Determination of Glycan Structure from Tandem Mass Spectra

Sebastian Böcker, Birte Kehr, and Florian Rasche

Abstract—Glycans are molecules made from simple sugars that form complex tree structures. Glycans constitute one of the most important protein modifications and identification of glycans remains a pressing problem in biology. Unfortunately, the structure of glycans is hard to predict from the genome sequence of an organism. In this paper, we consider the problem of deriving the topology of a glycan solely from tandem mass spectrometry (MS) data. We study, how to generate glycan tree candidates that sufficiently match the sample mass spectrum, avoiding the combinatorial explosion of glycan structures. Unfortunately, the resulting problem is known to be computationally hard. We present an efficient exact algorithm for this problem based on fixed-parameter algorithmics that can process a spectrum in a matter of seconds. We also report some preliminary results of our method on experimental data, combining it with a preliminary candidate evaluation scheme. We show that our approach is fast in applications, and that we can reach very well *de novo* identification results. Finally, we show how to count the number of glycan topologies for a fixed size or a fixed mass. We generalize this result to count the number of (labeled) trees with bounded out degree, improving on results obtained using Pólya's enumeration theorem.

Index Terms—Computational mass spectrometry, glycans, parameterized algorithms, exact algorithms, counting trees.

1 Introduction

Glycans are besides nucleic acids and proteins, the third major class of biopolymers and are built from simple sugars. Glycans may occur attached to proteins or lipids, or as free oligosaccharides in the cell plasma. Since simple sugars can have up to five linkage sites, glycans are assembled in a tree-like structure.

The elucidation of glycan structure remains one of the most challenging tasks in biochemistry, yet the proteomics field cannot be completely understood without these important post-translational modifications. Apweiler et al. [1] estimated that more than 50 percent of all eukaryotic proteins are glycosylated, i.e., carry a glycan modification.

One of the most powerful tools for glycan structure elucidation is tandem mass spectrometry (MS) [2], [3]. MS is a technology, which in essence allows to determine the molecular mass of input molecules. Because of its speed and accuracy, it has become a prime technology for the analysis of proteins, metabolites, and glycans. Put in a simplified way, the input of the experiment is a molecular mixture and the output a peak list, a list of masses and their intensities. In tandem MS, we select one type of molecules in the sample, fragment these parent molecules, and measure the masses of all fragments. Ideally, each peak should correspond to the mass of some sample molecule fragment, and its intensity to the frequency of that fragment in the mixture. The situation is, in fact, more blurred due to noise and other factors. Tandem MS can provide general structural information about the glycan, in particular its *topology* that can be represented as a labeled tree (see Fig. 1). Glycan mass spectra can be interpreted by searching a database of reference spectra [4], [5], but such databases are vastly incomplete. Glycan sequencing is more difficult than peptide sequencing in the sense that we try to resolve a tree structure instead of a linear string. Unlike peptide sequencing, the alphabet of monosaccharides (the glycan building blocks) can differ depending on the type of glycan we are analyzing.

In this paper, we focus on the problem of *de novo* interpretation of glycan tandem MS data relying on neither a database nor biological background knowledge about possible glycan structures. This is particularly useful for free oligosaccharides not attached to a protein, O-linked glycans, or bacterial membrane glycans, where biological background knowledge offers little help to reduce the space of possible structures. But our method can also be applied if biological knowledge strongly restricts this space as in the case of N-linked glycans. For example, we can use biological restrictions to filter out candidates after candidate generation, see below.

Recent approaches for *de novo* interpretation of tandem MS data usually build on two analysis steps, the first step being *candidate generation* (filtering) and the second step being *candidate evaluation* [6]. We perform the first step to cut down the huge number of topologies we have to consider. A good candidate generation algorithm will generate a small set of candidates, but will not miss the correct interpretation. During candidate evaluation, we only have to consider a small set of candidates, and decide which of these candidates best fits the measured data. For glycans, a naïve approach to generate candidates is to decompose the parent mass of the

- *S. Böcker and F. Rasche are with the Chair for Bioinformatics, Faculty for Mathematics and Computer Science, Friedrich-Schiller-Universität Jena, Ernst-Abbe-Platz 2, Jena 07743, Germany. E-mail: {sebastian.boecker, florian.rasche}@uni-jena.de.*
- *B. Kehr is with the Algorithmic Bioinformatics Group, Institute for Computer Science, Takustraße 9, Berlin 14195, Germany. E-mail: birte.kehr@fu-berlin.de.*

Manuscript received 7 Oct. 2009; revised 14 Dec. 2009; accepted 15 Feb. 2010; published online 10 Dec. 2010.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-2009-10-0177. Digital Object Identifier no. 10.1109/TCBB.2010.129.

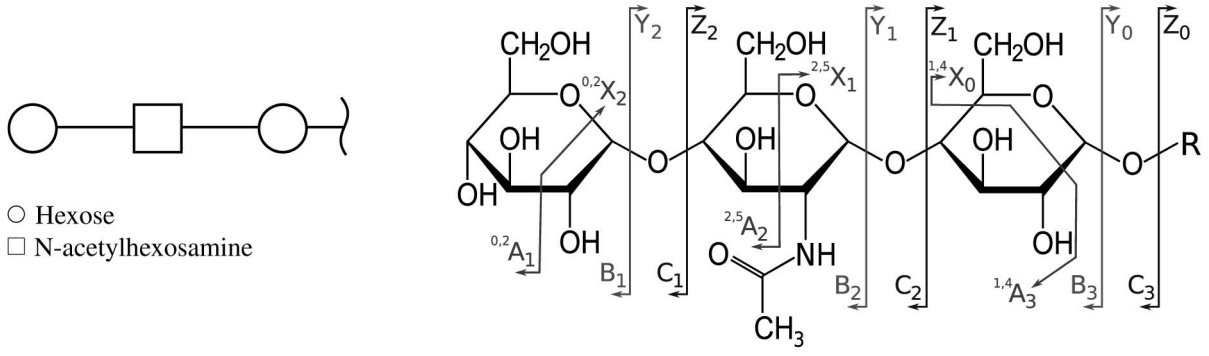

Fig. 1. (left) Topology of a glycan made from three monosaccharides and (right) fragments resulting from tandem MS analysis. Note that the topology of a glycan does not contain information about exact linkage positions.

glycan over the alphabet of monosaccharides [7], and then, to enumerate all topologies that have the correct multiplicities of monosaccharides. If glycans become larger, or if we analyze glycans that have a diverse branching structure, we either have to artificially restrict our search space [8], or we need a smarter way to generate candidates.

Both candidate generation and candidate evaluation rely on certain scoring schemes, that are usually less sophisticated for candidate generation because of running time constraints. Recent approaches for tandem MS interpretation typically use scoring schemes that are elaborate modifications of the peak counting score, where one simply counts the number of peaks that are common to sample spectrum and candidate spectrum [9]. Shan et al. [10] established that generating glycan topology candidates while avoiding peak double counts is an NP-hard problem. Existing approaches for glycan candidate generation can be subdivided into three categories: Some approaches enumerate all possible glycan topologies of the given parent mass [8], [11] and have to use very strict biological rules to cut down on the number of candidates, or enumerate all topologies that fulfill such strict biological rules [12]. Other tools use dynamic programming but simply ignore the problem of multiple peak counting caused by several glycan sub-topologies of the same mass [13]. Finally, Shan et al. [10] present a heuristic that avoids peak double counting. Note that many other approaches exist for glycan identification using MS, such as glycan identification by single-stage MS [5], [14], combining single-stage MS and tandem MS data [15], assimilation by MS^n fragmentation pathways [16], programs aiding an expert in identifying the correct glycan topology [17], or database searching tools such as the commercial program SimGlycan. We omit further details.

Here, we present a method that solves the candidate generation problem while at the same time avoiding multiple peak counting. Although the corresponding problem is NP-hard, we present an exact method that allows to process a glycan tandem mass spectrum in a matter of seconds, and guarantees that all top-scoring candidate topologies are found. Our algorithm is fixed-parameter tractable [18], where for our theoretical analysis, the parameter k is the "number of peaks in the sample spectrum." In practice, parameter k can be chosen arbitrarily and allows us to tune the methods, trading specificity for running time and memory consumption. We report some preliminary results on experimental data, showing that our candidate generation performs well in practice. We show that solving the simpler candidate generation problem, where one allows multiple peak counting, usually leads to poor results. We also report some preliminary results of our candidate evaluation.

We then consider the problem of counting glycan topologies and, more general, counting rooted trees with bounded degree. These trees may be labeled, meaning that each vertex has a usually nonunique label from a finite set. We present an algorithm with running time $O(n^3)$ for counting all glycan topologies with n vertices, and then, solve the related problem of counting glycan topologies for a given mass m. Our method can be easily generalized to rooted trees of bounded out degree d, resulting in a simple algorithm with running time $O(d^2n^3)$.

2 Preliminaries

The general concept of tandem MS is to filter a single ion species in a first mass analyzer, then fragment these ions, and finally determine the masses of the resulting fragments in a second analysis unit.[1] For glycans, collision-induced dissociation (CID) is often used as fragmentation technique, where the ions collide with an inert gas for fragmentation. The collision energy determines the collision strength. The higher the energy, the more and stronger atomic bonds break. Since glycosidic bonds between sugars are weak compared to bonds inside the monosaccharides, we can choose the energy so that mainly these bonds break.

There are three types of fragmentation that break the glycan topology, resulting in six types of ions [19], (see Fig. 1): X, Y, and Z-ions correspond to fragments that contain the monosaccharide attached to the peptide and are called *reducing end ions* or *reducing end fragments*. A, B, C-ions, in contrast, do not contain this monosaccharide. A and X-ions are cross-ring fragments that result from internal monosaccharide breakages; the exact breakage positions are denoted by an additional superscript. Using low-collision

1. In fact, MS does not filter one particular ion species but instead, selects a certain mass-over-charge range of ions. This can be a problem in glycan analysis, as there may be isomers, that is, compounds with the same molecular formula but different structural formulas. We will not address this problem in our presentation but instead, assume that a single ion species has been filtered.

energies, we predominantly generate B and Y ions, so we concentrate on these two types in our presentation.

Masses of molecules are measured in "Dalton" (Da), where 1 Da is approximately the mass of a neutron. We will often assume *integer masses* in our presentation for the sake of clarity. Accurate masses will be used in the scoring scheme.

We model a *glycan topology* as a rooted tree $T = (V, E)$, where the root is the monosaccharide attached to the peptide. Tree vertices are labeled with monosaccharides from a fixed alphabet Σ, where Σ depends on the biological background of the experiment. Every vertex has an out degree of at most four, because each monosaccharide has at most five linkages. Every element $g \in \Sigma$, and hence, every vertex in the tree T is assigned an integer mass $\mu(g)$. This is the mass of the monosaccharide g, minus 18 Da for the mass of H_2O removed in binding. A fragment T' of T is a connected subtree, and the mass of T' is the sum of masses of the constituting vertices. Let $M := \mu(T)$ be the *parent mass* of the glycan structure. To simplify our presentations, we ignore mass modifications, such as adding the terminal H_2O group, reducing end modifications, or the proton mass. These modifications can be easily incorporated into the presented methods.

If we restrict ourselves to simple fragmentation events, then fragmentation of the tree means removing a single edge. Hence, we can represent each simple fragmentation event by a vertex $v \in V$, where the subtree $T(v)$ induced by v represents the nonreducing end fragment, and the remainder of the tree is the reducing end fragment. The resulting nonreducing end fragments have the mass of a subtree of T induced by a vertex v, denoted as $\mu(v)$. For reducing end fragments, we subtract $\mu(v)$ from the parent mass M.

Our method will take into account all possible glycan topologies, deliberately ignoring all biological restrictions on, say, the amount of branching in the tree. It is well known that certain branching types are observed seldom in biological samples. For example, most monosaccharides show only one to three linkages. But instead of completely forbidding such structures, we prefer to incorporate biological restrictions into our scoring model, by subtracting a penalty if a structural rule is violated. In this way, we do not impede the discovery of rare structures that may diverge significantly from structural restrictions.

Our algorithm uses the concept of *fixed-parameter tractability* (FPT) [18]. This technique delivers exact solutions for an NP-hard problem in acceptable running time if the problem can be *parameterized*. That is, in addition to the problem size n, we introduce a parameter k of the problem instance, where typically $k \ll n$. A parameterized algorithm then restricts the superpolynomial growth of its running time to the parameter k, whereas the running time is polynomial in the problem size n. Example parameters are the size of the output or structural features of the input such as the tree width of graphs. Here, the problem size is the parent mass M, whereas k is the number of (intense) peaks in the sample spectrum.

3 Candidate Generation

In this section, we address the problem of candidate generation: given the experimental data, we want to generate a small set of candidate glycan topologies, containing the correct topology. This step can be seen as a filter, where the huge number of possible topologies is reduced to a small set of, say, hundred candidates that we will study in more detail. To generate these candidates, we first have to define an objective function or score, such that candidates that agree well with the experimental data also receive a high score. We then show how to find the candidate topology with maximal score and at the end of the section, a set of suboptimal candidates. Note that we deliberately use a very simple score in our presentation, for the sake of lucidity. A scoring that takes into account realistic experimental considerations, will be presented in Section 5.

Assume, we have given a topology T and we want to evaluate T against the sample mass spectrum. Given T, we can use some simple fragmentation model to generate a hypothetical *candidate spectrum*, and use an additive scoring scheme to rate the candidate spectrum against the sample spectrum. The scoring is required to be additive, so that optimal solutions can be assembled from optimal solution of subproblems using dynamic programming. Let $f(m)$ be the score, we want to assign if a peak at mass m is present in our candidate spectrum. In its simplest incorporation, f is the characteristic function telling us if a peak is present in the *sample* mass spectrum at mass m. Then, summing $f(m)$ over all peak masses m in the candidate spectrum, we count all peaks that are common to both the sample spectrum and the candidate spectrum. To this end, $f(m)$ is often referred to as *peak counting score*. We can also take into account expected peaks that are not present in the sample spectrum, by defining $f(m) = +1$, if a peak at mass m is present in the sample spectrum, and $f(m) = -1$ otherwise. Of course, for experimental data, we use more involved scoring taking into account, say, peak intensities (see Section 5).

To simplify our presentation, let us assume for the moment that all our mass spectra consist of nonreducing end ions only. It turns out that the general case is a simple generalization of this model. Let $T = (V, E)$ be a labeled tree. Shan et al. [10] introduce the scoring model $S'(T) := \sum_{v \in V} f(\mu(v))$, which unfortunately is not a peak counting score. Instead, for *every* subtree T' of T with mass $m' = \mu(T')$, we add $f(m')$ to the score. In this way, a tree that contains many subtrees of identical mass m' receives a high score, if $f(m')$ is large, even if it ignores all other peaks. We will show below how computations for this model can be transferred over to peak counting scores though.

To find the labeled tree T that maximizes the score $S'(T)$, we define $S'[m]$ to be the maximal score of any labeled tree with total mass m. It is easy to see that S' can be computed by the recurrence

$$S'[m] = f(m) + \max_{m_1+m_2+m_3+m_4+\mu(g)=m} S'[m_1] + S'[m_2] + S'[m_3] + S'[m_4], \tag{1}$$

where the maximum is taken over all $g \in \Sigma$ and $0 \le m_1 \le m_2 \le m_3 \le m_4 \le m$ [10]. The term $S'[m]$ corresponds to a subtree with a monosaccharide at its root. We initialize $S'[0] = 0$: If one of the m_j in (1) equals zero, then the monosaccharide at the root of the subtree has less than four bonds. The maximal score of any glycan topology is $S'[M]$ and we can back trace through the array S' to find the optimal labeled tree. Shan et al. simplify (1) to

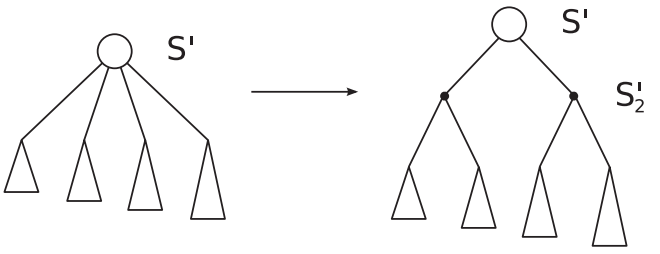

Fig. 2. In (1), we compute the score for appending up to four previously computed subtrees to one monosaccharide the root. Equation (2) reduces the complexity of the computation by appending two "headless" subtrees, which in turn consist of two subtrees each, to the root monosaccharide.

$$\begin{aligned} S'[m] &= f(m) + \max_{g\in\Sigma} \max_{m_1=0,\ldots,\left\lfloor\frac{m-\mu(g)}{2}\right\rfloor} S'_2[m_1] \\ &\quad + S'_2[m-\mu(g)-m_1], \qquad (2) \\ S'_2[m] &= \max_{m_1=0,\ldots,\left\lfloor\frac{m}{2}\right\rfloor} S'[m_1] + S'[m-m_1]. \end{aligned}$$

The term $S'_2[m]$ corresponds to a "headless" subtree without a monosaccharide at its root (see Fig. 2). Using (2), we can compute $S'[M]$ in time $O(|\Sigma|\cdot M^2)$. Equations (1) and (2) can easily be modified to take into account properties of the monosaccharide g, such as the number of links of g for the scoring.

An exact algorithm for the peak counting problem. Recall that we actually want to compute the peak counting score $S(T) := \sum_{m=0,\ldots,M} f(m)\cdot g_T(m)$, where $g_T(m)$ is the characteristic function of the labeled tree T. If T contains one or more subtrees $T(v)$ with mass $\mu(v)=m$, then $g_T(m)=1$ and $g_T(m)=0$ if no such subtree exists. Shan et al. [10] show that the following problem is NP-hard:

Glycan peak counting problem. Given a monosaccharide alphabet Σ, a sample scoring function $f(m)$ and a parent mass M, find the glycan topology T of mass $\mu(T)=M$ that maximizes $S(T)$.

We now modify recurrences (2) to find the labeled tree T that maximizes $S(T)$. To this end, note that the complexity of the problem only holds for mass spectra that contain a "large" number of peaks. But sample spectra are relatively sparse and contain only tens of peaks that have significant intensity. The number of simple fragments of a given glycan topology is only linear to its number of monosaccharides. Let k be the number of peaks in the sample spectrum. Here, k is the parameter of our problem, and we limit the running time explosion to this parameter, while maintaining a polynomial running time with respect to M. Choosing k as the number of peaks is solely done for the ease of presentation of our theoretical results. In the next sections, we show that parameter k can be arbitrarily chosen in application, trading specificity for running time and memory consumption of the method. For low k, the method produces more candidates that have a high score because of scoring peaks multiple times. In our evaluation in Section 6, we find that a moderate k such as $k=10$, leads to fast running times, in practice, but also generates only a moderate number of candidates.

In order to avoid multiple peak counting, we incorporate the set of explained peaks into the dynamic programming. This technique has been used frequently in algorithmics, see, for example, [20]. Let C^* be the set of peak masses in the sample spectrum, where $|C^*|=k$. For every mass $m\le M$ and every subset $C\subseteq C^*$, we define $S[C,m]$ to be the maximal score of any labeled tree T with total mass $\mu(T)=m$ where only the peaks from C are used to compute this score. At the end of our computations, $S[C^*,M]$ holds the maximal score of any labeled tree, where all peaks from C^* are taken into account for scoring. We now modify (2) for our purpose. We define $S_2[C,m]$ to be the score of a "headless" labeled tree with mass m using peaks in C. Using S_2, we can restrict the branching in the tree to bifurcations. We limit the recurrence of $S[C,m]$ to two "headless" subtrees with disjoint peak sets $C_1, C_2\subseteq C$, where C_1 is the subset of peaks explained by the first subtree and C_2 is the set of peaks explained by the second subtree. We require $C_1\cap C_2=\emptyset$ what guarantees that every peak is scored only once. Additionally, we demand $C_1\cup C_2 = C\setminus\{m\}$. Clearly, sets C that contain masses bigger than m need not be considered. We obtain the following recurrences:

$$\begin{aligned} S[C,m] &= \max_{g\in\Sigma} \max_{m_1=0,\ldots,\left\lfloor\frac{m-\mu(g)}{2}\right\rfloor} \max_{C_1\subseteq C\setminus\{m\}} \big\{ f(C,m) + S_2[C_1,m_1] \\ &\quad + S_2\big[C\setminus(C_1\cup\{m\}), m-\mu(g)-m_1\big]\big\}, \\ S_2[C,m] &= \max_{m_1=0,\ldots,\left\lfloor\frac{m}{2}\right\rfloor} \max_{C_1\subseteq C} S[C_1,m_1] + S\big[C\setminus C_1, m-m_1\big]. \end{aligned} \qquad (3)$$

Note that we delay the scoring of a peak at mass m, if m is not in C by extending the scoring function to $f(C,m)$. If $m\notin C$ but $m\in C^*$, then $f(C,m)=0$. Otherwise, set $f(C,m)=f(m)$. So, both peaks not in C^* and peaks in C are scored, whereas scoring of peaks in $C^*\setminus C$ is delayed.

We now analyze time and space that is needed for the computation of (3). One can easily see that the space required to store $S[C,m]$ is $O(2^k\cdot M)$. Using sparse dynamic programming, memory requirements tend to be much smaller in applications see Section 4. The time complexity for calculating the optimal solution increases by a factor of 3^k reaching $O(3^k\cdot|\Sigma|\cdot M^2)$, as there are 3^k possibilities to partition k peaks into the three sets C_1, C_2, and $C^*\setminus(C_1\cup C_2)$. The exponential running time factor can be reduced to 2^k [21], but the practical use seems to be limited due to the required overhead.

Theorem 1. *The GLYCAN PEAK COUNTING problem for monosaccharide alphabet Σ, k peaks, and parent mass M can be solved in time $O(3^k\cdot|\Sigma|\cdot M^2)$ and space $O(2^k\cdot M)$. Running time can be reduced to $O(2^k\cdot|\Sigma|\cdot M^2)$ using fast subset convolutions.*

Recall that we have limited our computations to the case, where only nonreducing end ions are present in the mass spectra. But whenever we generate a nonreducing end ion, then the corresponding reducing end ion must also be present in the mass spectrum for perfect data. We incorporate reducing end ions into the computation by "mirroring" the spectrum in a preprocessing step. We subtract all sample peak masses m from the parent mass M and insert the new mass $M-m$ into the spectrum. If a peak is already present, the intensities (or scores) can be added since this positively scores the presence of both reducing end and nonreducing

end ion. The spectrum now contains reducing end and nonreducing end peaks with same intensity for every observed peak even if only one was detected by the instrument. In consequence, we have to be careful that only one of the reducing and nonreducing end ion is scored. This is easily done by regarding the elements in C^* as colors and assigning corresponding reducing end and nonreducing end ion peaks the same color. In practice, the mass of a reducing end ion is often not decomposable, if regarded as nonreducing end ion mass because of the reducing end modification, and will not be considered for the score anyway.

To recover an optimal solution, we back trace through the dynamic programming matrix starting from entry $S[C^*, M]$. It is obvious that the maximal score will explain as many peaks as possible, but not necessarily all. More interestingly, we can also compute all solutions that deviate at most δ from the score of the optimal solution. This is done using recursive backtracking and allowing suboptimal scores in every backtracking step. We attach an error variable d to every partial solution generated. This variable d records the current deviation of the partial solution from an optimal solution and is updated in every backtracking step. If $d > \delta$, then the partial solution can be discarded. To generate a candidate set of reasonable size for subsequent analysis, we can iteratively adjust the parameter δ. Backtracking usually generates many isomorphic trees, which we remove from the final output. We can do so by encoding trees as strings, we omit the simple details. Running time of backtracking is $O(out \cdot 2^k \cdot Mn)$, where n is the maximal size of a glycan topology in the output and out is the number of generated trees including isomorphic trees, that is usually larger than the size of the final candidate set.

4 Algorithm Engineering and Heuristics

In this section, we present several modifications to the algorithm from the previous section, to make it run fast in application. We show how to make computations "sparse" both with respect to subsets $C \subseteq C^*$ and masses $m \leq M$. We also show how to swiftly process spectra that contain more than, say, ten peaks.

We usually do not add penalties for additional peaks that are not explained by our glycan, see the next section for a justification. Doing so, we can completely ignore additional peaks in our computation, and we have to compute $S[C, m]$ and $S_2[C, m]$ only for those sets $C \subseteq C^*$ that do not contain any additional peaks. This can be efficiently implemented by using hash maps to store these values and by restricting (3) to initialized entries. In this case, we have to start back tracing from the maximal entry $S[C, M]$ with $C \subseteq C^*$. Furthermore, when computing (3), sets C containing masses bigger than the current mass m need not be considered.

In case, the number of peaks in a glycan mass spectrum is too large, we can easily limit the exponential growth in memory and running time by choosing an appropriate k such as $k = 10$. Now, we use the k most intense peaks C^* in our explanation at most once, whereas we allow all other peaks to contribute multiple times to the score. As our results show, this is usually not a problem for candidate generation.

Using these engineering techniques, the prohibitive factor in the running time becomes M^2. We note that many masses m are not decomposable at all over the alphabet of monosaccharide masses. We can exclude these masses from our computation since there exists no subtree, which could explain them. A similar argumentation shows that the mass remainder $M - m$ must also be decomposable over the alphabet of monosaccharide masses, so we can also exclude many masses close to M from our computations. Doing so, we again reduce running time and memory requirements of our algorithm in practice.

5 Scoring for Candidate Generation

We noted that the scoring presented above is overly simplified, what was done to ease the presentation. We now describe some modifications that are needed to achieve good results on experimental data. Here, we take into account real-valued masses, mass deviations, and peak intensities. As noted above, our scoring has to be a simple additive scoring to allow for dynamic programming and we only score fragments that stem from simple fragmentation events.

For our scoring, we use real masses of fragments, so we first describe some modifications needed to do so. All the presented recurrences iterate over integer masses m, but monosaccharide and subtree masses are noninteger. To deal with real masses, we define $S[m]$ to be the maximal score of any labeled tree whose exact mass falls into the interval $[m - 0.5, m + 0.5)$ and additionally store the exact mass of the subtree with optimal score in this interval. We update a matrix entry $S[m]$ only if the new subtree mass (the sum of $\mu(g)$, and masses of two headless subtrees) falls in the current interval. Note that for integer mass m_1, we may have to consider the neighboring entries $\{m_1 - 1, m_1, m_1 + 1\}$ in the maximum (3), since the sum of corresponding exact masses might fall into the current interval.

To guarantee that the solution of the recurrence is optimal, we would have to assume that at most one peak can be found in every interval of size 1 Da, but this can be violated for candidate mass spectra. In this case, we store the highest scoring explanation, but we can no longer guarantee that the optimal solution of the original problem (the *true* solution) is found using the integer recurrence. In practice, this is usually not a problem: Rounding might lead to a different optimal solution but clearly, the true solution will be only slightly suboptimal. Since we are not interested in *the* optimal solution but instead, accept all solutions that are up to some δ away from optimality, chances are that the true solution will be part of the set of candidates we generate. A similar reasoning applies, if we limit exact computations to the k most intense peaks in the spectrum because only peaks of low intensity and small score will be used multiple times.

The basis of our peak score is its normalized intensity, assuming that a high intensity indicates a high probability that the peak is not noise. Mass spectrometrists assume that the mass error of a device roughly is normal distributed. To account for mass deviation $\Delta = |m_{\text{peak}} - m_{\text{fragment}}|$, we multiply the peak intensity with $\text{erfc}(\Delta/(\sigma\sqrt{2}))$, where erfc is the error function complement and σ the standard deviation of the measurement error, typically set to $\frac{1}{3}$ or $\frac{1}{2}$ of the mass accuracy.

We do not apply a penalty, if a peak in the sample spectrum is not explained by our candidate spectrum. This

may be justified by the fact that our scoring ignores fragments not resulting from a simple fragmentation event. But a closer analysis reveals that by scoring peak intensities, we, in fact, score each peak for not being noise. A candidate that does not explain an intense peak, declares this peak to be noise, although this is unlikely. The candidate then has a disadvantage compared to other candidates that can explain this peak. This scoring leads to good results as long as the intensity of peaks from simple fragmentation events is higher than that from nonsimple ones.

6 Evaluation of Glycan Topology Candidates

Once we have reduced the set of potential glycan topologies from the exponential number of initial candidates, to a manageable set of tens or hundreds of structures, we can now evaluate each candidate glycan topology using an in-depth comparison between its theoretical spectrum and the sample spectrum. This comparison can also take into account multiple-cleaved fragment trees, other ion series such as A/X and C/Z ions, or those X-ions that have lost parts of a monosaccharide. Since evaluation of candidate glycan structures is not the focus of this work, we only present a rather general scoring scheme, and we will not go into too much detail here. Still, we reach good identification results. Our scoring generalizes ideas of Goldberg et al. [8]. We stress that our evaluation approach leaves room for improvement.

The idea of our approach is to determine and score the fragmentation tree of the glycan, a representation of the consecutive fragmentation events. Therefore, we construct a fragmentation graph similar to the one used for metabolite identification in [22]. The fragmentation graph enables us to score all peaks that we can explain by fragmentation of the candidate glycan. Additionally, we can incorporate relationships between fragments. For example, a double-cleaved fragment receives a higher score if the intermediate single-cleaved fragment exists. We again avoid peak double counts by regarding peaks as colors, and not allowing to score a color twice. The *fragmentation graph* is constructed as follows: It contains a vertex for every subtree of the candidate tree whose mass deviates less than the instruments mass accuracy from a peak mass. Vertices whose subtrees correspond to the same peak are colored with the same color. Vertices are connected by a directed edge, if the tree of the descendant vertex is a subtree of the tree of the ancestor vertex. This results in a transitive colored digraph.

We now compute scores for the vertices and edges of the fragmentation graph. In contrast to candidate generation, we use the unfolded spectrum as there are separate vertices for reducing end and nonreducing end ions in the fragmentation graph. The vertex score comprises peak intensity and mass deviation. Additionally, we penalize for the number of fragmentation events necessary to produce the fragment from the fragmentation graph root. We refer to this number as the *fragmentation distance* x_r and score it by a slowly falling function we chose as $\mathrm{frag}_r(x_r) = 0.75^{x_r}$.

The edge score takes into account the fragmentation distance to the *parent vertex*. We assume that intermediate fragments should be observed in the spectrum. So, edges that represent multiple fragmentation events shall contribute less to the score than edges representing a single fragmentation event. We achieve this by using the subadditive function $\mathrm{frag}_e(x_e) = 1/x_e^2$, where x_e is the fragmentation distance of parent and child vertex. This function has been chosen ad hoc to avoid overfitting to the data. To simplify the score calculation for the algorithm, we pass on the vertex scores to all incoming edges multiplying it with the edge scores. The overall score of an edge is then $s(e) = int \cdot erfc \cdot \mathrm{frag}_r(x_r) \cdot \mathrm{frag}_e(x_e)$, where *int* is the peak intensity and *erfc* is the complementary error function of the mass deviation.

The *maximum colorful subtree* [22] of this weighted graph (the subtree that has maximal weight and uses at most one vertex per color) then is a hypothetical fragmentation tree. Unfortunately, finding this tree is NP-hard, and we refrain from using the exact FPT algorithm from [22] since it is too slow to be executed for each candidate. Therefore, we apply a greedy heuristic to determine a score. This heuristic tests all edges in descending order of score. If an edge can be added violating neither the tree nor the colorful property, it is attached to the partial tree. Due to the transitivity of the fragmentation graph, this procedure eventually results in a tree. The score of the candidate structure is the sum of the edge scores of the fragmentation tree.

We incorporate C/Z ions by creating vertices not only for every B and Y ion subtree mass that could explain a peak, but also for those masses shifted by the mass increase or decrease of C or Z ions. These vertices are also colored according to the mass of the peak they explain. Edge creation is then performed as usual. This ensures that C and Z ions are considered separately from their corresponding B and Y ions, but a peak may only be explained by either a B/Y or a C/Z ion.

7 Results on Experimental Data

To show that our method works for experimental data, we have implemented it and applied it to a set of 24 glycan mass spectra.

We implemented our algorithm in Java 1.5. Running times were measured on an Intel Core 2 Duo, 2.5 GHz with 3 GB memory. A set of carbohydrate side chains of the serine protease batroxobin from the venom of the viper Bothrops moojeni served us to test the program [23]. We used 24 spectra of N-glycans from recent investigations, where a protonated precursor ion was selected. The spectra were measured using a Bruker Daltonics ultraFlex TOF/TOF instrument with a MALDI ion source. Glycans are composed of fucoses (F, mass 146.06 Da), hexoses (H, 162.05 Da), and N-acetylhexosamines (N, 203.08 Da). The 24 glycans were not permethylated and do not contain sialic acids. Glycans were detached from the protein and the reducing end was marked by a two-aminopyridine modification resulting in a mass increase of 78 Da for reducing end ions. The raw data was baseline corrected and peaks were picked using the SNAP method provided by Bruker. We follow the naming convention from [23] for reporting the analyzed glycans. We use the fraction the glycan was extracted from (e.g., F2-5) followed by the composition of monosaccharides.

We used the following parameters for analyzing the spectra. We set $k = 10$, avoiding multiple peak counting for the ten most intense peaks. We allowed a mass deviation of 1.0 Da and chose $\sigma = 0.5$ Da as standard deviation of the measurement error. After normalizing the sum of peak

TABLE 1
Results of the Algorithm for 24 N-Glycans of Batroxobin [23]

glycan	parent mass	# peaks	# B/Y ion peaks	δ	running time	# cand.	rank eval.	biological cand.	biological eval.
H3N5F	1744.0	36	10	15%	0.74 s	126	3	6	1
H3N6F	1947.0	41	8	10%	1.20 s	184	14	5	3
H3N6	1801.0	28	8	7%	0.93 s	122	1	5	1
H4N4F	1703.0	66	15	22%	4.69s	115	1	8	1
H5N4F	1865.0	34	11	15%	5.60 s	154	1	16	1
F1-H3N5	1598.6	26	7	25%	0.61 s	121	20	5	2
F2-1-H3N4F	1541.0	16	6	10%	0.43 s	133	3	6	2
F2-2-H4N4	1557.6	21	2	20%	1.75 s	128	1	18	1
F2-4-H4N4F	1703.6	33	12	20%	1.84 s	150	1	5	1
F2-5-H5N4F	1865.7	29	11	11%	0.92 s	282	1	41	1
F2-5-H5N4	1719.6	23	7	10%	1.09 s	147	3	43	2
F3-H3N8F	2354.0	18	6	12%	5.21 s	383	83	2	2
F4-1-H3N5	1598.0	35	12	20%	1.37 s	154	1	7	1
F4-3-H3N4F2	1687.6	20	6	17%	2.65 s	145	3	13	1
F4-3-H3N5F	1744.0	41	13	20%	3.03 s	113	1	7	1
F4-3-H4N2F2	1849.0	24	8	4.5%	5.40 s	164	62	6	4
F4-4-H3N6F2	2092.0	11	8	9%	4.04 s	450*	66	4	2
F5-1-H3N6F	1947.0	92	11	6%	1.47 s	171	47	5	2
F5-1-H5N4F	1865.0	37	10	15%	3.14 s	169	12	22	3
F6-1-H2N4F	1379.0	38	11	60%	0.48 s	103*	1	2	1
F6-2-H5N4F	1865.0	34	11	15%	6.48 s	161	1	16	1
F7-2-H4N4F	1703.0	66	15	23%	4.66 s	146	1	10	1
F7-3-H3N6	1801.0	28	8	7%	0.91 s	122	1	5	1
F7-3-H4N5F	1907.0	29	9	9%	0.87 s	143	3	12	2

Running times including trace back. * *See text for details.*

intensities, we discarded all peaks with an intensity lower than 0.02. We chose the penalty for missing peaks as the average of the smallest intensity and the mean value of all peak intensities. We iteratively adjusted the score deviation δ for backtracking to obtain a candidate set of about 100-200 topologies. Results are shown in Table 1.

Average running time for candidate generation was 2.5 s without and 4.0 s including trace back. There were many spectra with less than 10 decomposable peaks what reduced running time. For the "loaded" spectra with $k = 10$, average running time was 3.6 s without and 5.5 s with trace back. In all except two cases, the candidate set contained the manually determined topology, see below. For 17 spectra the true topology was found in the TOP 50 of candidates and for 12 spectra even in the TOP 25 of candidates.

For F4-4-H3N6F2, we were not able to generate the correct topology with the described parameter set because there were no single-cleaved fragments of the trimannosyl core in the spectrum. Since this results in too many missing peaks, the algorithm can only reconstruct the correct topology, if we lower the penalty for missing peaks to a value slightly larger than the weakest peak's intensity. Because of the small number of peaks in the spectrum, we assume that the threshold for peak picking was set too low. In the case of F6-1-H2N4F, all expected peaks were present in the spectrum. Here, intensities of double-cleaved fragments resulting from cleaving fucose and another subtree were as high or even higher than some single fragmentation peaks. Subtracting the mass of fucose from the parent mass identified the unfucosylated structure correctly. We plan to incorporate an automated identification of such cases into our approach.

We also tested if avoiding peak double counting is needed for candidate generation. To this end, we set $k = 0$, so every peak could be counted an arbitrary number of times. Doing so, candidate generation produced the correct topology only for eight of the 24 spectra even if the candidate set was chosen to contain at least 500 structures. This shows that avoiding multiple peak counting is essential for the analysis. Certain glycan topologies do, in fact, create the same fragment mass several times. It must be understood that our approach does not *penalize* such topologies, but it also does not *reward* them. As the extreme case, consider a single leaf of the tree. If the corresponding peak has high intensity, then we reward trees for having identical labels at all leaves, which is surely not desirable. Finally, we tested if further increasing k could improve the results of candidate generation. But as it turned out, increasing k to the 15 most intense peaks did not improve the results. So, computations can be carried out with a moderate k such as $k = 10$ for glycans of this size without loosing specificity.

As already mentioned, evaluation of candidate glycan structures is not the focus of this work, so we just report some identification rates we were able to obtain after evaluation. The evaluation step ranked the true topology in all except four cases in the TOP 20, for 12 structures even on the first rank. Two topologies that were correctly identified are

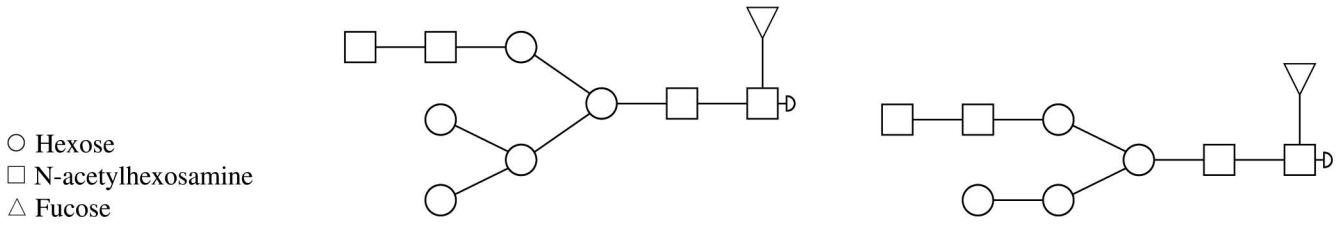

Fig. 3. The manually determined topologies of (left) H5N4F and (right) H4N4F. In both cases, the correct topology was identified by our algorithm without using biological prior knowledge.

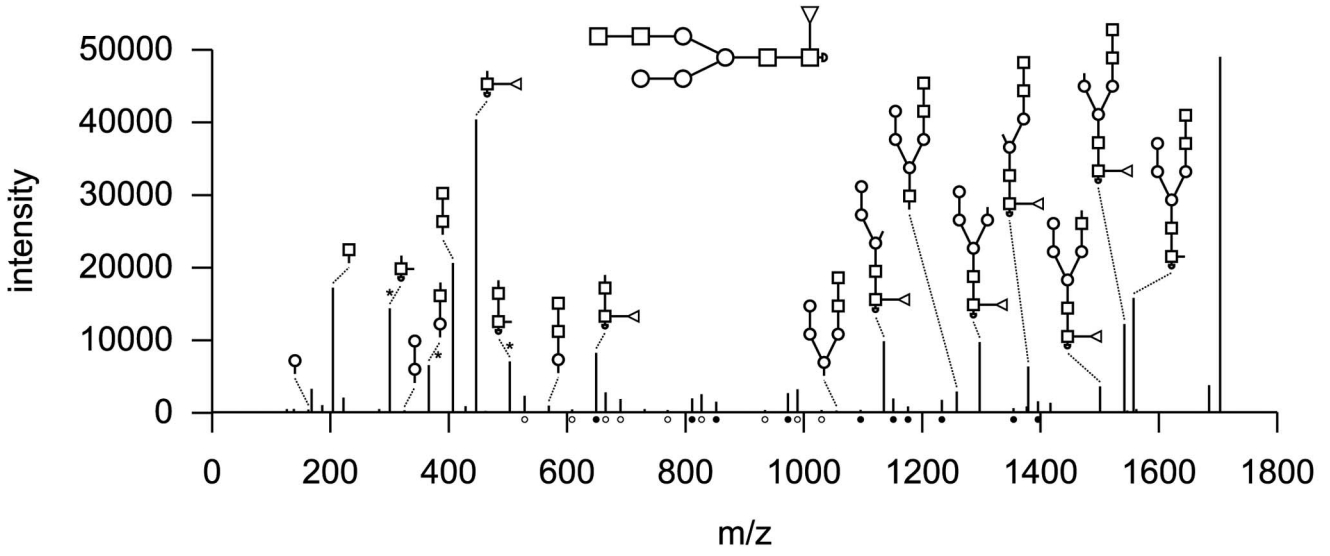

Fig. 4. The peak-picked spectrum of F7-2-H4N4F annotated with single-cleaved fragments of the correct glycan topology. Only the peak at $m/z \approx 147$, corresponding to a single protonated fucose is missing. Three intense peaks marked with $(*)$ correspond to double-cleaved fragments. Additionally, all peaks marked with filled circles can be annotated with double-cleaved fragments and peaks marked with unfilled circles can be annotated with fragments that stem from more than two cleavages.

displayed in Fig. 3, and a manually annotated spectrum for one of them is shown in Fig. 4. In many cases, top-scoring topologies are biologically impossible, that is, they do not contain the trimannosyl core that N-glycans generally share. Using this biological knowledge on the structure of the analyzed glycans, we significantly reduce the number of candidates and also improve our identification results. We always find the true solution in the TOP 4, and 14 true topologies reach rank one, see the right columns of Table 1.

In one case, our software suggested a candidate that had not been considered by the experimentalists for annotation. The experimentalists annotated the spectrum with a slightly different topology, which was second place in our analysis. We display these topologies in Fig. 5. Unfortunately, the spectrum does not contain sufficient information to distinguish between these candidates. To distinguish between the two candidates, we can record MS^3 data, where a fragment of the glycan is again fragmented. Unfortunately, such data was not available in our study.

8 Counting Glycan Topologies

We have mentioned above that the number of glycan topologies easily becomes prohibitive for enumerating all possible topologies. We now substantiate this claim, by computing the number of glycan topologies for a certain number of monosaccharides and for a certain mass. We first recapitulate some classical results for counting rooted trees of bounded degree, then present an algorithm for the exact and swift counting of glycan topologies.

Let $N[n, |\Sigma|]$ be the number of different glycan topologies with n vertices, where vertices are labeled with elements from Σ. Recall that a glycan topology corresponds to a rooted tree such that every vertex has out degree at most four. For $|\Sigma| = 1$, Otter [24] has shown that the number of rooted trees with out-degree at most four, asymptotically behaves like[2]

$$N[n, 1] \sim \tilde{t}(n) := 0.462103 \cdot 2.911038^n \cdot n^{-3/2}. \qquad (4)$$

This approximation is very accurate even for small n: For $n = 10$, we calculate $\tilde{t}(10) = 638.6$ whereas the true number is 643, so the relative error is well below one percent. We can estimate the number of glycan topologies over an arbitrary alphabet Σ by $|\Sigma|^n \cdot \tilde{t}(n)$ since every vertex can be colored with an individual color. This overestimates the number of trees, as we do not take into account isomorphic trees. For $|\Sigma| = 5$ and $n = 10$ we get $5^{10} \cdot \tilde{t}(10) = 6.24 \cdot 10^9$ whereas the true number is $3.10 \cdot 10^9$, so the true number is only half of what our rough estimate tells us. The relative error will become even larger as n increases.

The "classical way" of counting trees, is to use Pólya's enumeration theorem [25]. We want to count the number of rooted quaternary trees, where every vertex has exactly four children. Note that there is a bijection between rooted

2. Computing the two constants is somewhat involved, we omit the mathematical details.

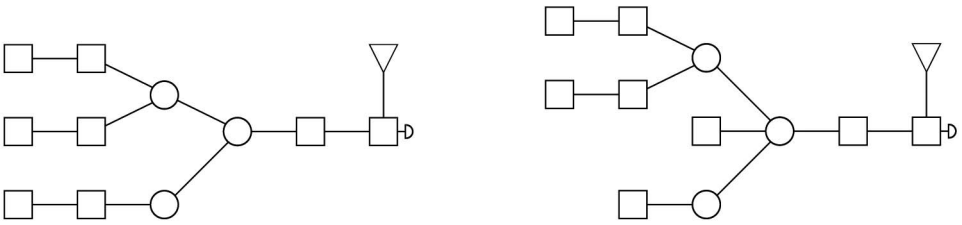

Fig. 5. Candidate topologies for F3-H3N8F: (left) The topology determined by the experimentalists and (right) the highest scoring topology of our algorithm.

quaternary trees with n nonleaf vertices and rooted trees with maximum out degree four and n vertices. To count this number, we need the cycle index of the symmetric group S_4 with four elements, which is

$$Z(S_4) = \frac{1}{24}\left(a_1^4 + 6a_1^2a_2 + 3a_2^2 + 8a_1a_3 + 6a_4\right).$$

Now, the generating function $T(z)$ of the set of rooted quaternary trees can be expressed via

$$T(z) = 1 + \frac{1}{24}z(T(z)^4 + 6T(z)^2T(z^2) + 3T(z^2)^2 + 8T(z)T(z^3) + 6T(z^4)).$$

For the number of trees $t(n) := N[n,1]$, this leads to the recurrence $t(0) = 1$ and

$$t(n) = \frac{1}{24}\left(\sum_{i+j+k+l=n-1} t(i)t(j)t(k)t(l) + 6\sum_{i+j+2k=n-1} t(i)t(j)t(k) + 3\sum_{2i+2j=n-1} t(i)t(j) + 8\sum_{i+3j=n-1} t(i)t(j) + 6\sum_{4i=n-1} t(i)\right). \quad (5)$$

One evaluation of (5) takes $O(n^3)$ time, so the recurrence needs $O(n^4)$ time to compute $t(n)$. We will show in the next section how to improve the running time to $O(n^3)$. Unfortunately, there seems to be no possibility to generalize these results to trees where nodes are (nonuniquely) labeled with elements from a finite set.

In the remainder of this section, we present a method for the exact computation of $N[n] := N[n, |\Sigma|]$ for a fixed alphabet Σ of arbitrary size. To reach an efficient recurrence, we distinguish four cases, corresponding to the out degree of the root vertex. To the root vertex, we attach a forest of one to four trees that in total have one vertex less than the new glycan topology. In our recurrence, we will also have to upper bound the number of vertices in each individual tree. To this end, let $N_i[n,k]$ be the number of forests consisting of i nonempty glycan trees, such that the total number of vertices in the forest is n and no tree in the forest has more than k vertices. Clearly, we can label the root with any element from Σ. Hence, the number of glycan topologies labeled with the alphabet Σ is

$$N[n+1] = |\Sigma| \cdot \left(N[n] + N_2[n,n] + N_3[n,n] + N_4[n,n]\right). \quad (6)$$

We initialize $N[0] = 1$ for the empty tree.

Obviously, the main difficulty is to compute the number of forests $N_i[n,k]$. Assume that $l \le k$ is the maximal size of any tree in such a forest, and let $j \le i$ be the number of trees of size l in this forest. We can choose $\binom{N[l]+j-1}{j}$ different trees of size l. Now, $i-j$ trees remain in the forest and these can have at most $l-1$ vertices each, and must have $n-jl$ vertices in total. From the definition above, we know that there exist $N_{i-j}[n-jl, l-1]$ such forests. Thus, we reach the recurrence

$$N_i[n,k] = \sum_{j=1}^{i}\sum_{l=\lceil n/i\rceil}^{\min\{k,\lfloor n/j\rfloor\}} \binom{N[l]+j-1}{j} \cdot N_{i-j}[n-jl, l-1], \quad (7)$$

for the number of different forests with i glycan trees, maximal tree size k, and n vertices in total. We have to initialize $N_i[n,k]$ depending on the number of trees i. For $i = 0$, we set $N_0[0,k] := 1$, and $N_0[n,k] := 0$ for all $n \ge 1$. For $i = 1$, we set $N_1[0,k] = 0$ for all $k \ge 0$, $N_1[n,k] := N[n]$ for $n \le k$, and $N_1[n,k] := 0$ otherwise. For $i \ge 2$, we set $N_i[n,k] := 0$ in case $i > n$ or $k \cdot i < n$ or $k = 0$ holds. All other values can be computed from recurrences (6) and (7). So, we have reached:

Lemma 1. *Using recurrences (6) and (7), the number of glycan topologies with n vertices over an alphabet Σ can be computed in time $O(n^3)$ and space $O(n^2)$.*

Table 2 displays the number of different glycan topologies for an alphabet size of one to five.

Finally, we show how to count the number of glycan topologies of a given mass M. One may be tempted to compute all possible monosaccharide compositions of mass M, then use the multinomial coefficient of the composition times $N[n]$ to compute the number of labeled glycan trees for each monosaccharide composition. Unfortunately, this again overestimates the true number of trees, as we have to take into account that siblings may induce isomorphic trees, in which case the true number of labellings is much smaller. In addition, this approach becomes prohibitive for large masses and large alphabets, as the number of decompositions explodes.

Here, we combine (1) with the above recurrences (6), (7), to reach a recurrence that is practically independent of $|\Sigma|$. We again assume integer masses $M = 0, 1, 2, \ldots$. Let Σ be the alphabet of monosaccharides. Let $\mathcal{N}[M]$ be the number of glycan topologies over Σ with mass M. Finally, let $\mathcal{N}_i[M,m]$ be the number of forests consisting of i nonempty glycan trees, such that the total mass of the trees in the forest is M, and no tree in the forest weights more than m. Similar to (6), we have

TABLE 2
Number of Glycan Topologies for an Alphabet Size of One to Five, Plus Rounded Approximation (4)

number of vertices	approx. using (4)	number of glycan topologies $\|\Sigma\|=1$	$\|\Sigma\|=2$	$\|\Sigma\|=3$	$\|\Sigma\|=4$	$\|\Sigma\|=5$
5	9	9	214	1485	5996	17850
10	639	643	416388	$2.09 \cdot 10^7$	$3.46 \cdot 10^8$	$3.10 \cdot 10^9$
15	72664	72917	$1.25 \cdot 10^9$	$4.51 \cdot 10^{11}$	$3.07 \cdot 10^{13}$	$8.24 \cdot 10^{14}$
20	$9.87 \cdot 10^6$	$9.88 \cdot 10^6$	$4.51 \cdot 10^{12}$	$1.16 \cdot 10^{16}$	$3.23 \cdot 10^{18}$	$2.61 \cdot 10^{20}$
25	$1.48 \cdot 10^9$	$1.48 \cdot 10^9$	$1.78 \cdot 10^{16}$	$3.29 \cdot 10^{20}$	$3.75 \cdot 10^{23}$	$9.08 \cdot 10^{25}$
30	$2.35 \cdot 10^{11}$	$2.35 \cdot 10^{11}$	$7.50 \cdot 10^{19}$	$9.89 \cdot 10^{24}$	$4.62 \cdot 10^{28}$	$3.36 \cdot 10^{31}$

$$\mathcal{N}[M] = \sum_{g\in\Sigma}(\mathcal{N}[M-\mu(g)] + \mathcal{N}_2[M-\mu(g), M] + \mathcal{N}_3[M-\mu(g), M] + \mathcal{N}_4[M-\mu(g), M]). \tag{8}$$

We initialize $\mathcal{N}[0] := 1$. For brevity, we assume $\mathcal{N}[M] = 0$ and $\mathcal{N}_i[M, m] = 0$ for $M < 0$.

Again, the main difficulty is to compute the number of forests $\mathcal{N}_i[M, m]$, what can be achieved by a variation of (7)

$$\mathcal{N}_i[M,m] = \sum_{j=1}^{i} \sum_{m'=\lceil M/i\rceil}^{\min\{m,\lfloor M/j\rfloor\}} \binom{\mathcal{N}[m']+j-1}{j} \cdot \mathcal{N}_{i-j}\big[M-jm', m'-1\big]. \tag{9}$$

Similar to above, we initialize $\mathcal{N}_i[M, m]$: For $i = 0$, we set $\mathcal{N}_0[0, m] := 1$, and $\mathcal{N}_0[M, m] := 0$ for all $M \geq 1$. For $i = 1$, we set $\mathcal{N}_1[M, m] := \mathcal{N}[M]$ for $M \leq m$, and $\mathcal{N}_1[M, m] := 0$ otherwise. For $i \geq 2$, we set $\mathcal{N}_i[M, m] := 0$ in case $m = 0$ or $i > M$ or $m \cdot i < M$. We reach:

Lemma 2. *Using recurrences (8) and (9), the number of glycan topologies with integer mass M over an alphabet Σ can be computed in time $O(|\Sigma|M + M^3)$ and space $O(M^2)$.*

9 Counting Rooted Trees with Bounded Degree

We now generalize our findings from the previous section to arbitrary rooted trees with out degree bounded by d. Note that these results do not apply for glycans, but for other structures that can be viewed as rooted trees of bounded degree.

First, let us use Pólya's enumeration theorem for counting trees. For out degree d, we have to compute the cycle index of the symmetric group S_d, which is

$$Z(S_d) = \frac{1}{d!}\sum_{(j)} \frac{d!}{\prod_{k=1}^{d} k^{j_k} j_k!} a_1^{j_1} a_2^{j_2} \cdots a_d^{j_d}. \tag{10}$$

The sum runs over all $(j_1, \ldots, j_d)$ satisfying $1 \cdot j_1 + 2 \cdot j_2 + \cdots + d \cdot j_d = d$. Hence, there exist $p(d)$ summands in (10), where $p(n)$ is the *partition number* of d, that is, the number of ways of writing d as a sum of positive integers.

For each partition $x_1 + \cdots + x_l = d$, we get a summand in the cycle index, and hence, a sum in the recurrence for the number of trees $t_d(n)$, compare to (5). These sums are of the form

$$\sum_{x_1 i_1 + \ldots + x_l i_l = n-1} t_d(i_1) \cdot t_d(i_2) \cdots t_d(i_l), \tag{11}$$

so we can compute this value in time $O(n^l)$ and in view of $l \leq d$, in $O(n^d)$. Doing so for all partitions, we reach a total running of $O(p(d)n^{d+1})$ for computing all values $t_d(0), \ldots, t_d(n)$. But we can do faster than that: We can rewrite (11) as

$$\sum_{i_1=0}^{\lfloor (n-1)/x_1\rfloor} t_d(i_1) \cdot \sum_{i_2=0}^{\lfloor (n-1-i_1x_1)/x_2\rfloor} t_d(i_2) \cdots \sum_{i_l=0}^{\lfloor (n-1-i_1x_1-\ldots-i_{l-1}x_{l-1})/x_l\rfloor} t_d(i_l).$$

Using convolution, we can compute these values in time $O(ln^2)$, and hence, $O(dn^2)$. With this trick, we reach a total running time of $O(d\,p(d)n^3)$.

Unfortunately, growth of $p(d)$ is superpolynomial in d: More precisely, we have

$$p(d) \sim \frac{1}{4d\sqrt{3}} \exp\Big(\pi\sqrt{2d/3}\Big) \approx 0.144337 \cdot \frac{1}{d} \cdot 13.001954^{\sqrt{d}},$$

for the asymptotic growth of $p(d)$ [26].

Now, let us take a look at our alternative method for counting trees, using (6) and (7). It is quite obvious how to generalize our findings, to count trees with maximal out degree d other than $d = 4$. Doing so, we reach a polynomial running time for counting trees of bounded out degree:

Lemma 3. *We can count the number of rooted trees with n vertices and maximal out degree d in time $O(d^2n^3)$ and space $O(dn^2)$. The same holds for labeled rooted trees, where vertices are (nonuniquely) labeled with elements from a finite set Σ.*

As a side note, we want to point out that recurrence (6), (7) is much simpler than a computation based on Pólya's enumeration theorem.

10 Conclusion

We have presented an approach for the automated analysis of glycan tandem mass spectra. We focused on the problem of candidate generation needed to reduce the search space of glycan structures. Despite the computational complexity of the candidate generation problem, our approach avoids peak double counting and solves the problem exactly using fixed-parameter techniques. We also

present a preliminary scoring scheme for candidate generation. Evaluation using experimental data shows that our method achieves swift running times and very good identification results. We plan to integrate our algorithms into a framework for MS or glycan analysis, such as SIRIUS [27] or GlycoWorkbench [17].

We have also presented an efficient recurrence for counting glycan trees and more generally, (labeled) rooted trees of bounded out degree, improving on results obtained using Pólya's enumeration theorem.

ACKNOWLEDGMENTS

The authors would like to thank Kai Maaß from the Biochemistry Department of the Justus-Liebig-Universität Gießen for providing the glycan mass spectra.

REFERENCES

[1] R. Apweiler, H. Hermjakob, and N. Sharon, "On the Frequency of Protein Glycosylation, as Deduced from Analysis of the SWISS-PROT Database," *Biochimica et Biophysica Acta,* vol. 1473, no. 1, pp. 4-8, 1999.

[2] A. Dell and H.R. Morris, "Glycoprotein Structure Determination by Mass Spectrometry," *Science,* vol. 291, no. 5512, pp. 2351-2356, 2001.

[3] J. Zaia, "Mass Spectrometry of Oligosaccharides," *Mass Spectrom. Rev.,* vol. 23, no. 3, pp. 161-227, 2004.

[4] K.K. Lohmann and C.-W. von der Lieth, "GlycoFragment and GlycoSearchMS: Web Tools to Support the Interpretation of Mass Spectra of Complex Carbohydrates," *Nucleic Acids Research,* vol. 32, pp. W261-W266, 2004.

[5] C.A. Cooper, E. Gasteiger, and N.H. Packer, "GlycoMod - A Software Tool for Determining Glycosylation Compositions from Mass Spectrometric Data," *Proteomics,* vol. 1, no. 2, pp. 340-349, 2001.

[6] J.A. Taylor and R.S. Johnson, "Sequence Database Searches via De Novo Peptide Sequencing by Tandem Mass Spectrometry," *Rapid Comm. Mass Spectrometry,* vol. 11, pp. 1067-1075, 1997.

[7] S. Böcker and Z. Lipták, "A Fast and Simple Algorithm for the Money Changing Problem," *Algorithmica,* vol. 48, no. 4, pp. 413-432, 2007.

[8] D. Goldberg, M. Bern, B. Li, and C.B. Lebrilla, "Automatic Determination of O-Glycan Structure from Fragmentation Spectra," *J. Proteome Research,* vol. 5, no. 6, pp. 1429-1434, 2006.

[9] T. Chen, M.-Y. Kao, M. Tepel, J. Rush, and G.M. Church, "A Dynamic Programming Approach to De Novo Peptide Sequencing via Tandem Mass Spectrometry," *J. Computational Biology,* vol. 8, no. 3, pp. 325-337, 2001.

[10] B. Shan, B. Ma, K. Zhang, and G. Lajoie, "Complexities and Algorithms for Glycan Sequencing Using Tandem Mass Spectrometry," *J. Bioinformatics and Computational Biology,* vol. 6, no. 1, pp. 77-91, 2008.

[11] S.P. Gaucher, J. Morrow, and J.A. Leary, "STAT: A Saccharide Topology Analysis Tool Used in Combination with Tandem Mass Spectrometry," *Analytical Chemistry,* vol. 72, no. 11, pp. 2331-2336, 2000.

[12] M. Ethier, J.A. Saba, M. Spearman, O. Krokhin, M. Butler, W. Ens, K.G. Standing, and H. Perreault, "Application of the StrOligo Algorithm for the Automated Structure Assignment of Complex N-Linked Glycans from Glycoproteins Using Tandem Mass Spectrometry," *Rapid Comm. Mass Spectrometry,* vol. 17, no. 24, pp. 2713-2720, 2003.

[13] H. Tang, Y. Mechref, and M.V. Novotny, "Automated Interpretation of MS/MS Spectra of Oligosaccharides," *Bioinformatics,* vol. 21, Suppl 1, pp. i431-i439, 2005.

[14] D. Goldberg, M. Sutton-Smith, J. Paulson, and A. Dell, "Automatic Annotation of Matrix-Assisted Laser Desorption/Ionization N-Glycan Spectra," *Proteomics,* vol. 5, no. 4, pp. 865-875, Mar. 2005.

[15] D. Goldberg, M. Bern, S. Parry, M. Sutton-Smith, M. Panico, H.R. Morris, and A. Dell, "Automated N-Glycopeptide Identification Using a Combination of Single- and Tandem-MS," *J. Proteome Research,* vol. 6, no. 10, pp. 3995-4005, 2007.

[16] A.J. Lapadula, P.J. Hatcher, A.J. Hanneman, D.J. Ashline, H. Zhang, and V.N. Reinhold, "Congruent Strategies for Carbohydrate Sequencing. 3. OSCAR: An Algorithm for Assigning Oligosaccharide Topology from MS^n Data," *Analytical Chemistry,* vol. 77, no. 19, pp. 6271-6279, 2005.

[17] A. Ceroni, K. Maass, H. Geyer, R. Geyer, A. Dell, and S.M. Haslam, "GlycoWorkbench: A Tool for the Computer-Assisted Annotation of Mass Spectra of Glycans," *J. Proteome Research,* vol. 7, no. 4, pp. 1650-1659, 2008.

[18] R. Niedermeier, *Invitation to Fixed-Parameter Algorithms.* Oxford Univ. Press, 2006.

[19] B. Domon and C.E. Costello, "A Systematic Nomenclature for Carbohydrate Fragmentations in FAB-MS/MS Spectra of Glycoconjugates," *Glycoconjugate J.,* vol. 5, pp. 397-409, 1988.

[20] S.E. Dreyfus and R.A. Wagner, "The Steiner Problem in Graphs," *Networks,* vol. 1, no. 3, pp. 195-207, 1972.

[21] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto, "Fourier Meets Möbius: Fast Subset Convolution," *Proc. ACM Symp. Theory of Computing (STOC '07),* pp. 67-74, 2007.

[22] S. Böcker and F. Rasche, "Towards De Novo Identification of Metabolites by Analyzing Tandem Mass Spectra," *Bioinformatics,* vol. 24, pp. i49-i55, 2008.

[23] G. Lochnit and R. Geyer, "Carbohydrate Structure Analysis of Batroxobin, a Thrombin-Like Serine Protease from Bothrops Moojeni Venom," *European J. Biochemistry,* vol. 228, no. 3, pp. 805-816, 1995.

[24] R. Otter, "The Number of Trees," *The Annals of Math.,* vol. 49, no. 3, pp. 583-599, 1948.

[25] G. Pólya, "Kombinatorische Anzahlbestimmungen Für Gruppen, Graphen und Chemische Verbindungen," *Acta Mathematica,* vol. 68, no. 1, pp. 145-254, 1937.

[26] G.H. Hardy and S. Ramanujan, "Asymptotic Formulae in Combinatory Analysis," *Proc. London Math. Soc.,* vol. s2-17, no. 1, pp. 75-115, 1918.

[27] S. Böcker, M. Letzel, Z. Lipták, and A. Pervukhin, "SIRIUS: Decomposing Isotope Patterns for Metabolite Identification," *Bioinformatics,* vol. 25, no. 2, pp. 218-224, 2009.

Sebastian Böcker received the PhD degree in mathematics at Bielefeld University, in 1999 on the topic from computational phylogeny. He joined SEQUENOM and worked in the industry for three years, both in the Hamburg subsidiary and the San Diego headquarters. In 2003, he became the head of the DFG-Funded Research Group "Computational MS" at Bielefeld University. Since 2006, he has been the Chair of Bioinformatics at Friedrich Schiller University in Jena. His research interests include computational phylogenetics, parameterized algorithms for computationally hard problems in bioinformatics, finding approximate gene clusters, and computational MS.

Birte Kehr received the Diploma degree in bioinformatics from the Friedrich Schiller University in Jena, Germany, in 2009. She is currently working in the Algorithmic Bioinformatics Group at the Freie Universität Berlin, Germany, and is a scholar of the International Max Planck Research School for Computational Biology and Scientific Computing. Her current research interests include the development of efficient algorithms for genome comparison.

Florian Rasche received the Diploma degree in bioinformatics from the Friedrich Schiller University in Jena, Germany, in 2008. He is working toward the PhD degree in the Bioinformatics Department at Jena University. His research interests include computational MS, in particular involving small molecules.