Tagger: Enhance Database Search Tools with De Novo Sequencing Tags

by

Qi Tang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2017

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Tandem mass spectrometry (MS/MS) is widely used in proteomics nowadays to identify peptides and proteins from a sequence database. In a classic procedure of MS/MS protein identification, proteins are digested into short peptides by enzymes. Then, a tandem mass spectrometer is used to measure the tandem mass spectra for the peptides. Finally, the spectra are interpreted by computer software to identify the sequences of peptides and proteins. A traditional approach for protein identification is MS/MS search approach, it directly searches MS/MS spectra in a sequence database to find the best matching. However, regular methods become too slow when both the mass spectrometry data and sequence database sizes are large. Therefore, a speed boost for the database search is needed.

This thesis studies the possibility of using de novo tag search to improve traditional database search methods and proposes novel software named "Tagger". As a tag-based method, it utilizes the de novo sequencing results from Novor software as its input and performs approximate sequence matches in the sequence database. During the search, peptide tags are first matched with substrings of protein sequences in the database and then a protein scoring function is used to get the final score of a protein from those substrings. After the identified protein list is provided, other methods can treat it as a new database for peptide identification.

According to the test results, the search speed is significantly increased by the ability of indexing de novo sequence tags. Compared with X!Tandem, one of the fastest database search tool for protein identification, Tagger is about 9 times faster. For protein identification, the sensitivity of Tagger is also similar to that of X!Tandem. This is inconsistent with the common belief that a tag based search is inferior in terms of sensitivity. The combination of Tagger and X!Tandem provide the most accurate and sensitive protein identification result. In peptide identification tests, X!Tandem, as well as MSGFDB and another tag-based method InsPecT, are used for comparison. Thanks to Tagger's high accuracy of protein identification, the size of identified protein list is much smaller than the whole database so the search can be done significantly faster. On the other hand, the number of identified peptides grows greatly after combining the results of Tagger and the three programs.

Acknowledgements

I would like to thank all the people who made this possible.

Foremost, I would like to express my sincere gratitude to my supervisor Prof. Bin Ma for the continuous support of my Master study and research, for his patience, enthusiasm, and intelligence. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better supervisor and mentor for my Master study.

Besides, I would like to thank the rest of my thesis committee: Daniel Berry and Andrew Doxey, for their insightful comments and questions.

I thank my fellow labmates in Bioinformatics Group: Chenyu Yao, Jianqiao Shen, Lian Yang, Rong Wang, and Tiancong Wang, for the heated discussions, and for all the fun we have had in the last two years.

Last but not least, I would like to thank my family for supporting me spiritually throughout my life.

Dedication

This is dedicated to the one I love.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

In proteomics research, we often need to identify proteins and peptides from sequence databases. In a typical bottom-up approach, proteins digested by enzymes are measured with tandem mass spectrometry (MS/MS) to generate thousands to millions of MS/MS spectra. These spectra are compared to the peptides in a protein sequence database to find highly confident peptide-spectrum matches (PSM). Proteins are inferred from the identified peptides. In today's proteomics labs, this protein identification process is carried out daily, and many database search software packages are available. Some of the commonly used packages include Sequest [1], Mascot [2], PEAKS DB [3], Byonic [4], X!Tandem [5], ProteinProspector [6], Andromeda [7], MSGFDB [8], and JUMP [9].

Given the imperfect quality of the data, false positive identifications are inevitable. This promoted the development of the target-decoy method for result quality control [10]. In this method, the decoy database is searched together with the target sequence database. After the search is finished, the number of decoy matches are used as an estimation of the number of false target matches. The false discovery rate (FDR) is then calculated as the ratio between the estimated false target matches and the total number of target matches. Because of the way that each decoy protein is introduced as a separate entity in the database, the target-decoy method is robust for estimating the protein FDR. However, for estimating the peptide FDR, the target-decoy method is incompatible with some of the search engines [11]. However, many of the problems of target-decoy can be fixed by the decoy fusion method [12] that is a slight modification of the target-decoy method.

When a sequence database is unavailable, one will have to rely on the de novo sequencing method to derive the peptide's sequence from the MS/MS spectrum directly. De novo sequencing tools include PEAKS [3], Novor [13], PepNovo [14], and pNovo [15]. A more detailed review of de novo sequencing tools can be found in the review articles [16, 17].

A traditional way to perform database search is the MS/MS search approach, in which MS/MS spectra are searched directly in a protein sequence database to find the best match. In the past 20 years, many efforts have been made to improve the accuracy and sensitivity of the database search software. This involves the use of sophisticated scoring functions that can better separate the true and false matches; and the consideration of variable post-translational modifications (PTM) and nonspecific enzyme cleavages. However, these efforts of making database search more accurate and sensitive also make the search slower. Meanwhile, the increase in both mass spectrometry data size and the sequence database size make it urgent to consider new approaches that can greatly improve the database search speed.

1.2 Research Objectives and Contributions

A promising approach to speed up database search is to use de novo sequence tags to assist the database search. For example, PEAKS DB [12] uses de novo sequence tags to select a shortlist of proteins from the database, and only carry out the database search on this shortlist. However, as the de novo sequencing speed used to be slower than database search, the speed benefit was bounded by the speed of de novo sequencing. For such a reason, most tag-based search algorithms are intended to search for mutated or modified peptides. These include FASTS [18], MS-Shotgun [19], MS-BLAST [20], SPIDER [21], GutenTag [22], and InsPecT [23]. It is widely known that when many variable PTMs are considered, database search becomes unacceptably slow and these approaches often require higher quality MS/MS spectra to make a confident assignment. However, for database search without an excessive numbers of PTMs, the benefit of employing a fully tag-based approach has not been studied.

With the recent availability of the Novor software for de novo sequencing, de novo sequencing becomes significantly faster than database search [13]. Therefore, the aforementioned concern about the extra time spent on de novo sequencing has been removed. However, there is a loss-of-sensitivity with the use of a tag-based approach for database search without an excessive numbers of PTMs. It is very likely that a spectrum's quality is barely enough to identify the peptide from a database, but not good enough to produce a long de novo sequence tag. If a long sequence tag is used to filter the database sequences,

then such spectra would not be identified. Particularly for this concern, the JUMP software [9] used tags of length 1 to filter the database sequences. The use of length 1 tags helped JUMP to improve its database search accuracy. However, a sequence tag of length 1 does not provide a significant gain in speed over the standard database search approach.

The rise of proteogenomics creates the need to search the mass spectrometry data against a DNA sequence database. In theory, one can translate the DNA database into a protein database with the genetic code, and identify proteins from it. However, the size of the protein database from such translation is significantly larger than a regular protein database. For example, the database of human Expressed Sequence Tag (EST) in NCBI GenBank contains 9×10^6 sequences while only 5.4×10^5 in the Uniprot database. This trend of searching proteomics data in a genomic database made it necessary to develop a general method to significantly speed up today's database search tools.

This thesis studies the possibility of using de novo tag search to improve traditional database searches and develops the Tagger software for this purpose. We take an approach different from that of some of the earlier works. Tagger focuses on the tag search step and leaves the de novo sequencing and peptide-spectrum matching steps to other existing software tools. This makes it possible to use Tagger to improve any existing database search tools. Given the wide adoption of a large number of different existing database search tools in the proteomics community, this flexibility is critical. In return, this flexibility allows Tagger to take advantage of the newest developments in de novo sequencing. In particular, the availability of the rapid de novo sequencing software, Novor, is pivotal for making the Tagger approach practical.

The novel contributions of the paper include: (1) a rigorous scoring function for evaluating tag-peptide matches, (2) an accurate scoring function to score proteins based on tag-peptide matches, and (3) a general strategy to use Tagger to improve other existing database search tools without changing them. The Tagger software can be used also independently of other database search tools for peptide and protein identification. Benchmark experiments were conducted to evaluate the Tagger approach by integrating Novor and Tagger for tag generation and tag search, while peptide-spectrum matching was done by X!Tandem, InsPecT, and MSGFDB, respectively. The results demonstrate that this approach not only improves the X!Tandem tool alone, but also outperforms the InsPecT and MSGFDB tools that integrate all the tag generation, tag search, and peptide-spectrum matching steps in one program.

1.3 Thesis Overview

Chapter 2 reviews the fundamental knowledge of protein and DNA, as well as the basics of mass spectrometry and de novo sequencing. Related research on tag-based database search are presented in Chapter 3. The detailed algorithms for searching peptides and proteins are shown in Chapter 4. The experiments are presented in Chapter 5. Finally, the conclusion and discussion are in Chapter 6. The Appendices provide some supply materials including the tests we did for choosing parameters and the proof of a lemma.

Chapter 2

Background

2.1 DNA and Protein

Deoxyribonucleic acid (DNA) exists in every nucleus of an organism, it contains the whole genetic information of the creature. DNA is made up from 4 different bases (nucleotides), adenine (A), thymine (T), guanine (G) and cytosine (C), which is true for all the creatures on the earth that contains DNA. DNA has a double-stranded structure, the bases on one strand of DNA form base pairs with a second strand of DNA to form the double helix. But the base pairs that can be formed have limitations; adenine (A) can only form a base pair with thymine (T) and guanine (G) can only form a base pair with cytosine (C). So when we know the sequence of bases on 1 strand of DNA, we also know the sequence of bases on the other strand of DNA [24].

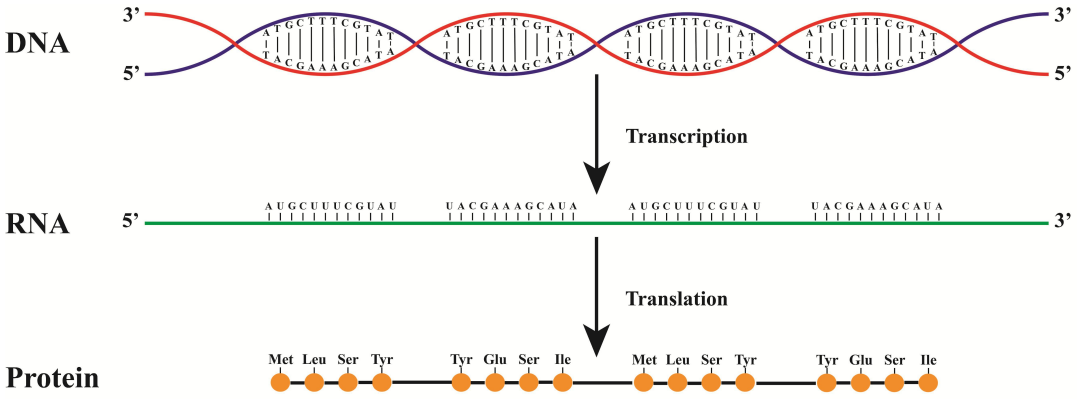


Figure 2.1: An example that shows the procedure from DNA to Protein [24].

A protein is made from amino acids, these form a strand. There are total 20 different amino acids. A protein strand is derived from DNA after two steps: transcription and translation, which is shown in Figure 2.1. During transcription, messenger RNA is made from DNA. RNA, short for Ribonucleic Acid, is synthesized in the nucleus and is very similar to DNA. The synthesis of RNA also involves the use of bases, but in RNA synthesis uracil (U) is used instead of thymine (T). The sequence of RNA corresponds to the sequence of DNA from which the RNA is synthesized. Then in translation step, proteins are made from mRNAs. One amino acid is added to the protein strand for every three bases in the RNA, which is called codons. Figure 2.2 shows the table of translation rules.

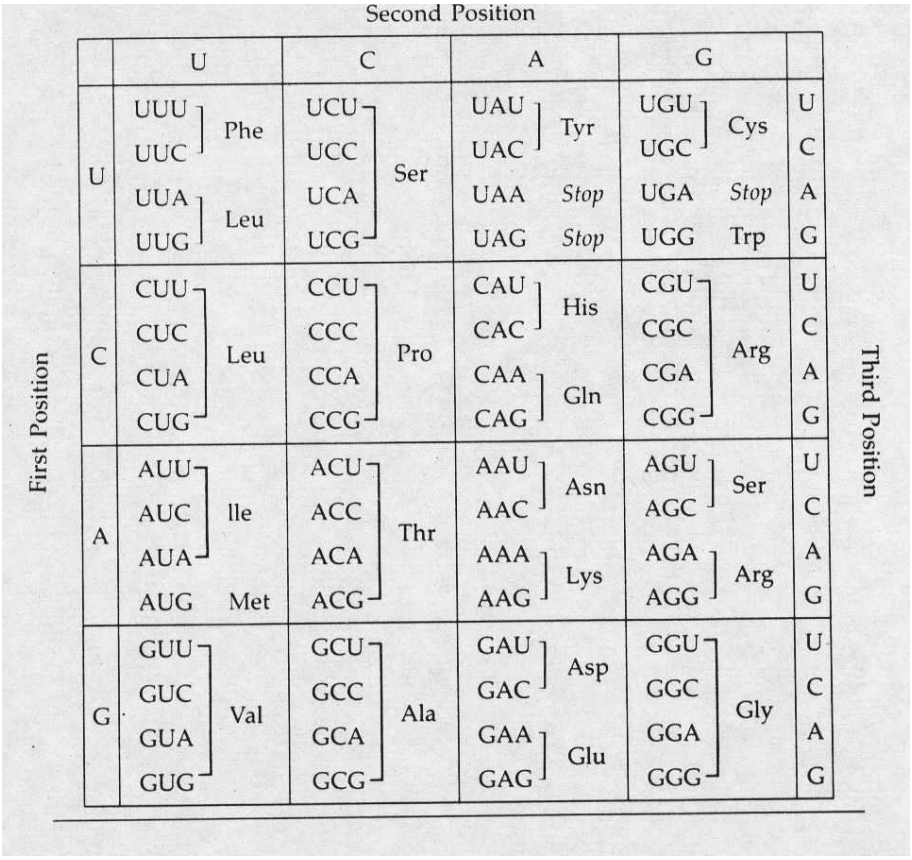


Figure 2.2: The codon table that shows the rule of translation [24].

2.2 Basics of Mass Spectrometry (MS)

Tagger doesn't actually deal with mass spectra while searching databases, they are taken by Novor software during tag generation step. We only introduce the basic idea of MS in this section.

2.2.1 Mass Spectrometry (MS)

Mass spectrometry (MS) is an analytical technique that ionizes chemical species and sorts the ions by their mass to charge ratio. In other words, a mass spectrum measures the

masses within a sample. Mass spectrometry is used in many different fields and is applied to pure samples as well as complex mixtures. The mass is usually measured in Dalton (Da), which is 1/12 of the mass of a carbon atom.

A mass spectrum is provided by a mass spectrometer, which is a plot of the ion signal as a function of the mass-to-charge ratio. These spectra are used to determine the elemental or isotopic signature of a sample, the masses of particles and of molecules, and to explain the chemical structures of molecules, such as peptides, proteins, and other chemical compounds. A simple sample of mass spectrum is shown in Figure 2.3

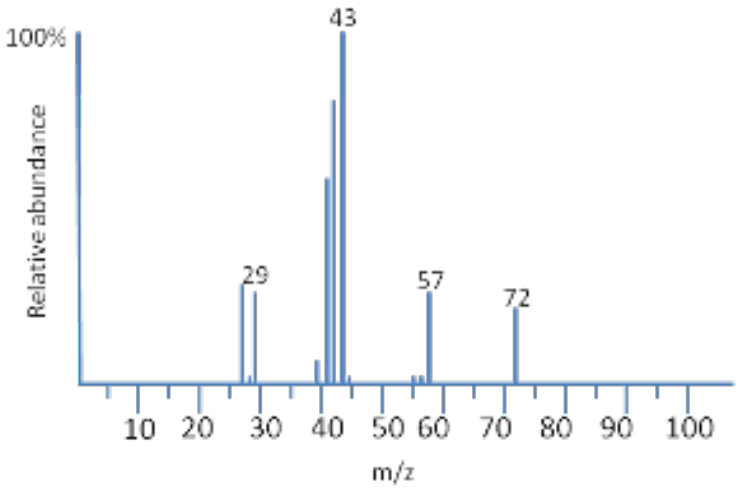


Figure 2.3: A sample of mass specturm.

In a typical MS procedure, a sample is ionized, for example by bombarding it with electrons. The sample's molecules may break charged fragments in this way. These ions are then separated according to their mass-to-charge ratio, typically by accelerating them and subjecting them to an electric or magnetic field because ions of the same mass-to-charge ratio will undergo the same amount of deflection [25]. The ions are detected by a mechanism capable of detecting charged particles, such as an electron amplifier. Results are then displayed as spectra of the relative abundance of detected ions as a function of the mass-to-charge ratio. The atoms or molecules in the sample can be identified by correlating known masses to the identified masses or through a characteristic fragmentation pattern [26]. Figure 2.4 shows a typical MS procedure.

There are many types of ionization methods used in mass spectrometry methods. Electron ionization (EI) is one of the most classic methods. It is done by volatilizing the sample

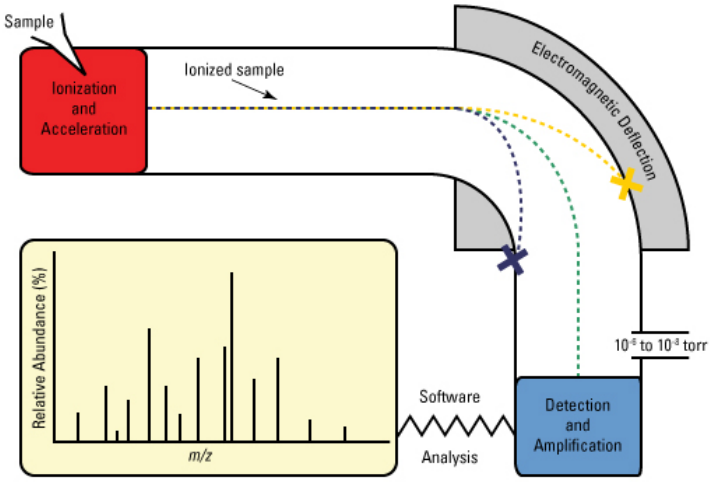


Figure 2.4: A typical MS procedure [27].

directly in the source that is placed in a vacuum system and then the gas phase sample is bombarded by an electron beam. This method was first proposed in 1918 by Sir Arthur J. Dempster [28]. EI is a reproducible method that can provide structural information of the sample. But the sample used by EI must be volatile and stable, and the molecular ions are often missing.

Electrospray ionization (ESI) is now becoming the most popular ionization technique. The sample is sprayed across a high potential difference from a needle into an orifice in the interface. Heat and gas flows are used to desolvate the ions existing in the sample [29]. ESI is best for multiple-charged ions and compatible for MS/MS. But it is not good for uncharged ions and the ion current is relatively low.

Matrix Assisted Laser Desorption Ionization (MALDI) is a technique of ionization in which the sample is bombarded with a laser. The matrix chromophore absorbs and distributes the energy of a laser, thus produced a plasma, vaporates and ionize the sample [30]. MALDI is a rapid method that works well for singly charged ions but is has difficulty doing MS/MS [31].

2.2.2 Tandem Mass Spectrometry (MS/MS)

Tandem mass spectrometry, also MS/MS, involves multiple steps of mass spectrometry selection, with some form of fragmentation happening in between the stages. In a tandem mass spectrometer, ions are formed in the ion source and separated by the mass-to-charge ratio in the first stage of mass spectrometry (MS1). Ions of a particular mass-to-charge ratio (precursor ions) are selected and fragment ions (product ions) are created by collision-induced dissociation, ion-molecule reaction, photodissociation, or other processes. The resulting ions are then separated and detected in the second stage of mass spectrometry (MS2) [32]. Figure 2.5 shows the schematic of MS/MS. There are many different methods

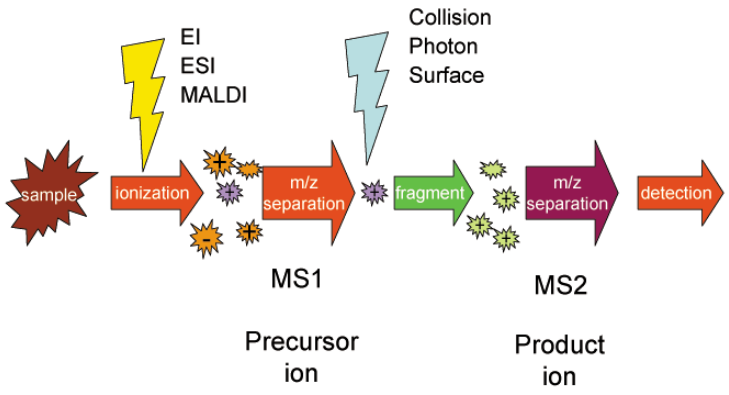


Figure 2.5: Schematic of tandem mass spectrometry [32].

for molecule fragmentation. Collision-induced dissociation (CID) is a mass spectrometry technique for molecular ions fragmentation in the gas phase [33]. The molecular ions are usually accelerated to high kinetic energy by electrical potential, then they can collide with neutral molecules. Some of the kinetic energy is converted into internal energy in the collision, which will cause dissociation occurring at amide bonds and the fragmentation of the molecular ion into smaller pieces. Then a tandem mass spectrometer can be used to analyze the fragment ions. CID works better for small, low-charged peptides.

Electron-transfer dissociation (ETD) is a method of fragmenting multiply-charged gaseous macromolecules in a mass spectrometer between the stages of tandem mass spectrometry (MS/MS)[34]. ETD is used extensively with polymers and biological molecules such as proteins and peptides for sequence analysis. It can transfer an electron and then cause peptide backbone cleavage into c- and z-ions while leaving labile post-translational modifications

(PTM) intact. The technique only works well for higher charged peptide or polymer ions ($z > 2$). However, relative to CID, ETD is advantageous for the fragmentation of longer peptides or even entire proteins, which makes it important for top-down proteomics.

Another alternative type of fragmentation method is the beam-type CID or high-energy collision dissociation (HCD). Unlike the traditional ion trap CID, the fragmentation pattern of HCD uses higher activation energy and shorter activation time [35]. HCD generates b- and y-type fragment ions as well. HCD has no low mass cut-off limitation and is able to provide high mass accuracy MS2 spectra. It has been successfully applied for de novo peptide sequencing and providing more detailed ion series. Also, certain diagnostic ions specific for HCD could be recognized for PTMs identification in PTMs studies [36].

2.3 De Novo Peptide Sequencing

De novo sequencing in proteomics means the process of deriving peptide sequences from tandem mass spectra without the assistance of a sequence database [16]. De novo sequencing is frequently used in proteomics research to require new peptides from MS/MS data. It is helpful to sequence an entire protein and assist database search analysis [12]. There are many software packages available for de novo sequencing: PEAKS contains a full pack of functions including de novo sequencing, database search, PTM identification, homology search and quantification in data analysis [3]; CycloBranch is a tool for de novo sequencing of non-ribosomal peptides (i.e. linear, cyclic, branched and branch-cyclic) [37]; NovoHMM is another method of de novo sequencing which use hidden Markov model (HMM) in a Bayesian framework as a new way to solve the problem [38]; PepNovo uses a probability network which reflects the chemical and physical rules of the peptide fragmentation [14]. A more detailed introduction to de novo methods can be found in [16].

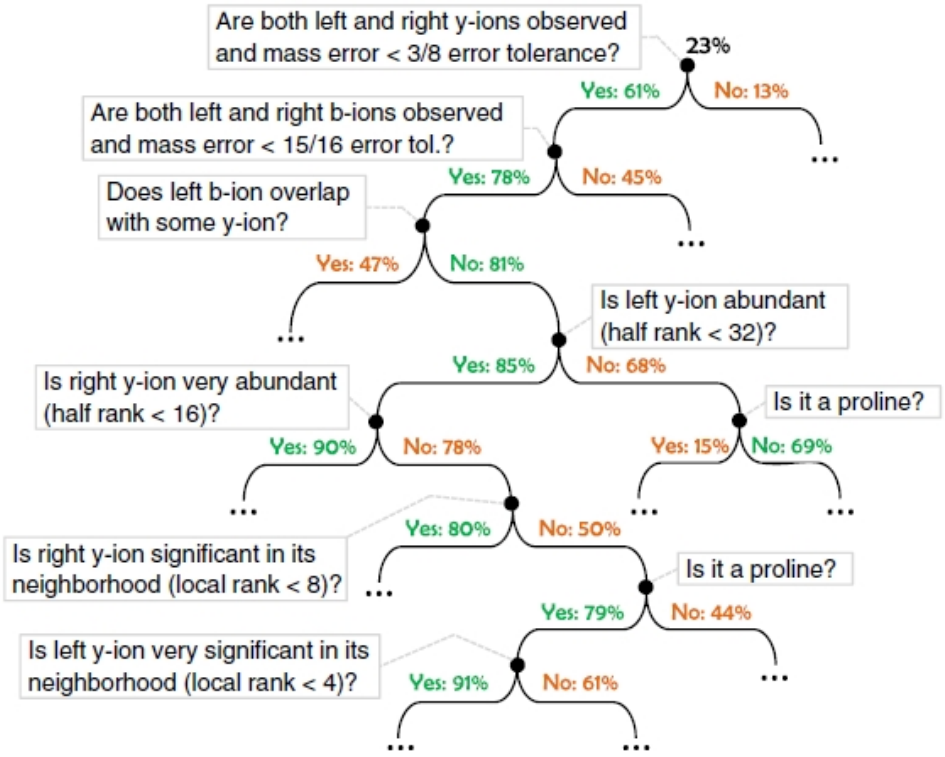


Figure 2.6: A small portion of decision tree learned by Novor [13].

Although there are different kinds of de novo sequencing methods in the market, the speed of de novo sequencing is not satisfying. In a typical proteomics progress, de novo sequencing with today's software usually takes longer than database search, which is not acceptable as we want to use the result of de novo sequencing as a way to reduce running time. Recently, a de novo sequencing named Novor is proposed, it is based on two decision trees built with machine learning [13]. Figure 2.6 shows part of the decision tree learned by the machine learning algorithm of Novor. Novor is able to improve de novo speed greatly and retain similar accuracy as other tools. More than 300 MS/MS spectra can be sequenced by Novor on a laptop computer per second, which opens the possibility to develop a fast speed protein identification method using de novo results.

Chapter 3

Related works

Although tag-based approach for database search doesn't have a history as long as the classic MS/MS approach. Many great pioneer works are already there and inspire us a lot. We will review some of them in this section.

3.1 FASTS, MS-Shotgun and MS-BLAST

FASTS [18], MS-Shotgun [19], and MS-BLAST [20] are three tag search programs modified from the general purpose homology search programs: FASTA [39], Shotgun [40], and BLAST [41]. Compare with their "parent" programs, they make an adjustment of parameters and consider multiple tags at the same time to improve the accuracy of protein identification. Figure 3.1 shows a workflow of FASTS. In the figure, there is another program FASTF, it is designed by the same author as FASTS and searches the database with mixed peptide sequences. Despite the development compared with their "parents", none of the three methods can deal with de novo sequencing errors. These errors can happen everywhere, which may affect scoring functions and result in wrong answers.

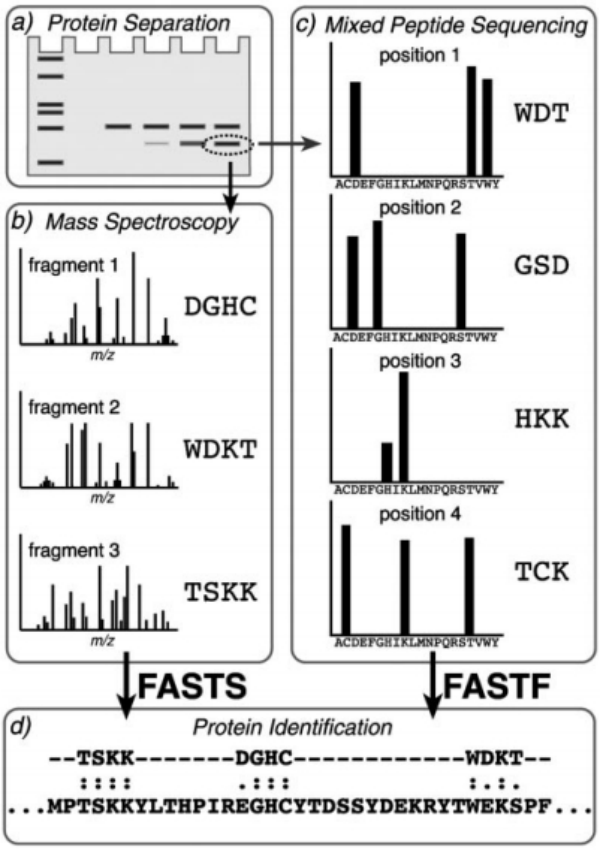


Figure 3.1: The workflow of FASTS [18].

3.2 OpenSea

Opensea is a software package that can match error-containing sequence tags with a database to locate the modified or homologous proteins [42]. Homologous mutations and de novo sequencing errors are both considered during the search. Nevertheless, OpenSea doesn't allow sequencing errors and mutations to occur ar the same positions. And it requires that the de novo sequencing errors have a minimum length of 3. Figure 3.2 shows the mass-based alignment scheme of OpenSea. The matches are done using a simple greedy algorithm so not all optimal matches can be found.

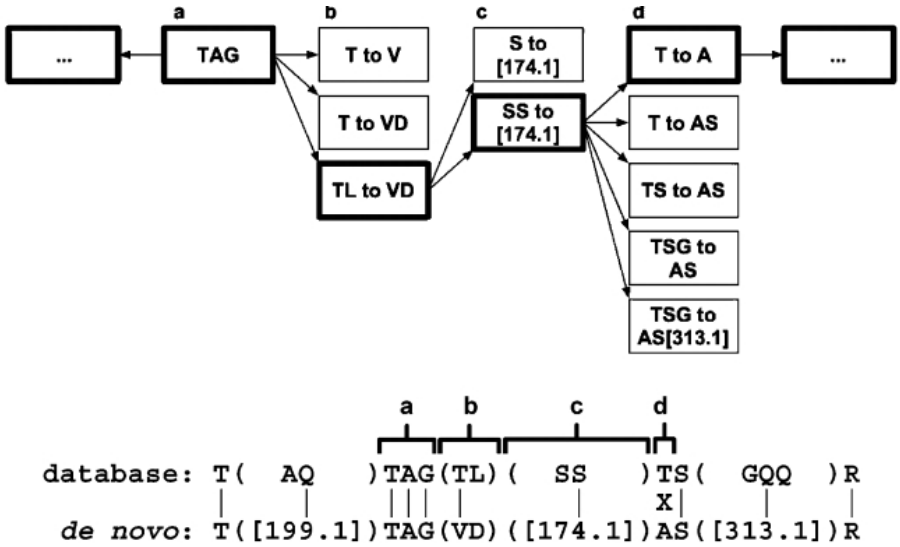


Figure 3.2: The mass-based alignment scheme of OpenSea [42].

Also, in the research paper of OpenSea, the authors demonstrated that de novo approach can get better results than MS/MS search approach for protein identification, which strengthen our faith in designing a tag-based software.

3.3 InsPecT

The InsPecT program [23] is another classic use of de novo tags. Different from the other tag search programs, InsPecT's goal is to perform unrestrictive PTM search on the exact sequence database of the studied organism. Tag matching is only used as a filtration step of the Inspect algorithm, while the scoring of the identified peptides is still based on the original spectrum. Similar ideas are also employed in the PEAKS DB [12] and JUMP [9] software. All of the three tools use their own algorithms to generate the de novo sequence tags, and their own scoring functions to evaluate the peptide-spectrum matches. This prevents the use of these tools to improve other traditional database search tools; as well, these software tools cannot take advantage of the newly improved de novo sequencing software made available by the research community.

3.4 SPIDER

SPIDER is a sequence tag searching method which takes both de novo sequencing errors and homology mutations into consideration [21]. Unlike OpenSea, it allows the errors and mutations to appear at same positions of the sequences. Moreover, SPDIER can be extended directly to identify proteins with PTMs. It also has the ability to "guess" the real sequence with most probability by combining the information given by the homolog of the real sequence and the partially correct sequence.

SPIDER's alignment model is among three sequences, the sequence tag, the real peptide sequence, and the homology of the real sequence in a database. It uses a dynamic programming algorithm to find the alignment and offers an efficient way to find the optimal alignment. Although SPIDER works fine with normal-sized data, running time still becomes a problem when dealing with huge data files.

Chapter 4

Methodology

4.1 Method Overview

The overall procedure from tandem mass spectrum to identified peptides and proteins includes the following five steps: tag generation, tag search, protein shortlisting, peptide-spectrum matching, and result combination. Our Tagger software takes care of the tag search, protein shortlisting, and result combination steps; whereas the tag generation and peptide-spectrum matching steps are performed with other software tools. Tagger provides the simple mechanism to support the changing of different software tools for tag generation and peptide-spectrum matching. The functions of each step are summarized in the rest of this section, with details provided in the next few sections.

Step 1. The *tag generation* step computes a de novo sequence for each tandem mass spectrum in the data file. These de novo sequences are called tags and provided to Tagger with a simple text file in the csv format.

Step 2. In the *tag search* step, Tagger searches a large protein sequence database to find the best matching database peptide for each tag. A nontrivial index structure is built to facilitate the efficient searching. A rigorous but efficient scoring function is proposed to evaluate each peptide-tag match.

Step 3. In the *protein shortlisting* step, the proteins that contain at least one tag matches are scored and sorted according to their scores. A short list of proteins are selected and output as a FASTA file. The second and third steps only rely on the sequence tags and do not need the original spectrum data. The first three steps are very efficient and take minimum amount of time.

Step 4. The *peptide-spectrum matching* is conducted by calling separate database search software. The original spectrum data, as well as the short list of proteins in the FASTA file computed in step 3, are provided to the database search software as input. Since the FASTA file is usually significantly smaller than the original database, this step is more efficient than searching the whole database with the same software.

Step 5. The *result combination* step combines the peptide and protein identification results made in the third and fourth steps together. As the results are already provided in former steps, no additional computation is needed, this step can be done in several seconds.

The fourth step is usually the most time-consuming step. If desired, the procedure can stop at any of steps 3, 4, and 5. For example, if the process stops at the third step, the tag matches and protein shortlists are used as the peptide and protein identification results. However, the search quality increases if the last two steps are included.

4.2 Tag Generation

In theory, any de novo sequencing tool can be used to generate tags for Tagger. However, the speed requirement of de novo sequencing makes the Novor software the only choice at the time being. Novor was reported to be at least 10 times faster than other de novo sequencing tool. With a speed of more than 300 spectra per second on a personal computer, Novor makes the time complexity of this tag generation step negligible. Thus, in this work, the free Novor software (Version 1.05, Rapid Novor Inc., Waterloo, ON, Canada) is used for de novo sequencing. Novor reports only one de novo sequence for each spectrum in a plain text file. This file is parsed to retrieve the spectrum scan number, de novo sequence, as well as Novor's percentage score on the sequence and on each individual amino acids, respectively. The de novo sequences with score less than 10% are removed.

4.2.1 Tag Preprocessing

De novo sequencing often cannot disambiguate two possible amino acids (or modified amino acids) that have the identical or similar mass values. Also, the output of Novor can't be taken by our method directly because it shows possible or fixed PTMs in sequences while standard protein databases don't have PTMs in them. Therefore, a simple preprocess step is performed to unify the amino acids with very similar mass values. All amino acids with similar mass are substituted with a representative amino acid. Table 4.1 shows the list of substitutions Tagger makes.

Table 4.1: Amino acids with similar mass are substituted with the bolded amino acid code.

Nominal mass	Amino acids
113	**L**, I
128	**Q**, K
131	**F**, Oxidized M

4.3 Tag Search

After modifying the input, the next step is to match those tags with peptide sequences in the database by assigning matching score to tag-peptide matches. As the database might be very large, it will be way too slow to search every protein in the database to find the match. So we only calculate matching score for substrings of protein sequences that are similar to the de novo tag, which we called candidates. First, we use a seeding method to find match candidates. The candidates are then divided into different blocks and the matching scores are calculated based on the probabilities of the blocks.

A tag with its highest-scored peptide sequence is called a final match. There may exist several final matches for a tag because the highest score can be shared by more than one peptides and the same peptide may appear many times in the database.

4.3.1 Finding Candidates

A seeding strategy is used to help quickly identify the potential tag-peptide matches. A match between a k-mer of a de novo tag and a k-mer of the sequence database is called a *hit*. A hit suggests that there is a potential tag-peptide match nearby and will trigger a more rigorous examination. Hits can be found very efficiently by utilizing a hash table.

The hash table is only dependent on the input tag list, so it can be built before the search begins. The procedure of building the hash table is as following:

Buliding the Hash Table

input: a list of tags L and an integer k
output: a hash table that contains the indexes of the k-mers

Set the hash table to an empty table H
For a *tag* in L
    For a *kmer* in *tag*
        If H doesn't contain *kmer* **Then**
            Traverse L to find all indexes of *kmer*
            Add *kmer* and its indexes to H
        End If
    Next *kmer*
Next *tag*

As shown above, all the k-mers from the query tags are indexed in the hash table. Then each k-mer in the sequence database is used to query the hash table to find hits. The use of a hash table to find hits has been well studied in the homology searching area [43, 16]. The time complexity of finding all the hits is $O(m + n + K)$, where m is the total length of the query tags, n is the total length of the database sequences, and K is the number of hits.

Once a hit is found, a hit extension procedure is triggered. Suppose a hit is found for position i of a de novo tag T, and position j of a database sequence S. Tagger linearly examines the amino acids immediately after position j, until it finds a substring $S[j + 1, \ell]$ such that the total residue mass is approximately equal to the total residue mass of $T[i + 1, |T|]$. Here $|T|$ indicates the length of T. If such ℓ cannot be found, then the hit is regarded as a random match and discarded. If successful, the same procedure is repeated by growing the hit to its left. Figure 4.1 illustrates a successfully extended hit. A successfully extended hit provides a tag-peptide matching candidate.

4.3.2 Improving Seeding Sensitivity

In our seeding strategy, we require the k-mer to be matched with exactly the same k-mer. This helps reducing the searching space but has its own disadvantages. Due to the imperfect spectrum quality, we can't avoid errors in de novo sequencing results. In addition,

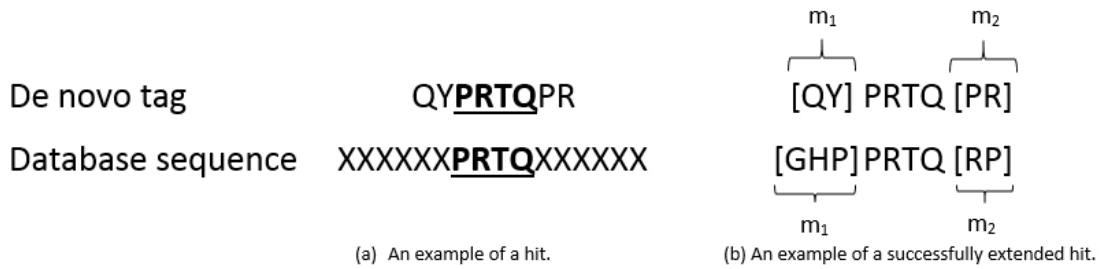


Figure 4.1: Two simple examples of a hit and extended hit.

it is commonly found in de novo results that a block with certain mass is substituted or permutated into another block because it is hard for MS/MS to distinguish blocks with very similar mass values.

To address this problem, we need to provide better sensitivity for the search, which can be done by decreasing the value of k in our seeding method. According to the definition, if two sequences have a k-mer hit then they also have a hit for any k' less than k. That is because any subsequence of a shared sequence is also shared, which means the search spaces of lesser k contain those of larger ones, so basically the search results using larger k won't be better than lesser ones.

On the other hand, we must consider the running time of the method. For example, decrease k by 1 will make the search space 20 times larger because one less amino acid need to be matched and there are total 20 kinds of amino acids. Considering both accuracy and speed, we came out with the idea of "neighbor table" to make a tradeoff.

The main idea of the neighbor table is using an approximate seeding strategy to compress the search space while not affecting the results much. More specifically, the matching of a k-mer and a k'-mer are regarded as a hit if they have the same total residue mass, and share at least j identical amino acids when aligned together. A neighbor table is used to facilitate the efficient finding of the hits under this definition. By adjusting the values of k, k' and j, one can find the sweet point for balancing search speed and sensitivity.

This strategy is to tolerate the possible de novo sequencing errors in the de novo sequence tags. It's common in de novo sequencing that some 2 or 3 mers are replaced by other ones of the same mass. By the definition of the neighbor match, we can tolerate those errors if they don't show up too much. Figure 4.2 shows the updated workflow for searching potential matches after applying neighbor tables. Instead of being searched directly in the tags, every k'-mer in the database is searched against the neighbor table

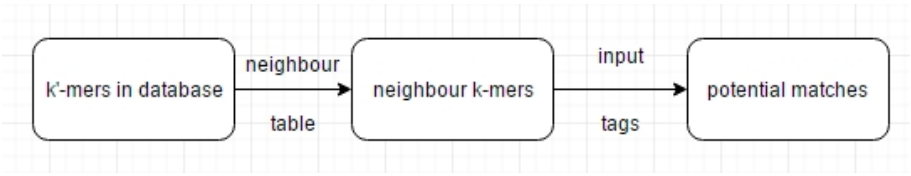


Figure 4.2: The workflow for finding potential matches using neighbor table.

to find their neighbors, then the neighbors are used as seeds to search the input tags for potential matches.

We must commit that building a neighbor table is time-consuming because every k'-mer is searched with all 20^k k-mers. However, neighbor tables are independent of both the input tags and protein sequence databases so we don't have to build it repeatedly. Actually, every neighbor table represents a search strategy, we store different neighbor tables in text files and use them for different purposes. As a neighbor only need to be created once, we can always do it in spare time so the running time won't be a problem.

4.3.3 Tag-Peptide Matching Score

The candidate finding step will find many database peptide matches for each query de novo tag. These candidate matches are scored to find the best matching peptide. For each tag-peptide match, the probability that such a match can occur randomly in the sequence database is first calculated with the procedure described in the following.

Suppose $S = a_1a_2 \dots a_n$ is a peptide sequence. Let $m(a)$ denote the mass of an amino acid residue a. The total residue mass of S, denoted by $m(S)$, is calculated by $m(S) = \sum_{i=1}^{n} m(a_i)$. The sequence consists of the k residues, $a_1a_2 \dots a_k$, are called the length-k prefix of S. In particular, the 0-th prefix is an empty sequence and the n-th prefix is the complete sequence S. The total mass of the first k residues, $m_k(S) = \sum_{i=1}^{k} m(a_i)$, is called the k-th prefix mass of S.

Suppose $T = a_1a_2 \dots a_n$ is a de novo sequence tag computed from a spectrum, and $P = b_1b_2 \dots b_{n'}$ is the real peptide sequence for the same spectrum. Since most (if not all) de novo sequencing software tools use the spectrum's precursor mass as a constraint, we require that $|m(T) - m(P)| \leq \Delta$. Here $\Delta > 0$ is a user-specified parameter that corresponds to the mass spectrometry instrument's precursor ion mass error tolerance. Due to the imperfect data quality and errors in the de novo sequencing, T and P may not match exactly. However, most of the errors, if not all, are the *mass gap errors* [16]. Each

mass gap error replaces a block of adjacent amino acids $b_j b_{j+1} \dots b_{j'}$ on the peptide P by a few other amino acids $a_i a_{i+1} \dots a_{i'}$ that have approximately the same mass.

The tag T and peptide P can be aligned together according to their prefix masses to find out the mass gap errors. The concept is fairly intuitive and Figure 4.3 illustrates such alignment with an example. More precisely, two prefix masses $m_i(T)$ and $m_j(P)$ are called *matched*, if (1) $|m_i(T) - m_j(P)| \leq \delta$, and (2) at least one of a_i and a_{i+1} has amino acid confidence score $\geq 5\%$. Here $\delta > 0$ is a user-specified parameter that corresponds to the instrument's fragment ion mass error tolerance. The majority of today's mass spectrometry instruments have $\delta \geq \Delta$. Suppose $0 = i_1 < i_2 < \ldots < i_k = n$ and $0 = j_1 < j_2 < \ldots j_k = n'$ are the indices of all the matching prefix masses of T and P, respectively. Then the alignment between T and P are divided into $k-1$ blocks by the k indices. If both sequences T and P contribute one amino acid in a block and the two amino acids match, the block is called an *exact matching block*. Otherwise, the block is called a *mass gap block*. The number, types, and lengths of the blocks provide a quality assessment of the alignment. Intuitively, the alignment in Figure 4.3(b) has a better quality than the alignment in Figure 4.3(a). This intuition can be formalized as follows.

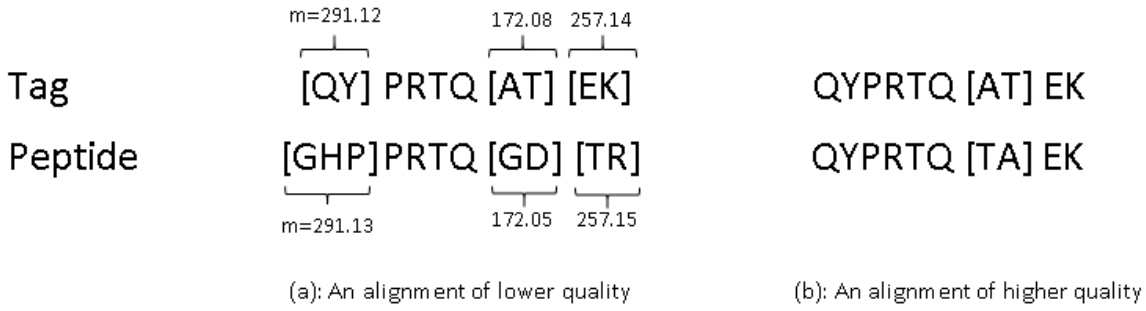


Figure 4.3: An example of tag-peptide matches and the blocks.

Consider a block consisting of two partial sequences t and p, from the tag T and a random peptide P from the database, respectively. If the block is an exact matching block consisting of a single amino acid x. The block is significant in the sense that a random sequence provides the same amino acid x at this position. The probability that this event happens is the amino acid frequency of x in the database, denoted by $p(x)$.

If the block is a mass gap block, let $m(t)$ denote the total residue mass of t. The block is significant in the sense that the mass of a random sequence p matches $m(t)$. Let $x_1 x_2 \dots$

be a randomly generated infinitely long amino acid sequence. Define

$$P(m) = Pr\left(\text{there is } i \text{ such that } |m - \sum_{j=1}^{i} m(x_j)| \leq \delta\right).$$

The value of $P(m)$ is calculated with the following recurrence relation:

$$P(m) = \begin{cases} 0, & \text{if } m < -\delta, \\ 1, & \text{if } -\delta \leq m \leq \delta, \\ \sum_{\text{every amino acid } x} p(x) \times P(m - m(x)), & \text{otherwise.} \end{cases} \tag{4.1}$$

Notice that the calculation of $P(m)$ only requires the frequency $p(x)$ for each amino acid x. This is easily computed for each protein sequence database. Therefore, Tagger precomputes $P(m)$ and store the values in memory for later use.

Therefore, given an alignment between T and P. The significance of an exact matching block with a single amino acid x is defined as $p(x)$, and the significance of a mass gap block is defined by $P(m(t))$, where t is the portion of tag T in the block. Suppose the significance of all blocks are $p_1, \ldots, p_k$. Then the score of the peptide-tag match is defined by

$$score(T, P) = -\log_2\left(\Pi_{i=1}^{k} p_i\right) = \sum_{i=1}^{k} -\log_2 p_i. \tag{4.2}$$

We call this score as the *bit score*. It is the basic version of tag-peptide match score and only needs some minor changes for the final use.

As the bit score is based on the alignment between T and P. We first need to ensure the existence of mass gap blocks and exact residue matches in a match candidate before we move on to find the alignment.

Lemma 1.

If two peptide sequences have the same mass, they can always be divided into n mass gap blocks and m exact residue matches, where $n + m > 0$.

The proof of *Lemma 1* can be found in AppendixC. From the hit extension procedure, it is obvious that if a hit is successfully expanded, the tag and peptide must have approximately the same mass. So for every tag-peptide match candidate, we designed an algorithm to find mass gap blocks and exact residue matches, which is shown below in pseudocode.

Mass Gap Blocks and Exact Residue Matches

input: amino acid sequence *tag* and *peptide*
output: positions of mass gap blocks and exact residue matches

Set A_1, A_2 to the first amino acid of *tag* and *peptide*
While(A_1 **or** A_2 **is not** the end of the sequence){
    If $M_A(A_1)$ is equal to $M_A(A_2)$ **Then**
        If the amino acid before A_1 is same as the one before A_2 **Then**
            an exact residue match is found
        Else
            a mass gap block is found
        End If
        Set A_1 to the next amino acid of A_1, if possible
    Else If $M_A(A_1)$ is less than $M_A(A_2)$ **Then**
        Set A_1 to the next amino acid of A_1, if possible
    Else
        Set A_2 to the next amino acid of A_2, if possible
    End If
}

In the algorithm above, both sequences are traversed only once so the time complexity is $O(n + m)$, where n is the length of the tag and m is the length of the peptide. The algorithm above doesn't include the check of amino acid confidence scores, which can be done simply by traversing the tag and merging the blocks if the requirement is not met. Once the mass alignment is determined, the bit score can be calculated using 4.2.

In some cases, the bit scores of long low-quality matches will be larger than some short but high-quality matches because longer tags tend to have more blocks when aligned. In order to address this problem, we use the number of blocks to further evaluate the tag-peptide match. We believe that exact matching block is better than mass gap block in quality. As an exact matching block contains 1 amino acid, higher quality matches tend to have more blocks. Denote the number of blocks between T and D as k, and the length of T as l. The tag-peptide matching score is defined as following.

$$matchscore(T, P) = \frac{k}{l} \times score(T, P). \tag{4.3}$$

This is our formula for tag matching score. Here, $score(T, P)$ is the bit score defined

in 4.2. By applying this, matches with more blocks will have more chances to get higher score. In tag search procedure, for every tag, we calculate the score for every potential match and the highest scored matches are used for protein scoring. However, not all of them can be used in the next stage. According to our matching score function, matches with lower score are more likely to be fake because of short lengths or lacking exact residue matches. So we need a threshold to decide whether a match should be taken to protein scoring. The simplest way to apply a threshold is the FDR, during this step, peptides within 20%FDR are used in the next step.

4.4 Protein Shortlisting

Suppose a protein has n significant tag-peptide matches. Denote them by p_1, p_2, ..., p_n. Assume their scores are s_1, s_2, ..., s_n. Because some of these peptides heavily overlap each other (or even identical), a direct sum of these peptide scores did not perform well for separating true and false proteins. Thus, the following strategy for redundancy removal is used.

Without loss of generality, suppose their scores are such that $s_1 \geq s_2 \geq \ldots \geq s_n$. Let l_i be the length of peptide p_i. Let l'_i be the length of peptide p_i that do not overlap with any peptide p_j for $1 \leq j < i$. The protein score is then calculated as

$$\sum_{i=1}^{n} s_i \times \frac{l'_i}{l_i}. \tag{4.4}$$

Basically, every p_i will update the score of the protein it belongs to. After going through all of them, a list of proteins with their scores is provided. We use those proteins within 30%FDR to output a FASTA file. This serves as the short list of proteins for later steps of the search.

4.5 Peptide-Spectrum Matching

In this step, Tagger will then call a third-party database search tool to search the original spectrometry data in the FASTA file generated in the protein shortlisting step. Since all mainstream database search tools support customized protein databases in FASTA file format, this allows the flexibility to use Tagger to enhance different third-party database

search tools. In our experiments, some of the most widely used free database search tools, including X!Tandem [5], MS-GFDB [8], and Inspect [23], are used to demonstrate Tagger's performance.

4.6 Result Combination

Although the procedure can stop at the peptide-spectrum matching step, the peptide identification result can be further improved by combining Tagger's result with the third-party database search tool's result. Since both results have been computed already in the previous steps, the combination step takes only minimum additional time. There have been different strategies for combining multiple engines' search results in the literature [44]. However, for the flexibility to use any third-party engines, we chose to make the method as simple as possible. The main idea of the method is to union the two engines' results together. We use different strategies to combine the result of protein and peptide search and the details are presented in the next chapter.

Chapter 5

Results and Discussions

We choose three database search tools as benchmarks to test our method: (1) X!Tandem [5] is a widely used free software for protein identification. It is among the best maintained free software tools in proteomics. It is also well known for its relatively fast speed as a conventional database search tool. That fits our speed comparison purpose well. In addition, X!tandem provides results of protein identification in detail, which makes it possible for us to compare the protein search results; (2)InsPecT [5] is a tag-based database search method, we include it in our test for the comparison of other tag-based methods; (3)MSGFDB [8] is the advanced version of InsPecT, it outperforms InsPecT in both searching speed and accuracy. Unluckily, InsPecT and MSGFDB don't provide scores for proteins as they are designed for peptide search. As a result, we only use X!Tandem to compare protein search results, when comparing peptide results, all three benchmarks are included.

5.1 Datasets and Databases

Three datasets are used to compare performance: (1) Proteomics Dynamic Range Standard (UPS2): This standard dataset is an enhancement of original Universal Proteomics Standard (UPS1). A complex mixture of 49 human proteins has been formulated into a dynamic range of concentrations, ranging from 500 amoles to 50 pmoles. This data were generated by Vogel et.al to confirm purposes in their research of protein and mRNA concentration [45]. The dataset is available in the MS/MS data repository at Marcotte's lab at the University of Texas, Austin (www.marcottelab.org/MSdata). We chose datafile MSups_15ul.RAW.gz in dataset 13 for our test; (2) U2OS: This dataset is from the proteomeXchange data repository (ID: PXD001220). It is produced by Kirkwood et.al in

their study of human osteosarcoma (U2OS) cells [46]. We chose file PT1541S1F16.raw for the test; (3) Tumor1: This dataset was downloaded from the proteomeXchange data repository (ID: PXD001676). The data was produced by Sethi et.al in their research of paired colorectal cancer and non-tumorigenic tissues [47]. One data file Tumor-1_raw.zip, consisting of 10 raw files was used. Table 5.1 shows the size of the three datasets.

Table 5.1: Number of spectra in the three datasets, respectively.

	Size
UPS2	9424
U2OS	36159
Tumor1	57743

The Uniprot database is used as the main sequence database for testing. This database is downloaded from http://www.uniprot.org/uniprot/. The 115 common lab contaminant proteins from the cRAP (Common Repository of Adventitious Proteins) database were also appended to the Uniprot database [48]. There are about 5.5×10^5 proteins in the database and they are manually reviewed. As a medium-sized sequence database with low redundancy, the Uniprot database is used for both sensitivity and speed comparison.

Large query datasets can be tested on the Uniprot database, but one may also want to know the performance of Tagger while searching large databases. So database NCBI-NR database is used for further comparing the running time. This database is downloaded from ftp://ftp.ncbi.nih.gov/blast/db/FASTA/nr.gz. The NCBI-NR database encompasses sequences from both, non-curated and curated databases from many sources. It contains more than 10^8 protein sequences and is about 60GB in size. This database is mainly used to show Tagger's capability of searching huge databases.

5.2 Test Procedure and Parameters

Tagger was implemented in Java without much code optimization. The versions and download sites of other methods are shown in Table 5.2.

All tests were done on a PC with an AMD X8 FX-8320 (3.8GHZ, 8 cores) processor and 16GB memory. Tagger, Novor, X!Tandem, and MSGFDB are capable for multithreaded tasks and all 8 cores were involved during the search. However, InsPecT cannot use multiple cores so only 1 core was used.

Table 5.2: Versions and websites of the softwares used.

	Version	Site
Novor	1.5.571	http://www.rapidnovor.com/free-academic/
X!Tandem	VENGEANCE	http://www.thegpm.org/
InsPecT	2012/01/09	http://proteomics.ucsd.edu/Software/Inspect/
MSGFDB	7780	http://proteomics.ucsd.edu/Software/MSGFDB/

As none of the methods can use raw format input, we converted files of datasets into MGF format using RawConverter (version: 1.0.0.0) from The Scripps Research Institute. The following parameters were used for all the four software: precursor error tolerance = 15 ppm (InsPecT only accepts Da as the parameter so we used 0.5 Da instead), fragment ion error tolerance = 0.5 Da, fixed modification = carbamidomethyl of Cys, and variable modification = oxidation of Met.

For Tagger, the parameter we want to discuss is the neighbor table. After optimization, Tagger used 4-mer with at least 2 precise matches ($k = 4$, $k' = \{3, 4, 5\}$ and $j = 2$) as the seed to generate hits. Details about the optimization of these parameters are provided in AppendixB.

5.3 Main Results

In this section, we present the results in three aspects: (1)the running time required by Tagger and other methods on the three datasets; (2) the protein shortlisting sensitivity; (3) accelerating and enhancing peptide search.

5.3.1 Running Time Comparison

Table 5.3 shows the size of the shortlists provided by Tagger. Table 5.4 shows the running time of Tagger and other methods searching the UniProt database, on the three datasets, respectively. We divided Tagger's running time by steps so it can be easier to understand. And for illustration purpose, a breakdown of the time spent on each step of MSGFDB on the U2OS data is given in Figure 5.1. During the tests, the protein search results of Tagger within 30%FDR were used as the protein shortlist. The table shows that Tagger is much faster than other methods when used alone, also, as the shortlists are much smaller than the whole database, the running time was greatly reduced for X!Tandem, InsPecT, and

MSGFDB as well. Step 5 of Tagger was omitted in the table because the result combination only takes 3-5 seconds, which is negligible in comparison to other steps.

Table 5.3: Number of proteins in the shortlists of three datasets, respectively.

	Shortlist size
UPS2	8231
U2OS	32753
Tumor1	50249

Table 5.4: The running time (seconds) required by Tagger and other three methods, on the three datasets, respectively.

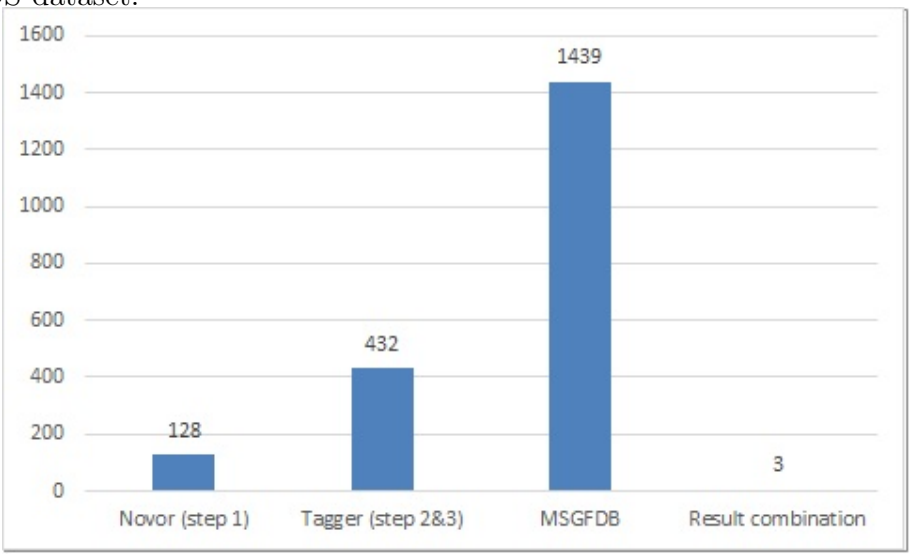
	UPS2	U2OS	Tumor1
Novor (Step 1 of Tagger)	44	128	217
Step 2 and 3 of Tagger	96	432	680
X!Tandem alone	957	3962	7405
X!Tandem on shortlist	78	450	819
InsPecT alone	17706	99244	more than 2 days
InsPecT on shortlist	766	11043	21250
MSGFDB alone	3409	7608	10081
MSGFDB on shortlist	258	1439	2309

We also used the four methods to search the NCBI-NR database to test their ability to search huge databases. The results are shown in Table 5.5, it is surprising that Tagger was the only method that can get the work done, other methods even failed on the smallest dataset UPS2. As Tagger doesn't need to read the whole database into memory, it actually has no limitation on database size.

Table 5.5: Results on searching the UPS2 dataset on NCBI-NR database

	UPS2
Tagger	Search completed after 2 hours
X!Tandem	Ran out of memory after 5 hours
InsPecT	Searched less than 3% after 2 days
MSGFDB	Java heap space error

Figure 5.1: The time breakdown (seconds) for each step of the MSGFDB configuration on the U2OS dataset.



5.3.2 Protein Shortlisting Sensitivity

The protein shortlisting step downsizes the protein database purely by the de novo sequence tags. If a protein is not ranked high enough to be kept in the shortlist, then all its peptides cannot be found in the later peptide-spectrum matching step. Thus, we need to watch the number of confident proteins identified by Tagger at its step 3.

Suppose a database search engine, say, X!Tandem, can identify n proteins from the original database at 1% FDR, without using Tagger. Then Tagger should identify a comparative number (or more) proteins at 1% FDR at its step 3. Otherwise the search sensitivity may be undermined by this protein shortlisting.

Figure 5.2 shows the Venn diagram of the proteins identified by X!Tandem and the first three steps of Tagger, on the three datasets, respectively. The figure shows that Tagger actually found slightly more proteins than X!Tandem. Figure 5.3 shows the FDR curves of X!Tandem and Tagger on the three datasets. These figures justified the appropriateness of this protein shortlisting step.

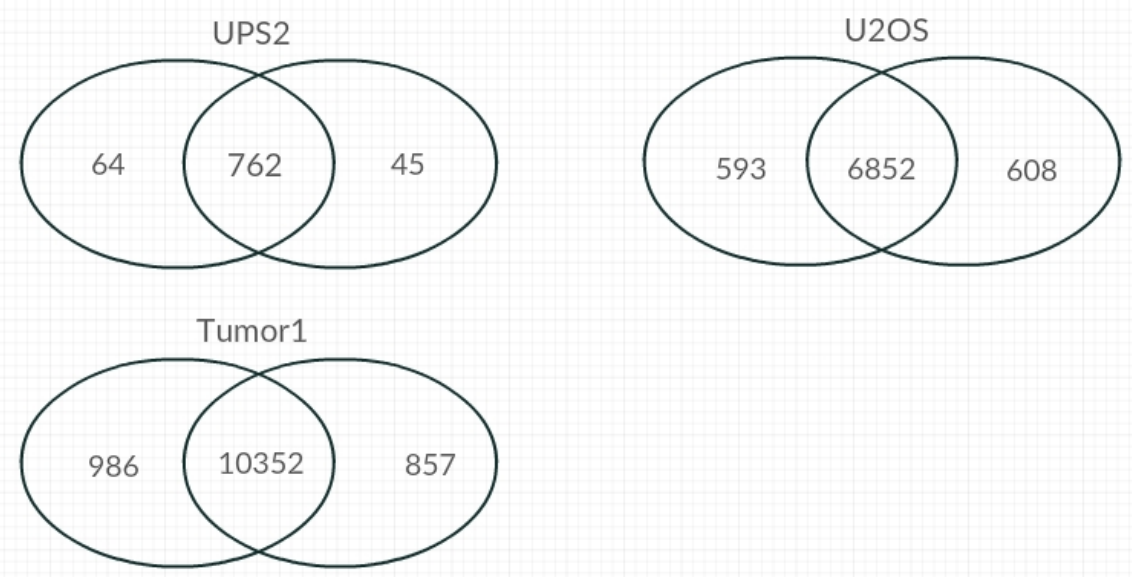


Figure 5.2: The Venn Diagram of the identified proteins with 1% FDR by Tagger (left) and X!Tandem (right), on the three datasets, respectively.

5.3.3 Protein Results Combination

As Tagger can find the same amount (or even more) proteins than X!Tandem, the results of step 3 are already of enough quality for protein identification. However, if we already have the results of X!Tandem searching the whole database, then the results of Tagger can be used to quickly improve the results of X!Tandem.

After applying 1% FDR to the results of Tagger and X!Tandem, we first studied the distribution of decoy protein by drawing a graph in which every point represented a decoy protein that was identified by either Tagger or X!Tandem, the x-axis value is the score of the protein given by Tagger and the y-axis value is the score of X!Tandem, as shown in Figure 5.4.

As shown in the graphs, most of the points are on either x or y-axis, which means they are only identified by one method. It inspired us to use the intersection of two result sets as the new result. This strategy takes only the common proteins found by both Tagger and X!Tandem. This gives a smaller list of proteins, but with much reduced FDR. One could achieve the same effect by providing a more stringent filtration on X!Tandem alone. To compare which approach gives a better sensitivity, we adjusted X!Tandem's score threshold to meet the actual FDR of the intersection strategy.

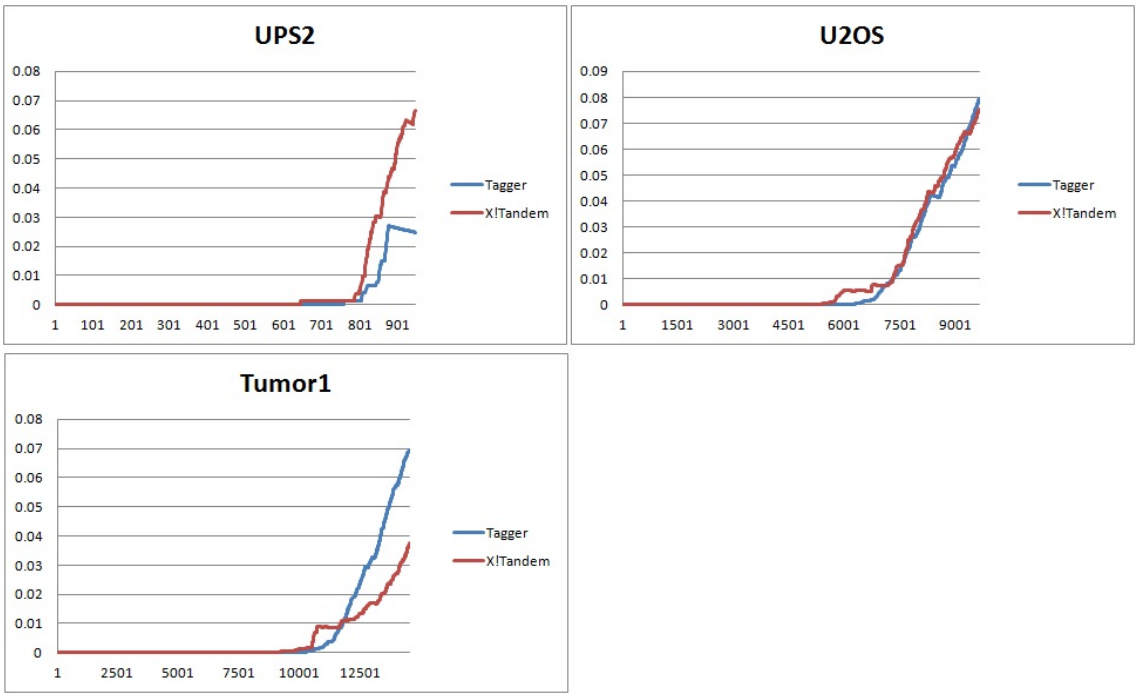


Figure 5.3: The protein FDR curves of X!Tandem and Tagger, on the three datasets, respectively.

And naturally, the union of two set was used, this strategy reports all proteins found by any of Tagger and X!Tandem. This will increase the FDR. To offset the FDR increase, we first set the FDR of Tagger and X!Tandem to be 0.5%. Then the union guarantees an FDR to be no more than 1%.

Table 5.6: Size and FDR of different sets.

	$T_{0.5\%} \cup X_{0.5\%}$	$X_{1\%}$	$T_{1\%}$	$T_{1\%} \cap X_{1\%}$	$X_{compare}$
UPS2	853, 0.63%	807, 1%	826, 1%	762, 0%	657, 0%
U2OS	7958, 0.84%	7460, 1%	7445, 1%	6852, 0.055%	5552, 0.055%
Tumor1	12087, 0.98%	11209, 1%	11338, 1%	10352, 0.046%	9409, 0.046%

Table 5.6 shows the number of proteins identified with different strategies, as well as the actual FDR of the results. The table shows that the combined use of the two software could significantly improve the sensitivity of the search over X!Tandem, at the same level or better FDR.

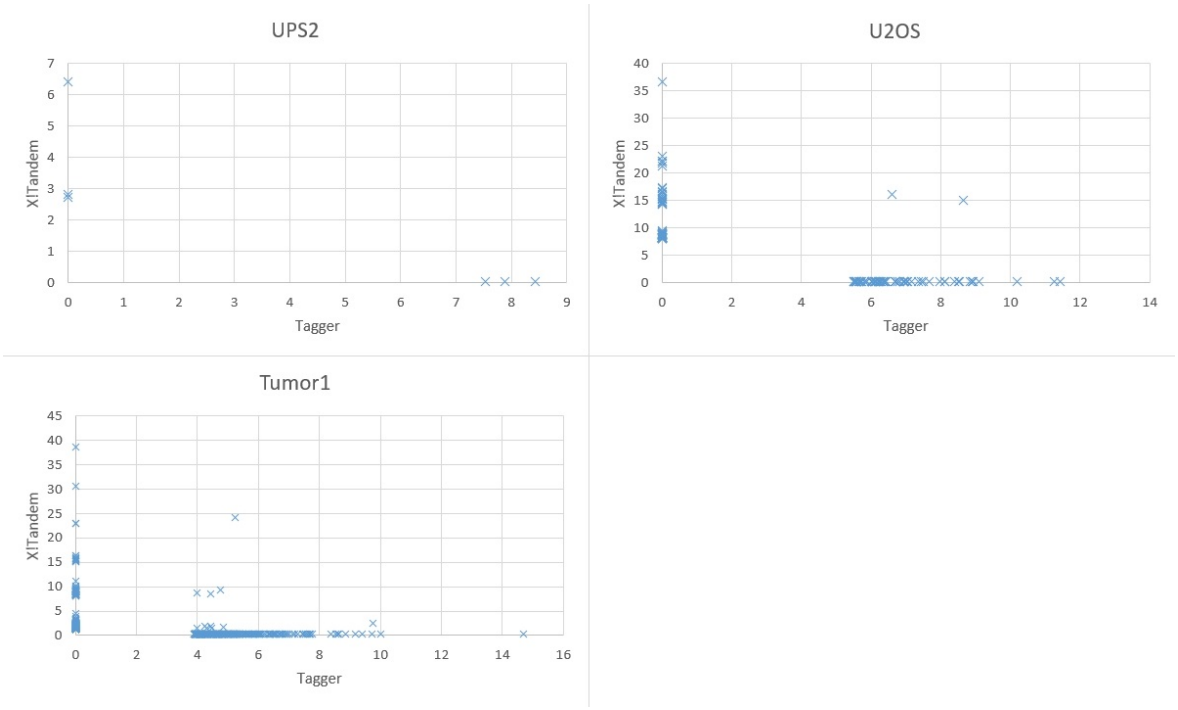


Figure 5.4: The distribution of decoy proteins.

5.3.4 Peptide Identification Sensitivity

The main goal of Tagger is to enhance the other search engines identifying more peptides. Table 5.7 shows the number of PSMs provided by different search engines at 1%FDR. The table shows that unlike the protein identification results, Tagger only found less than half of the proteins than MSGFDB when used alone. As a result, the peptide identification results of step 2 and 3 can't be used directly. Step 4 and 5 are necessary while doing peptide identification.

Here we want to discuss the strategy of result combination in step 5. As Tagger can't provide as much PSMs as other methods, we give low confidence on the peptides identified by Tagger. During the result combination, Tagger's results at 0% FDR were unioned with the result of other software at 1% FDR. This strategy only adds target peptides into the results of other methods, thus, the FDR won't be worse and the FDR curve will be moved parallel through the x-axis compared with the original one.

A few different configurations of using Tagger with different search engines are compared

Table 5.7: PSMs identified by the four softwares at 1% FDR.

	UPS2	U2OS	Tumor1
Tagger	1555	7912	11291
X!Tandem	3223	14848	24594
InsPecT	2109	6520	N/A
MSGFDB	3966	15279	28105

by using the number of PSMs at 1% FDR and the search speed. The following list explains the different configurations:

- **X!Tandem**: X!Tandem is used to search the complete protein database. Tagger is not used at all.
- **X!Tandem (Tagger accelerated)**: X!Tandem is used to search the protein shortlist computed by Tagger. Therefore the search speed is accelerated. This setting includes the first 4 steps of Tagger workflow.
- **X!Tandem (Tagger enhanced)**: On top of X!Tandem (Tagger accelerated), Tagger's peptide identification results are combined with X!Tandem's results. Therefore both the search speed and the search sensitivity are enhanced. This setting includes all the steps of Tagger workflow.
- **MS-GFDB, MS-GFDB (Tagger accelerated), MS-GFDB (Tagger enhanced)**: These settings are similar to the X!Tandem settings, except that X!Tandem is replaced with MS-GFDB.
- **Inspect, Inspect (Tagger accelerated), Inspect (Tagger enhanced)**: These settings are similar to the X!Tandem settings, except that X!Tandem is replaced with Inspect.

Table 5.8 shows the number of identified PSMs at 1% FDR, as well as the running time, by using each of the configurations. Figure 5.5 shows the FDR curves of different configurations using X!Tandem. The curves for other benchmark software have similar shapes and are not included here. From both Table 5.8 and Figure 5.5, one can conclude that Tagger can greatly enhance both the search sensitivity and search speed.

Table 5.8: The number of PSMs identified with 1% FDR and the running time for different configurations of using Tagger. The running time includes all steps from the raw data to the output of the result.

	UPS2	U2OS	Tumor1
X!Tandem	3223 (15m57s)	14848 (1h6m)	24591 (2h3m)
X!Tandem (Tagger accelerated)	3794 (3m38s)	15381 (16m50s)	27343 (28m36s)
X!Tandem (Tagger enhanced)	4074 (3m40s)	16548 (16m53s)	29896 (28m40s)
MS-GFDB	3966 (56m49s)	15279 (2h7m)	28105 (2h48m)
MS-GFDB (Tagger accelerated)	4420 (6m38s)	16797 (33m19s)	31011 (53m26s)
MS-GFDB (Tagger enhanced)	5070 (6m41s)	19835 (33m24s)	35105 (53m32s)
Inspect	2109 (4h55m)	6520 (27h40m)	N/A
Inspect (Tagger accelerated)	2666 (15m06s)	6842 (3h13m)	N/A
Inspect (Tagger enhanced)	3178 (15m07s)	9953 (3h13m)	N/A

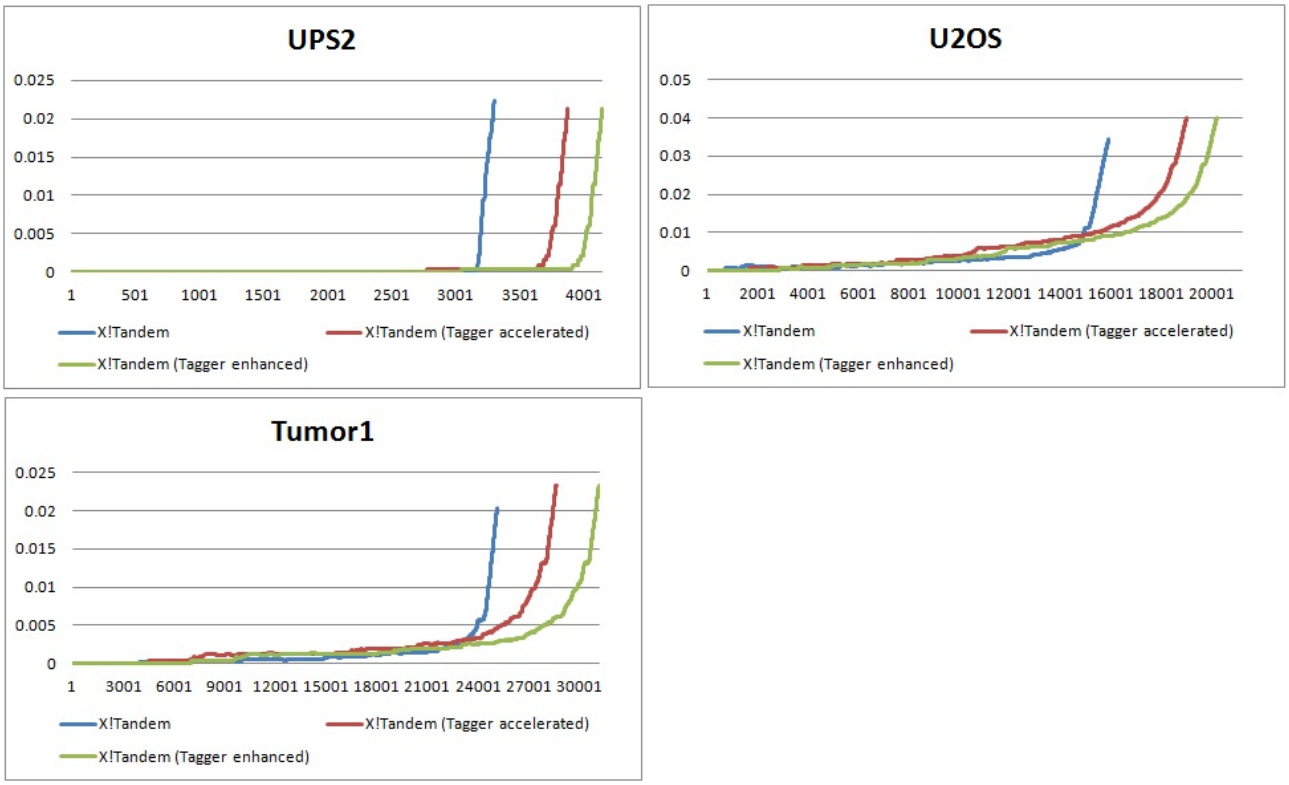


Figure 5.5: The peptide FDR curves of different configurations of using Tagger on the U2OS dataset.

Chapter 6

Conclusion and Discussion

In the past more than 20 years, the proteomics research community has developed a large number of database search engines. Many of these engines have accumulated a large user base. However, the rapid growth of mass spectrometry data size and the protein database size has made the database search increasingly slow with these existing search engines. We develop a general strategy and the Tagger software to improve the speed of any existing database search software, without making changes to the existing software.

Tagger's speed improvement is made possible due to the free availability of the rapid de novo sequencing software Novor. Other existing de novo sequencing software is too slow for the purpose. However, if other fast-speed de novo sequencing tools become available in the future, they can be easily plugged into Tagger's workflow to replace Novor.

Tagger's acceleration of the search speed does not compromise the search quality. On the contrary, for each of the search engines tested in this work, the search accuracy and sensitivity were significantly increased as the same time of the speed improvement. This is rather surprising as the protein shortlisting step creates the possibility to lose some true positive proteins. The rigorous peptide-tag scoring function and the protein scoring function proposed here are instrumental for keeping the true positives on the short list. During the course of this work, several scoring functions have been implemented and tested. The scoring functions used in this paper are both the best-performing and among the least complicated.

The peptide-tag scoring function is very different from most scoring functions used in today's database search software. Therefore, Tagger's own peptide identification results complement the results of the other search engines. This further contributes to the search quality in the final result combination step. The result combination step uses a very simple

strategy to combine Tagger's own results with that of the other search engine. The main consideration here is to keep the integration simple. So one can easily plug another search engine into Tagger's workflow. However, once the database search software is decided, one can start developing better ways to combine the results. It is likely that a more sophisticated result combination step can lead to further improvement of search quality. The Tagger software should be useful to any proteomics researchers who use database search software to identify peptides from mass spectrometry.

References

[1] Eng, J.K. *et al.* (1994) An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. *J. Am. Soc. Mass Spectrom.*, **5**, 976-989.

[2] Perkins, D.N. *et al.* (1999) Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis.*, **20**, 3551-3567.

[3] Ma, b. *et al.* (2003) PEAKS: powerful software for peptide de novo sequencing by tandem mass spectrum. *Rapid Commun. Mass Spectrom.*, **17**, 2337-2342.

[4] Bern, M. *et al.* (2012) Byonic: Advanced Peptide and Protein Identification Software. *Curr. Protoc. Bioinform.*, **40**, 13.20.113.20.14.

[5] Craig, R. and Beavis, R.C. (2004) Tandem: matching proteins with tandem mass spectra. *Bioinformatics*, **20**, 1466-1467.

[6] Clauser, K.R. *et al.* (1996) MALDI/PSD Peptide Fragment-Ion Tagging and Mass Fingerprinting of Placental Proteins from 2D Page. *44th ASMS Conference of Mass Spectrometry and Allied Topics*, Seattle, Washington.

[7] Cox, J. *et al.* (2011) Andromeda: A Peptide Search Engine Integrated into the MaxQuant Environment. *J. Proteome Res.*, **10**, 17941805.

[8] Kim, Sangtae *et al.* (2010) The Generating Function of CID, ETD and CID/ETD Pairs of Tandem Mass Spectra: Applications to Database Search. *Molecular & Cellular Proteomics.*, **9**, 2840-2852.

[9] Wang, X. *et al.* (2014) JUMP: A Tag-based Database Search Tool for Peptide Identification with High Sensitivity and Accuracy. *Mol Cell Proteomics.*, **13**, 36633673.

[10] Vaudel, M. *et al.* (2011) Peptide identification quality control. *Proteomics.*, **11**, 21052114.

[11] Gupta, N. *et al.* (2011) Target-Decoy Approach and False Discovery Rate: When Things May Go Wrong. *J. Am. Soc. Mass Spectrom.*, **22**, 11111120.

[12] Zhang, J. *et al.* (2012) PEAKS DB: De Novo sequencing assisted database search for sensitive and accurate peptide identification. *Mol. Cell. Proteomics.*, **11**, M111.010587.

[13] Ma, B. (2015) Novor: Real-Time Peptide de Novo Sequencing Software. *J. Am. Soc. Mass Spectrom.*, **26**, 1885-1894.

[14] Frank, A. and Pevzner, P. (2005) PepNovo: De Novo Peptide Sequencing via Probabilistic Network Modeling. *Anal. Chem.*, **77**, 964-973.

[15] Chi, H. *et al* (2010) pNovo: De novo Peptide Sequencing and Identification Using HCD Spectra. *J. Proteome Res.*, **9**, 27132724.

[16] Ma, B. and Johnson, R. (2012) De novo sequencing and homology searching. *Mol. Cell. Proteomics.*, **11** , O111.014902.

[17] Hughes, C. *et al.* (2010) De novo sequencing methods in proteomics. *Methods.Mol.Biol.*, **604**, 105121.

[18] Mackey, A.J. *et al.* (2002) Getting More for Less: Algorithms for Rapid Protein Identification with Multiple Short Peptide Sequences.*Mol. Cell. Proteomics.*, **1**, 139-147.

[19] Huang, L. *et al.* (2001) Functional Assignment of the 20 S Proteasome from Trypanosoma brucei Using Mass Spectrometry and New Bioinformatics Approaches. *J. Biol. Chem.*, **276**, 28327-28339.

[20] Shevchenko, A. *et al.* Charting the proteomes of organisms with unsequenced genomes by MALDI-quadrupole timeof-flight mass spectrometry and BLAST homology searching. *Anal Chem.,* **73**, 1917-26.

[21] Han, Y. *et al.* (2005) SPIDER: Software for Protein Identification from Sequence Tags with De Novo Sequencing Error. *J. Bioinformatics Computational Biol.*, **3**, 697-716.

[22] Tabb, D.L. *et al.* (2003) GutenTag: High-Throughput Sequence Tagging via an Empirically Derived Fragmentation Model. *Anal.Chem.*, **75**, 64156421.

[23] Tanner, S. *et al.* (2005) InsPecT: identification of posttranslationally modified peptides from tandem mass spectra. *Anal. Chem.*, **77**, 4626-4639.

[24] ScienceExplained. (2016) DNA, RNA and protein – the Central Dogma. http://science-explained.com/theory/dna-rna-and-protein/.

[25] Sparkman, O. David. (2000) Mass spectrometry desk reference. *Pittsburgh: Global View Pub.*, ISBN 0-9660813-2-3.

[26] Wikipedia. (2016) Mass spectrometry – Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Mass_spectrometry.

[27] ThermoFisher Scientific. (2016) Overview of Mass Spectrometry for Protein Analysis. https://www.thermofisher.com/ca/en/home/life-science/protein-biology/protein-biology-learning-center/protein-biology-resource-library/pierce-protein-methods/overview-mass-spectrometry.html.

[28] Dempster, A. J. (1918) A new Method of Positive Ray Analysis. *Physical Review.*, 11, 316-325.

[29] Fenn, J.B. *et al.* (1989) Electrospray ionization for mass spectrometry of large biomolecules. *Science.*, **246**, 64-71.

[30] Karas, M. *et al.* (1987) Matrix-assisted ul-traviolet laser desorption of non-volatile compounds. *Int. J. Mass Spectrom. Ion Process* **78**, 53-68.

[31] Emory College of Art and Sciences. (2016) Mass Spectrometry Ionization Methods. http://chemistry.emory.edu/msc/tutorial/mass-spectrometry-ionization.html.

[32] Wikipedia. (2016) Tandem mass spectrometry – Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Tandem_mass_spectrometry.

[33] Mitchell Wells, J. *et al.* (2005) CollisionInduced Dissociation (CID) of Peptides and Proteins. *Meth Enzymol.*, **402**, 148-85.

[34] Coon, J.J. *et al.* (2005) Tandem mass spectrometry for peptide and protein sequence analysis. *BioTechniques.*, **38**, 519, 521, 532.

[35] Olsen, J.V. *et al.* (2005) Higher-energy C-trap dissociation for peptide modification analysis. *Nat. Methods.*, **4**, 709-12.

[36] Quan, L. and Liu, M. (2013) CID, ETD and HCD Fragmentation to Study Protein Post-Translational Modifications. *Mod Chem appl.*, **1**, 1.

[37] Novk, J. *et al.* (2015) CycloBranch: De Novo Sequencing of Nonribosomal Peptides from Accurate Product Ion Mass Spectra. *J. Am. Soc. Mass Spectrom.*, **26**, 1780-1786.

[38] Ficher, B. *et al.* (2005) NovoHMM: A Hidden Markov Model for de Novo Peptide Sequencing. *Anal. Chem.*, *77*, 7265-7273.

[39] Pearson, W.R. and Lipman, D.J. (1988) Improved tools for biological sequence comparison. *Natl.Acad.Sci.,* **85**, 2444-2448.

[40] Pegg, S.C. and Babbitt, P.C. (1999) Shotgun: getting more from sequence similarity searches. *Bioinformatics.,* **15**, 729-740.

[41] Altschul, S.F. *et al.* (1990) Basic local alignment search tool. *J.Mol.Biol.*, **215**, 403-410.

[42] Searle, B.C. *et al.* (2004) High-throughput identification of proteins and unanticipated sequence modifications using a mass-based alignment algorithm for MS/MS de novo sequencing results. *Anal.Chem.,* **76**, 2220-2230.

[43] Ma, B. *et al.* (2002) PatternHunter: faster and more sensitive homology search. *Bioinformatics.*, **18**, 440445.

[44] Shteynberg, D. *et al.* (2013) Combining Results of Multiple Search Engines in Proteomics. *Molecular & Cellular Proteomics.*, **12**, 2383-2393.

[45] Vogel, C. *et al.* (2010) Sequence signatures and mRNA concentration can explain two-thirds of protein abundance variation in a human cell line. *Mol. Syst. Biol.,* **6**, 400.

[46] Kirkwood, K.J. *et al.* (2013) Characterization of native protein complexes and protein isoform variation using size-fractionation-based quantitative proteomics. *Mol. Cell. Proteomics.,* **12**, 3815-73.

[47] Sethi, M.K. *et al.* (2015) Quantitative proteomic analysis of paired colorectal cancer and non-tumorigenic tissues reveals signature proteins and perturbed pathways involved in CRC progression and metastasis. *J. Proteomics.*, **126**, 54-67.

[48] The Global Proteome Machine. (2016) cRAP protein sequences. http://www.thegpm.org/crap/.

[49] Boguski, M.B. *et al.* (1993) dbEST *Nat Genet.*, **4** 332-3.

[50] Liu, C. *et al.* (2006) Peptide sequence tag-based blind identification of post-translational modifications with point process model. *Bioinformatics.*, **22**, e307-e313.

[51] Han, X. *et al.* (2011) PeaksPTM: mass spectrometrybased identification of peptideswith unspecifiedmodifications. *J. Proteome Res.*, **10**, 2930-2936.

[52] Ning, Z. *et al.* (2001) SSAHA: A Fast Search Method for Large DNA Databases. *Genome Res.*, **11**, 1725-1729

[53] Bailey T.L. *et al.* (2006) MEME: discovering and analyzing DNA and protein sequence motifs. *Nucl. Acids Res.*, **34**, W369-W373

w

Appendix A

APPENDICES

A.1 AppendixA: Software and Hardware used

Software list

- Novor 1.5.571
  −de novo sequencing.

- RawConverter 1.0.0.0
  −converting RAW data to MGF file.

- X!Tandem VENGEANCE
  −database search.

- InsPecT 2012/01/09
  −database search.

- MSGFDB 7780
  −database search.

- Eclipse Luna 4.4.2
  −implementation of Tagger in Java.

- Microsoft Office Excel 2013
  −analysing results and drawing graphs.

Hardware list

- Personal Computer: AMD X8 FX-8320@3.8GHZ, 16GB RAM

A.2 AppendixB: Parameters of the neighbor table

A neighbor table is a hash table, whose key is a k'-mer and the value is a list of k-mers. Two sequences of the same mass should have at least j exact matches to be considered as neighbors. So there are three parameters to be discussed: k', k and j.

Parameter k'

Parameter k' is the most flexible among the three parameters. We only have one minimum constraint for k': the value of k' must contain k because neighbors are most likely to have the same length.

We can add extra values to k' as a feature for better results. After searching all sequences of the same length k, it is reasonable to search the sequence of length $k+1$ and $k-1$. If two amino acids have the same mass, their lengths won't differ a lot, especially for short sequences. As the k-mers are subsequences of de novo tags, their lengths are relatively short. Thus, we set $k' = \{k-1, k, k+1\}$.

Paremeter k

Different k'-mers have different neighbors, but they all have the same length k. Parameter k is the key parameter because those k-mers are used directly to search for potential matches. So we designed some tests to help us choosing the proper value of k.

If we increase the value of k by 1, then the search space can be compressed to approximately 1/20. We also want to know the quality of result after increasing k. In the classic BLAST algorithm [41], a seed of length 3 was used to find matches. Inspired by that, we first configured Tagger to use exact 3-mer match while seeking potential matches, then the UniProt database was searched with UPS2 dataset by Tagger and X!Tandem separately. Before we compare the results, we noticed that the running time of Tagger was almost the same as X!Tandem, which is not acceptable because our research goal is to build a fast speed method. According to the test, the search space of using 3-mer exact match is too large and we have to increase the value of k.

Next, we check the effect on the results after increasing k. In order to do that, we used a result file of searching the UniProt database with UPS2 dataset, which was done by PEAKS DB (thanks to my supervisor Prof Ma for offering the result file). There were total 4300 PSMs in the database, then they were examined one by one to determine whether

they can be matched as potential matches using seeds of different lengths. Table A.1 shows the main results of the tests.

Table A.1: Number and percentage of potential matches.

	Number	Percentage
k=3	4066	95%
k=4	3832	89%
k=5	2357	55%
k=6	975	23%

Although there is no guarantee that PEAKS DB can match all spectra to the correct protein, it reflects the inner structure within protein sequences and we think the result is more convincing than using randomly generated data. According to the results, the match rate starts to drop rapidly when $k = 5$. The score function is only performed on candidates and low match rate will surely lead to bad search results. Finally, we chose $k = 4$ to build the neighbor table. If needed, we can consider adding more features to the neighbor to further increasing the sensitivity of the search if it won't affect the running time greatly.

Paremeter j

As k is set to 4 and k' set to $\{3, 4, 5\}$, we set j to 2 as it is the only reasonable choice. If we set j to 3, then it has no meaning to include 3 in k' because it is impossible for a 3-mer to have 3 exact matches with a 4-mer. Moreover, if a 4-mer has 3 exact matches with another 4-mer, then the 4th amino acid will also be matched as an exact match, which means the two 4-mers are the same. It is also not wise to set j to 1 because it will allow too many sequences into the neighbor list. We did a test in which we created a neighbour table using $k = 4$, $k' = \{4\}$, $j = 1$. After checking the table, we found every k'-me had by average more than 60 neighbors! It means the search space is over 60 times larger than using the exact 4-mer match. The search space of using exact 3-mer match is 20 times larger than 4-mer and it is already too large for a fast speed method. The running time will be completely intolerable if we set j to 1. As a result, $j = 2$ is the best choice.

A.3 AppendixC: Proof of Lemma 1

The proof is quite simple, denote the length of one of the two sequences as l (it doesn't matter which one is chosen), then *Lemma 1* can be rewritten as follows:

- For an amino acid sequence S of length l ($l > 0$) and a sequence S' that has the same mass as S, they can always be divided into n mass gap blocks and m exact residue matches, where $n + m > 0$.

According to the definition of mass gap blocks, if there is no exact residue matches or minor blocks within S and S', then S and S' are matched as a mass gap block. *Lemma 1* is true for this situation because S and S' can be considered as 1 mass gap block and 0 exact matches. We focus on the situation in which exact matches or minor mass gap blocks occur between S and S'.

Note one of the minor blocks or exact matches as g. Then S and S' can be divided into three parts: the sequence on the left side of g, denoted as S_l and S'_l, g itself, and the sequence on the right side of g, denoted as S_r and S'_r. As mentioned before, g is either mass gap block or exact match, so *Lemma 1* is true for g.

Obviously, S_l and S'_l have the same mass, so do S_r and S'_r. If we can prove that *Lemma 1* is true for S_l and S'_l as well as S_r and S'_r, then *Lemma 1* is also true for S and S'. As g can occur in any location of S, the length of S_l and S_r varies from 0 to $l - 1$. We only discuss the length from 1 to $l - 1$ because empty sequence has no meaning. **As a result, if *Lemma 1* is true for any $l' < l$, then it is also true for l.**

Now the problem is quite clear, we only need to prove that *Lemma 1* is true for $l = 1$. The proof is trival: denote the length of S' as l', if $l' = 1$ then S and S' are matched as an exact match. If $l' > 1$ than they are matched as a mass gap block. So *Lemma 1* holds for $l = 1$. And it is also true for any integer $l > 1$, *Lemma 1* is proved.