Antilope—A Lagrangian Relaxation Approach to the *de novo* Peptide Sequencing Problem

Sandro Andreotti, Gunnar W. Klau, and Knut Reinert

Abstract—Peptide sequencing from mass spectrometry data is a key step in proteome research. Especially *de novo* sequencing, the identification of a peptide from its spectrum alone, is still a challenge even for state-of-the-art algorithmic approaches. In this paper, we present ANTILOPE, a new fast and flexible approach based on mathematical programming. It builds on the spectrum graph model and works with a variety of scoring schemes. ANTILOPE combines Lagrangian relaxation for solving an integer linear programming formulation with an adaptation of Yen's k shortest paths algorithm. It shows a significant improvement in running time compared to mixed integer optimization and performs at the same speed like other state-of-the-art tools. We also implemented a generic probabilistic scoring scheme that can be trained automatically for a data set of annotated spectra and is independent of the mass spectrometer type. Evaluations on benchmark data show that ANTILOPE is competitive to the popular state-of-the-art programs PepNovo and NovoHMM both in terms of runtime and accuracy. Furthermore, it offers increased flexibility in the number of considered ion types. ANTILOPE will be freely available as part of the open source proteomics library OpenMS.

Index Terms—Computational proteomics, *de novo* peptide sequencing, Lagrangian relaxation, discrete optimization.

1 Introduction

Mass spectrometry-based high throughput identification of peptides and proteins is a key step in most proteomics research experiments. It requires fast algorithmic solutions with good identification capabilities. Depending on the initial situation of the experiment, two general strategies exist: database-assisted and *de novo* identification. If a database for the studied proteins exists the first method is usually preferred over *de novo* sequencing. The crucial step in database search algorithms like INSPECT [1], SEQUEST [2], Mascot [3], and OMSSA [4] is to filter the database based on different methods. INSPECT generates peptide sequence tags (PST) and keeps only those candidate peptides containing the tag as a subsequence. SEQUEST uses the parent mass as filter criterion. After filtering, the query spectrum is scored against the remaining candidates and a ranking of possible identifications is produced. In addition to the quality of the spectrum, database search methods clearly depend on the correctness and completeness of the database, and hence on the availability of a suitable set of peptides or transcripts for the studied organism. Even if this is the case, factors like alternative splice variants and mutations can lead to missing identifications.

- *S. Andreotti and K. Reinert are with the Algorithmische Bioinformatik, Institut für Informatik, Freie Universität Berlin, Takustr. 9, Berlin 14195, Germany, and with the International Max Planck Research School for Computational Biology and Scientific Computing (IMPRS-CBSC). E-mail: andreott@inf.fu-berlin.de, knut.reinert@fu-berlin.de.*
- *G.W. Klau is with the Life Sciences group, Centrum Wiskunde & Informatica (CWI), PO Box 94079, 1090 GB Amsterdam, Netherlands, and with the Netherlands Institute for Systems Biology (NISB). E-mail: gunnar.klau@cwi.nl.*

Manuscript received 20 May 2010; revised 15 Nov. 2010; accepted 27 Feb. 2011; published online 22 Mar. 2011.
For information on obtaining reprints of this article, please send E-mail to: tcbb@computer.org, and reference. IEEECS Log Number TCBB-2010-05-0125. Digital Object Identifier no. 10.1109/TCBB.2011.59.

In such situations, *de novo* sequencing algorithms provide an alternative as they infer the sequence from the spectrum itself without any information collected in databases. In recent years, many algorithms and software packages were published, with the most popular being PEAKS [5], PepNovo [6], NovoHMM [7], Lutefisk [8], Sherenga [9], EigenMS [10], and PILOT [11]. Most of them use the graph-theoretical approach introduced by Bartels [12] and construct a so-called N-C spectrum graph which is then used to search for the correct sequence. See Fig. 1.

Using this formulation, the *de novo* peptide sequencing problem can be formulated as the search for the longest antisymmetric path, an NP-complete problem [13]. PepNovo solves a special case of this problem by restricting the construction of the spectrum graph, which enables it to apply a dynamic programming algorithm proposed by Chen et al. [14], [15]. The restrictions limit the possible interpretations of each peak to at most one N-terminal (usually b-ion) and one C-terminal (usually y-ion) ion type. Liu et al. [16] use tree decomposition to solve the restricted problem. Bafna and Edwards [17] propose a variant of the dynamic programming approach that also allows for more interpretations leading to a polynomial algorithm of a higher degree. Their algorithm is still limited to so-called *simple* ion types, excluding doubly and triply charged ions that can also aid the identification process. PILOT [11] uses an integer linear programming (ILP) formulation for the longest antisymmetric path problem that is flexible and can be extended to overcome these restrictions on the cost of efficiency. This allows for more interpretations of each peak which can lead to improved identification in situations, where the prominent b- and y-ions are missing. Furthermore, the ILP formulation can be easily extended in several ways by simply adding or modifying constraints to further restrict or modify the set of possible solutions. The approach also allows for global reasoning such as limiting the number of a certain amino acid type for each prediction, or, as

1545-5963/12/$31.00 © 2012 IEEE Published by the IEEE CS, CI, and EMB Societies & the ACM

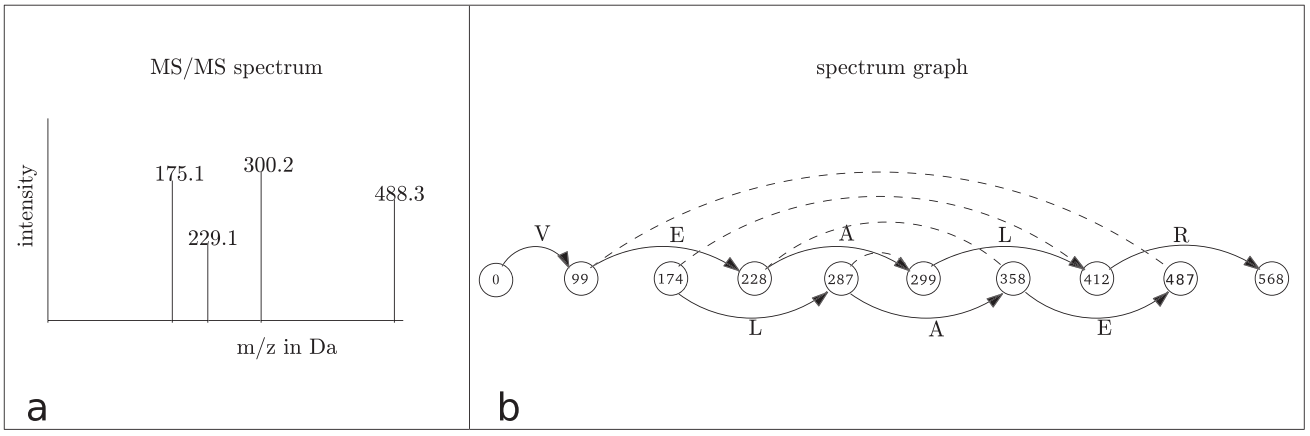


Fig. 1. Spectrum graph generation. (a) Simplified tandem mass spectrum of the peptide VEALR. Rounded *m/z* values in Da are presented on top of each peak. (b) The corresponding spectrum graph with two nodes being generated for each peak. One under the assumption of being a b-ion, the other under the assumption of being a y-ion. It is obvious that the path starting at node s with mass 0 and ending at node t with mass 568 encodes the correct peptide sequence. The undirected edges connecting complementary nodes are drawn as dashed lines.

implemented in PILOT, to bound the total mass error of the predicted peptides.

The main contribution of this work is an improvement of this approach by an extension that retains most of the flexibility and leads to a remarkable improvement in running time. Instead of focusing on computing one antisymmetric path, we propose a novel algorithm to find the k best antisymmetric paths. We achieve this by applying the Lagrangian relaxation technique to the problem and solving the subproblems with an elegant variant of Yen's k shortest paths algorithm. Lagrangian relaxation was already successfully applied to biological problems such as sequence alignment [18], protein [19], and RNA [20] structural alignment or protein threading [21].

An additional contribution of this paper is a generic probabilistic scoring scheme that can be trained automatically for a data set of annotated spectra and is independent of the mass spectrometer type. The performance of both, *de novo* and database search approaches, depends on a good scoring function to model prediction quality. Currently used scoring functions range from rather simple peak intensity-based scoring to statistical models including Bayesian networks. The latter show a better performance, but require retraining for different spectrometer types and, thus depend on reliable annotated data sets. Our flexible scoring scheme allows for user controlled training on supplied annotated data sets. The topology of the network can either be defined by the user or, following the approach proposed by Datta and Bern [22], learned from the given data set directly. We extend this approach by considering ion intensities and cleavage positions similar to the PepNovo scoring in order to account for shifts of the fragmentation patterns between different m/z regions along the spectrum.

Our software ANTILOPE (ANTIsymmetric path search with Lagrangian Optimization for PEptide identification), an implementation of the improved approach, is freely available as part of upcoming releases of the open source proteomics library OpenMS [23].

The structure of the remainder of this paper is as follows. Section 2 describes our new method. In Section 3, we compare our tool with the state-of-the-art tools PepNovo, NovoHMM, LutefiskXP, PILOT, and PEAKS. Finally, in Section 4, we discuss our results and future work.

2 Novel *de novo* Peptide Sequencing Algorithm

This section describes our new approach to the *de novo* sequencing problem. At first, we formally introduce the graph-theoretic formulation and the resulting ILP formulation our method ANTILOPE is based on. Then, we present our new algorithmic approach to find the k best solutions of the ILP. Finally, we explain the scoring model of ANTILOPE.

2.1 Graph-Theoretical Formulation

Bartels introduced the transformation of a tandem mass spectrum into the so-called *spectrum graph*, a now commonly used data structure in graph-theoretical approaches to the *de novo* sequencing problem [6], [9], [16], see Fig. 1. Using this data structure, the original problem amounts to finding a longest path with certain properties in this graph.

When a peptide P is fragmented by collision induced dissociation (CID), it usually breaks along the backbone between two neighboring amino acids into a pair of N-terminal (prefix) and C-terminal (suffix) fragments. We define the residual mass of P as the sum of the monoisotopic masses of all amino acid residues in P. By parent mass M_P, we denote the total mass of P, which is the residual mass, plus 18 Da for an additional water molecule. Depending on the exact fragmentation position, different types of fragment ions are produced that have a certain mass offset compared to the prefix residue mass (PRM) or suffix residue mass. Besides the types presented in Fig. 2, also neutral loss variants, e.g., loss of water or ammonia, of

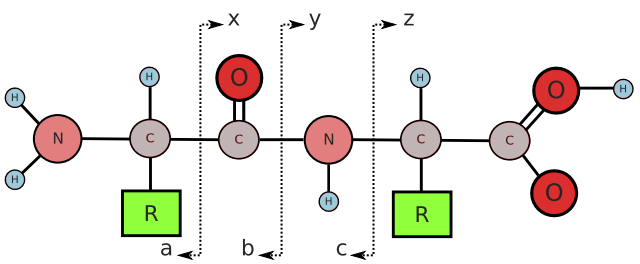


Fig. 2. Peptide fragmentation along the backbone. This figure displays the most prominent fragmentation positions for the generation of pairs of b/y-ion, a/x-ion, and c/z-ion in the backbone of a peptide.

several ion types are observed frequently as well as multiply charged ions. The fragmentation process is still not fully understood and which types are generated with which intensity depends on many factors.

The spectrum graph G, consists of a set of nodes V, a set of directed edges E_D, and a set of undirected edges E_U. In the original definition the spectrum graph does not contain the set of undirected edges, which is a slight modification by Liu et al. [16] who termed this the *extended spectrum graph*. In the spectrum graph, each node corresponds to some possible prefix residue mass of the peptide to be identified. Directed edges represent amino acids and connect nodes, if their mass difference can be explained by some amino acid. Two nodes that lead to contradicting interpretations of some mass peak are called complementary and are connected by an undirected edge.

Given the tandem mass spectrum of some unknown peptide the construction of the spectrum graph is as follows: each peak s with mass m_s in the input spectrum generates a set of nodes. If we consider k different N-terminal ion types (e.g., b-ion and a-ion) with mass offsets $\delta_1, \ldots, \delta_k (+1\ \text{Da}$ for b-ions, $-27\ \text{Da}$ for a-ions) from the PRM, then peak s generates k nodes with masses $m_s - \delta_1, \ldots, m_s - \delta_k$. For C-terminal ion-types with offsets $\delta_1, \ldots, \delta_k$, additional k nodes with masses $M_p - 18 - (m_s - \delta_1), \ldots, M_p - 18 - (m_s - \delta_k)$ are generated. Each of these nodes represents the prefix residue mass under the assumption that s was generated by an ion of a certain type. Clearly at most one of these nodes can represent the true PRM, therefore, they are all contradicting each other and are connected by undirected edges. Whenever the mass difference of two nodes v_i and v_k equals the mass of some amino acid $\alpha(\pm\epsilon)$, we connect v_i and v_k via a directed edge (v_i, v_k) labeled with α. Finally, we add two so-called *goalpost* nodes s and t, with masses 0 and $P_M - 18\ \text{Da}$, respectively.

If the spectrum of some peptide P is complete, i.e., fragment ion peaks are abundant for each possible cleavage site of P, then there exists a node for each PRM of P. Therefore, the correct sequence of P is obtained by finding the s-t path of nodes corresponding to the true prefix sequences of P and by concatenation of the edge labels along this path. Each node in the spectrum graph has a score that represents the reliability of that node to correspond to a true PRM.

However, simply looking for the longest path in the graph often leads to infeasible solutions, namely, if two nodes that were generated by the same peak are included in the path, since in general only one of them corresponds to a true PRM. This problem is aggravated when the score of each node is directly related to the intensity of the generating peak. In such a scenario, a high-intensity peak generates several high scoring nodes and a longest path search then tends to include a pair of complementary nodes in the longest path leading to a contradicting N- and C-terminal interpretation of the same peak. Such an infeasible path is called symmetric because the pairs of forbidden pairs of N-terminal and C-terminal nodes form a symmetric structure, which can be seen in Fig. 1. To solve the *de novo* sequencing problem, we, hence, have to search for *antisymmetric* paths. These are paths without contradicting nodes. They, therefore, do not contain pairs of nodes that are connected by an undirected edge. See Fig. 3 for an example.

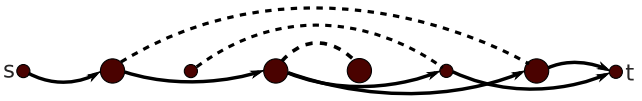


Fig. 3. Symmetric path example. This figure sketches schematically the situation when an infeasible symmetric path would be preferred over a feasible antisymmetric solution. Assuming that the small nodes have a score of 1 and the bold nodes have a score of 2, the illegal s-t path scores higher than the legal one. Therefore, in this example, a simple longest path search yields infeasible solutions.

Most *de novo* sequencing algorithms generate one pair of complementary nodes for each peak assuming it being either a b-ion or y-ion. These pairs form a nested non-interleaving structure allowing for efficient computation. But, although b- and y-ions are usually the most abundant in CID spectra, there are cases in which both of them are missing and, therefore, no correct node is generated in this case. Therefore, it is promising to include nodes for other interpretations, especially in the low and high mass range of the spectrum, where fragmentation is usually less complete.

While the longest antisymmetric path problem is NP-complete for general directed graphs [13], there exist polynomial algorithms for the special case, where the noninterleaving property is satisfied. The polynomial algorithm proposed by Chen et al. [14] uses dynamic programming to compute an optimal solution to the longest antisymmetric path with noninterleaving forbidden pairs. In a second paper, Lu and Chen [15] extended this approach to compute suboptimal solutions by constructing a so-called matrix spectrum graph and applying depth-first search and a backtracking algorithm. In contrast, the ILP formulation presented in the next section does not depend on such a nested structure and corresponds to the *de novo* sequencing problem for any desired set of ion types.

2.2 Integer Linear Programming Formulation

Our algorithm is based on the following ILP formulation [24], which is very similar to the one Floudas and DiMaggio used for their tool PILOT [11]. Our formulation models the problem by means of zero-one variables for each edge. We put the score of each node on all its outgoing directed edges. As the graph is acyclic, this is a safe transformation

$$\max \sum_{(v_i,v_k)\in E_D} c_{i,k} x_{i,k}, \tag{1}$$

$$\sum_{(v_s,v_k)\in E_D} x_{s,k} = 1, \tag{2}$$

$$\sum_{(v_k,v_t)\in E_D} x_{k,t} = 1, \tag{3}$$

$$\sum_{(v_i,v_k)\in E_D} x_{i,k} - \sum_{(v_k,v_j)\in E_D} x_{k,j} = 0 \qquad \forall k \in V \setminus \{v_s, v_t\}, \tag{4}$$

$$\sum_{v_i \in e} \sum_{(v_i,v_k)\in E_D} x_{i,k} \leq 1 \qquad \forall e \in E_U, \tag{5}$$

$$x_{i,k} \in \{0, 1\}. \tag{6}$$

We introduce a binary variable $x_{i,k}$ for every directed edge $(v_i, v_k) \in E_D$, which has value one if edge (v_i, v_k) is part of the path (active) and zero otherwise (inactive). The objective function (1) maximizes the summed score of all active directed edges. For the two goalposts s and t, the two constraints (2) and (3) assure that exactly one active edge leaves s and one enters t. Together with the flow conservation constraints (4), they establish a correspondence between feasible solutions of the ILP and s-t paths in the graph. An optimal solution of the ILP consisting of objective function (1) and constraints (2), (3), and (4) corresponds to a longest s-t path, still possibly symmetric and therefore infeasible for the *de novo* sequencing problem. Therefore, we add another constraint (5) that makes sure that for each pair of contradicting nodes at most one will be selected. The difference of our model to the one proposed by Floudas and DiMaggio is twofold. First, we do not introduce variables for nodes as they are not required. This does not change the general structure of the formulation and has no strong effect on the time required for solving. Second, we do not formulate a constraint that prevents the exact mass of the predicted sequence to deviate from the measured parent mass by more than a certain threshold value (usually 2.5 Da). We argue that in our algorithm it is more promising to defer this filtering to a later stage of the algorithm. Since, we add edges that correspond to pairs and triples of amino acids which often represent several possible combinations of amino acids, there is no exact mass which could be used in such a constraint. Therefore, we perform the filtering at a later stage when we have created the candidate superset.

2.3 Applying Lagrangian Relaxation

While linear programming (LP) problems can be solved in polynomial worst case time, adding integrality constraints makes them generally NP-hard and the resulting integer linear programs (ILPs) require different algorithmic solution approaches. One common method is to first solve the LP relaxation and then investigate the obtained solution. If the solution is fractional, one has to resort to techniques like branch-and-bound or branch-and-cut using upper and lower bounds obtained from heuristics and from the relaxed solution.

We apply a different kind of relaxation method, *Lagrangian relaxation*, which yields in many cases much more efficient algorithms than those based on LP relaxations because it can exploit structural knowledge of the problem. Lagrangian relaxation is motivated by the experience that many hard integer programming problems correspond to a significantly easier problem that has been complicated by an additional set of constraints. To obtain the efficiently computable Lagrangian problem, the complicating constraints are removed and replaced by a penalty term in the objective function. The relaxed problem obtained that way is called the Lagrangian problem and can often be solved efficiently.

The Lagrangian relaxation of the *de novo* sequencing ILP (1)-(6) is straightforward as it is very obvious that the antisymmetry constraints form the class of *hard* constraints that complicate the computationally *easy* problem of a longest path search in a directed acyclic graph (DAG). We can solve this relaxed problem by means of a simple standard algorithm, which can be found in reference material [25].

We apply Lagrangian relaxation by dropping the antisymmetry constraint (5) and moving it to the objective function to penalize its violation. This leads to the Lagrangian problem

$$\begin{aligned} Z(\lambda) = \max & \sum_{(v_i,v_k)\in E_D} x_{i,k}\left(c_{i,k} - \sum_{e\in E_U,\, v_i\in e} \lambda_e\right) + \sum_{e\in E_U} \lambda_e \\ & \sum_{(v_s,v_k)\in E_D} x_{s,k} = 1 \\ & \sum_{(v_k,v_t)\in E_D} x_{k,t} = 1 \\ & \sum_{(v_i,v_k)\in E_D} x_{i,k} - \sum_{(v_k,v_j)\in E_D} x_{k,j} = 0 \qquad \forall k \in V \setminus \{v_s, v_t\} \\ & x_{i,k} \in \{0,1\}. \end{aligned} \tag{7}$$

The vector λ holds the Lagrangian multipliers, nonnegative real numbers that define the weight of the penalty term.

Lemma 1. *The Lagrangian problem (7) can be solved in linear time and space.*

Proof. Solving the Lagrangian problem consist of the following steps: first, we simply subtract from each edge weight $c_{i,k}$ the value λ_e, for all undirected edges e incident to node v_i. Then, we apply the linear time $O(|V| + |E_D|)$ longest path search algorithm for DAGs on the graph with the modified edge weights. Finally, we add the value of $\sum_{e\in E_U} \lambda_e$ to the score obtained from the longest path search algorithm. Obviously, each of the steps requires only linear time and space. □

By restricting the Lagrangian multipliers to nonnegative values one can easily show that the value of the solution of the Lagrangian problem is an upper bound to the optimal value of the original problem [26]. In order to obtain a tight bound, the strategy is to find the values for the Lagrangian multipliers that minimize $Z(\lambda)$, which means solving the dual problem

$$Z_D = \min_{\lambda \geq 0} Z(\lambda).$$

We apply the efficient iterative subgradient optimization algorithm, computing sequences of multipliers λ^t, where $t = 0, 1, 2, \ldots$ denotes the iteration. We start with $\lambda_e^0 = 0$, for all $e \in E_U$ and in each iteration t we compute the subgradients $S_e^t = 1 - \sum_{v_i\in e} \sum_{(v_i,v_k)\in E_D} x_{i,k}$, for all $e \in E_U$ and update the Lagrangian multipliers according to formula

$$\lambda_e^{t+1} = \max\{0, \lambda_e^t - \theta^t S_e^t\}. \tag{8}$$

One crucial factor with a huge influence on the performance is the step-size θ. The subgradient method converges to the optimal solution Z_D, if the step-size satisfies the following conditions [27]:

$$\lim_{k\to\infty} \theta^k = 0 \quad \text{and} \quad \lim_{k\to\infty} \sum_{i=1}^{k} \theta^i = \infty.$$

A formula that is widely used for step-size computation because it shows good performance in practice is given by

$$\theta^t = \frac{\gamma^t(Z(\lambda^t) - Z^*)}{\sum_{e \in E_U} (S_e^t)^2}, \quad (9)$$

where Z^* is the value of the best solution to the original problem that was computed yet and γ^t defines a decreasing adaption parameter.

2.4 Suboptimal Solutions

A straightforward strategy to compute suboptimal solutions, also implemented in PILOT [11], is to cut off previous solutions by an additional constraint. A known drawback of this approach is that solving time may increase dramatically after generating a few suboptimal solutions. We suggest a different strategy and overcome this problem by means of an algorithm which, for a given number k, directly computes the k longest paths. We use the algorithm by Yen [28] that was originally designed to compute the k shortest paths without cycles on general directed graphs.

Yen's algorithm is a deviation algorithm based on the fact that the ith shortest path P_i, will coincide with every shorter path $P_{i-1} \dots P_1$ up to some node until it deviates. The farthest node from the source s with this property is called *deviation node* $d(P_i)$.

The strategy to find the $i+1$st shortest s-t path P_{i+1} is, starting at $d(P_i)$, to compute for each node v_j^i of P_i the shortest path to t, that deviates from P_i at node v_j^i. Therefore, a shortest path from v_j^i to t is computed which is not allowed to use the edge (v_j^i, v_{j+1}^i). This shortest path from v_j^i to t is then concatenated with the prefix $(v_1^i \dots v_{j-1}^i)$ of P_i to obtain the shortest s-t path that deviates from P_i at node v_j^i. This path is added to a candidate set X. After the shortest deviating paths of P_i have been computed, the shortest path in the candidate set X corresponds to the $i+1$st shortest s-t path P_{i+1} and is removed from X.

Yen's algorithm performs an additional trick to guarantee for paths without cycles that we do not discuss here. For a more detailed description of this algorithm and variants please refer to reference material [28], [29].

Our problem differs in a few points from the original problem solved by Yen's algorithm, so it requires a few adaptations. While Yen's algorithm is designed for general directed graphs that may contain cycles, we are working on a DAG. This simplifies the problem as we do not have to worry about cycles and can simply transform the shortest path problem into a longest path problem. Note that the longest path problem is NP-complete in graphs with cycles. A second difference is that we have the additional condition to find antisymmetric paths. Therefore, every time the shortest path algorithm is called in the Yen's algorithm, we replace this by solving the Lagrangian relaxation formulation for the longest antisymmetric path search. The following theorem and its proof capture the main algorithmic result of this paper.

Theorem 1. *The combination of our Lagrangian relaxation-based algorithm for antisymmetric paths and a modification of Yen's algorithm solves the problem of computing the k longest antisymmetric paths in time $O(kls(|E| + |V|))$, where l is the length of the longest path and s is the total number of subgradient iterations.*

Proof. In iteration $i+1$ of Yen's algorithm the computed path deviating from P_i at node v_j^i must satisfy two conditions in order to form an antisymmetric path in G.

1. There are no two nodes in the path from v_j^i to t that are in conflict.
2. None of the nodes in the computed path from v_j^i to t is in conflict with some node from the prefix of path P_i up to node v_j^i.

The first condition is satisfied by the Lagrangian relaxation formulation itself, because if applied to the subgraph $v_j^i \dots t$, every feasible solution corresponds to an antisymmetric path from node v_j^i to t. To meet the second condition, it is sufficient to remove all nodes from the subgraph $v_j^i \dots t$ that are connected via an undirected edge with some node of the prefix $s \dots v_j^i$ of P_i before we compute the longest antisymmetric path. This trick also simplifies the longest antisymmetric path search for increasing j as the possibilities to generate an infeasible solution are decreasing.

The complexity of Yen's algorithm for computing the k-longest paths in a DAG is $O(k|V|(|E| + |V|))$. The first factor $|V|$ comes from the fact that, in a general graph, one path can possibly contain all $|V|$ nodes. In the case of peptide sequencing, the length of a path equals the length of the predicted peptide which usually does not exceed a length of 30 for typical experimental settings. In the longest antisymmetric path version using Lagrangian relaxation, the $O(|E| + |V|)$ DAG longest path algorithm gets iteratively called during subgradient optimization algorithm. Therefore, the complexity of our formulation for identification of a peptide containing l amino acids is $O(kls(|E| + |V|))$ with s being the number of iterations during subgradient optimization. □

Note that the value of s is possibly exponential if the subgradient optimization does not converge and the complete branch and bound tree has to be enumerated. Nevertheless, in the results section, we will show that for our peptide sequencing formulation, on average, only very few iterations are required which leads to a practically efficient algorithm.

2.5 Scoring Model

We use a probabilistic scoring based on a Bayesian network similar to the scoring model of PepNovo. Bayesian networks are directed acyclic graphs, where nodes represent random variables and the edges represent conditional dependencies between variables. The variables in our model are the ion types $t \in T$ that are considered by our scoring model and the possible values for each variable is the intensity. Therefore, as a first step, we normalize the intensity of all peaks to discrete values as defined by Dančík et al. [9] by using their rank as intensity.

The usage of Bayesian networks for scoring nodes in the spectrum graph is motivated by the observation that fragmentation events are not independent. For example, the probability of observing a strong b-ion is not independent of the abundance and intensity of the complementary y-ion.

Unlike for the PepNovo algorithm, where the structure of the probabilistic network is predefined leading to a fixed set of accounted conditional dependencies, we implemented a flexible scoring scheme where the network topology can be either defined by the user or it can be learned during the training process automatically.

For inference and training of the Bayesian network, we used the Bayesian Network Classifiers in the machine learning suite Weka [30]. Similar to PepNovo, we discretize the relative position of a cleavage into several (default 3) equally sized regions r to account for the different intensity distributions in the center and terminal regions usually observed in CID spectra. For each of the regions, we train a Bayesian network using some training set of tandem mass spectra with known peptide identification. For each training spectrum, we construct the node set of the spectrum graph and select an equal number of true positives (vertices representing a true PRM) and false positives (vertices not representing a PRM). For each of the selected nodes, we look for witnessing peaks at their calculated positions and record their normalized intensity to obtain the training vectors for the Bayesian network. Each of the training vectors has one additional entry, the class label, which is T for true positives and F for false positives. We select only those ion types of the witness set for the network training that appear in at least t percent of the true positive samples of the training set, where the threshold parameter t can be defined by the user. For each of these selected types, we then add a node in the Bayesian network. One additional node for the class is added. During the network training, the structure (set of directed edges) of the network is learned and once the structure is fixed the conditional probability tables are learned from the training data. While the user can control a huge range of possible options for the Weka Bayesian network classifier training through our program, we set as the default training algorithm the K2-HillClimber and the Bayesian metric for local scoring [31]. For a user defined network topology the first step is skipped and only the conditional probability tables are computed.

Once the network is trained, we score a node v in the spectrum graph by looking for peaks in the spectrum at the calculated masses for the selected ion types to obtain the set of intensity observations I^v. Using the trained Bayesian network *BN*, we then compute the log likelihood ratio as

$$LLR(v) = \log \frac{\Pr(I^v \mid BN, \text{class} = T)}{\Pr(I^v \mid BN, \text{class} = F)}, \tag{10}$$

where $\Pr(I^v \mid BN, \text{class} = X \in \{T, F\})$ is the probability of observation I^v under model *BN* when the class variable is set to X. In contrast to Bern and Datta who obtain their false positive samples from perturbation of the correct PRM, we only take false positive PRM for which a node was created during the spectrum graph construction. We chose that approach, since we want the Bayesian network to discriminate between correct and false nodes in the spectrum graph. By just perturbing the true PRM one will very likely generate false positive training samples containing only zero intensity entries, which will never be the case for a node in the spectrum graph as it requires at least one peak to be generated.

Additional to the Bayesian network, we also use a simple intensity rank score $S_R(v)$ as it is also used by INSPECT [1]. This score is the ratio between two probabilities, the probability that a peak with a certain intensity rank corresponds to a certain ion type (e.g., a b-ion) and the the probability that a randomly chosen peak was generated by that ion type. As these values differ between different mass regions of a spectrum, we split the spectrum into three equally spaced mass regions and estimate the probabilities for each of them separately using the same training data as for the Bayesian network. For example, if we generate a node for a peak of rank 4, and this node interprets the peak as a b-ion, then $S_R(v)$ is the log ratio between the probability that a rank 4 peak is a b-ion and the probability that any random peak is a b-ion.

The final score $s(v)$ for each node v of the spectrum graph is then computed as

$$s(v) = LLR(v) + S_R(v). \tag{11}$$

Nodes having negative scores correspond to unreliable PRMs and are removed from the graph in order to reduce the size of the spectrum graph and speed up the candidate generation process. Since our formulation is working with edge weights, we move the node scores onto the edges, such that each directed edge gets the score of its left node. In the filtered spectrum graph, we compute the predefined number of suboptimal solutions, each corresponding to one antisymmetric path. To account for missed cleavages, we also add edges corresponding to pairs and triples of amino acids to the spectrum graph. For each of the generated candidates, in a second step, we try to resolve the pairs and triples of amino acids. Therefore, we generate all possible combinations and permutations of amino acids to generate a candidate superset. The candidates in that superset are then rescored by a refined version of a shared peaks count, where we reward abundant witness peaks and penalize missing ones.

Given a candidate sequence we look for witnessing peaks in the query spectrum and give a bonus if one was found or a penalty if it is missing. Further, we check whether the peak is a primary isotopic peak, a secondary isotopic peak or a lone peak. A peak is called a primary isotopic peak if we find a child peak at offset 1 Da for a singly charged ion or 0.5 Da for a doubly charged ion. Equivalently, a peak is called a secondary isotopic peak if it has a parent peak with offset -1 Da for a singly charged ion or -0.5 Da for a doubly charged ion. If a peak is neither primary isotopic nor secondary isotopic then it is a lone peak. If a witnessing peak is a primary peak, we add another bonus to the score while we charge a penalty if it is a secondary peak. While the reward and penalty score are actually user parameters, we will offer a generic algorithm to estimate reasonable values in future versions. The candidates are then reranked according to this score and the predefined number of candidates is returned.

3 Results

In this section, we present and discuss our computational results. We compared ANTILOPE with state-of-the-art alternative peptide identification software with respect to running time and quality.

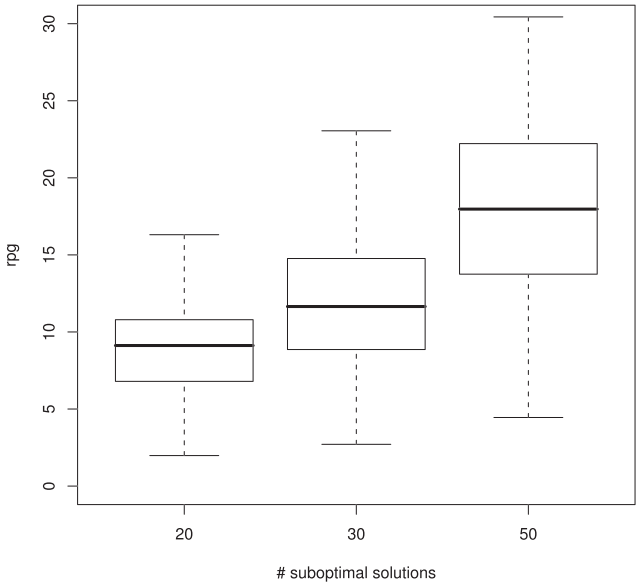


Fig. 4. Running time comparison between Lagrangian relaxation and ILP formulation for computation of 20, 30, and 50 suboptimal solutions of 100 benchmark spectra. Box-and-whisker plots display median, quartiles, and extrema of the distribution of relative performance gains $rpg = \mathrm{runtime(ILP)}/\mathrm{runtime(Lagrange)}$. ANTILOPE outperforms the CPLEX-based method for all spectra and all numbers of suboptimal solutions. The advantage increases with the number of suboptimal solutions. The considered spectrum graphs contained between 80 and 200 nodes.

3.1 Efficiency

The major contribution of this work is the new algorithmic approach based on Lagrangian relaxation. We will first give a thorough analysis of the performance and compare it to the ILP formulation (1)-(6). Like implemented in PILOT, we generate the suboptimal solutions by introducing additional constraints that cut off previous solutions. We implemented our algorithm in C++ and use the OpenMS [23] library that offers convenient data structures and algorithms to handle and manipulate spectral data. For the ILP formulation, we use the commercial CPLEX [32] solver software (version 9.0), which is in general the fastest solver available. We were not able to directly compare to PILOT because the software is not available upon request. In Fig. 4, we compare the running times of the ILP and our Lagrangian relaxation formulation on a set of 100 tandem mass spectra from the ISB data set [33]. In this comparison, we only consider the time required to generate the set of candidate sequences and ignore the preprocessing of the spectrum, the spectrum graph generation, and rescoring as these steps are independent of the applied algorithm. We compared the running time required to generate the top scoring 20, 30, and 50 candidates for each spectrum. The figure shows that our approach significantly outperforms the ILP formulation on all instances and the performance gain increases with the number of candidates to be generated. Our algorithm is, on average, ≈ 9 times faster for the best 20 candidates, for 30 and 50 candidates the average advantage increases to a factor of ≈ 12 and ≈ 18. While the runtime for of the ILP formulation for the top 50 candidates was usually above 2 seconds, the Lagrangian relaxation formulation requires, on average, only a few tenths of a second.

In a closer analysis, we investigated the convergence behavior of our Lagrangian relaxation formulation. It reveals that for each Lagrange problem solved during the path ranking algorithm only very few iterations of the subgradient optimization are required. The path ranking algorithm maintains a list of previously detected candidate paths together with their scores. Since the score $Z(\lambda)$ of the Lagrange problem is an upper bound to the score Z_{IP} of the best possible feasible solution, subgradient optimization can be aborted as soon as $Z(\lambda)$ falls below the lowest score in the candidate list. If the Lagrangian relaxation does not converge and cannot be aborted after 100 iterations, we apply a branching step. We use the best infeasible path found during the subgradient optimization and arbitrarily choose one node v_b involved in a conflict. Then, we generate two subproblems, one forcing v_b to be in the path and one forbidding v_b to be selected. We found that only for a very small fraction of the Lagrange problems a branching step had to be performed, and the depth of branch-and-bound trees never exceeded a value of 3. It is necessary to mention that the performance strongly depends on the scoring function used, since a good scoring function will not generate many high scoring nodes for the same peak and only the correct one should receive a significantly high score. Therefore, a good scoring function does not only affect the identification performance, but also affects the complexity of the candidate generation.

3.2 Sequencing Performance

We compared the performance of ANTILOPE to four noncommercial *de novo* sequencing tools, LutefiskXP, NovoHMM, PILOT,[1] PepNovo, and the commercial software PEAKS.[2] We used two measures, accuracy and recall, to assess their performance. Accuracy denotes the fraction of correctly predicted amino acid residues compared to all predicted residues. Recall is the fraction of correctly predicted residues compared to the total number of residues of the correct peptide sequences. When looking at suboptimal solutions for each algorithm, we looked for the prediction with the highest recall and reported the values of this prediction for recall and precision. In case of multiple predictions with the same recall value, we report the values for the one with the highest precision among them. As benchmark set, we chose tandem mass spectra from the ISB data set [33] that were generated by an ESI-ion trap mass spectrometer by Thermo Finnigan and spectra from the open proteomics database. This set of reliably annotated spectra from tryptic peptides has already been used for training of PepNovo and NovoHMM. We created a training set of 1214 spectra from doubly charged precursor ions of unique peptides to train the scoring model. During the Bayesian network training for each of the 3 mass sectors, only the ion types that had a peak in at least 20 percent of the true positive training samples are selected for the corresponding Bayesian network. The topologies of the Bayesian networks together with a brief discussion can be found in the supplementary material which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2011.59. The parameters to score the peptide spectrum matches for the candidate sequences in the superset were chosen as follows: for an abundant b- or y-ion, we awarded

1. As PILOT was not available, the identifications for the test data were generated by the authors of PILOT.
2. We used the PEAKS Online 2.0 web interface.

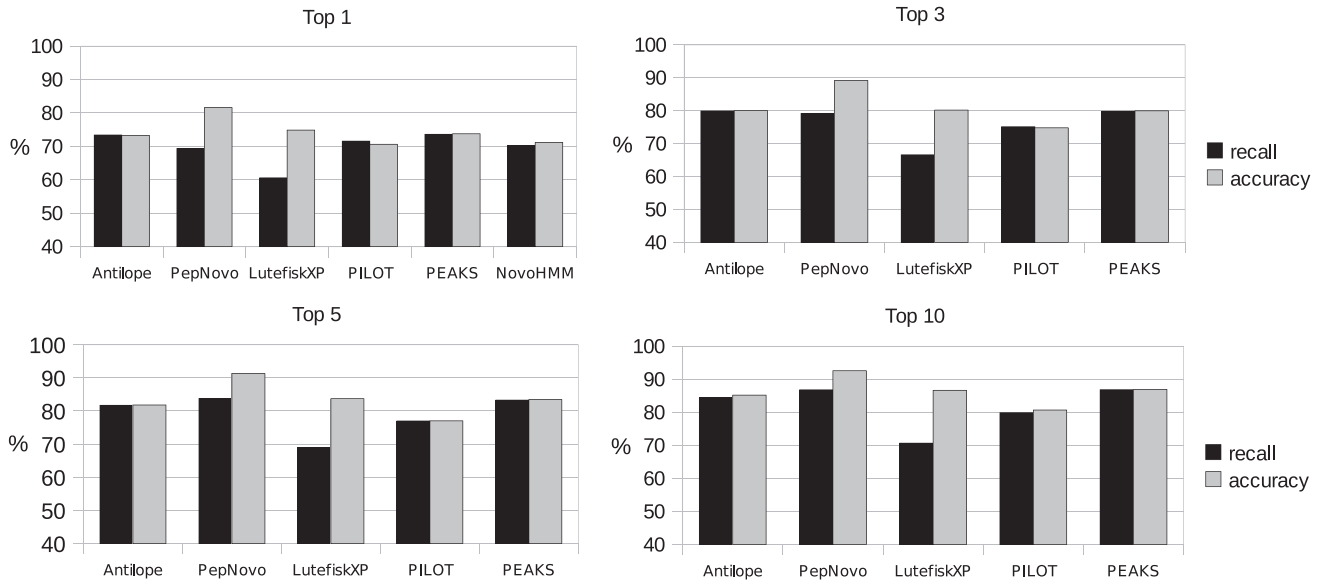


Fig. 5. Benchmark. Comparison of accuracy and recall of ANTILOPE with NovoHMM, PepNovo, PILOT, PEAKS, and LutefiskXP. We compare the accuracy and recall of the best prediction among the top 1, 3, 5, and 10 ranked candidates returned by each tool. As the best prediction, we consider the one with the best recall among the candidates. Since NovoHMM generates only one candidate per spectrum, it appears only in the first plot. Discussion in text.

the score $PSM_b = PSM_y = 1$, doubly charged b- or y-ions scored 0.5, a-ions 0.3 and all neutral losses were awarded a score of 0.2. Isotopic peaks for some type t were awarded a score of $PSM_t \cdot 0.2$. If some peak was missing the penalty of $PSM_t \cdot 0.5$ was subtracted from the score. When a peak was classified as a secondary peak its score is reduced to $PSM_t \cdot 0.8$. The score for some peak is then weighted with the relative m/z distance between the expected and the observed m/z value using a linear function.

The test set consists of 200 spectra of peptides (peptides in training and test data set disjunct) with a molecular mass of at most $1{,}600$ Da and an average peptide length of 10 residues. We score a predicted amino acid as correct, if its predicted starting mass position does not deviate by more than 2.5 Da from the correct starting mass position. Further, in our evaluation, we do not discriminate between the amino acids Q/K and I/L since their masses cannot be distinguished.

To compare the tools, we do not only look at the top hit, but, we also look at the accuracy and recall for the best hit in the top 3, 5, and 10 candidates. The results are presented in Fig. 5. Since NovoHMM only generates one candidate per spectrum it appears only in the first plot. Looking only at the top hit, the recall of ANTILOPE ($\approx$73.4%) is only marginally lower than of PEAKS ($\approx$73.7%), but slightly better than that of PILOT ($\approx$71.5%), NovoHMM ($\approx$70.2%), and PepNovo ($\approx$69.4%). The recall of LutefiskXP ($\approx$60.6%) is much lower than for all other tools. Since ANTILOPE and NovoHMM both compute complete sequences, they have almost equal values for accuracy and recall, while for LutefiskXP and PepNovo these values differ as they allow for gaps in their predicted sequences. If we go over from the top hit to the best 3, 5, and 10 candidates, we observe that in terms of recall, ANTILOPE is always very close to PepNovo and PEAKS (equal for the top 3, 2.5 percent advantage of PepNovo and PEAKS for the top 10) and always approximately 4 percent better than PILOT. In terms of accuracy, PepNovo ($\approx$92%) has a better performance than ANTILOPE, PEAKS, and PILOT since it allows for partial peptide predictions. The accuracy of LutefiskXP is slightly better than for ANTILOPE and PILOT, but this accuracy is achieved at a much lower recall which is between 12 and 14 percent lower in all four cases. The four tools ANTILOPE, LutefiskXP, NovoHMM, and PepNovo are comparable in terms of runtime which is usually between 0.5 and 1.5 seconds per spectrum. The running time of PILOT as reported by the authors was, on average, around 9 seconds per spectrum. We cannot directly estimate the runtime of PEAKS since the identification is performed via a web interface.

4 Conclusion

We proposed a new algorithmic approach to solve the longest antisymmetric path problem by means of Lagrangian relaxation, combined with a polynomial algorithm for suboptimal solutions. Using this approach, the algorithm is flexible and not restricted to the nested structure of the spectrum graph and solves this problem much faster than an LP relaxation-based method for the same formulation. Therefore, for our tool ANTILOPE, the candidate generation is no longer the bottleneck as the most time consuming step is the reranking phase since the number of possible candidates can easily explode if several double and triple amino acid edges are selected. In terms of sequencing performance, ANTILOPE is already competitive to available state-of-the-art programs PepNovo and PEAKS while it outperforms LutefiskXP and NovoHMM especially if we also consider suboptimal solutions. For long peptides PepNovo still has a small advantage, which is mostly due to the fact that the current version of ANTILOPE produces only complete annotations without gaps.

Actually, we only generated two nodes for each peak, one for a b- and one for a y-ion. Generating nodes for all ion types decreased the performance as this always lead to some high scoring, but false nodes and, thus to wrong interpretations. Nevertheless, we are sure that generating more nodes can lead to better identifications in combination with a refined scoring scheme. The algorithmic framework is flexible enough to work with mass spectra generated by different

kind of mass spectrometers. So, the user can define for which ion types a node shall be generated. This can lead to improved identification performance for different data sets. Combined with a scoring function trained on a representative set of spectra, the ability of our algorithm to directly model multiply charged ions can lead to an improvement over the other algorithms when analyzing tandem mass spectra obtained from higher charged precursor ions.

For the future, we plan to improve our algorithm in several directions. We will include support for identification of peptides containing posttranslational modifications. Further, we want to support combinations of complementary fragmentation techniques like CID together with electron transfer dissociation (ETD) or CID with electron capture dissociation (ECD), which can improve the identification [22], [34]. In these applications, the flexibility of our formulation may become a major advantage over existing programs.

To improve the performance for spectra of longer peptides, we will extend ANTILOPE in a way that it can produce partial predictions allowing for gaps at the terminals. This, together with a machine learning strategy for the rescoring like the rank-boosting algorithm used by PepNovo, should lead to a further improvement. ANTILOPE is freely available as part upcoming releases of the open source proteomics library OpenMS [23] allowing for convenient integration into experimental workflows.

ACKNOWLEDGMENTS

S. Andreotti is the corresponding author. K. Reinert and G.W. Klau are shared last authors.

REFERENCES

[1] S. Tanner, H. Shu, A. Frank, L.C. Wang, E. Zandi, M. Mumby, P.A. Pevzner, and V. Bafna, "Inspect: Identification of Posttranslationally Modified Peptides from Tandem Mass Spectra," *Analytical Chemistry,* vol. 77, no. 14, pp. 4626-4639, 2005.

[2] J.K. Eng, A.L. McCormack, and J.R. Yates, "An Approach to Correlate Tandem Mass Spectral Data of Peptides with Amino Acid Sequences in a Protein Database," *J. Am. Soc. for Mass Spectrometry,* vol. 5, no. 11, pp. 976-989, 1994.

[3] D.N. Perkins, D.J. Pappin, D.M. Creasy, and J.S. Cottrell, "Probability-Based Protein Identification by Searching Sequence Databases Using Mass Spectrometry Data," *Electrophoresis,* vol. 20, no. 18, pp. 3551-3567, 1999.

[4] L.Y. Geer, S.P. Markey, J.A. Kowalak, L. Wagner, M. Xu, D.M. Maynard, X. Yang, W. Shi, and S.H. Bryant, "Open Mass Spectrometry Search Algorithm," *J. Proteome Research,* vol. 3, no. 5, pp. 958-964, 2004.

[5] B. Ma, K. Zhang, C. Hendrie, C. Liang, M. Li, A. Doherty-Kirby, and G. Lajoie, "Peaks: Powerful Software for Peptide De Novo Sequencing by MS/MS," *Rapid Comm. Mass Spectrom,* vol. 17, pp. 2337-2342, 2003.

[6] A. Frank and P. Pevzner, "PepNovo: De Novo Peptide Sequencing via Probabilistic Network Modeling," *Analytical Chemistry,* vol. 77, no. 4, pp. 964-973, 2005.

[7] B. Fischer, V. Roth, F. Roos, J. Grossmann, S. Baginsky, P. Widmayer, W. Gruissem, and J.M. Buhmann, "NovoHMM: A Hidden Markov Model for De Novo Peptide Sequencing," *Analytical Chemistry,* vol. 77, no. 22, pp. 7265-7273, 2005.

[8] 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, no. 9, pp. 1067-1075, 1997.

[9] V. Dančík, T.A. Addona, K.R. Clauser, J.E. Vath, and P. Pevzner, "De Novo Protein Sequencing via Tandem Mass-Spectrometry," *J. Computational Biology,* vol. 6, pp. 327-341, 1999.

[10] M. Bern and D. Goldberg, "De Novo Analysis of Peptide Tandem Mass Spectra by Spectral Graph Partitioning," *J. Computational Biology,* vol. 13, no. 2, pp. 364-378, 2006.

[11] P.A. DiMaggio and C.A. Floudas, "De Novo Peptide Identification via Tandem Mass Spectrometry and Integer Linear Optimization," *Analytical Chemistry,* vol. 79, no. 4, pp. 1433-1446, 2007.

[12] C. Bartels, "Fast Algorithm for Peptide Sequencing by Mass Spectroscopy," *Biological Mass Spectrometry,* vol. 19, no. 6, pp. 363-368, 1990.

[13] H.N. Gabow, S.N. Maheshwari, and L.J. Osterweil, "On Two Problems in the Generation of Program Test Paths," *IEEE Trans. Software Eng.,* vol. SE-2, no. 3, pp. 227-231, Sept. 1976.

[14] 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.

[15] B. Lu and T. Chen, "A Suboptimal Algorithm for De Novo Peptide Sequencing via Tandem Mass Spectrometry," *J. Computational Biology,* vol. 10, no. 1, pp. 1-12, 2003.

[16] C. Liu, Y. Song, B. Yan, Y. Xu, and L. Cai, "Fast De Novo Peptide Sequencing and Spectral Alignment via Tree Decomposition," *Proc. 11th Pacific Symp. Biocomputing (PSB),* pp. 255-266, 2006.

[17] V. Bafna and N. Edwards, "On De Novo Interpretation of Tandem Mass Spectra for Peptide Identification," *Proc. Seventh Ann. Int'l Conf. Research Computational Molecular Biology (RECOMB),* pp. 9-18, 2003.

[18] E. Althaus and S. Canzar, "A Lagrangian Relaxation Approach for the Multiple Sequence Alignment Problem," *J. Combinatorial Optimization,* vol. 16, no. 2, pp. 127-154, 2008.

[19] A. Caprara, R. Carr, S. Istrail, G. Lancia, and B. Walenz, "1001 Optimal PDB Structure Alignments: Integer Programming Methods for Finding the Maximum Contact Map Overlap," *J. Computational Biology,* vol. 11, no. 1, pp. 27-52, 2004.

[20] M. Bauer, G.W. Klau, and K. Reinert, "Accurate Multiple Sequence-Structure Alignment of RNA Sequences Using Combinatorial Optimization," *BMC Bioinformatics,* vol. 8, article 271, 2007.

[21] S. Balev, "Solving the Protein Threading Problem by Lagrangian Relaxation," *Algorithms in Bioinformatics,* LNCS, I. Jonassen and J. Kim, eds., vol. 3240, pp. 182-193, Springer Berlin/Heidelberg, 2004.

[22] R. Datta and M. Bern, "Spectrum Fusion: Using Multiple Mass Spectra for de Novo Peptide Sequencing," *J. Computational Biology,* vol. 16, no. 8, pp. 1169-1182, 2009.

[23] M. Sturm, A. Bertsch, C. Groepl, A. Hildebrandt, R. Hussong, E. Lange, N. Pfeifer, O. Schulz-Trieglaff, A. Zerck, K. Reinert, and O. Kohlbacher, "OpenMS—An Open-Source Software Framework for Mass Spectrometry," *BMC Bioinformatics,* vol. 9, article 163, 2008.

[24] S. Andreotti, "Fast De Novo Sequencing with Mathematical Programming," master's thesis, Freie Universität Berlin, Jan. 2008.

[25] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms,* second ed. MIT Press, 2001.

[26] D. Bertsimas and J. Tsitsiklis, *Introduction to Linear Optimization.* Athena Scientific, 1997.

[27] M. Held, P. Wolfe, and H.D. Crowder, "Validation of Subgradient Optimization," *Math. Programming,* vol. 6, pp. 62-88, 1974.

[28] J.Y. Yen, "Finding the *K* Shortest Loopless Paths in a Network," *Management Science,* vol. 17, pp. 712-716, 1971.

[29] E. Martins and M. Pascoal, "A New Implementation of Yen's Ranking Loopless Paths Algorithm," *Quarterly J. Belgian, French and Italian Operations Research Societies,* vol. 1, pp. 121-133, 2003.

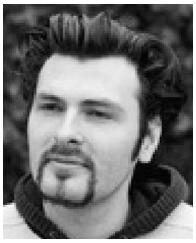
[30] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten, "The WEKA Data Mining Software: An Update," *SIGKDD Explorations Newsletter,* vol. 11, pp. 10-18, http://doi.acm.org/10.1145/1656274.1656278, Nov. 2009.

[31] R.R. Bouckaert, "Bayesian Network Classifiers in Weka," technical report, Dept. of Computer Science, The University of Waikato, http://www.cs.waikato.ac.nz/~ml/publications/2004/uow-cs-wp-2004-14.pdf, Oct. 2004.

[32] IBM,"cplex," http://www.cplex.com, 2011.

[33] A. Keller, S. Purvine, A.I. Nesvizhskii, S. Stolyar, D.R. Goodlett, and E. Kolker, "Experimental Protein Mixture for Validating Tandem Mass Spectral Analysis," *OMICS: A J. Integrative Biology,* vol. 6, pp. 207-212, 2002.

[34] A. Bertsch, A. Leinenbach, A. Pervukhin, M. Lubeck, R. Hartmer, C. Baessmann, Y.A. Elnakady, R. Müller, S. Böcker, C.G. Huber, and O. Kohlbacher, "De Novo Peptide Sequencing by Tandem MS Using Complementary CID and Electron Transfer Dissociation," *Electrophoresis,* vol. 30, no. 21, pp. 3736-3747, 2009.



Sandro Andreotti received the MSc degree in bioinformatics from Freie Universität Berlin, Germany, in 2008, where he is currently working toward the PhD degree in the algorithmic bioinformatics group of Knut Reinert. His research focuses on computational proteomics and discrete optimization.

Gunnar W. Klau received the PhD degree in computer science in 2001 from Saarland University, Germany. Currently, he heads the Life Sciences group at CWI, the national research center for mathematics and computer science in the Netherlands. His research interests include combinatorial algorithms and discrete optimization in computational biology.

Knut Reinert received the PhD degree in 1999 from Saarland University, Germany. He worked from 1999 to 2002 for Celera Genomics and took part in the sequencing of the human genome. Currently, he is a professor for algorithmic bioinformatics at the FU Berlin in Germany. His research interests include developing algorithms for sequence analysis and proteomics.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.