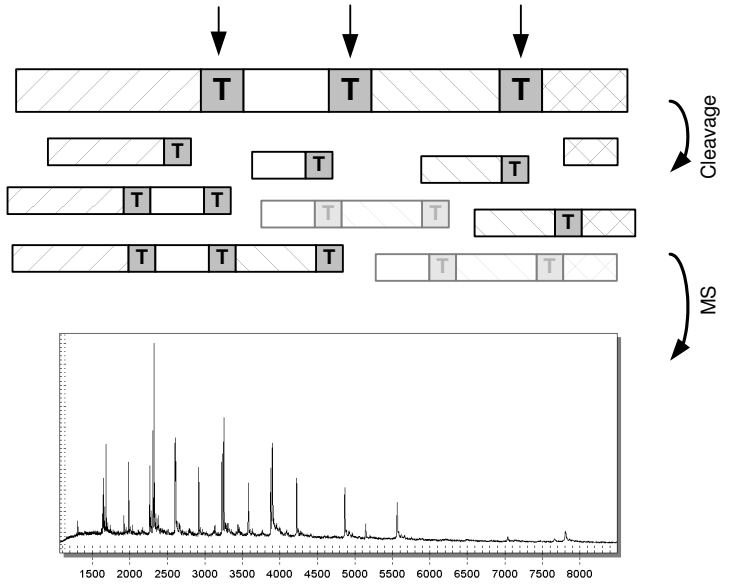

FIGURE 1. Partial cleavage using RNAse A with dCTP, rUTP, and dTTP. Gray fragments indicate that *corresponding peaks* might not be detected in the sample mass spectrum.

a list of signal peaks with masses and intensities. We can repeat the above procedure, as well as the following analysis steps, using cleavage reactions specific to all four bases.

If the sample sequence is known, then exact chemical results of the used cleavage reactions and, in particular, the masses of all resulting fragments are known in advance and can be simulated by an *in silico* experiment. Clearly, this holds up to a certain extent only, leading to the detection of false positive, and the non-detection of false negative peaks in the sample mass spectrum.

Having said that, we can also solve the inverse problem: For every peak detected in the sample mass spectrum, we can calculate one or more base compositions (that is, DNA molecules with unknown order but known multiplicity of bases) that could have created the detected peak, taking into account the inaccuracy of the mass spectrometry read. Therefore, we obtain a list of base compositions (or compomers, see the next section) and their intensities, for every incorporated cleavage method.

In real life, several limitations characteristic for mass spectrometry and partial cleavage make the problem of de-novo sequencing from mass spectrometry data more challenging:

- Current mass spectrometers limit the mass range in which particles can be detected: Signals above 8 000 Da ($\approx$ 25 nt) tend to get lost in the spectrum.
- Because MS spectra are noisy, it is often impossible to distinguish between signal peaks with low intensities and "noise peaks" randomly found in the spectrum.
- Using partial cleavage results in an *exponential decay* (in the number of uncleaved cut bases) of signal intensities in the mass spectrum, so peaks from fragments containing many uncleaved cut bases will be difficult or impossible to detect.
- Peak intensities are comparatively hard to predict by an *in silico* simulation of the cleavage reaction (Böcker, 2003c).

Here, we have listed only those limitations relevant in the context of false negative peaks, see (Böcker, 2003b) for a more detailed list. In this paper, the last limitation above is of particular interest to us: Potentially, the intensity of a peak in a sample mass spectrum is so weak that this peak cannot be detected in the "noise" of the mass spectrum. A sensitive peak detection algorithm can reduce the number of false negative peaks, but it cannot completely eliminate them in all cases. The biochemical and physical causes for the variation of peak

intensities are not completely understood, but it is believed that one of the causes are distinct ionization characteristics of different biomolecules.

3. Methods

Mostly we will follow the notation of (Böcker, 2003b) and assume that the reader is familiar with the basic concepts presented there.

3.1. **The compomer spectrum.** Let $s = s_1 \dots s_n$ be a string over the alphabet Σ where $|s| = n$ denotes the *length* of s. We denote the concatenation of strings a, b by ab, the empty string of length 0 by ϵ.

If $s = axb$ holds for some strings a, x, b then x is called a *substring* of s, a is called a *prefix* of s, and b is called a *suffix* of s. We define the *number of occurrences* of x in s by:

$$\mathrm{ord}_x(s) := \max\bigl\{k \,:\, \text{there exist } s_0, \dots, s_k \in \Sigma^* \text{ with } s = s_0 x s_1 x \dots x s_k\bigr\}$$

Hence, x is a substring of s if and only if $\mathrm{ord}_x(s) \geq 1$.

For strings $s, x \in \Sigma^*$ we define the *string spectrum* $\mathcal{S}(s, x)$ of s by:

$$\mathcal{S}(s,x) := \bigl\{y \in \Sigma^* \,:\, \text{there exist } a, b \in \Sigma^* \text{ with } s \in \{yxb, axyxb, axy\}\bigr\} \cup \{s\} \tag{1}$$

So, the string spectrum $\mathcal{S}(s, x)$ consists of those substrings of s that are bounded by x or by the ends of s. In this context, we call s *sample string* and x *cut string*, while the elements $y \in \mathcal{S}(s, x)$ will be called *fragments* of s (under x). We use special characters $\mathbf{0}$, $\mathbf{1}$ to uniquely denote start and end of the sample string.

We use the following mathematical representation of base compositions: We define a *compomer* to be a map $c : \Sigma \to \mathbb{Z}$, where $\mathbb{Z}$ denotes the set of integers. We say that c is a *natural* compomer if $c(\sigma) \geq 0$ holds for all $\sigma \in \Sigma$. For the rest of this paper, we assume that all compomers are natural compomers, unless explicitly stated otherwise. Let $\mathcal{C}_+(\Sigma)$ denote the set of all natural compomers over the alphabet Σ. Clearly, $\mathcal{C}_+(\Sigma)$ is closed with respect to addition, as well as multiplication with a scalar $n \in \mathbb{N}$, where $\mathbb{N}$ denotes the set of natural numbers *including* 0. We denote the canonical partial order on the set of compomers over Σ by $\preceq$, that is, $c \preceq c'$ if and only if $c(\sigma) \leq c'(\sigma)$ for all $\sigma \in \Sigma$. Furthermore, we denote the *empty compomer* $c \equiv 0$ by 0.

For $\Sigma = \{\sigma_1, \dots, \sigma_k\}$ we use the notation $c = (\sigma_1)_{i_1} \dots (\sigma_k)_{i_k}$ to represent the compomer $c : \sigma_j \mapsto i_j$ omitting those characters σ_j with $i_j = 0$. Since the characters $\mathbf{0}$, $\mathbf{1}$ appear at most once in any fragment, we usually omit the indices for these two characters.

The function $\mathrm{comp} : \Sigma^* \to \mathcal{C}_+(\Sigma)$ maps a string $s = s_1 \dots s_n \in \Sigma^*$ to the compomer of s by counting the number of characters of each type in s. Formally, we define $\mathrm{comp}(s) : \Sigma \to \mathbb{N}$ by

$$\mathrm{comp}(s)(\sigma) := \bigl|\{1 \leq i \leq |s| \,:\, s_i = \sigma\}\bigr| \quad \text{for all } \sigma \in \Sigma.$$

For example, set $\Sigma := \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}$ and $c := \mathrm{comp}(\mathrm{ACCTA})$, then $c(\mathrm{A}) = 2$, $c(\mathrm{C}) = 2$, $c(\mathrm{G}) = 0$, and $c(\mathrm{T}) = 1$ or, equivalently, $c = \mathrm{A}_2\mathrm{C}_2\mathrm{T}_1$. The *compomer spectrum* $\mathcal{C}(s, x)$ of s consists of the compomers of all fragments in the string spectrum:

$$\mathcal{C}(s,x) := \mathrm{comp}\bigl(\mathcal{S}(s,x)\bigr) = \bigl\{\mathrm{comp}(y) \,:\, y \in \mathcal{S}(s,x)\bigr\} \tag{2}$$

Recall that the problem of reconstructing a string s from its compomer spectra $\mathcal{C}(s, x)$, $x \in \mathcal{X}$, cannot be used for experimental MS data: This approach does not take into account the limitations of mass spectrometry and partial cleavage mentioned in the previous section. In particular, signals from fragments y with $\mathrm{ord}_x(y)$ above a certain threshold will be lost in the noise of the mass spectrum. Hence, for strings s, x and $k \in \mathbb{N} \cup \{\infty\}$, we define the k-string spectrum of s by:

$$\mathcal{S}_k(s,x) := \{y \in \mathcal{S}(s,x) \,:\, \mathrm{ord}_x(y) \leq k\} \tag{3}$$

The integer k is called the *order* of the string spectrum. The k-compomer spectrum of s is defined by:

$$\mathcal{C}_k(s,x) := \operatorname{comp}\big(\mathcal{S}_k(s,x)\big) = \big\{\operatorname{comp}(y) \,:\, y \in \mathcal{S}(s,x),\ \operatorname{ord}_x(y) \le k\big\} \tag{4}$$

If the cut string is a single character $x \in \Sigma$, we infer $\mathcal{C}_k(s,x) = \{c \in \mathcal{C}(s,x) \,:\, c(x) \le k\}$.

In applications, a tiny peak detected in a mass spectrum can account for several compomers in the corresponding compomer set, and trying to minimize the number of unexplained detected peaks contradicts the experimental observation. To this end, it makes more sense to find all "good" strings with compomer spectra that are subsets of the measured compomer (or mass) spectra. Accordingly, we define in (Böcker, 2003b):

Sequencing From Compomers (SFC) Problem. For a fixed order $k \in \mathbb{N} \cup \{\infty\}$, let $\mathcal{X} \subseteq \Sigma^*$ be the set of cut strings and, for all $x \in \mathcal{X}$, let $\mathcal{C}_x \subseteq \mathcal{C}_+(\Sigma)$ be a compomer set. Finally, let $S \subseteq \Sigma^*$ be the set of sample string candidates. Now, find all strings $s \in S$ that satisfy $\mathcal{C}_k(s,x) \subseteq \mathcal{C}_x$ for all $x \in \mathcal{X}$.

Clearly, the formulation of SFC does not capture the problem of false negative peaks in the sample mass spectrum: As described in Section 2, there is a number of factors that might lead to the non-detection of a peak in a sample mass spectrum even though theory predicts it to be present. In such cases, the set of "measured" compomers $\mathcal{C}_x \subseteq \mathcal{C}_+(\Sigma)$ will also be missing a compomer that corresponds to the false negative peak or, formally: the set $\mathcal{C}_k(s,x) \setminus \mathcal{C}_x$ is non-empty. Then, the correct sample string is no solution of this instance of SFC. But other, incorrect strings might be solutions of this instance, further aggravating the problem.

3.2. **Weighted compomers.** Let $\mathbb{R}$ denote the set of real numbers and $\mathbb{R}_{\ge 0}$ that of non-negative real numbers. To overcome the shortcoming of SFC, we introduce a *compomer weight function*

$$w_x^{\text{comp}} : \mathcal{C}_+(\Sigma) \to \mathbb{R}_{\ge 0} \tag{5}$$

where $w_x^{\text{comp}}(c)$ represents the "chance" that the peak corresponding to the compomer c is missing in a sample mass spectrum of the cleavage reaction with cut string $x \in \Sigma^*$. Intuitively, w_x^{comp} can be used to *penalize* for missing compomers. In the following, we will limit our attention to compomer weight functions with $w_x^{\text{comp}}(c) \ge 0$ for all $c \in \mathcal{C}_+(\Sigma)$. The simplest weight function $w_x^{\text{comp}} \equiv 1$ corresponds to counting false negative peaks, see below. Another simple but reasonable weight function — capturing the aspect of exponential decay of peak intensities in partial cleavage experiments — is defined by

$$w_x^{\text{comp}}(c) := r^{c(x)} \tag{6}$$

where $x \in \Sigma$ is the cut string of length 1, and the constant $r \in [0,1]$ corresponds to the portion of uncleaved cut bases. If we have some stochastic model to compute the probability that a peak corresponding to some compomer c will be missing from the sample spectrum, then a straightforward choice for $w_x^{\text{comp}}(c)$ is the log-likelihood of this event.

So, w_x^{comp} provides a penalty measure for compomers that are missing from an *arbitrary* sample mass spectrum of the cleavage reaction with cut string $x \in \Sigma^*$. Now, we concentrate on a *fixed* sample mass spectrum: We define a weight function $w_x : \mathcal{C}_+(\Sigma) \to \mathbb{R}_{\ge 0}$ that takes into account if we have observed a compomer in the fixed sample mass spectrum. A simple way of doing so is to transform the sample mass spectrum for cut string x, into a set of observed compomers $\mathcal{C}_x \subseteq \mathcal{C}_+(\Sigma)$ as described in Section 2: that is, for every peak detected in the mass spectrum with mass m, we add all those compomers to $\mathcal{C}_x$ that have masses sufficiently close to m. Then we define

$$w_x(c) := \big(1 - \chi_{\mathcal{C}_x}(c)\big) \cdot w_x^{\text{comp}}(c) = \begin{cases} 0 & \text{for } c \in \mathcal{C}_x \\ w_x^{\text{comp}}(c) & \text{for } c \notin \mathcal{C}_x \end{cases} \tag{7}$$

for all $c \in \mathcal{C}_+(\Sigma)$, where $\chi_{\mathcal{C}_x}$ is the characteristic function of $\mathcal{C}_x$. In general, w_x may also consider peak intensities and peak masses in the sample mass spectrum: If we expect a peak corresponding to some compomer c to have higher intensity then observed in the sample mass spectrum, then we can assign some weight $w_x(c) > 0$ to represent this unexpected intensity loss. Motivated by (7), we call $w_x : \mathcal{C}_+(\Sigma) \to \mathbb{R}_{\geq 0}$ a *characteristic compomer weight*.

Example 1. Let $s := \mathbf{0}\mathrm{CTGATCCGCTATCCTGG}\mathbf{1}$ be the sample string, $x := \mathrm{T}$ the cut string, and $k = 1$ the order. Suppose that the set of observed compomers

$$\mathcal{C}_\mathrm{T} := \{\mathbf{0}\mathrm{C}_1, \mathbf{0}\mathrm{A}_1\mathrm{C}_1\mathrm{G}_1\mathrm{T}_1, \mathrm{A}_1\mathrm{G}_1, \mathrm{A}_1\mathrm{C}_3\mathrm{G}_2\mathrm{T}_1, \mathrm{C}_3\mathrm{G}_1, \mathrm{A}_1, \mathrm{A}_1\mathrm{C}_2\mathrm{T}_1, \mathrm{C}_2, \mathrm{C}_2\mathrm{G}_2\mathrm{T}_1\mathbf{1}, \mathrm{G}_2\mathbf{1}\} \subsetneq \mathcal{C}_1(s, x)$$

that was generated from the detected peaks of some sample mass spectrum, is missing the compomer $\mathcal{C}_1(s, x) \setminus \mathcal{C}_\mathrm{T} = \{\mathrm{A}_1\mathrm{C}_3\mathrm{G}_1\mathrm{T}_1\}$. If we use the compomer weight function (6) with $r := \frac{1}{2}$ and the characteristic compomer weight w_T from (7), then:

- $w_\mathrm{T}(c) = 0$ holds for all $c \in \mathcal{C}_\mathrm{T}$
- $w_\mathrm{T}(c) = 1$ holds for all $c \in \mathcal{C}_+(\Sigma)$ with $c \notin \mathcal{C}_\mathrm{T}$ and $c(\mathrm{T}) = 0$
- $w_\mathrm{T}(c) = \frac{1}{2}$ holds for all $c \in \mathcal{C}_+(\Sigma)$ with $c \notin \mathcal{C}_\mathrm{T}$ and $c(\mathrm{T}) = 1$
- in particular, $w_\mathrm{T}(\mathrm{A}_1\mathrm{C}_3\mathrm{G}_1\mathrm{T}_1) = \frac{1}{2}$

A straightforward way to define a "false negative peak penalty" for a sample string candidate s, is to sum up the weights $w_x(c)$ of all compomers $c \in \mathcal{C}_k(s, x)$. But this does not capture the multiplicity of compomers in the compomer spectrum $\mathcal{C}_k(s, x)$: For $w_x(c) > 0$, two strings s, s' with $c \in \mathcal{C}_k(s, x) \cap \mathcal{C}_k(s', x)$ will receive the same penalty for generating the compomer c, even though c might correspond to a single fragment in s and to many fragments in s'.

Example 2. For $s' := \mathbf{0}\mathrm{CTGATCCTAGTCCTGG}\mathbf{1}$ with $x = \mathrm{T}$ and $\mathcal{C}_\mathrm{T}$ from in Example 1, we calculate $\mathcal{C}_1(s', x) \setminus \mathcal{C}_\mathrm{T} = \{\mathrm{A}_1\mathrm{C}_2\mathrm{G}_1\mathrm{T}_1\}$. So, the cardinality $|\mathcal{C}_1(s', x) \setminus \mathcal{C}_\mathrm{T}| = 1$ is the same as for the correct sample string s in Example 1, and in case $w_\mathrm{T}(\mathrm{A}_1\mathrm{C}_2\mathrm{G}_1\mathrm{T}_1) \approx w_\mathrm{T}(\mathrm{A}_1\mathrm{C}_3\mathrm{G}_1\mathrm{T}_1)$, we would have to regard s and s' as strings of comparable "quality". But this contradicts the observation that the compomer $\mathrm{A}_1\mathrm{C}_2\mathrm{G}_1\mathrm{T}_1$ is generated three times by s'.

One way to solve this problem is to modify string and compomer spectra to be multisets instead of simple sets. Here, we use a different approach that allows us to use regular sets: We define the *multiplicity* of some compomer $c \in \mathcal{C}_+(\Sigma)$ with respect to $s, x \in \Sigma^*$ by

$$\mathrm{mult}_{s,x}(c) := \left|\left\{(a, y, b) \in (\Sigma^*)^3 \,:\, c = \mathrm{comp}(y) \text{ and } s \in \{yxb, axyxb, axy, y\}\right\}\right| \tag{8}$$

Informally, $\mathrm{mult}_{s,x}(c)$ simply counts the number and multiplicity of fragments y in $\mathcal{S}(s, x)$ such that $c = \mathrm{comp}(y)$ holds. For $s' = \mathbf{0}\mathrm{CTGATCCTAGTCCTGG}\mathbf{1}$ from Example 2 we calculate $\mathrm{mult}_{s',\mathrm{T}}(\mathrm{A}_1\mathrm{C}_2\mathrm{G}_1\mathrm{T}_1) = 3$.

This enables us to define a sensible "false negative peak penalty" $w_{k,x} : \Sigma^* \to \mathbb{R}_{\geq 0}$ by:

$$w_{k,x}(s) := \sum_{c \in \mathcal{C}_k(s,x)} \mathrm{mult}_{s,x}(c) \cdot w_x(c) \tag{9}$$

For $w_x^\mathrm{comp} \equiv 1$ and $w_x = 1 - \chi_{\mathcal{C}_x}$ defined in (7), $w_{k,x}(s)$ counts the number of compomers (with multiplicities) missing from the sample compomer set $\mathcal{C}_x$.

Example 3. For $s, s', x = \mathrm{T}, \mathcal{C}_\mathrm{T}$, and w_T from Examples 1 and 2, we calculate

$$w_{1,\mathrm{T}}(s) = \mathrm{mult}_{s,x}(\mathrm{A}_1\mathrm{C}_3\mathrm{G}_1\mathrm{T}_1) \cdot w_\mathrm{T}(\mathrm{A}_1\mathrm{C}_3\mathrm{G}_1\mathrm{T}_1) = 1 \cdot \tfrac{1}{2} = \tfrac{1}{2}$$
$$\text{and} \quad w_{1,\mathrm{T}}(s') = \mathrm{mult}_{s',x}(\mathrm{A}_1\mathrm{C}_2\mathrm{G}_1\mathrm{T}_1) \cdot w_\mathrm{T}(\mathrm{A}_1\mathrm{C}_2\mathrm{G}_1\mathrm{T}_1) = 3 \cdot \tfrac{1}{2} = \tfrac{3}{2}.$$

Hence, s' is penalized stronger than the correct sample string s, as desired.

We use (9) to establish a weighted version of SFC taking into account false negative peaks. Recall that we do not use the compomer sets $\mathcal{C}_x$ for doing so, because their information is included in the characteristic compomer weights using, say, equation (7).

Weighted Sequencing from Compomers (WSC) Problem. For a fixed order $k \in \mathbb{N} \cup \{\infty\}$, let $\mathcal{X} \subseteq \Sigma^*$ be the set of cut strings and, for all $x \in \mathcal{X}$, let $w_x : \mathcal{C}_+(\Sigma) \to \mathbb{R}_{\geq 0}$ be the characteristic compomer weight for cut string x. Finally, let $S \subseteq \Sigma^*$ be the set of sample string candidates. Now, find all strings $s \in S$ minimizing

$$\varphi(s) := \sum_{x \in \mathcal{X}} w_{k,x}(s) \tag{10}$$

where $w_{k,x}$ is defined in (9).

The following two generalizations of WSC are evident: In applications, we will usually extend our search to strings s such that $\varphi(s)$ is sufficiently close to the minimal weight. To do so, we define a nondecreasing *weight delta function* $\delta_{\text{weight}} : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ and search for all strings $s \in S$ such that

$$\varphi(s) \leq \big(1 + \delta_{\text{weight}}(\varphi_{\min})\big) \cdot \varphi_{\min} \tag{11}$$

where $\varphi_{\min} \in \mathbb{R}_{\geq 0}$ denotes the minimum of φ on S. Though the definition allows for an arbitrary function δ_{weight}, this will generally be a constant or linear function. Secondly, we can limit our search to strings $s \in S$ such that $\varphi(s) \leq b$ holds for some given threshold $b \in \mathbb{R}_{\geq 0}$.

It is clear that SFC can be seen as a special case of every version of WSC: For an instance of SFC, we set $w_x^{\text{comp}} \equiv 1$ and use w_x from (7). For $\delta_{\text{weight}} \equiv 0$ and $b := 0$, a string $s \in S$ is a solution of SFC if and only if it is a solution of WSC. So, every version of WSC is at least as hard as SFC.

3.3. **The de Bruijn graph.** A *directed graph* consists of a set V of vertices and a set $E \subseteq V^2 = V \times V$ of edges. An edge (v, v) for $v \in V$ is called a *loop*. We limit our attention to finite directed graphs with finite vertex sets. A *walk* in G is a finite sequence $p = (p_0, p_1, \ldots, p_n)$ of elements from V with $(p_{i-1}, p_i) \in E$ for all $i = 1, \ldots, n$, and $|p| := n$ denotes the *length* of p. An *edge weighting* of a directed graph with edge set E is a function $\tilde{w} : E \to \mathbb{R}$; in the following, we concentrate on edge weightings such that $\tilde{w}(e) \geq 0$ holds for all edges $e \in E$.

For an alphabet Σ and an order $k \geq 1$, the *de Bruijn graph* $B_k(\Sigma)$ is a directed graph with vertex set $V_k := \Sigma^k$ and edge set

$$E_k := \big\{(u, v) \in V_k^2 \,:\, u_{j+1} = v_j \text{ for all } j = 1, \ldots, k-1\big\}$$

where $u = (u_1, \ldots, u_k)$ and $v = (v_1, \ldots, v_k)$. We use the Cartesian product notation $v = (v_1, \ldots, v_k)$ instead of the string notation $v = v_1 \ldots v_k$ for the sake of lucidity. In the following, we denote an edge $\big((e_1, \ldots, e_k), (e_2, \ldots, e_{k+1})\big)$ of $B_k(\Sigma)$ by $(e_1, \ldots, e_{k+1})$ for short.

For a cut string $x \in \Sigma^1$ of length 1, a *compomer alphabet* over (Σ, x) is a subset

$$\Sigma_x \subseteq \{c \in \mathcal{C}_+(\Sigma) \,:\, c(x) = 0\} \cup \{*\} \tag{12}$$

where $* \in \Sigma_x$ denotes a special source character we require to be an element of every compomer alphabet. Note that we can *add* compomer characters $c, c' \in \Sigma_x$: For the source character $* \in \Sigma_x$ we formally define $c + * = * + c = *$ for every compomer c.

Recall that the edges of the de Bruijn graph $B_k\big(\Sigma_x \setminus \{*\}\big)$ are $(k+1)$-tuples of compomers over the alphabet Σ. We use the notation

$$e_{[i,j]} := e_i + \text{comp}(x) + e_{i+1} + \text{comp}(x) + \cdots + e_{j-1} + \text{comp}(x) + e_j \quad \in \mathcal{C}_+(\Sigma) \tag{13}$$

for $1 \leq i \leq j \leq k+1$ to denote the compomer corresponding to parts of an edge $e = (e_1, \ldots, e_{k+1})$ of $B_k(\Sigma_x)$, if the reference to the cut string x is clear. Note that $e_{[i,j]} = *$ holds if and only if there exists an index $i' \in [i, j]$ such that $e_{i'} = *$. Otherwise, we have $e_{[i,j]}(x) = j - i$ in case $x \in \Sigma^1$.

For sample string $s \in \Sigma^*$ and cut string $x \in \Sigma^*$, we call strings $s_0, \ldots, s_l \in \Sigma^*$ satisfying

$$s = s_0 x s_1 x s_2 x \ldots x s_l \tag{14}$$

and $\text{ord}_x(s_j) = 0$ for all $j = 0, \ldots, l$ an *x-partitioning* of s. For $x \in \Sigma^1$, there exists exactly one x-partitioning of s.

Example 4. The T-partitioning of $s := \mathbf{0}$CTGATCCGCTATCCTGG$\mathbf{1}$ from Example 1 is

$$(s_0, s_1, s_2, s_3, s_4, s_5) = (\mathbf{0}\mathrm{C}, \mathrm{GA}, \mathrm{CCGC}, \mathrm{A}, \mathrm{CC}, \mathrm{GG}\mathbf{1}).$$

Let Σ be an alphabet, $x \in \Sigma^1$ a cut string of length 1, and Σ_x a compomer alphabet over (Σ, x). A string $s \in \Sigma^*$ is called *compatible* with a walk $p = p_0 \dots p_{|p|}$ in the de Bruijn graph $B_k(\Sigma_x)$ if the x-partitioning $s_0, \dots, s_l \in \Sigma^*$ of s from (14) satisfies $l = |p|$ and

$$p_j = \big(c_{j-k+1}, c_{j-k+2}, \dots, c_j\big) \quad \text{for } j = 0, \dots, l\,, \tag{15}$$

where $c_j := \mathrm{comp}(s_j)$ for $j = 0, \dots, l$, and $c_{-j} := *$ for all integers $j > 0$. Note that we have modified the definition of compatibility from (Böcker, 2003b) to take into account the source character $*$. This will allow us to state Theorem 1 below in a formally simple way.

Remark 1. If p is compatible with some string s then $(*, \dots, *)$ is the first vertex of p.

Proposition 2. *For an alphabet Σ and a cut string $x \in \Sigma^1$, let Σ_x be a compomer alphabet over (Σ, x), and $s \in \Sigma^*$ a sample string. Then there exists a walk p in the de Bruijn graph $B_k(\Sigma_x)$ compatible with s if and only if $\mathrm{comp}(s_j) \in \Sigma_x$ holds for all $j = 0, \dots, l$, where $s_0, \dots, s_l \in \Sigma^*$ is the unique x-partitioning of s. Furthermore, there exists at most one such walk p.*

Proposition 3. *Let Σ_x be a compomer alphabet over (Σ, x). For every walk p in the de Bruijn graph $B_k(\Sigma_x)$, there exist one or more strings $s \in \Sigma^*$ compatible with p.*

3.4. **Weighted sequencing graphs.** We now generalize the concept of directed sequencing graphs (Böcker, 2003b) to take into account compomer weights of false negative peaks. For a characteristic compomer weight $w_x : \mathcal{C}_+(\Sigma) \to \mathbb{R}_{\geq 0}$, a cut string x, and a compomer alphabet $\Sigma_x \subseteq \{c \in \mathcal{C}_+(\Sigma) : c(x) = 0\} \cup \{*\}$, we define the *weighted sequencing graph* $G_k(x, \Sigma_x; w_x)$ of order $k \geq 1$ as follows: This is an edge-weighted directed graph, consisting of the de Bruijn graph $B_k(\Sigma_x)$ of order k, together with an edge weighting $\tilde{w}_x : E_k \to \mathbb{R}_{\geq 0}$ defined by

$$\tilde{w}_x(e_1, \dots, e_{k+1}) := \sum_{i=1}^{k+1} w_x(e_{[i,k+1]}) \tag{16}$$

Example 5. Let $x := \mathrm{T}$ be the cut string and w_T be the characteristic compomer weight from Example 1. Set the compomer alphabet

$$\Sigma_x := \mathcal{C}_0(s, x) = \{\mathbf{0}\mathrm{C}_1, \mathrm{A}_1\mathrm{G}_1, \mathrm{C}_3\mathrm{G}_1, \mathrm{A}_1, \mathrm{C}_2, \mathrm{G}_2\mathbf{1}\}$$

for s also from Example 1. We have depicted the weighted sequencing graph $G_1(\mathrm{T}, \Sigma_\mathrm{T}; w_\mathrm{T})$ in Figure 2.

Given a walk $p = (p_0, \dots, p_l)$ in a directed graph G with edge weighting $\tilde{w}_x$, we define the *weight* of p by

$$\tilde{w}_x(p) := \sum_{j=1}^{l} \tilde{w}_x(e_j)\,, \qquad \text{where } e_j := (p_{j-1}, p_j) \text{ for all } j = 1, \dots, l. \tag{17}$$

Theorem 1. *Let $s \in \Sigma^*$ be a string, $x \in \Sigma^1$ a cut string, and $w_x : \mathcal{C}_+(\Sigma) \to \mathbb{R}$ a characteristic compomer weight. Suppose we are given a walk p in the weighted sequencing graph $G_k(x, \Sigma_x; w_x)$ where Σ_x is a compomer alphabet over (Σ, x). If s and p are compatible, then*

$$w_k(s, x) = \tilde{w}_x(p) \tag{18}$$

holds, where $w_k(s, x)$ is defined in (9) and $\tilde{w}_x$ denotes the edge weighting of $G_k(x, \Sigma_x; w_x)$.

Proof. From Remark 1 we know that $(*, \dots, *)$ is the first vertex of p. We use induction on $l = |p|$, where $p = \big((*, \dots, *)\big)$ is clearly compatible with $s = \epsilon$, and both have zero weight.

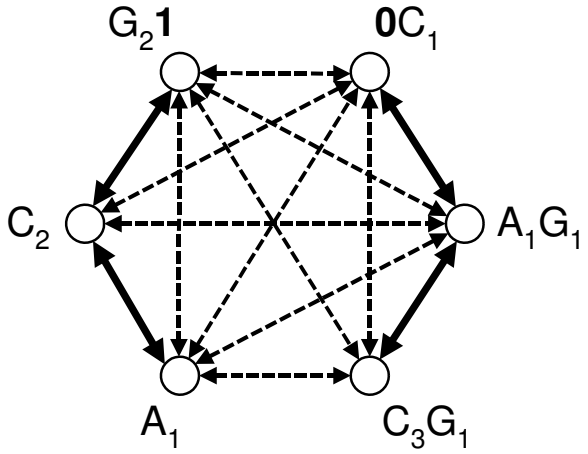

FIGURE 2. The weighted sequencing graph $G_1(\mathrm{T}, \Sigma_\mathrm{T}; w_\mathrm{T})$ from Example 5. Solid edges e represent an edge weight $\tilde{w}(e) = 0$, while dashed edges e' represent an edge weight $\tilde{w}(e') = \frac{1}{2}$.

Since s, p are compatible we know $|p| = l$, and (15) hold for $p = (p_0, \ldots, p_l)$ and the x-partitioning $s_0, \ldots, s_l$ of s. Suppose that $w_k(s', x) = w(p')$ holds for $p' = p_0 \ldots p_{l-1}$ and $s' = s_0 x s_1 x \ldots x s_{l-1}$.

Let $j_0 := \max\{0, l-k\}$. Now, $\mathcal{S}_k(s, x)$ contains $\mathcal{S}_k(s', x)$ as well as $s_j x s_{j+1} x \ldots x s_l$ for all $j = j_0, \ldots, l$. Formally,

$$\mathrm{mult}_{s,x}(c_j^*) = \mathrm{mult}_{s',x}(c_j^*) + 1$$

holds for all $c_j^* := \mathrm{comp}(s_j x s_{j+1} x \ldots x s_l)$ where $j = j_0, \ldots, l$. For all remaining compomers $\mathcal{C}_+(\Sigma)$ the multiplicity does not change between s' and s. Inserting into (9) gives us

$$w_k(s, x) - w_k(s', x) = \sum_{j=j_0}^{l} \left(\mathrm{mult}_{s,x}(c_j^*) - \mathrm{mult}_{s',x}(c_j^*)\right) \cdot w_x(c_j^*) = \sum_{j=j_0}^{l} w_x(c_j^*)\,.$$

On the other hand, let $e := (p_{l-1}, p_l)$ be the last edge of p. We derive $e = (e_1, e_2, \ldots, e_{k+1}) = (c_{l-k}, c_{l-k+1}, \ldots, c_l)$ what implies

$$e_{[i,k+1]} = \mathrm{comp}(e_i x \ldots x e_{k+1}) = c_{l-k+i-1} + \mathrm{comp}(x) + \cdots + \mathrm{comp}(x) + c_l$$

for $i = 1, \ldots, k+1$. So, $e_{[i,k+1]} = *$ and $w_x(e_{[i,k+1]}) = 0$ holds for $l - k + i - 1 < 0$ or, equivalently, for $i \leq k - l$; while $e_{[i,k+1]} = c_{l-k+i-1}^*$ for $i > k - l$. Using the index transformation $l - k + i - 1 \mapsto j$ we calculate

$$\begin{aligned}\tilde{w}_x(p) - \tilde{w}_x(p') = \tilde{w}_x(e) &= \sum_{i=1}^{k+1} w_x\big(e_{[i,k+1]}\big) = \sum_{i=\max\{1,k-l+1\}}^{k+1} w_x\left(c_{l-k+i-1}^*\right) \\ &= \sum_{j=\max\{l-k,0\}}^{l} w_x\left(c_j^*\right) = \sum_{j=j_0}^{l} w_x\left(c_j^*\right)\end{aligned}$$

and finally conclude $w_k(s, x) - w_k(s', x) = \tilde{w}_x(p) - \tilde{w}_x(p')$. □

Clearly, weighted de Bruijn graphs are a generalization of directed sequencing graphs, that are subgraphs of de Bruijn graphs: By choosing an edge weighting of 1 for edges not present in the directed sequencing graph, and 0 for all other edges, both graphs contain the same information. But note that there exists no immediate correspondence between directed

sequencing graphs $G_k(\mathcal{C}, x)$ and weighted sequencing graphs $G_k(x, \Sigma_x; w_x)$: Given an edge $(e_1, \ldots, e_{k+1}) \in (\Sigma_x)^{k+1}$ of the de Bruijn graph, (16) uses only the weights of compomers $e_{[i,k+1]}$ for $i = 1, \ldots, k+1$ while for directed sequencing graphs, all $e_{[i,j]}$ for $1 \le i \le j \le k+1$ are taken into account. This is used to thin out directed sequencing graphs, but has no direct equivalent in the setting of weighted sequencing graphs.

We must point out that in the definition of weighted sequencing graphs, we assume the compomer alphabet Σ_x to be known beforehand. But this is not the case in applications where peaks corresponding to fragments with no internal cut base might be missing from the sample mass spectrum. On the other hand, our constructions are based on the assumption that $\mathcal{C}_0(s, x) \subseteq \Sigma_x$ holds for the correct sample string s. If this condition is violated then the correct sample string cannot be constructed using weighted sequencing graphs. Work on this topic is currently in progress.

4. Algorithm

The algorithm presented in this section evolved from the one presented in (Böcker, 2003b). Let Σ be a constant and finite alphabet where $\mathbf{0}, \mathbf{1} \in \Sigma$ uniquely denote the first and last character of our sample strings. Let $\mathcal{X} = \Sigma^1 \setminus \{\mathbf{0}, \mathbf{1}\}$ be the set of cut strings, and $k \in \mathbb{N}$ the fixed order. We suppose that we know a compomer alphabet Σ_x such that $\mathcal{C}_0(s, x) \subseteq \Sigma_x$ holds for the correct sample string s. We are given characteristic compomer weights $w_x : \mathcal{C}_+(\Sigma) \to \mathbb{R}_{\ge 0}$ for $x \in \mathcal{X}$ that were generated from sample mass spectra, and a set $S \subseteq \Sigma^*$ of strings. We want to solve the Weighted Sequencing from Compomers Problem in the form that we search for all strings $s \in S$ with $\varphi(s) \le b$ for some threshold $b \in \mathbb{R}$, and (11) in case a solution exists. For the sake of brevity, we define the nondecreasing function $b' : \mathbb{R}_{\ge 0} \to \mathbb{R}_{\ge 0}$ by $b'(x) := \min\{b, (1 + \delta_{\text{weight}}(x)) \cdot x\}$ and search for strings s with $\varphi(s) \le b'(\varphi_{\min})$. We further concentrate on the case that

$$S = \left\{ s \in \Sigma^* \,:\, l_{\min} \le |s| \le l_{\max}, \text{ and } s = \mathbf{0}\, s'\, \mathbf{1} \text{ for some } s' \in (\Sigma \setminus \{\mathbf{0}, \mathbf{1}\})^* \right\} \tag{19}$$

contains all strings of length in a given interval, which is especially relevant for applications.

To solve WSC, we present a depth-first search that backtracks through sequence space, moving along the edges of the sequencing graphs in parallel. In this way, we implicitly build walks in the weighted sequencing graphs of order k that are compatible with the constructed strings. By Theorem 1, every such string s has the same weight $\varphi(s)$ as the sum of weights of the compatible walks. This allows us to do a branch-and-bound check by stopping the recursion as soon as the resulting string has weight above one of the thresholds, because all edge weights are non-negative.

4.1. Recursively building the sequencing graphs.

First, we have to build the sequencing graphs $G_x := G_k(x, \Sigma_x; w_x)$ for $x \in \mathcal{X}$. This means that for every edge e of the de Bruijn graph $B_k(\Sigma_k)$, we have to calculate and store the edge weight $\tilde{w}_x(e)$. Assume that we can calculate the characteristic compomer weight w_x in constant time. The trivial approach using (16) needs $O\big(|\Sigma_x|^{k+1}\, k\big)$ time. A faster method of generating $G_k(x, \Sigma_x; w_x)$ is to iteratively build the graphs $G_\kappa(x, \Sigma_x; w_x)$ for $\kappa = 1, \ldots, k$, what can be done in $O\big(|\Sigma_x|^{k+1}\big)$ time for $|\Sigma_x| \ge 2$.

4.2. The depth-first search.

We make use the of the following notations: s is the current string that will be a prefix of all string candidates constructed in subsequent recursion steps. $\psi \in \mathbb{R}_{\ge 0}$ denotes the weight of the current prefix string s, and $\psi_{\min} \in \mathbb{R}_{\ge 0} \cup \{\infty\}$ denotes the weight of the best solution found so far. Clearly, $\psi_{\min} \ge \varphi_{\min}$ always holds. As we want to construct only strings s satisfying $\varphi(s) \le b'(\varphi_{\min})$, we can stop the recursion as soon as ψ is too large. To this end, $\psi_{\text{bound}} := b'(\psi_{\min}) \in \mathbb{R}_{\ge 0} \cup \{\infty\}$ denotes the current weight bound. h_x denotes the weight change that is added to ψ if we append the character $x \in \Sigma \setminus \{\mathbf{0}\}$ to s. For $x \ne \mathbf{1}$, h_x equals the weight of some edge in G_x. Next, $\tilde{h}_x \ge h_x$ denotes the induced weight

change if we append the character $x \in \Sigma \setminus \{\mathbf{0}, \mathbf{1}\}$: Appending x will force edge transitions in G_σ for $\sigma \neq x$ in subsequent recursion steps. Finally, v_x denotes the active vertex in G_x.

Now, we start the recursion with $s := \mathbf{0}$, $\psi := 0$, $\psi_{\min} := \infty$, and $\psi_{\text{bound}} := b$. We initialize the current vertices $v_x := (*, \ldots, *)$ for all $x \in \mathcal{X}$.

In the *recursion step*, let s be the current prefix string, ψ its weight, $\psi_{\min}$ the best solution weight, and ψ_{bound} the current weight bound. For all $x \in \mathcal{X}$, let v_x be the current active vertices in the sequencing graph G_x. Let s_x be the unique string satisfying $\text{ord}_x(s_x) = 0$ such that either xs_x is a suffix of s, or $s_x = s$ if $\text{ord}_x(s) = 0$. Set $c_x := \text{comp}(s_x)$.

- If $|s| + 1 \geq l_{\min}$ then calculate the weight change $h_{\mathbf{1}}$ appending $\mathbf{1}$.
  - If $\psi + h_{\mathbf{1}} \leq \psi_{\text{bound}}$ then **output** $s\mathbf{1}$ as a string candidate with weight $(\psi + h_{\mathbf{1}})$.
  - If $\psi + h_{\mathbf{1}} \leq \psi_{\min}$ then $\psi_{\min} \leftarrow \psi + h_{\mathbf{1}}$ and $\psi_{\text{bound}} \leftarrow b'(\psi + h_{\mathbf{1}})$.
- If $|s| < l_{\max}$, then calculate the weight change h_x and the induced weight change $\tilde{h}_x$ appending x, for all $x \in \Sigma \setminus \{\mathbf{0}, \mathbf{1}\}$. For every character x satisfying $\psi + \tilde{h}_x \leq \psi_{\text{bound}}$ do a recursion step: Replace s by the concatenation sx; replace ψ by $\psi + h_x$; and in the sequencing graph G_x, replace the active vertex $v_x = (v_1, v_2, \ldots, v_k)$ by $(v_2, \ldots, v_k, c_x)$ that is a vertex of G_x.
- Return to the previous level of recursion.

Here the weight change h_x and induced weight change $\tilde{h}_x$ of a character $x \in \mathcal{X}$ can be calculated as follows: Firstly, if $c_x \in \Sigma_x$ then let $h_x := \tilde{w}_x(v_1, \ldots, v_k, c_x)$ where $v_x = (v_1, \ldots, v_k)$ is the active vertex in G_x. For $c_x \notin \Sigma_x$ we set $h_x := \infty$. Secondly, let

$$\tilde{h}_x := h_x + \sum_{\sigma \in \mathcal{X} \setminus \{x\}} \tilde{h}_{x,\sigma}$$

where $\tilde{h}_{x,\sigma}$ for $\sigma \neq x$ is defined as follows: Let $v_\sigma = (v_1, \ldots, v_k)$ be the active vertex in G_σ. Let E_σ denote the set of edges $(v_1, \ldots, v_k, c'_\sigma)$ in G_σ starting in v_σ and satisfying $c_\sigma \preceq c'_\sigma$. If $E_\sigma = \emptyset$ then set $\tilde{h}_{x,\sigma} := \infty$. Otherwise, we define $\tilde{h}_{x,\sigma} := \min_{e \in E_\sigma} \tilde{w}_\sigma(e)$.

Finally, the weight change $h_{\mathbf{1}}$ of the end character $\mathbf{1} \in \Sigma$ is calculated as follows: For all $x \in \mathcal{X}$, let $v_x = (v_1, \ldots, v_k)$ be the active vertex in G_x. Set $c'_x := c_x + \mathbf{1}_1$, where $\mathbf{1}_1$ denotes the compomer containing exactly one end character. Then, we set

$$h_{\mathbf{1}} := \sum_{x \in \mathcal{X}} \tilde{w}_x(v_1, \ldots, v_k, c'_x)$$

where, in analogy to above, we set $h_{\mathbf{1}} := \infty$ if there exists at least one $x \in \mathcal{X}$ such that $c'_x \notin \Sigma_x$.

As a post-processing step of the algorithm, we can sort out all string candidates s with weight $\varphi(s) > b'(\psi_{\min})$. Example 6 shows a single step of the algorithm.

Example 6. Let $\Sigma = \{\mathbf{0}, \text{A}, \text{C}, \text{G}, \text{T}, \mathbf{1}\}$ be the DNA alphabet. Suppose we enter the recursion step of the above algorithm with current prefix string $s = \mathbf{0}\text{GACAGGCTCTTA}$ and weight $\psi = \psi_{\text{bound}} - 2$. Portions of the weighted sequencing graphs of order $k = 2$ and, in particular, the active vertices and their successors are displayed in Fig. 3. Here, we have omitted some edges e leaving the active vertex with weight $\tilde{w}_x(e) > 2$, because no such edge will be used in the calculations of h_x and $\tilde{h}_x$ for $x \in \mathcal{X}$. Now, the weight changes and induced weight changes for $x \in \mathcal{X}$ are as follows:

- For the character A we have $\tilde{h}_\text{A} \geq h_\text{A} = \tilde{w}_\text{A}(\text{C}_1, \text{C}_2\text{G}_2\text{T}_3, 0) = 3$, so $\psi + \tilde{h}_\text{A} > \psi_{\text{bound}}$ and A is not appended to s.
- For C we have $h_\text{C} = \tilde{w}_\text{C}(\text{A}_1\text{G}_2, \text{T}_1, \text{A}_1\text{T}_2) = 0$, but $\tilde{h}_{\text{C,A}} = \tilde{w}_\text{A}(\text{C}_1, \text{C}_2\text{G}_2\text{T}_3, \text{C}_2\text{G}_2\text{T}_3) = 1.5$, $\tilde{h}_{\text{C,G}} = \tilde{w}_\text{G}(\text{A}_2\text{C}_1, 0, \text{A}_1\text{C}_3\text{T}_5) = 0$, and $\tilde{h}_{\text{C,T}} = \tilde{w}_\text{T}(\text{C}_1, 0, \text{A}_1\text{C}_2) = 1$. Hence, $\tilde{h}_\text{C} = 0 + 1.5 + 0 + 1 = 2.5$ still prohibits to append the character C in view of $\psi + \tilde{h}_\text{C} > \psi_{\text{bound}}$. The reasoning behind this is as follows: At some recursion step, we will have to append the character A, forcing an edge transition in G_A. But among all

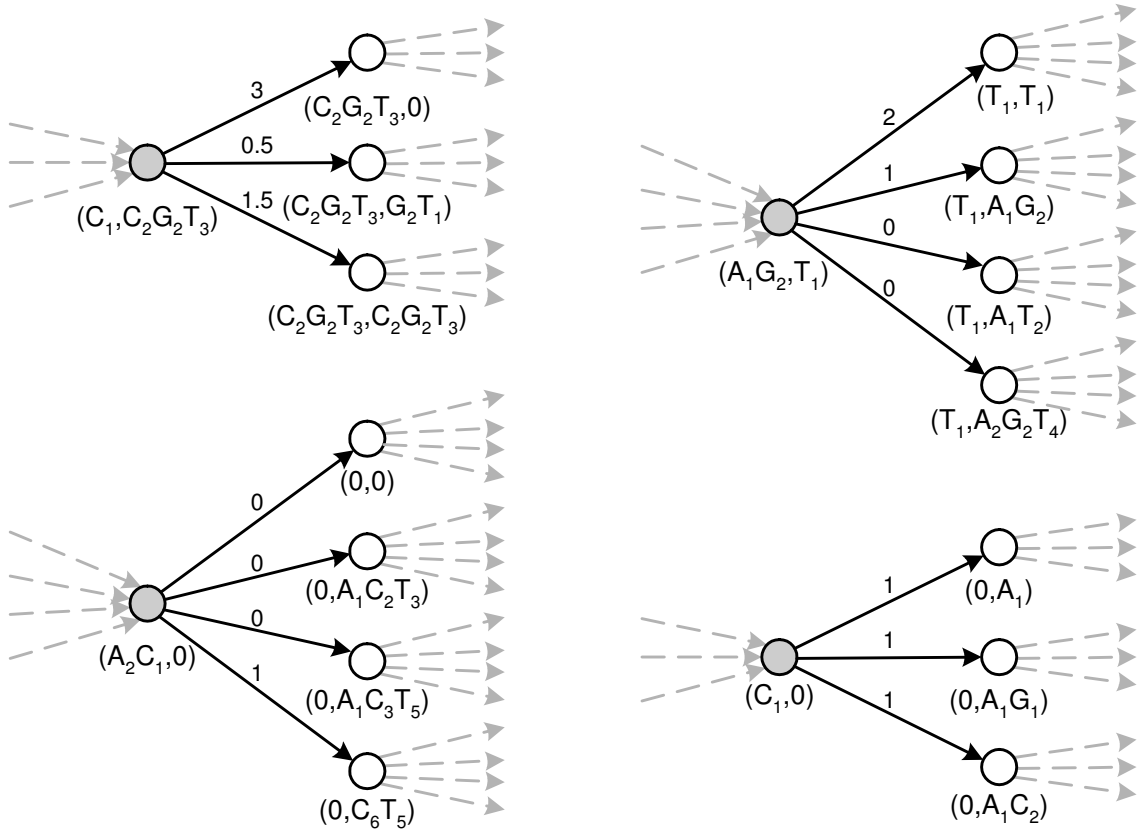

FIGURE 3. One step of the algorithm as described in Example 6: Portion of the weighted sequencing graphs G_{A}, G_{C} (top, left and right) and G_{G}, G_{T} (bottom, left and right). Active vertices marked in gray, some edges e with weight $\tilde{w}(e) \geq \psi_{\mathrm{bound}} - \psi$ omitted.

edges of G_{A} starting in $(\mathrm{C}_1, \mathrm{C}_2\mathrm{G}_2\mathrm{T}_3)$ that come into question, $(\mathrm{C}_1, \mathrm{C}_2\mathrm{G}_2\mathrm{T}_3, \mathrm{C}_2\mathrm{G}_2\mathrm{T}_3)$ has minimal weight. The edge $(\mathrm{C}_1, \mathrm{C}_2\mathrm{G}_2\mathrm{T}_3, \mathrm{G}_2\mathrm{T}_1)$ is not considered in this calculation: At this future point of the recursion, the newly added character C will be a prefix of s_{A} and, hence, $\mathrm{C}_1 \preceq c_{\mathrm{A}}$. But $\mathrm{C}_1 \not\preceq \mathrm{G}_2\mathrm{T}_1$, so after appending the character C, the edge traversed next in G_{A} cannot be $(\mathrm{C}_1, \mathrm{C}_2\mathrm{G}_2\mathrm{T}_3, \mathrm{G}_2\mathrm{T}_1)$.

- The character G will be appended, because $h_{\mathrm{G}} = \tilde{w}_{\mathrm{G}}(\mathrm{A}_2\mathrm{C}_1, 0, \mathrm{A}_1\mathrm{C}_2\mathrm{T}_3) = 0$, $\tilde{h}_{\mathrm{G,A}} = \tilde{w}_{\mathrm{A}}(\mathrm{C}_1, \mathrm{C}_2\mathrm{G}_2\mathrm{T}_3, \mathrm{G}_2\mathrm{T}_1) = 0.5$, $\tilde{h}_{\mathrm{G,C}} = \tilde{w}_{\mathrm{C}}(\mathrm{A}_1\mathrm{G}_2, \mathrm{T}_1, \mathrm{A}_2\mathrm{G}_2\mathrm{T}_4) = 0$, $\tilde{h}_{\mathrm{G,T}} = \tilde{w}_{\mathrm{G}}(\mathrm{C}_1, 0, \mathrm{A}_1\mathrm{G}_1) = 1$ and, finally, $\tilde{h}_{\mathrm{G}} = 0 + 0.5 + 0 + 1 = 1.5$. Hence, we have $\psi + \tilde{h}_{\mathrm{G}} = \psi + 1.5 < \psi_{\mathrm{bound}}$.
- The character T will also be appended, because we calculate $h_{\mathrm{T}} = \tilde{w}_{\mathrm{T}}(\mathrm{C}_1, 0, \mathrm{A}_1) = 1$, $\tilde{h}_{\mathrm{T,A}} = \tilde{w}_{\mathrm{A}}(\mathrm{C}_1, \mathrm{C}_2\mathrm{G}_2\mathrm{T}_3, \mathrm{G}_2\mathrm{T}_1) = 0.5$, $\tilde{h}_{\mathrm{T,C}} = \tilde{w}_{\mathrm{C}}(\mathrm{A}_1\mathrm{G}_2, \mathrm{T}_1, \mathrm{A}_2\mathrm{G}_2\mathrm{T}_4) = 0$, $\tilde{h}_{\mathrm{T,G}} = \tilde{w}_{\mathrm{G}}(\mathrm{A}_2\mathrm{C}_1, 0, \mathrm{A}_1\mathrm{C}_3\mathrm{T}_5) = 0$ and, finally, $\tilde{h}_{\mathrm{G}} = 1 + 0.5 + 0 + 0 = 1.5$. Again, we infer $\psi + \tilde{h}_{\mathrm{G}} = \psi + 1.5 < \psi_{\mathrm{bound}}$.

In total, we conclude that the characters G, T will be appended to s in two recursion steps: For $x = \mathrm{G}$ we replace s by $\mathbf{0}\mathrm{GACAGGCTCTTAG}$ and update the active vertex $v_{\mathrm{G}} \leftarrow (0, \mathrm{A}_1\mathrm{C}_2\mathrm{T}_3)$ in G_{G}, while ψ stays constant. For $x = \mathrm{T}$ we replace s by $\mathbf{0}\mathrm{GACAGGCTCTTAT}$ and update the active vertex $v_{\mathrm{T}} \leftarrow (0, \mathrm{A}_1)$ in G_{T} and weight $\psi \leftarrow \psi + h_{\mathrm{T}} = \psi + 1$.

Theorem 2. *For all $x \in \mathcal{X} := \Sigma \setminus \{\mathbf{0}, \mathbf{1}\}$, let w_x be characteristic compomer weights satisfying $w_x(c) \geq 0$ for all compomers c. Let Σ_x be a compomer alphabet over (Σ, x). For a fixed order*

k and S as defined in (19)*, the algorithm of this section will return all strings $s \in S$ and their weights $\varphi(s)$ that are solutions of WSC and satisfy $\mathcal{C}_0(s,x) \subseteq \Sigma_x$.*

Proof. Let s be an output string of the algorithm and ψ the associated weight. Traversing through the sequencing graphs, we have implicitly constructed walks p_x in G_x for all $x \in \mathcal{X}$ that are compatible with s, and from the construction we also infer that ψ equals the sum of weights of these walks. Using Theorem 1 we conclude

$$\psi = \sum_{x\in\mathcal{X}} \tilde{w}_x(p_x) = \sum_{x\in\mathcal{X}} w_k(s,x) = \varphi(s).$$

We will show below that in case $\varphi_{\min} \leq b$, the algorithm outputs at least one string s with $\varphi(s) = \varphi_{\min}$, so $\psi_{\min} = \varphi_{\min}$ holds when the algorithm terminates. After post-processing, clearly $\psi \leq b'(\psi_{\min}) = b'(\varphi_{\min})$ holds for every output string with weight ψ.

Now, at any stage of the algorithm we have $\varphi_{\min} \leq \psi_{\min}$, because $\psi_{\min}$ denotes the minimum weight found so far, while $\varphi_{\min}$ is the global minimum of φ. Hence, every string s with $\varphi(s) \leq b'(\varphi_{\min})$ also satisfies $\varphi(s) \leq \psi_{\text{bound}}$ in view of $b'(\varphi_{\min}) \leq b'(\psi_{\min}) = \psi_{\text{bound}}$, because b' is a nondecreasing function.

It remains to be shown that all strings $s \in S$ that satisfy $\varphi(s) \leq b'(\varphi_{\min})$ and $\mathcal{C}_0(s,x) \subseteq \Sigma_x$, are constructed by the algorithm. To this end, let $s \in S$ be such a string. By Proposition 2, there exist unique walks p_x in G_x compatible with s, for every $x \in \mathcal{X}$. We will show by induction that every proper prefix s' of s is an input to the recursion step of the algorithm.

To this end, let s'_x, s_x be strings with $\text{ord}_x(s_x) = 0$ such that either $s' = s'_x x s_x$ holds, or $s'_x = \epsilon$ and $s_x = s$ if $\text{ord}_x(s) = 0$. Using again Proposition 2, there exist a unique walk in G_x compatible with s'_x, and from the uniqueness of such walks we infer that this walk p'_x is a sub-walk of p_x. Since all edge weights are non-negative, we have $\tilde{w}_x(p'_x) \leq \tilde{w}_x(p_x)$. One can easily check that entering the recursion, the active vertex in G_x is the last vertex of p'_x, denoted v'_x. In addition, ψ equals $\sum_{x\in\mathcal{X}} \tilde{w}_x(p'_x)$ at this point. Let u'_x denote the vertex in p_x that succeeds the last vertex of p'_x.

The induction basis is trivial for $s' = \mathbf{0}$. Assume that $s' = s''x$ for some $s'' \in \Sigma^*$ and $x \in \Sigma$. Let p'_σ denote the sub-walks in G_σ corresponding to s' as defined above, and let p''_σ denote the sub-walks corresponding to s''. One can easily see that $p'_\sigma = p''_\sigma$ for all $\sigma \neq x$, while p'_x equals p''_x extended by the single vertex $u''_x = v'_x$.

We have to show that x is an admissible character satisfying $\psi + \tilde{h}_x \leq b'(\varphi_{\min})$. Now, the active vertex in G_σ is $v''_\sigma = v'_\sigma$ and $e_\sigma := (v'_\sigma, u'_\sigma)$ is an edge of p_σ and, hence, of G_σ. From the compatibility of s with p_σ we infer that $c'_\sigma \preceq \text{comp}(s_\sigma) = c_\sigma$ where c'_σ denotes the last component of the vector u'_σ. This implies $e_\sigma \in E_\sigma$ and, hence, $\tilde{h}_{x,\sigma} = \min_{e\in E_\sigma} \tilde{w}_\sigma(e) \leq \tilde{w}_\sigma(e_\sigma)$. This proves

$$\begin{aligned}
\psi + \tilde{h}_x &= \sum_{\sigma\in\mathcal{X}} \tilde{w}_\sigma(p''_\sigma) + h_x + \sum_{\sigma\in\mathcal{X}\setminus\{x\}} \tilde{h}_{x,\sigma} \\
&= \tilde{w}_x(p''_x) + h_x + \sum_{\sigma\in\mathcal{X}\setminus\{x\}} \left(\tilde{w}_\sigma(p''_\sigma) + \tilde{h}_{x,\sigma}\right) \\
&\leq \tilde{w}_x(p'_x) + \sum_{\sigma\in\mathcal{X}\setminus\{x\}} \tilde{w}_\sigma(p_\sigma) \\
&\leq \sum_{\sigma\in\mathcal{X}} \tilde{w}_\sigma(p_\sigma) \;=\; \varphi(s) \;\leq\; b'(\varphi_{\min}) \;\leq\; \psi_{\text{bound}}
\end{aligned}$$

as claimed.

We conclude that s' with $s'\mathbf{1} = s$ is also an input of the recursion step. We can show that at this point, ψ equals $\varphi(s) - h_{\mathbf{1}}$, and it follows that $s'\mathbf{1} = s$ is an output of the algorithm. □

What are time and space requirements of the described algorithm? Because there can be exponentially many solutions to WSC, the worst-case runtime is also exponential in the number of detected peaks, as well as the maximal length of an output string $l_{\max}$. In addition, the runtime can still be exponential even if there is a unique solution to WSC, or no solution at all. Since the de Bruijn graph $B_k(\Sigma_x)$ has $|\Sigma_x|^{k+1}$ edges, we need $O(m^{k+1})$ memory to store the weighted sequencing graphs, where $m := \max\{|\Sigma_x| : x \in \mathcal{X}\}$. For $n := \max\{|s| : s \in S\}$ we need $O(n)$ memory in the recursion part of the algorithm. The critical factor is obviously storing the sequencing graphs and in general prohibits the use of orders $k > 2$: For $k = 3$ and $|\Sigma_x| = 200$ we have to store $6.4 \cdot 10^9$ edge weights in memory. Work on this problem is currently in progress.

The complete process of de-novo sequencing from mass spectrometry data can now be performed as follows: Firstly, we generate detected compomer sets $\mathcal{C}_x$ for all $x \in \mathcal{X}$ as described in (Böcker, 2003b). These sets are used in (7) to define characteristic compomer weights w_x that, in turn, allow us to build weighted sequencing graphs G_x. We use the algorithm of Section 4.2 to generate all sample string candidates s that are solutions to WSC satisfying $\mathcal{C}_0(s, x) \subseteq \Sigma_x$. Clearly, we can further evaluate the generated sample string candidates by, say, an appropriate likelihood measure, taking into account MS data from all cleavage reactions.

Recall that for DNA sequencing, a heuristic used to analyze the mass spectrometry data may or may not find the correct sample string. But this is not acceptable in the setting of de-novo sequencing.

5. Results

We use two types of simulated mass spectrometry data to test the algorithm; application of the method to "real-world" mass spectrometry data is in preparation. Firstly, we generate random sample DNA sequences with uniform base distribution. Secondly, we use a region of 4 Mb around human ApoE (Lai et al., 1998) and randomly cut out sample DNA of the desired length.

For our initial evaluation, we set the order of our sequencing graph to be $k = 2$, and choose a sample DNA length of 200 nt. We simulate four cleavage reactions based on "real-world" RNAse cleavage, where we generate only fragments of order at most k, assuming that peaks from fragments of order $k+1$ and higher cannot be detected in the mass spectrum. Then, we calculate masses of all resulting fragments and disturb peak masses by at most $\delta_{\text{mass}} = 0.3$ Da, the *mass accuracy* of the measurement.

We address false negative peaks in the following way: By Theorem 2 we have to guarantee $\Sigma_x \subseteq \mathcal{C}_0(s, x)$ for the correct sample string s, so we assume that all fragments $y \in \mathcal{S}(s, x)$ of order $\text{ord}_x(y) = 0$ can be detected in the simulated sample spectrum. Choose a number n of false negative peaks. We remove a total of n peaks from the four simulated spectra, where every peak corresponds to a fragment $y \in \mathcal{S}(s, x)$ with $\text{ord}_x(y) \geq 1$. Doing so, we have to take into account fragment multiplicities, see (8) and (9). We conduct these simulations for $n = 0$ corresponding to no false negative peaks, and for $n = 5, 10, 15$ false negative peaks. Here, five false negative peaks represent approximately 1.25% of the peaks in the initially simulated sample mass spectra. Even this small ratio of false negative peaks is reasonable for applications, because we can use an extremely sensitive peak detection, since simulations indicate that the detection of false positive peaks does not interfere with our method.

Next, we transform the spectrum into a set of compomers $\mathcal{C}_x$ as indicated in Section 2: For every peak in the simulated mass spectrum, we calculate all compomers of order at most k that might possibly create a peak with mass at most δ_{mass} off the perturbed signal mass. Note that we do not simulate false positives (additional) peaks here. Finally, we use the characteristic compomer weight w_x as defined in (7), where $w_x^{\text{comp}} \equiv 1$ corresponds to counting peaks. We use the algorithm from Section 4 to construct all string s with $\varphi(s) \leq b := n$, where we choose

the length bounds $l_{\min} := 190$ and $l_{\max} := 210$. For every parameter set, 1000 runs were conducted.

# ambiguous	Random sequence data				ApoE sequence data			
bases	$n = 0$	$n = 5$	$n = 10$	$n = 15$	$n = 0$	$n = 5$	$n = 10$	$n = 15$
0	96.4	95.1	91.9	88.9	83.8	81.0	76.7	71.5
2	3.2	4.0	6.3	6.5	7.1	8.7	9.8	8.4
3	0	0.4	0	0.3	0.4	0.3	0.3	0.3
4	0.2	0	0.5	0.9	2.3	2.6	2.8	2.1
5	0.1	0	0.1	0.3	0.2	0.4	0.1	0.5
6–10	0.1	0	0.2	0.5	3.0	2.6	2.1	1.9
11+	0	0.3	0.6	0.9	2.9	2.9	3.3	3.1
undecidable	0	0.2	0.4	1.7	0.3	1.5	4.9	12.2
runtime	2 ms	3 s	8 s	53 s	5 s	26 s	80 s	200 s

TABLE 1. Results of the simulations for $k = 2$, $l = 200$, and $\delta_{\text{mass}} = 0.3$. For a number m of ambiguous bases, we list the percentage of input strings where the output shows m ambiguous bases. See text for details.

We present the results of our simulations in Table 1. Here we provide the percentage of strings that were constructed with a certain number of *ambiguous* bases: An ambiguous base is a column in the multiple alignment of all output strings, where the aligned output strings differ. As for SFC, we found no case of a single ambiguous base. To limit the runtime of the branch-and-bound algorithm, we stop the algorithm as soon as $5 \cdot 10^7$ branching events are reached after approximately 20 minutes runtime, see below. In all other cases, the correct input string was among the output string candidates by design. The average "runtime" of the branch-and-bound algorithm for one input string was measured on an UltraSparc III processor with 750 MHz.

One can see that reconstruction "accuracy" decreases for increasing numbers of false negative peaks. But this comes as no surprise: Informally, a high number of false negative peaks moves the problem into the direction of spectrum order $k = 1$. We have seen in (Böcker, 2003a) that for $k = 1$, even short strings of length 100 bp cannot be uniquely recovered from their mass spectra in most cases. The increase of "undecidable" input strings, on the other hand, might limit the presented approach to a small number of false negative peaks. We assume that this effect is less pronounced for $k \geq 3$.

6. Discussion and improvements

We have introduced the Weighted Sequencing from Compomers Problem that stems from the analysis of mass spectrometry data from partial cleavage experiments. WSC extends the Sequencing From Compomers Problem introduced in (Böcker, 2003b) by taking into account false negative peaks in the sample mass spectra. Although WSC is computationally difficult in general, we have introduced an approach to perform de-novo sequencing from such data. The introduced method uses weighted de Bruijn graphs to construct all DNA sequences that are "compatible" with the observed mass spectra. We tested the performance of our approach on simulated mass spectrometry data from random and biological sequences. Simulation results indicate that the presented approach is capable of reconstructing the correct sequence in many cases if the ratio of false negative peaks is small, and ambiguities are often limited to a small number of bases. So, this approach may enable de-novo sequencing even when false negative peaks must be taken into account in the mass spectrometry data.

As noted in Section 5, our simulations are only a first step in evaluating the power of the presented approach. A more thorough simulation analysis is currently in progress, in particular for spectrum order $k = 3$. In addition, to guarantee a reasonable runtime in the

string construction recursion, better branch-and-bound conditions are necessary. Finally, the condition $\mathcal{C}_0(s,x) \subseteq \Sigma_x$ can be too restrictive in applications. Work on this is also in progress.

Acknowledgments

Additional programming for the implementation of the presented method was provided by Matthias Steinrücken. I want to thank Zsuzsanna Lipták and Hans-Michael Kaltenbach for proofreading earlier versions of this manuscript.

References

Bains, W. and Smith, G. C. (1988). A novel method for nucleic acid sequence determination. *J. Theor. Biol.*, 135:303–307.

Böcker, S. (2003a). Sequencing from compomers: Using mass spectrometry for DNA de-novo sequencing of 200+ nt. Submitted.

Böcker, S. (2003b). Sequencing from compomers: Using mass spectrometry for DNA de-novo sequencing of 200+ nt. Extended abstract. In *Proceedings of WABI 2003*, Budapest, Hungary. Available from `http://www.cebitec.uni-bielefeld.de/~boecker/`.

Böcker, S. (2003c). SNP and mutation discovery using base-specific cleavage and MALDI-TOF mass spectrometry. *Bioinformatics*, 19:i44–i53. Supplemental for ISMB 2003.

França, L. T. C., Carrilho, E., and Kist, T. B. L. (2002). A review of DNA sequencing techniques. *Q. Rev. Biophys.*, 35(2):169–200.

Hartmer, R., Storm, N., Böcker, S., Rodi, C. P., Hillenkamp, F., Jurinke, C., and van den Boom, D. (2003). RNAse T1 mediated base-specific cleavage and MALDI-TOF MS for high-throughput comparative sequence analysis. *Nucl. Acids. Res.*, 31(9):e47.

Jett, J. H., Keller, R. A., Martin, J. C., Marrone, B. L., Moyzis, R. K., Ratliff, R. L., Seitzinger, N. K., Shera, E. B., and Stewart, C. C. (1989). High-speed DNA sequencing: An approach based upon fluorescence detection of single molecules. *J. Biomol. Struct. Dynam.*, 7:301–309.

Karas, M. and Hillenkamp, F. (1988). Laser desorption ionization of proteins with molecular masses exceeding 10,000 Daltons. *Anal. Chem.*, 60:2299–2301.

Köster, H., Tang, K., Fu, D.-J., Braun, A., van den Boom, D., Smith, C. L., Cotter, R. J., and Cantor, C. R. (1996). A strategy for rapid and efficient DNA sequencing by mass spectrometry. *Nat. Biotechnol.*, 14(9):1084–1087.

Lai, E., Riley, J., Purvis, I., and Roses, A. (1998). A 4-Mb high-density single nucleotide polymorphism-based map around human ApoE. *Genomics*, 54(1):31–38.

Lysov, Y., Floretiev, V., Khorlyn, A., Khrapko, K., Shick, V., and Mirzabekov, A. (1988). DNA sequencing by hybridization with oligonucleotides. *Dokl. Acad. Sci. USSR*, 303:1508–1511.

Maxam, A. M. and Gilbert, W. (1977). A new method for sequencing DNA. *Proc. Nat. Acad. Sci. USA*, 74(2):560–564.

Rodi, C. P., Darnhofer-Patel, B., Stanssens, P., Zabeau, M., and van den Boom, D. (2002). A strategy for the rapid discovery of disease markers using the MassARRAY system. *BioTechniques*, 32:S62–S69.

Ronaghi, M., Uhlén, M., and Nyrén, P. (1998). Pyrosequencing: A DNA sequencing method based on real-time pyrophosphate detection. *Science*, 281:363–365.

Sanger, F., Nicklen, S., and Coulson, A. R. (1977). DNA sequencing with chain-terminating inhibitors. *Proc. Nat. Acad. Sci. USA*, 74(12):5463–5467.

von Wintzingerode, F., Böcker, S., Schlötelburg, C., Chiu, N. H., Storm, N., Jurinke, C., Cantor, C. R., Göbel, U. B., and van den Boom, D. (2002). Base-specific fragmentation of amplified 16S rRNA genes and mass spectrometry analysis: A novel tool for rapid bacterial identification. *Proc. Natl. Acad. Sci. USA*, 99(10):7039–7044.

Bisher erschienene Reports an der Technischen Fakultät
Stand: 2003-09-12

94-01 Modular Properties of Composable Term Rewriting Systems
(Enno Ohlebusch)

94-02 Analysis and Applications of the Direct Cascade Architecture
(Enno Littmann, Helge Ritter)

94-03 From Ukkonen to McCreight and Weiner: A Unifying View of Linear-Time Suffix Tree Construction
(Robert Giegerich, Stefan Kurtz)

94-04 Die Verwendung unscharfer Maße zur Korrespondenzanalyse in Stereo Farbbildern
(André Wolfram, Alois Knoll)

94-05 Searching Correspondences in Colour Stereo Images – Recent Results Using the Fuzzy Integral
(André Wolfram, Alois Knoll)

94-06 A Basic Semantics for Computer Arithmetic
(Markus Freericks, A. Fauth, Alois Knoll)

94-07 Reverse Restructuring: Another Method of Solving Algebraic Equations
(Bernd Bütow, Stephan Thesing)

95-01 PaNaMa User Manual V1.3
(Bernd Bütow, Stephan Thesing)

95-02 Computer Based Training-Software: ein interaktiver Sequenzierkurs
(Frank Meier, Garrit Skrock, Robert Giegerich)

95-03 Fundamental Algorithms for a Declarative Pattern Matching System
(Stefan Kurtz)

95-04 On the Equivalence of E-Pattern Languages
(Enno Ohlebusch, Esko Ukkonen)

96-01 Static and Dynamic Filtering Methods for Approximate String Matching
(Robert Giegerich, Frank Hischke, Stefan Kurtz, Enno Ohlebusch)

96-02 Instructing Cooperating Assembly Robots through Situated Dialogues in Natural Language
(Alois Knoll, Bernd Hildebrand, Jianwei Zhang)

96-03 Correctness in System Engineering
(Peter Ladkin)

96-04 An Algebraic Approach to General Boolean Constraint Problems
(Hans-Werner Güsgen, Peter Ladkin)

96-05 Future University Computing Resources
(Peter Ladkin)

96-06 Lazy Cache Implements Complete Cache
(Peter Ladkin)

96-07 Formal but Lively Buffers in TLA+
(Peter Ladkin)

96-08 The X-31 and A320 Warsaw Crashes: Whodunnit?
(Peter Ladkin)

96-09 Reasons and Causes
(Peter Ladkin)

96-10 Comments on Confusing Conversation at Cali
(Dafydd Gibbon, Peter Ladkin)

96-11 On Needing Models
(Peter Ladkin)

96-12 Formalism Helps in Describing Accidents
(Peter Ladkin)

96-13 Explaining Failure with Tense Logic
(Peter Ladkin)

96-14 Some Dubious Theses in the Tense Logic of Accidents
(Peter Ladkin)

96-15 A Note on a Note on a Lemma of Ladkin
(Peter Ladkin)

96-16 News and Comment on the AeroPeru B757 Accident
(Peter Ladkin)

97-01 Analysing the Cali Accident With a WB-Graph
(Peter Ladkin)

97-02 Divide-and-Conquer Multiple Sequence Alignment
(Jens Stoye)

97-03 A System for the Content-Based Retrieval of Textual and Non-Textual Documents Based on Natural Language Queries
(Alois Knoll, Ingo Glöckner, Hermann Helbig, Sven Hartrumpf)

97-04 Rose: Generating Sequence Families
(Jens Stoye, Dirk Evers, Folker Meyer)

97-05 Fuzzy Quantifiers for Processing Natural Language Queries in Content-Based Multimedia Retrieval Systems
(Ingo Glöckner, Alois Knoll)

97-06 DFS – An Axiomatic Approach to Fuzzy Quantification
(Ingo Glöckner)

98-01 Kognitive Aspekte bei der Realisierung eines robusten Robotersystems für Konstruktionsaufgaben
(Alois Knoll, Bernd Hildebrandt)

98-02 A Declarative Approach to the Development of Dynamic Programming Algorithms, applied to RNA Folding
(Robert Giegerich)

98-03 Reducing the Space Requirement of Suffix Trees
(Stefan Kurtz)

99-01 Entscheidungskalküle
(Axel Saalbach, Christian Lange, Sascha Wendt, Mathias Katzer, Guillaume Dubois, Michael Höhl, Oliver Kuhn, Sven Wachsmuth, Gerhard Sagerer)

99-02 Transforming Conditional Rewrite Systems with Extra Variables into Unconditional Systems
(Enno Ohlebusch)

99-03 A Framework for Evaluating Approaches to Fuzzy Quantification
(Ingo Glöckner)

99-04 Towards Evaluation of Docking Hypotheses using elastic Matching
(Steffen Neumann, Stefan Posch, Gerhard Sagerer)

99-05 A Systematic Approach to Dynamic Programming in Bioinformatics. Part 1 and 2: Sequence Comparison and RNA Folding
(Robert Giegerich)

99-06 Autonomie für situierte Robotersysteme – Stand und Entwicklungslinien
(Alois Knoll)

2000-01 Advances in DFS Theory
(Ingo Glöckner)

2000-02 A Broad Class of DFS Models
(Ingo Glöckner)

2000-03 An Axiomatic Theory of Fuzzy Quantifiers in Natural Languages
(Ingo Glöckner)

2000-04 Affix Trees
(Jens Stoye)

2000-05 Computergestützte Auswertung von Spektren organischer Verbindungen
(Annika Büscher, Michaela Hohenner, Sascha Wendt, Markus Wiesecke, Frank Zöllner, Arne Wegener, Frank Bettenworth, Thorsten Twellmann, Jan Kleinlützum, Mathias Katzer, Sven Wachsmuth, Gerhard Sagerer)

2000-06 The Syntax and Semantics of a Language for Describing Complex Patterns in Biological Sequences
(Dirk Strothmann, Stefan Kurtz, Stefan Gräf, Gerhard Steger)

2000-07 Systematic Dynamic Programming in Bioinformatics (ISMB 2000 Tutorial Notes)
(Dirk J. Evers, Robert Giegerich)

2000-08 Difficulties when Aligning Structure Based RNAs with the Standard Edit Distance Method
(Christian Büschking)

2001-01 Standard Models of Fuzzy Quantification
(Ingo Glöckner)

2001-02 Causal System Analysis
(Peter B. Ladkin)

2001-03 A Rotamer Library for Protein-Protein Docking Using Energy Calculations and Statistics
(Kerstin Koch, Frank Zöllner, Gerhard Sagerer)

2001-04 Eine asynchrone Implementierung eines Microprozessors auf einem FPGA
(Marco Balke, Thomas Dettbarn, Robert Homann, Sebastian Jaenicke, Tim Köhler, Henning Mersch, Holger Weiss)

2001-05 Hierarchical Termination Revisited
(Enno Ohlebusch)

2002-01 Persistent Objects with O2DBI
(Jörn Clausen)

2002-02 Simulation von Phasenübergängen in Proteinmonoschichten
(Johanna Alichniewicz, Gabriele Holzschneider, Morris Michael, Ulf Schiller, Jan Stallkamp)

2002-03 Lecture Notes on Algebraic Dynamic Programming 2002
(Robert Giegerich)

2002-04 Side chain flexibility for 1:n protein-protein docking
(Kerstin Koch, Steffen Neumann, Frank Zöllner, Gerhard Sagerer)

2002-05 ElMaR: A Protein Docking System using Flexibility Information
(Frank Zöllner, Steffen Neumann, Kerstin Koch, Franz Kummert, Gerhard Sagerer)

2002-06 Calculating Residue Flexibility Information from Statistics and Energy based Prediction
(Frank Zöllner, Steffen Neumann, Kerstin Koch, Franz Kummert, Gerhard Sagerer)

2002-07 Fundamentals of Fuzzy Quantification: Plausible Models, Constructive Principles, and Efficient Implementation
(Ingo Glöckner)

2002-08 Branching of Fuzzy Quantifiers and Multiple Variable Binding: An Extension of DFS Theory
(Ingo Glöckner)

2003-01 On the Similarity of Sets of Permutations and its Applications to Genome Comparison
(Anne Bergeron, Jens Stoye)

2003-02 SNP and mutation discovery using base-specific cleavage and MALDI-TOF mass spectrometry
(Sebastian Böcker)

2003-03 From RNA Folding to Thermodynamic Matching, including Pseudoknots
(Robert Giegerich, Jens Reeder)

2003-04 Sequencing from compomers: Using mass spectrometry for DNA de-novo sequencing of 200+ nt
(Sebastian Böcker)

2003-05 Systematic Investigation of Jumping Alignments
(Constantin Bannert)

2003-06 Suffix Tree Construction and Storage with Limited Main Memory
(Klaus-Bernd Schürmann, Jens Stoye)