APPLICATION OF COMBINATORIAL METHODS TO PROTEIN IDENTIFICATION IN PEPTIDE MASS FINGERPRINTING

Leonid Molokov
Department of Computer Science and Engineering, Chalmers University of Technology, Gothenburg, Sweden

Keywords: Bioinformatics, Proteomics, Peptide mass fingerprinting, Error removal, Union editing, Set cover, Hitting set, Simulations.

Abstract: Peptide Mass Fingerprinting (PMF) for long has been a widely used and reliable method for protein identification. However it faced several problems, the most important of which is inability of classical methods to deal with protein mixtures. To cope with this problem, more costly experimental techniques are employed. We investigate, whether it is possible to extract more information from PMF by more thorough data analysis. To do this, we propose a novel method to remove noise from the data and show how the results can be interpreted in a different way. We also provide simulation results suggesting our method can be used for analysis of small mixtures.

1 INTRODUCTION

Proteomics is an important and fast developing field in bioinformatics, concerned in protein identification and quantification. As proteins are complex molecules, often one can only measure the properties of their side-products or components. Proteomics studies how to produce smaller molecules, what properties of theirs to investigate and how to recover information about initial proteins from these known properties. One of the most extensively used techniques in the field is peptide mass fingerprinting (PMF). The key idea of the method is to break proteins into pieces, measure masses of the fragments and use them to identify the initial proteins. Protein fragmentation is believed to preserve through repeating experiments thus making possible analysis of the observed set of masses. Although in past decades the method was widely used, now it loses to shotgun proteomics, because of these two major problems:

1. Mass ambiguity. There are peptides with nearly identical masses, with difference being less than a sensitivity of the measurement tool.
2. Single protein analysis. Many widely-used analysis methods such as Mascot (Perkins et al., 1999), Ms-Fit (Clauser et al., 1999) and ProFound (Zhang and Chait, 2000) treat the sample as if it was a single protein. Analysis of more complex mixtures is deemed to be too difficult due to reasons that we are going to discuss below.

Mass ambiguity leads to less specificity in protein identification, and is an experimental setup problem, which we hardly can solve. On the contrary, single protein limitation is put by the method interpreting the data, what was already recognized in earlier works (He et al., 2009),(Jensen et al., 1997) with attempts to provide solutions for protein mixtures.

But why was this limitation put? Consider a set of masses $M = \{m_1, m_2, \ldots, m_k\}$ observed in the experiment. Let us make an assumption that the experiment was held in perfect conditions and each particular mass m corresponds to some protein p, whose mass spectrum $M_p \subset M$, is fully represented in the experimental spectrum. It is also reasonable to put some limitations on the cardinality of initial set of proteins $P = \{p_1, p_2, \ldots, p_n\}$, that produced spectrum M: we assume $n \leq c$ (where c is some integer constant). We also have a collection of *theoretical spectra* $M_p, p \in P_{db}$ for each protein p in database. One can see that the question, whether there exist a set of proteins P_s such that $card(P) \leq c$ and $M \subseteq \bigcup_{p \in P_s} M_p$ explaining the spectrum M, is precisely a SET COVER problem, which is known to be NP-complete. We have an enumeration problem, i. e. describe all the possible satisfying sets (if there are any) which is even harder. Nevertheless, simulations show that in practical cases, most proteins have unique masses, what means quite optimistic real running time. We will cover that in section "Simulation". Worst case analysis is not hopeless as well, if one fixes the

Molokov L..
APPLICATION OF COMBINATORIAL METHODS TO PROTEIN IDENTIFICATION IN PEPTIDE MASS FINGERPRINTING.
DOI: 10.5220/0003102703070313
In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval* (KDIR-2010), pages 307-313
ISBN: 978-989-8425-28-7

maximum number of occurrence (degree) of an element (mass) in different sets (proteins), the problem is fixed-parameter tractable (FPT), for details, see (Damaschke and Molokov, 2009). It seems that masses with high degrees are not informative. In extreme case where a degree of particular mass is equal to number of all possible proteins (omnipresent mass), it can be safely removed from problem formulation, as any protein mixture will explain it. Otherwise, almost for sure there exists a mass that explains a subset of proteins that mass with big degree does. What leads us to conclusion, that one can drop high-degree masses from consideration (and thus bound the the maximum degree of an element). Also, obtained solutions can be verified later to explain these masses.

The aforementioned computational problem surprisingly reappears in works dedicated to protein inference in shotgun proteomics, (Nesvizhskii and Aebersold, 2005; Aebersold and Mann, 2003), where instead of mass spectra, the peptide sequences are used. Again the goal is identification of proteins, and the problem was recognized as NP-hard in (Aebersold and Mann, 2003).

One can see, that in both cases, whether the proteins are being identified by observed masses or peptides sequences, the presence of errors should be taken into account. Indeed, some observations (masses, sequences) could be lost or altered the way that equipment cannot detect them, and some represent noisy peaks, peptide fragments that were inserted as a consequence of flaws in the experiment. A significant amount of noise can be removed as described in (Samuelsson et al., 2004), but these procedures do not guarantee error-free output. A way to fix it is to reformulate the problem as SET COVER WITH MISSING ELEMENTS : given a spectrum M, a database $M_p, p \in P_{db}$ of theoretical mass spectra and numbers k, c, find out if there's a set of proteins P_s s. t.

$$card(P) \leq c, |M \triangle \bigcup_{p \in P_s} M_p| \leq k.$$

Let us refer to it in future as SCME. If $k = 0$, this is a classical SET COVER problem.

Unfortunately if $k \geq 10$, resulting space of possible solutions contains so many sets and elements, that it is infeasible to find solutions in reasonable time. It might be the reason why scoring candidate set is preferred to exact enumeration of solutions in both classical single-protein schemes and in attempts to cope with more complex mixtures, (He et al., 2009),(Jensen et al., 1997). There is an implicit assumption that there exists a set of proteins (or a single protein) that explains the data best; hence, a scoring function is introduced, and a set of proteins on which the function reaches its optimal value is searched (unfortunately, it has an expected problem, that only local optimum can be guaranteed). We considered a different possibility, that there might be several such sets, and it might be interesting to observe all possible solutions, to see, e. g., which proteins present in all or in most of solutions (omnipresent proteins), which seem to appear one instead of another, and which are only observed in a few solutions. The first type of proteins are the most important ones, because if one assumes that initial set of proteins is among listed solutions, then omnipresent proteins are contained in initial set of proteins as well.

One way to reduce the complexity of SCME is to use some extra information from the data. We already know that without errors, the mass spectrum should be precisely the union of the theoretical spectra of some set of proteins. This knowledge gives rise to a different approach: first, remove the errors, then try to solve the now simple (according to what simulations show) problem. To remove errors, we want to solve another computational problem, namely UNION EDITING, which we formulate below: *Given set M, a collection of its subsets $M_p, p \in P_{db}$ and integers k and l, find some complete union M' which is obtained from M by at most k insertions and l deletions, that is,*

$$|M' \setminus M| \leq k, \ |M \setminus M'| \leq l.$$

(A complete union notion will be defined in "Formalization" section for the sake of consistency.) Again, in the enumeration version we want all such sets M'. It appears that a simple branching algorithm for solving UNION EDITING, that we will show later, works successfully and sufficiently fast.

Therefore, the approach of enumerating the possible satisfying protein sets seems promising to us. We are going to present algorithm solving the problem with simulation results saved for the last section of this article.

2 FORMALIZATION

2.1 Preprocessing

It is time go into more details about how one would formulate the biological problem in mathematical terms. We will refer to protein set (mixture) $P = \{p_1, p_2, \ldots, p_k\}$ as *initial protein set*. As a result of mass spectroscopy, P generated a set of masses $M = \{m_1, m_2, \ldots, m_c\}$ (*theoretical spectrum of P*), which was changed by experimental errors into M' (*empirical spectrum of P*). We want to understand what P was.

Let us say that P (set of proteins) explains M (theoretical spectrum of P), if $M = \bigcup_{p \in P} M_p$. If, however, there is a $P' \subset P$, that explains M fully, i. e. produces the same spectrum, it is impossible to say, given the data, whether P or P' generated M. We will use Occam's Razor principle in a similar way as in (Nesvizhskii and Aebersold, 2005). More precisely, we think it is reasonable to seek only *minimal* protein sets, that is, sets with the property that if one vertex is removed from the protein set, it no longer explains M. It is also clear, that one has to limit the size of candidate mixtures, because experimental setup attempts to reduce the number of proteins under investigation.

As was mentioned before, in our model we assume that throughout the experiment, ideal spectrum M was altered by k insertions and and l deletions. Let us denote set of inserted masses by M_{ins} and set of deleted masses by M_{del} ($M_del \subseteq M$). What we in reality observe is empirical spectrum M', from which we want to infer what possible spectra M could be. Some insertions can be removed almost immediately, for the first preprocessing steps, see (Samuelsson et al., 2004), however, some will still remain. Unless we specify boundaries for possible amounts of errors (reflecting the quality of experiment), it is impossible to say, what M was, because M' could drift too far from it's ancestor. I. e. we have to make the additional assumption, that k and l are limited by constants k_0 and l_0. As soon as we do it, however, we get an immediate outcome: none of the proteins $P_{big} : \{p \in P_{db} | card(M_p \backslash M') > l_0$ or $card(M' \backslash M_p) > k_0\}$ were in P. It is a useful observation, because if a mass can be only produced by proteins in P_{big}, it could not be in M.

In other words, we have a hypergraph H, whose vertices represent masses, and each of its edges represents protein. A vertex representing a mass is incident to an edge representing a protein iff the protein has the mass in its spectrum. Essentially, $H = (V, E)$, where $V = M_{all}$, $E = P_{db}$, where M_{all} represent all the masses in the database. Given $M' \subseteq M_{all}$, a set of observed masses, k_0 and l_0 as integer boundaries for the numbers of insertions and deletions respectively, find $M \subseteq M_{all}$ such that $|M \backslash M'| < l_0$, $|M' \backslash M_p < k_0|$ and $M = \bigcup_{p \in P_s} M_p$, where P_s is some subset of P_{db}. In this error correction problem we do not limit the size of P_s by constant c, as it can be done on a later stage, where one solves the Set Cover problem for the recovered masses spectra.

As was said, the list of possible M for H and for $H' = (M_{all}, P_{db} \backslash P_{big})$ is the same (follows immediately from problem formulation), what gives substantial reduction of edges. Can we do anything else? The answer is yes, but we have to make an additional assumption. Consider a mass (vertex) $m \in M_{all}$. If m is not adjacent to any mass (vertex) from M in hypergraph H', then it is one of thousands of other masses in the database, about presence of which we have no available evidence. We assume that no such mass was initially in the experiment, because we cannot make any definite conclusion about these masses. It will allow us to shrink the hypergraph to edges adjacent only to M'. Let us denote such a hypergraph $H'(M')$.

The aforementioned assumption is strong and can lead to exclusion of some proper solutions. How drastic is the difference to full list of solutions? Imagine we indeed have $M_{iso} \subseteq M : M_{iso} \cap \bigcup_{e \in H'(M')} e = \emptyset$. I. e. edges of $H(M')$ and $H(M_{iso})$ do not intersect (they form different connectivity components). Let us imagine we are solving the full problem, consider hypergraph $H(M') \cup H(M_{iso})$. After union editing, which can only add or remove vertices, these two components will be still disconnected, what means that the SET COVER instance will contain also two connectivity components in its hypergraph. It is obvious that every solution for this instance of SET COVER can be split into two parts, each of which explaining one component. If a solution part size for $H(M_{iso}$ is small and fixed, solutions for another component will represent a part of some solution for the whole hypergraph. It suggests that as a result of our limitation, we can only lose some proteins from the solutions, but we can't get any extra unwanted proteins.

To summarize, we have made the following assumptions:

- The mixture size is limited by constant, i. e. Set Cover size is limited by parameter c.
- The true mixture was altered by a limited number of k_0 insertions and l_0 deletions
- Proteins are observed at least partially, i. e. there are no entirely lost proteins (if so, we simply ignore them). This does not affect what we learn about other proteins.

2.2 Algorithm for Union Editing Enumeration

In this subsection we will give a simple branching algorithm that is able to solve the UNION EDITING problem.

Let us call a vertex v of some set M' complete in hypergraph $H = (V, E)$ (where not necessarily $M' \subseteq V$), if $\exists e \in E \, s.t. \, v \in e, e \subseteq M'$. Remember, that we are interested in sets M' that are a union of some subfamily of M_p, $p \in P_{db}$, let us call such sets *complete unions*. It follows from definition that M' is

complete union iff every vertex $v \in M'$ is complete. That immediately gives an idea of a branching algorithm: to guarantee that a particular internal vertex v is complete, we have to either remove v from M'; or choose some edge $e, v \in e$ and add all its external vertices (that were not in M'). Whenever we remove the vertex, we assume it was added by mistake, thus we should count this occurrence as a separate insertion. Same way, when we add the vertex, we consider it was deleted erroneously, and thus we should count it as a deletion. Therefore, throughout the branching process, we shall maintain counters for allowed insertions and deletions (k and l are given in the problem instance). The moment some counter turns to be negative, the current branch should be terminated.

Pseudocode of the algorithm is given below:

```
Input:   hypergraph H=(V,E),
   integers k, l (insertions, deletions),
   set of initial vertices M⊆V.

Find all complete unions M':
  M'⊆V, |M\ M'|≤k and |M'\ M|≤l.

begin
  FindCompleteUnion(H,k, l, M, M')
end

FindCompleteUnion(H, k, l, M, M')
begin
  if M=∅ and 0 ≤ kl
    output M'
    terminate
  end if
  while k≥ 0 and l≥ 0 do
    choose arbitrary vertex v in graph H
    B:=∅
    for all edges e∈ E incident to v
      A':={vertices incident to e} \ M
      FindCompleteUnion(H'(V,E), k,
       l−|A'|, M, M'∪A')
      B:=B∪{e}
    end for
    FindCompleteUnion(H'(V,E\B),
     k−1, l, M\{v}, M'\{v})
  end while
end
```

Note that we stop whenever we find any feasible solution, even if k and l are not exhausted. By doing that, we stick to the previously chosen approach: find only minimal solutions of the instance. But what would minimality mean in this case?

In the introduction section, we mentioned, that we want to use only information coming from the empirical spectrum. What means, if $\exists e \in E : e \cap \{v\} = \emptyset$, $e \subseteq M'$, i. e. if there is an edge in M' that is not explained by an initial set of vertices M, then M' is not a minimal solution. Note, that it can be obtained from solution that does not have such edges by several removals of vertices - one can say, that non-minimal solution assumes more deletions then necessary.

Imagine an opposite situation, that a solution adds more insertions, than it should be. Keeping in mind that all solutions are complete unions, we can conclude that all vertices which are removed in non-minimal solution, were complete (we remove only vertices from M, and if vertex $v \in M$ was not complete, then it should be either completed by other vertices or removed in every possible solution). Hence, we obtain another condition: none of the initially complete vertices should be removed.

Definition 1. *Solution M' of instance of* UNION EDITING $(H = (V,E), k, l, M)$ *is not minimal if* $\exists M''$ *— another solution, such that* $(M'' = M' \backslash e,\ e \in E,\ e \cap M = \emptyset)$ *or* $(M'' = M' \cup e,\ e \in E,\ e \subseteq M)$. *All other solutions are minimal.*

Theorem 1. *FindCompleteUnion(H, k, l, M, M) outputs solutions of (H, k, l, M) instance of* UNION EDITING *and among them lists all minimal solutions.*

We do not mention any upper bounds on running time. It can be shown that CLIQUE can be reduced to a special case of UNION EDITING, thus the problem is (not surprisingly) NP-hard. What means it will be hard to get optimistic bounds for the general case. For our inputs though the algorithm runs fast enough to forget about computational complexity (which itself is an interesting separate topic to discuss for this problem), and these results we are going to cover in the next section.

3 SIMULATIONS

The main question of the simulations was whether it is feasible to get a list of minimal solutions for regular problem instance in reasonable time. We are also interested in the false positives and in the recall (here, the rate of detected proteins from the initial protein set), as the practical motivation dictates us to maximize the latter and get rid of the former.

To answer these questions, we generated uniformly at random 10 initial protein sets of sizes 5, 10, 15 and 20. Then we obtained a theoretical mass spectrum from our database (see "Data" subsection) which we modified by deleting 1 ... 9 of the theoretical masses uniformly at random and then inserting 1 ... 9 masses, also chosen uniformly at random. Here our simulation approach significantly deviated from standard (well described in (He et al., 2009)), where random numbers are inserted instead of masses from the database and masses are modified by noise instead of being deleted. The noise rate is, of course, higher,

reaching 50 % of noise peaks. One can translate this procedure into ours by accepting masses within some sliding window: i.e. consider mass m observed if $\exists m' \in M_{all} : |v(m) - v(m')| < d$, where $v(m)$ represent the actual mass value and d is the sliding window parameter, M_{all} is the set of all known masses. This way we will get also the initial spectrum modified by few insertions and deletions but the distribution of these insertions/deletions will be different.

We did not exclude the masses outside of detectable range (usually around 500-5000 Da). Of course, it gave us more specificity, because we should have also removed them from the database, thus making protein theoretical spectra less distinguishable. However, it is rather an equipment problem, than computational, because if we lose elements and thus decrease the cardinality of sets in an instance of either SET COVER or UNION EDITING, they cannot become harder. And we still would list all the possible solutions. So the undetectable masses were kept to increase the computational challenge of the problem.

As soon as the empirical sample is generated, we apply an implementation of algorithm described in previous section to it. Because error bounds are in real cases unknown, we used maximal parameter and twice the maximal parameter as a bound. I. e. if 3 insertions and 0 deletions were introduced, we gave 6 insertions and 6 deletions as input. We also paid attention to the few insertions $(0 \ldots 3)$ case as it seem to describe best the real situation - noise can be filtered in many ways, including knowledge of the protein origin. E. g. if we know that the mixture contained only yeast protein, then masses corresponding to peptides from other organisms can be removed, etc.

Although the initial plan was to formulate outputs of UNION EDITING instances as instances of SET COVER problem and solve them by branching algorithm described in (Damaschke and Molokov, 2009), it turned out later that this is not the best idea. As the number of the obtained solutions (possible theoretical spectra) for UNION EDITING instances with high error bounds is too big, we would waste too much time on going through all of them. First we need to do some categorization of errors, as there are might be omnipresent masses, etc. - the same way it was with proteins. It is still an open question, how one can exactly do it, but it seems rather time consuming to find set covers for all possible theoretical spectra. According to running times, given in tables 1 and 2, excluding the time spent for sample preprocessing, each full run would take more than a day!

Instead, we solve the SET COVER only for the initial protein sets. More precisely, as an input we used: universe M (union of mass spectra of initial proteins), and sets $\{p \in P_{db} | M_p \cap M\}$ as covering sets. Parameter c is not known and should be selected by the investigator. We used value 20 for it.

We will cover the results of both algorithms in subsection "Performance".

3.1 Performance

For all drawn samples, SET COVER instances appeared to be trivial - they were solved in a matter of seconds. The reason behind it was that proteins had few shared masses and there were many unique masses - the ones that appear only in one protein among chosen candidates. Results for one of the samples are given in table 1.

The running times of implementation of algorithm from (Damaschke and Molokov, 2009), suggest that we can use it as a tool to solve SET COVER instances on a regular basis.

Table 1: Results of implementation of the branching algorithm for SET COVER enumeration.

$\|P\|$	c	Time	Recall
20	20	1.934s	85%
15	20	2.075s	80%
10	20	0.982s	80%
5	20	0.244s	80%

In this table, $|P|$ represents cardinality of the initial set of proteins (sample size), c represents a parameter, the bound for solution size, Time stands for running time and Recall is obtained as a rate of detected proteins in initial set of proteins. But what proteins are counted as detected? If we consider only ones that are present in all solutions as hits, then as was said in introduction, we can guarantee that every such protein was in initial set of proteins. Therefore, the number of false positives obtained by our method will be 0 (it is not provided in table). Recall, however, reduces. Although we list the initial set of proteins in the solution, not all it's proteins are omnipresent.

The following table summarizes results of the branching algorithm given in previous section.

Here "Sample" column contains sample ids, k and l are numbers of actual insertions and deletions, whilst k_0 and l_0 are bounds specified for them.

First part of the table demonstrates the dependency between running time and the number of initial proteins $|P|$. Although there were no insertions made at all, high bound for their number lead to high running time, which rapidly increases with growing sample size. Second part of the table proves that it is not the actual number of errors that is important for the running time, but the error bounds we provide. E. g.

Table 2: Results of implementation of the branching algorithm for UNION EDITING.

Sample	k	l	$\|P\|$	k_0	l_0	Time
1	0	9	5	9	9	0.172s
1	0	9	10	9	9	19s
1	0	9	15	9	9	1102s
1	0	9	20	9	9	3572s
1	0	4	15	8	8	1260s
3	6	6	15	6	6	181s
2	9	9	5	9	9	0.634s
2	9	9	5	18	18	39s

for 12 errors in sample 3, we assume only 12 (6 insertions and 6 deletions), compared to 18 (9 insertions and 9 deletions) in previous part of the table. Third part of the table gives the most optimistic results. If $|P|$ is sufficiently small, even with high error bounds, reaching 36 - approximately 20% of initial sample size, running time is within one minute range. That means that our approach can be used directly for classical PMF, where a low number of proteins in mixture is guaranteed by separation procedures.

3.2 Discussion

Results raise several questions, which are not yet answered. The first one and the most important would be: is it possible to reduce the running time for big initial protein sets? It is clear we will get some exponential function, but the base of exponent could be improved by intelligent branching rules. We may consider using techniques similar to ones used for solving HITTING SET problem in (Fernau, 2006). We may also save time by finding a way to browse through solution space (list of possible solutions for high error bounds reaches order of 10^4) of UNION EDITING.

Another good question is, how well the method behaves on real data. It is hard to assess what are the real numbers of insertions and deletions in regular experiments, but one could try solving the instance for known protein mixtures.

Finally, it might be good to check whether it is possible to reformulate problem into counting and thus try some approximation scheme for SCME. Although we can't get a good polynomial time approximation scheme (Lund and Yannakakis, 1994), some randomized approximation scheme might work in polynomial time.

3.3 Data

Protein database uses protein sequences from SwissProt version of 2005, with humans as species of origin. Theoretical spectra were obtained by simulating trypsin digestion of proteins with no miscleavages (what assumes that trypsin always cuts where it should and does not miss any markers). For computations we used regular machine with Intel Core2 Duo T5800 processor, 2.00GHz with 3GB RAM available.

4 CONCLUSIONS

Simulations show that the method can be applied to standard PMF setup with small protein mixtures to improve the results obtained by classical methods. There is also a perspective of application in shotgun proteomics, which considers bigger protein mixtures. It requires modifications of the algorithm for solving UNION EDITING that will reduce its running time. A more intelligent way of enumeration of outputs of the algorithm might also provide an insight to a faster solution.

ACKNOWLEDGEMENTS

Work was partially supported by Swedish Research Council (Vetenskapsradet), grant no. 2007-6437, "Combinatorial inference algorithms : parameterization and clustering". Author also would like to express gratitude to Swedish Institute for former support and funding, and to his supervisor, Peter Damaschke, for valuable input in research and studies.

REFERENCES

Aebersold, R. and Mann, M. (2003). Mass spectrometry-based proteomics. *Nature*, 422:198–207.

Clauser, K., Baker, P., and Burlingame, A. (1999). Role of accurate mass measurement (+/- 10 ppm) in protein identification strategies employing ms or ms/ms and database searching. *Anal. Chem.*, 71(14):2871–2882.

Damaschke, P. and Molokov, L. (2009). The union of minimal hitting sets : combinatorial parameterized bounds and counting. *J. Discrete Algorithms*, 7:391–401.

Fernau, H. (2006). Parameterized algorithms for hitting set: The weighted case. In *CIAC'06*, volume 41, pages 332–343.

He, Z., Yang, C., Yang, C., Qi, R. Z., Tam, J., and Yu, W. (2009). Optimization-based peptide mass fingerprinting for protein mixture identification. In *RECOMB'09*, pages 16–30. LNCS 5541.

Jensen, O. N., Podtelejnikov, A. V., and Mann, M. (1997). Identification of the components of simple protein

mixtures by high-accuracy peptide mass mapping and database searching. *Anal. Chem.*, 69(23):4741–4750.

Lund, C. and Yannakakis, M. (1994). On the hardness of approximating minimization problems. *Journal of the ACM*, 41(5):960–981.

Nesvizhskii, A. I. and Aebersold, R. (2005). Interpretation of shotgun proteomic data: The protein inference problem. *Mol. Cel. Proteomics*, 4:1419–1440.

Perkins, D. N., Pappin, D. J. C., Creasy, D. M., and Cottrell, J. S. (1999). Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis*, 20:3551–3567.

Samuelsson, J., Dalevi, D., Levander, F., and Rognvaldsson, T. (2004). Modular, scriptable and automated analysis tools for high-throughput peptide mass fingerprinting. *Bioinformatics*, 20(18):3628–3635.

Zhang, W. and Chait, B. T. (2000). Profound: an expert system for protein identification using mass spectrometric peptide mapping information. *Anal. Chem.*, 72(11):2482–2489.