MASS SPECTROMETRY OF BIOLOGICAL MATERIALS

Second Edition, Revised and Expanded

edited by

BARBARA S. LARSEN
CHARLES N. McEWEN
E.I. du Pont de Nemours and Co., Inc.
Wilmington, Delaware

MARCEL DEKKER

MARCEL DEKKER, INC. NEW YORK • BASEL • HONG KONG

Library of Congress Cataloging-in-Publication Data

Mass spectrometry of biological materials / edited by Barbara S. Larsen. Charles N. McEwen. --2nd ed., rev. and expanded.
p. cm.
Includes bibliographical references and index.
ISBN 0-8247-0157-7 (alk. paper)
1. Mass spectrometry. 2. Biomolecules--Analysis. 3. Proteins-Analysis. I. Larsen, Barbara Seliger. II. McEwen, Charles N.
QP519.9.M3M38 1998
572 '.36--dc21 97-52814
CIP

The publisher offers discounts on this book when ordered in bulk quantities. For more information, write to Special Sales/Professional Marketing at the address below.

This book is printed on acid-free paper.

MARCEL DEKKER, INC.
270 Madison Avenue, New York, New York 10016
http://www.dekker.com

Current printing (last digit):
10 9 8 7 6 5 4 3 2 1

PRINTED IN THE UNITED STATES OF AMERICA

17

Network-Based Bioinformatics in Protein Mass Spectrometry

David Fenyö
Rockefeller University, New York, New York

Ronald C. Beavis*
The Skirball Institute of Biomedical Research, New York University Medical School, New York, New York

**Current affiliation:* Eli Lilly & Company, Indianapolis, Indiana.

I. INTRODUCTION

Proteins are directly responsible for almost all of the metabolic processes that occur within a cell. Specialized proteins are used for the generation of structural elements in a cell or an entire organism. The elaborate structure of DNA in a cell, now referred to as its genome, is meant to record the structure of an organism's proteins, both to serve as a template for the construction of copies of those proteins and to pass that crucial information on to the next generation. The proteins encoded by a genome can be organized into larger structures that are used as cellular machinery for performing specific tasks in a cell, such as transcribing DNA into more protein (which requires several discrete protein-based machines), transporting other molecules through membranes, or automatically repairing damage to cellular subsystems. When genetic material is passed from one generation to the next, it is also necessary to pass a complete working set of protein-based machinery for the reading of the DNA strands and performing all other necessary steps in cell metabolism.

Individual proteins are composed of discrete polypeptide chains, called subunits, that may be linked together either covalently or by the local structure of the surrounding solvent molecules. The linear sequence of individual amino acid residues in a particular polypeptide chain is referred to as the "primary structure" of the polypeptide. This primary structure is what is directly encoded by an organism's genome. Also encoded in the genome are the instructions to make protein-based machines for the express purpose of modifying particular residues in other proteins. These modifications occur after a polypeptide has been and are therefore called "posttranslational" modifications.

It should be noted that the word "protein" is normally used very loosely to describe a number of different physical or conceptual objects. This confusing state of affairs exists because the word protein was used originally to describe materials present in living organisms that performed particular tasks or catalyzed specific reactions, without reference to what these materials were chemically. Now that the fundamental structure of the molecules that carry out these tasks is known, the functionally derived names for proteins may be either inappropriate or slightly misleading, but they carry with them the weight of history. The realization that a particular protein may be a small part of a much larger machine makes the functional definition of a protein even more difficult. Regardless of shifts in philosophy about the true name and function of a protein, the component subunit polypeptide chains are real physical objects that are amenable to study and analysis.

This chapter describes the application of mass spectrometry to the analysis of the subunit polypeptide chains that are the building blocks of a protein (or protein-based machine). These polypeptide chains are the direct products of genome transcription and therefore their original structure can be predicted with

a knowledge of an organism's genome. Until recently, mass spectrometry was seen as a possible method for the determination of the primary structure of a polypeptide without reference to genomic information, in a manner analogous to Edman sequence analysis. The task has proved to be difficult, because of the unpredictable gas-phase cleavage behavior of peptides. Fortunately, genomic sequencing methods have made this task unnecessary in a growing list of species. The complete nucleic acid sequences that in turn correspond to all of the polypeptide sequences produced by those organisms are now known with a fair degree of reliability.

The existence of a truly huge amount of nucleic acid sequence information has opened up the field of protein analysis by mass spectrometry in unforeseen directions. It has, however, introduced a new problem: How can mass spectroscopists gain access to this information in a format that is appropriate for their work? Sequence databases were originally distributed in book form, but the paper format soon became too difficult to use because there is no possibility of properly indexing and searching the sequences for particular patterns. The compact disk read-only memory (CD-ROM) format became the standard method in the early 1990s, but this computerized format has become cumbersome because of the rapid rate of sequence generation. To stay current, these CD-ROMs needed to be replaced on a monthly basis, which is too costly. In the latter half of the 1990s, the distribution of databases has been almost completely replaced by electron-based formats (communications network transport). Using the currently popular Internet protocol (discussed later), it is possible to easily send information from one computer network to another, allowing the exchange of gigabytes of information with the same ease as making a telephone call.

Philosophically, all of the new applications of mass spectrometry can be viewed in terms of a simple hypothesis–confirmation model of experimental design [1]. If a hypothesis can be made about the structure of a polypeptide, an experiment can be designed in such a way that a mass spectrometric measurement will either confirm or disprove that hypothesis. A straightforward case (see Fig. 1) would be a polypeptide sample in which the amino acid sequence is thought to be known, such as a peptide produced by recombinant DNA methods. The known sequence can then be treated as a hypothesis: If this hypothetical structure is true, then cleaving that polypeptide with a sequence-specific protease should result in a collection of peptides with a certain set of allowed molecular masses. Performing the experimental cleavage and measuring the masses of the resulting peptides either will confirm the hypothetical structure, or will suggest that some parts of that hypothetical structure are incorrect. Hypotheses about those incorrect portions of the structure can then be formulated and tested by further experimentation. This type of experimental design leads to either the determination of the posttranslational modifications that have

been made to the polypeptide or to the discovery of errors in the nucleic acid sequence used as the basis of the original hypothesis.

The use of the combination of computers, analytical chemistry, and databases has been christened "bioinformatics." The following sections discuss the fundamental aspects of network-based resources that can be used to help analyze protein mass spectra, using bioinformatics methods, rather than manual calculations and analysis. The structure and use of the protein sequence databases currently available are reviewed, along with the current methods of accessing these data. These databases are the core of current protein chemistry network-based applications, so understanding the differences between databases is very important. These databases are currently available in the form of network-connected resources. Some discussion of the state-of-the-art of network information retrieval technology will also be presented. Details of hardware and software construction are fleeting; however, the general framework of server–client relationships and the principals of network-based computing are sufficiently important that they are discussed in some detail.

II. PROTEIN SEQUENCE DATABASES

Databases have become the preferred method for storing both polypeptide amino acid sequences and the nucleic acid sequences that code for these polypeptides. The databases come in a variety of different types that have advantages and disadvantages when viewed as the hypothesis for a polypeptide identification experiment. The properties of the most common databases currently in use are listed in Table 1.

While the "database entry" for an amino acid sequence may appear to be a simple text file to a user browsing for a particular polypeptide, many databases are being organized into very flexible, complicated structures [2]. The detailed implementation of the database on a particular system may be based on a collection of simple text files (a "flat-file" database), a collection of tables (a "relational" database), or it may be organized around concepts that stem from the idea of a protein, gene, or organism (an "object-oriented" database). The organization of the database is of more concern to programmers than it is to the user, and it does not directly affect the usefulness of the database for protein analysis and identification.

Any protein sequence database contains a collection of amino acid sequences represented by a string of single-letter codes for the residues in a polypeptide, starting at the N-terminus of the sequence. The letter codes may be either upper case or lower case, depending on the database interface. These codes may contain nonstandard characters to indicate ambiguity at a particular site (such as "B" indicating that the residue may be "D" or "N"). All of the sequences have a unique number–letter combination associated with them that

Table 1 Sequence Databases Currently Used for Protein Bioinformatics

Database	Peptide sequences	Nucleic acid sequences	Peptide annotation	Redundancy	Sequence reliability	Entries (/1000)
SWISSPROT [3]	Yes	No	Complete	Low	Excellent	59
PIR [4]	Yes	No	Complete	Moderate	Excellent	95
EMBL [5]	No	Yes	Some	High	Good	1438
TREMBL [5]	Yes	No	Some	High	Good	137
GENBANK [6]	No	Yes	Some	High	Good	1766
GENPEPT [6]	Yes	No	Some	High	Good	262
OWL [7]	Yes	No	Some	Low	Good	210
dbEST [8]	No	Yes	None	Very high	Low	1317

Note: The sequence reliability for GENBANK and EMBL refers to the normal nucleic acid sequences stored in those databases. These databases also include expressed sequence tag data, which are not as reliable.

is used internally by the database to identify the sequence, usually referred to as the accession number for the sequence (more technically referred to as a "key attribute" for the database entry).

A. Redundancy

A sequence database may contain multiple copies of a particular sequence (redundant entries) or it may have been constructed so that there are no multiple copies—that is, it is nonredundant. The presence of redundant sequences in a database may have several origins. The cause may be a historical accident, such as the entry of the same sequence under more than one "protein" name because of confusion about the function of the protein, or it may be deliberate—for example, multiple entries for a polypeptide sequences that has been spliced together after translation. The presence of many copies of the same sequence in a database has the effect of making the database larger, resulting in slower searches. It also may result in multiple hits in a protein identification experiment, while all of the hits are actually the same molecule. Database managers are currently engaged in the process of removing as many redundant entries as they can find from their databases. Some databases exist only to be nonredundant collections of sequence entries from other redundant databases.

B. Annotation

Databases may contain a combination of amino acid sequences, comments, literature references, and notes on known posttranslational modifications to the sequence. A database that contains all of these elements is referred to as "anno-

tated." Other databases only contain the sequence, an accession number, and a descriptive title. Annotation of each entry is obviously very time-consuming and difficult to maintain without errors; therefore annotated databases usually have many fewer sequence entries than nonannotated ones. Annotation also implies that some functional or structural information is known about the mature protein, as opposed to a sequence that is known only from the translation of a stretch of nucleic acid sequence. Even the best annotated databases now include large numbers of entries that have very little real information about the mature protein other than some reference to who sequenced and translated the nucleic acid sequence.

Annotated databases are technically superior for performing protein identification searches, because they contain information about the true form of the mature protein, making search misses caused by posttranslational modifications less likely. They also can screen out spurious hits that could occur if parts of the sequence that are not present in the mature protein are included in the search. The nucleic acid sequences of many polypeptides contain stretches of sequence that are removed either immediately following translation of the preprotein (signal peptides) or when an inactive proprotein is activated by the removal of the corresponding propeptide. Including these pre- and propeptides in the search may lead to errors.

Nonannotated databases have the tremendous advantage of being simpler to maintain. This simplicity means that new sequences are incorporated more quickly and the effort necessary to verify all new entries is not required. Several very large amino acid sequence databases (GENPEPT and TREMBL) are simply translations of corresponding large nucleic acid databases (GENBANK and EMBL). The number of entries in these translated databases makes them very attractive for protein identification searches because the chance of finding positive hits is much greater than in the much smaller annotated databases. It is necessary to be much more careful about interpreting the results from searching this type of database because no information about the mature polypeptide is available. Even with this caveat, the large number of polypeptide sequences available in nonannotated databases and the current ascendancy of molecular biology have resulted in their use as the hypothetical data set for comparison with experiment in protein identification searches.

C. Sequence Reliability

It is usually assumed that if something is published, it has been proofread and it is letter perfect. Protein sequence databases are generally not proofread and they are far from perfect. The problems stem from several sources. A major problem is that the data is difficult to proof, once it is manually entered into the database. The sequences are a string of random-looking capital letters and

it is difficult to pick out typing errors in this type of text. Database creators have different schemes in place to screen out typing errors, but they remain a consistent problem in all databases and they tend to migrate from database to database. If the detailed sequence of a protein is necessary for comparison with the primary structure of a sample, it is always necessary to check the original source of the data (frequently a journal article) to verify the sequence. Relying too heavily on the accuracy of protein sequence database has led to misunderstandings and confusion between groups examining the same protein.

The automated calling and transmission of sequences into a database directly from nucleic acid sequencing machines has practically eliminated human error in the data entry in some types of sequences, such as expressed sequence tags. This type of entry introduces a new class of error into a database, however. In order to achieve high-throughput sequencing, it is not possible to investigate portions of a particular sequence that cannot be called reliably. The current generation of sequence calling software will place an N at any position where the sequence cannot be called with confidence. Nucleic acid databases are accumulating sequence containing N's faster than they are accumulating sequences that are completely characterized. Automated sequencing software is also written with the idea that it will make errors in calling the sequence at some rate. This error rate will be unevenly distributed in particular sequences. Sequences called from good, strong signals will be very nearly perfect, while sequences called from poor quality data will have a very high error rate. Both sequences will appear the same in the database; no measure of signal strength or residue assignment confidence exists in current databases. It is therefore wise to treat all sequences that are not derived from journal publications as raw data, that is, with a certain amount of skepticism.

Another source of "reliability" problems arises even in cases where peer-reviewed journal articles describing a sequence exist. Most complete sequences of mature proteins are the results of several publications that may disagree in detail. Many "conflicts" exist between different references on the same protein. Annotated databases usually include some indication of these conflicts between sequences, but the sequence actually entered in the database represents one or the other of the sequences published. Database searching and sequence matching software does not take these conflicts into account: It considers the sequence as entered in the database as true. This situation is frequently the case for proteins where both protein sequencing and nucleic acid sequencing has been performed. Generally, the nucleic acid sequence seems to be more reliable, but nucleic acid sequencing gives no indication of posttranslational modifications or sequence conflicts that arise because of real variation in protein sequence within a population. The proportion of database entries with sequence conflicts is decreasing, because most sequencing is now only done once by nucleic acid sequencing from homogeneous cell populations. This style of se-

quence determination eliminates the possibility of the discovery and reporting of conflicts.

D. Protein Identification

Frequently, the sequence of a particular polypeptide is not known, even though the complete genomic sequence of the organism is available. This problem exists because there is no simple, direct method of connecting an experimentally observed polypeptide with the genomic sequence; that is, a band on a sodium dodecyl sulfate polyacrylamide gel electrophoresis (SDS-PAGE) gel cannot be connected to a gene without considerable effort. It is this particular problem that has attracted the attention of protein chemists and their mass spectrometers. A number of mass spectrometer-based experiments have been formulated to try to connect a stained band on a gel with genomic information. All of these experiments are the result of the realization that the sum total of the genomic information from an organism can be treated as one big hypothesis. If one is confronted with an unknown polypeptide, it is reasonable to make the hypothesis that this polypeptide sequence is encoded by the known nucleic acid sequence, along with thousands of other polypeptides. If it is possible to generate a set of mass spectrometric data points that can be tested against this hypothetical nucleic acid sequence by some algorithm, then there is the possibility of linking the real polypeptide with the portion of genome that produced it.

The process of linking molecular mass data with a translated genomic sequence has come to be called mass spectrometric "protein identification"—a rather inaccurate but evocative name [9–13]. Various clever strategies have been formulated for producing mass spectrometric information that is sufficiently unique to identify a particular stretch of genomic sequence. While there is no a priori reason to believe that such strategies can in fact be devised, the sequences of real polypeptide subunits greatly aid the process. The sequences of real subunits are extremely varied, even in molecules with similar three-dimensional shapes. The fact that amino acid residues with very similar physical properties have different molecular masses means that sequences that perform similar but discrete tasks in a cell will produce mass spectrometric data that can distinguish between the two molecular species.

The general pattern of the experiments used to identify proteins is similar to that illustrated in Fig. 1, but with the addition of bioinformatic database searching. Some combination of specific and/or nonspecific peptide bond cleavage experiments is performed and the experiments are mass analyzed. The determined masses are then compared against the collection of hypothetical polypeptide sequences in a genome. A match between the experimental results and the hypothetical sequences is then made on the basis of some reasonable algorithm that scores the probability that a particular sequence could give rise to

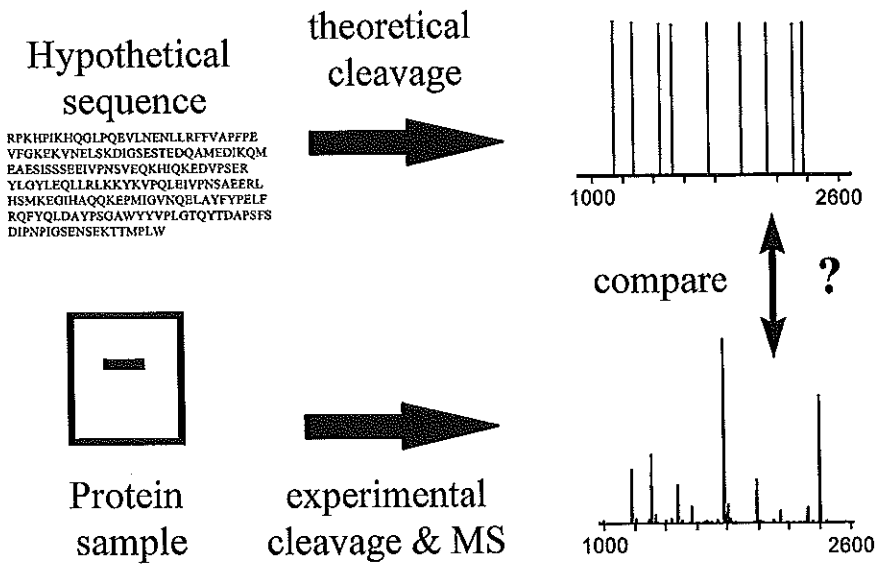

Figure 1 Typical hypothesis-driven, protein mass spectrometry experiment. Hypothetical sequences are screened against an observed protein cleavage pattern. A good match between the two patterns constitutes "identification"—that is, the hypothetical sequence is assumed to be the measured protein's primary structure.

the experimental results. The original polypeptide is then "identified" with the hypothetical sequence that produces the best score.

Any information about a polypeptide's partial sequence can be used in combination with the measured masses of peptides generated by proteolysis to constrain the search for a polypeptide. Several strategies using multidimensional mass spectrometry (MS/MS) have been demonstrated. Figure 2 illustrates this type of bioinformatics experiment. A single peptide from a protein digest is subjected to MS/MS measurement and the observed pattern of fragment ions is compared to the patterns of fragment ions predicted from database sequences. This type of comparison can be very constraining, resulting in high-confidence identification of the peptide with a known sequence. The approach is very attractive, requiring only one or two peptides to identify a protein sequence [14–20].

Figure 3a shows a MALDI ion trap mass spectrometry (ITMS) spectrum of a tryptic digest of an unknown protein from *Saccharomyces cerevisiae* that was observed as a spot on a gel. The spectrum has more than a dozen major peaks. If the corresponding masses are used to search all *S. cerevisiae* sequences in OWL with ProFound, a list of the proteins that are most likely to

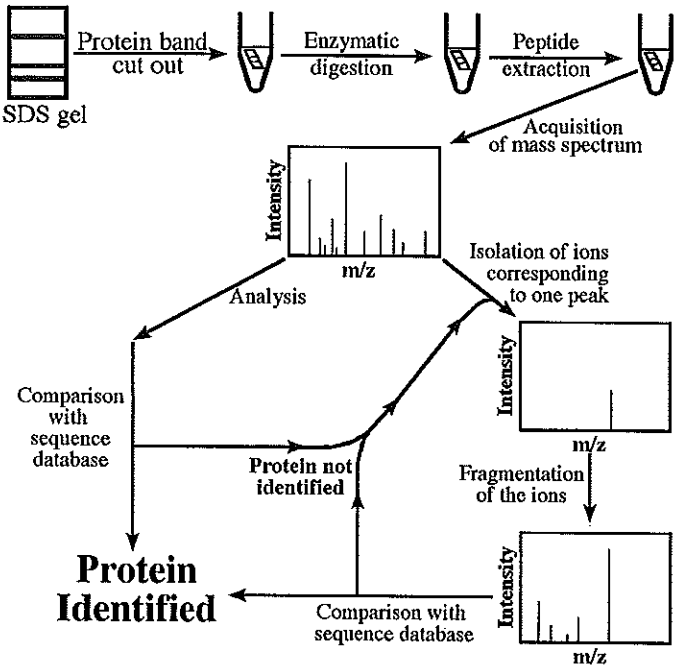

Figure 2 Flow chart showing the steps in a complete protein identification MS/MS experiment.

give the observed tryptic map is obtained (Table 2). In this example, subunit P130 of eukaryotic initiation factor 4F (IF42_YEAST) is the most probable protein. To further increase the confidence in the identification, the ions with $m/z = 2596$ were isolated and fragmented (Fig. 3b). The spectrum contains three major fragment ions. The peak at $m/z = 2468$ is loss of the C-terminal lysine and contains little information. The two other fragment peaks, on the other hand, correspond to fragmentation at the C-terminal side of acidic amino acids. If a database is searched for proteins that have tryptic peptides with mass 2595 Da that fragment at the C-terminal side of acidic amino acids to give rise to b or y ions with mass 1984 and 2337 Da there is only one yeast protein (IF42_YEAST) in the public databases that agree with this information. The tryptic peptide is AQPISDIYEFAYPENVERPDIK and the two fragment ions correspond to fragmentation on the C-terminal side of the aspartic acids at residues 6 and 20, respectively. If a theoretical trypsin digest of IF42_YEAST is compared with the peptide map, all major peaks can be assigned to tryptic peptides from IF42_YEAST or to peptides from autolysis of trypsin.

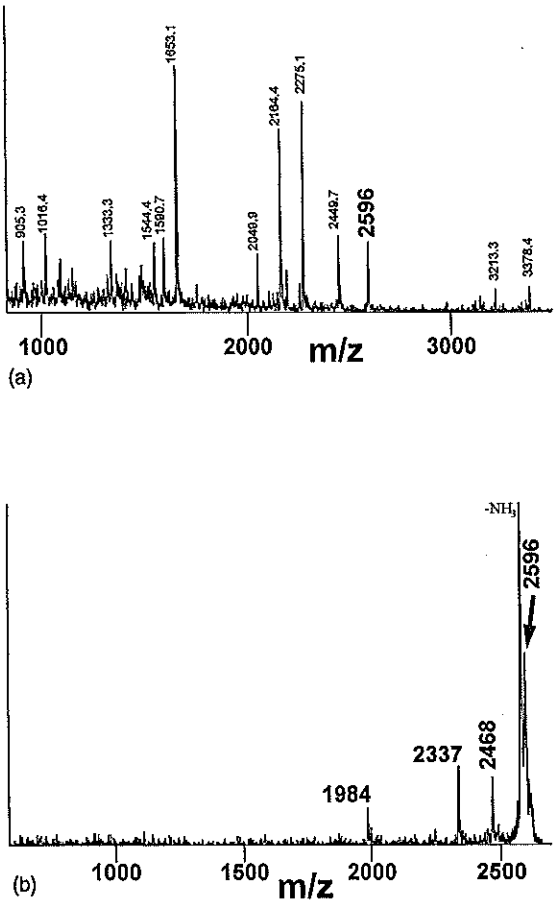

Figure 3 (a) MALDI-ITMS spectrum of a tryptic digest of an unknown protein from *S. cerevisiae*. (b) MALDI-ITMS/MS spectrum if peptides at $m/z = 2596$ (unpublished results, Jun Qin, NIH).

The observed molecular mass or isoelectric point of a polypeptide can also be used to constrain a mass spectrum pattern-matching search, although care must be taken not to apply this type of constraint too firmly. When nonannotated nucleotide sequence databases are used (such as TREMBL or GENPEPT), subsequent processing can greatly alter the pI or molecular mass of a protein, so much so that no identification can be made. For example, the small, highly conserved protein ubiquitin (SWISSPROT accession number P02248) has a molecular mass of 8.6 kD, which is the mass that would be measured by

Table 2 Result from a ProFound Search of All *S. Cerevisiae* Proteins in OWL with Masses from Fig. 3a

1. IF42_YEAST (probability = 7.9e-01) EUKARYOTIC INITIATION FACTOR 4F SUBUNIT P130
2. NPR1_YEAST (probability = 2.8e-02) NITROGEN PERMEASE REACTIVATOR PROTEIN
3. S63138 (probability = 1.9e-02) PROBABLE PROTEIN KINASE NPR1
4. RL3E_YEAST (probability = 1.8e-02) 60S RIBOSOMAL PROTEIN L30E
5. SCYOR206W (probability = 1.5e-02) HYPOTHETICAL PROTEIN SCYOR206W
6. CBF5_YEAST (probability = 1.4e-02) CENTROMERE/MICROTUBULE BINDING PROTEIN CBF5
7. S52893 (probability = 9.5e-03) HYPOTHETICAL PROTEIN YMR044W

Note: m/z = 2164 and 2275 were not used in the search, because they are from autolysis of trypsin.

a mass spectrometer or a gel. A simple keyword search of the translated-nucleotide database GENPEPT results in several sequences for the same protein [accession numbers M26880 (77 kD), U49869 (25.8 kD) and X63237 (17.9 kD)]. None of these nucleotide-translated sequences give the correct molecular mass or pI, so using those parameters to limit a search would result in missing the database sequence altogether. Only annotated databases that fully outline known modifications can be used when the properties of the mature protein are being used to constrain a search.

All of the protein identification strategies just outlined are currently available as CGI programs (discussed later) that can be accessed using a browser. Table 3 is a list of the programs available, as well as some of the relevant characteristics of the programs. The list of available programs will change with time, as will their features, but the table should remain a reasonable starting place for several years.

E. Software Strategies for Protein Identification

Because there is a range of CGI software available for protein identification, this application can serve as an example of the types of choices that must be made to create a bioinformatics application for mass spectrometry. Some protein identification software takes the original protein sequence database as input and calculates all the proteolytic masses every time an identification is performed. This has the advantage that the software can be easily modified to take into account different kinds of additional information (see previous section). Performing the mass calculations "on the fly" works well when the search is performed using mass spectrometric fragmentation information. However, for

Table 3 CGI Programs Currently Used for Protein Identification

CGI program	Data type	Databases	Cleavage chemistry	Additional search parameters	Help	AI	Report format
PepFrag [21]	MS/MS	10 (aa+na)	7 Enzymes + CNBr	Protein mass, peptide masses, taxonomy, missed cleavage sites, daughter ion types, mass accuracy	Pages + examples	No	Active + tools
ProFound [22]	Masses	1 (aa)	5 Enzymes + CNBr	Protein mass, taxonomy, missed cleavage sites, mass accuracy	Pages + examples	Yes	Active + tools
MS-Fit [23]	Masses	5 (aa+na)	9 Enzymes + CNBr	Protein mass, taxonomy, missed cleavage sites, mass accuracy	1 Example	No	Active
MS-Tag [24]	MS/MS	5 (aa+na)	9 Enzymes + CNBr	Protein mass, taxonomy, missed cleavage sites, daughter ion types, mass accuracy	1 Page	No	Active
PeptideSearch [25]	Masses	1 (na)	7 Enzymes +CNBr	Protein mass, mass accuracy	None	No	Active
PeptideSearch [26]	MS/MS	1 (na)	7 Enzymes +CNBr	Protein mass, daughter ion types, mass accuracy	None	No	Active
Mass Search [27]	Masses	2 (aa+na)	11 Enzymes + 7 chemical	Protein mass	1 example	No	Dead
Mowse [28]	Masses	1 (aa)	6 Enzymes + CNBr	Protein mass, amino acid composition, mass accuracy	1 Page	Yes	Active

Note: Abbreviations: aa, amino acid sequence database; na, nucleic acid sequence database; CNBr, cyanogen bromide; and AI, any type of artificial intelligence/expert system used to rate results. An "active" report format has links that you can follow to get the sequence of a matched protein, and "tools" refer to additional programs to help make use of that sequence information. A "dead" report just lists the matching proteins' names and the user is expected to find the sequence information and pass judgment on the results without any assistance.

peptide mapping speed becomes an issue. It is important that the search is fast (<1 min) so that after obtaining some experimental data a search can immediately be performed and evaluated, so that a decision can be made whether the information is enough or if more experiments are necessary. One way of increasing the speed for peptide mapping is to first make a secondary database that contains the masses of all possible proteolytic peptides that can arise from enzymatic digestion of all the proteins in the original database. This secondary database is then used for the search. Unfortunately, the secondary proteolytic peptide mass databases usually take much more space than the original protein sequence database.

The simplest way to present the results of a database search with mass spectrometric information is to list all proteins that match the constraints given. This type of search is possible when the experimental data does not contain much noise (e.g., when searching with mass spectrometric fragmentation information). Since it is not always possible to separate proteins well, peptide maps often contain more than one protein; therefore, peptide mapping algorithms usually rank the proteins in the sequence database according the how well they fit the list of masses provided in the search. This ranking can in the simplest case be according to the number of masses that match proteolyic peptides in the protein. More sophisticated algorithms take into account the size of the protein, the spread of the proteolytic peptide mass errors, and the relative position of the proteolytic peptides in the protein.

For mass spectrometric protein identification the ideal situation occurs when the DNA sequence of the complete genome of the organism of interest is available and all protein coding regions have been predicted correctly. In this case, the search can be performed on a database of the translated protein sequences. However, even with organisms having their genomes completed it can happen that no known proteins match the experimentally obtained mass spectrometric constraints. In this instance, all the six reading frames of the complete DNA sequence must be translated without regard to whether a region of DNA is considered to code for proteins or not. If a nucleic acid sequence database is searched in this way, it is possible to find proteins that have not been predicted from the DNA sequence or when a frame shift has occurred in a predicted protein. To be able to find a protein in this way more information is necessary than when the protein database is searched because a lot of noise will be introduced by the six-reading-frame translation of both coding and noncoding regions.

If the protein of interest is posttranslationally modified, the identification problem becomes more complicated. Most protein identification algorithms can handle modifications that are present at all occurrences of an amino acids (e.g., alkylation of cysteines where the chemistry can be tuned so every cysteine residue is modified). Other modifications, like phosphorylation, that give a known mass shift can be handled but increase the noise level, because many more peptide masses have to be considered. Modifications like glycosylation,

on the other hand, cannot be handled by the search algorithm because there are many possibilities for the mass shifts. Usually posttranslational modifications do not interfere with the identification because in most cases only a few of the proteolytic peptides bear modifications; however, they do become a significant issue when the detailed structure of a molecule is to be determined..

III. NETWORK-BASED ACCESS TO INFORMATION

A. Computer Networks

Computers have been connected together with communication lines for many years. Originally, these lines allowed the transfer of data from one computer to another, or from a terminal to a particular computer. The network consisted of a continuous wire connection between the computers and terminals, taking the name "network" from the electrical engineering word that refers of a set of interconnected wires.

Modern usage of the term "computer network" refers to a much more complicated arrangement of computers, wires, and/or fiber-optic cables and software [29]. The wires themselves are no longer "the network": They are now the method for "connecting to the network." In a typical, modern analytical laboratory, the same set of wire cabling will carry AppleTalk, Microsoft, and Internet protocol network messages. Telephone lines can be used for the same purpose. The word "network" no longer has a precise meaning out of context; it is used in conjunction with other words to fully describe what type of technology is being used to communicate between computers. For the purposes of this discussion, "network" designates the combination of cables, software, and computers necessary to send a message from one place to another.

B. The Internet Protocol

At the same time that protein sequence information proliferated, a revolutionary development in computerized network information access made it possible for everyone to have nearly instant access to that information. This development was the creation of a simple, standardized protocol for individual computer networks to communicate with each other. With the widespread use of this protocol, any computer that can be attached to a telephone line can connect to a local network that communicates with any other publicly accessible network. The concurrent development of powerful, inexpensive computers and mass data storage devices has lead to a rapid rate of change in information storage and access technologies. To someone working with these rapidly changing technologies, William Gibson's description of using a future global information access system as the projection of disembodied consciousness into a consensual hallucination seems to have come true.

The protocol for internetwork communication that has given rise to this consensual hallucination is called the Internet Protocol (IP) [30]. While the popular media frequently discusses the "Internet" as though it is a physical object that can be examined and judged, it does not exist at any level other than the imagination. The Internet Protocol was designed to allow any computer network to exchange information with any other computer network, regardless of the architecture of the networks communicating with each other. Therefore, using IP, network software companies are free to maintain the proprietary nature of their local-area network (LAN) or wide-area network (WAN) systems without affecting IP communication. IP deals strictly with networks; the characteristics of individual computer operating systems are not included in the protocol specification. Each computer must have a layer of translation software that mediates its communication with IP-compliant message. This layer is called the transport control protocol (TCP): it intercepts all IP messages and reformats them in such a way that a computer's operating system can interpret the message. The combination of TCP and IP software on a computer is called a "TCP/IP stack." The standard nature of the inputs and outputs from this stack makes it much simpler for programmers to write software that deals with network communication, rather than having to write interfaces for a large number of proprietary network specifications.

The Internet Protocol's success is based on three things. First, it is free. It was developed for the U.S. government and the use of the protocol does not require the payment of royalties. Second, it is straightforward to implement on any computer or network. As any user of UNIX will attest, the combination of free availability and straightforward implementation leads to software popularity, regardless of the obvious shortcomings of the system [31]. Finally, IP addresses the problem of transporting information between two networks over unreliable physical connections (such as telephone lines) that may be slower or faster than the networks that are trying to communicate. IP does this by breaking up any data into a set of small, formatted blocks that are referred to as "packets." These packets contain the information about where the information is going and how they are to be reassembled into the whole message when they are received. These packets are sent out from the source computer, through a local-area network to a special computer that is called a "router." The router computer examines each packet that arrives and reads its destination address. The routing computer will be able to send the packet to the correct computer if it is within its own LAN; otherwise, it will pass the packet on to another router. The packet continues to pass from router to router until one is found that has the destination computer attached to its LAN. The destination computer's TCP/IP stack then reassembles the message from the packets it receives and passes that message up to the computer's operating system. The process of passing packets through routers that have lookup tables that allow them to guess where to send the message next means that individual routers do not need to keep an

exhaustive list of all of the computers in the world on IP-compliant networks, which greatly simplifies the maintenance of the system. Packets are free to take any path available between two networks that is currently available; routers are free to make decisions about which path will be fastest or the most reliable on a packet by packet basis. Therefore, even if an intermediate pathway is interrupted or busy during the transmission of a message, subsequent packets are free to find other ways to their destination without loosing data. Because individual data packets can be switched to different routes based on load, it is possible to squeeze much more information down a physical component (such as a fiber optic) than can be done using a protocol that requires a continuous connection, such as is made with a conventional telephone call or FAX message.

C. Client–Server Interactions

Internet Protocol data packets can be used by a variety of software that communicates with the TCP/IP stack on a computer. By convention, part of the address for a computer specifies a "port" number, which determines what piece of software receives the information from the TCP/IP stack [10]. The file transfer protocol (FTP), hypertext transfer protocol (HTTP) and the simple mail transfer protocol (SMTP) refer to popular methods for sending information to and from the TCP/IP stack. When information is being dispensed using these protocols, even though many router computers are necessary for the message to be passed along, only the sending and the receiving computer are important. These computers are referred as either being a "client" or a "server." A client software on the "client" computer requests information or action from the server software on the "server" computer, which responds appropriately to the request. Figures 4 and 5 illustrate typical configurations for hypertext and e-mail transmission using the appropriate combination of client–server protocols and IP networking.

It must be stressed that the client and server designations refer to software running on a particular machine rather than to any special characteristics of a particular computer or operating system. A single computer may be acting as a client and a server simultaneously, depending on what software it is running. Because requests for service can occur at any time, server software usually runs continuously in the background on a computer. Operating systems that can run programs simultaneously (referred to as "multithreaded" systems) are appropriate for running server software because they can respond immediately to requests. Operating systems that allow multithreading, such as UNIX or Microsoft's Windows NT, are therefore the most common choice to run server software. Single-threaded operating systems, such as the Macintosh OS, can be used to run server software but the performance of the server can be seriously compromised by the operating system.

The most flexible of these client–server software combinations is the hypertext transfer protocol (HTTP). The protocol evolved around the idea of

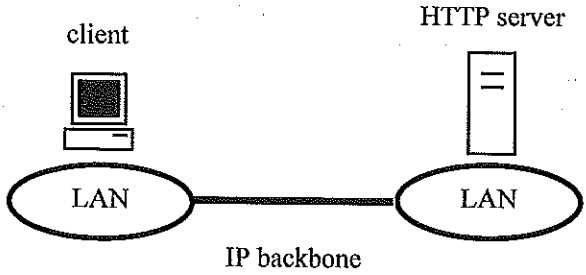

Figure 4 Typical network arrangement for the World Wide Web client–server relationship. The client computer is attached to a local-area network (LAN) of some configuration. The client's browser software makes a request using the HTTP format, which is routed out of the LAN, through the IP backbone (which can be local or international in scope). The HTTP request is finally received by the server, which is attached to its own LAN. The server then responds to the client, sending its message back to the client over the same IP backbone, even though the physical routing of the message may be very different.

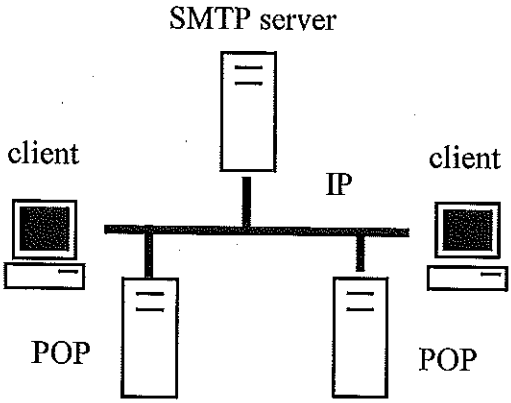

Figure 5 More complicated client–server configuration, commonly used to transmit e-mail. To send e-mail, a client machine will send a request to a simple mail transfer protocol (SMTP) server. The request contains the e-mail message and the address of the recipient's post office protocol (POP) server. The STMP server then sends the message to the POP server. When the recipient "checks" its e-mail, the client machine makes a request of the messages currently stored by the POP server, which responds to the request by sending the messages to the client.

being able to transfer elaborately formatted text pages to client software and to use highlighted portions of that text on the browser screen to request other pages of text from a specified server. A page with this property is referred to as "hypertext." The IP address of the new page and the location of that page on the server computer's mass storage device are easily specified using a simple text formatting language (the hypertext markup language, HTML), so anyone can create pages and place them as files on a computer running server software. HTTP has become ubiquitous because the protocol is royalty-free, it is straightforward to implement, and the client software (called a browser) is easy to use and either free or very inexpensive.

D. CGI, Databases, and Protein Analysis

In addition to being able to transfer pages of hypertext, HTTP has been extended to allow the client to request the server to run another piece of software that will accept some parameters from the client, perform some operation using those parameters, and respond to the client using HTML-formatted hypertext as output. The mechanism that the server uses for communicating with this additional software is called the common gateway interface (CGI), frequently referred to as "cgi-bin" because of a quirk in the conventional way for naming UNIX directories. The original HTTP specification has been extended to allow the inclusion of pictures along with text, and it also allows the server to respond to a request in such a way that the client will start a piece of software (called a "plug-in" or "helper" application) to receive and interpret the information that the server pushes back to the client. The clever combination of server-side CGI software and client-side browsers and plug-ins allows for very sophisticated interaction between the two computers in ways that were extremely difficult in the past. Rather than the clumsy process of manually "logging on" to a remote computer, running a remote program, downloading the results of that program, and reformatting it in such a way that it could be displayed on a local computer, HTTP hides the entire interaction behind a secure, easy-to-use interface that produces attractive displays.

The use of CGI programs to manipulate databases held on a remote server is potentially the most powerful application of network-based computing to protein analysis. Maintaining accurate protein sequence databases and developing sequence analysis tools is a time-consuming specialized task that is beyond the average computer user. Therefore, databases are held on servers that can be queried by users from anywhere that they can obtain a network connection and any computer capable of running a browser, including the newest generation of television receivers. Some programs exist that still attempt to perform database searches locally; however, this type of computing is a temporary transition step between isolated computers and network-aware software. Even the

mention this type of database computing will hopefully seem quaint by the time this book is published.

E. Helpers and Plug-Ins

The CGI programs that have dominated the discussion so far revolve around programs that run on a remote server, but that receive input and "push" output back to a browser running on the user's local computer. Programs also exist that run on the local computer too, but are linked to the browser in such a way that when certain types of data is pushed at the browser, the browser responds by starting these other programs to deal with it. Applications of this type are called "helpers" if they run as separate, stand-alone programs, or they are called "plug-ins" if they require the browser to run. Many of these applications exist for multimedia data types, such as compressed audio and video formats. Many common programs can be used as helper applications, because the mechanism used by current browsers to start a helper is to download the data to be read into a file and then start up the helper with a command line parameter that points it to the appropriate downloaded temporary file. Therefore, any program that creates a specific type of file can be used as a helper application to display and manage downloaded files of compatible types. For example, a word processing application can be used as a helper to view documents created with any compatible word processor.

Unfortunately, mass spectrometry has no commercially available plug-ins or helper applications made specifically for that purpose. The necessary multipurpose Internet mail extension (MIME) codes necessary to start these programs in response to server data pushes have not been agreed upon. Each mass spectrometer manufacturer uses its own data format, and the manufacturers do not seem to have grasped the importance of people viewing data at remote sites. These data formats do not generally economize on data storage, resulting in very large files that are impractical to send over current networks without unacceptible delays. The programs that are supplied for the analysis of mass spectra in relation to known sequences are also based around the idea of an isolated computer running only the manufacturer's software.

One group (R. J. Lancashire, C. Muir, and H. Reichgelthas) has developed a helper application around the JCAMP-DX data format [32], which is available for Win16 and Win32 operating systems [33]. The program was designed for mass spectra obtained from small molecules, but it will certainly deal with protein mass spectra. Unfortunately, mass spectrometer manufacturers have specifically chosen not to support the JCAMP-DX format, so converting spectra into this format may not be easy. Instead of JCAMP-DX, manufacturers have opted for the ANSI standard OFS file format. This file format does not support any type of data compression, resulting in extremely large data files for

many types of protein and peptide mass spectra. These files are not suitable for network exchange because of their size, but at least the data format is platform independent (i.e., the byte order of the data is the same regardless of the operating system that was used to create the file).

The authors of this chapter have also developed helper applications for dealing with protein mass spectra and sequences, the Protein Analysis Worksheet (PAWS) [34] and "m/z" [35], and they are available for Win16, Win32, and Macintosh operating systems. "m/z" is a mass spectrum viewing program that natively uses a highly compressed data format that was designed for data warehousing of large numbers of complex protein mass spectra [1]. It can accept files in the native formats used by most time-of-flight mass spectrometer manufacturers. PAWS is a protein sequence analysis tool that uses fuzzy-logic-based rules to interpret the results of protein mass spectroscopy experiments. PAWS can read data from any of the commonly used protein sequence database entry formats.

IV. RESOURCES OF PROTEIN MS INTERPRETATION

A. General Considerations

Software produced by scientific laboratories is almost always written by graduate students or postdoctoral fellows with little or no formal training in software engineering principles. The programs involved are written to solve particular problems associated with the research in a laboratory and are normally written to be used by the author. The code that results is usually very specific for a particular operating system (OS), because the authors use the details of the operating system to optimize the performance of the software. Distributing this type of software to other laboratories is very difficult: Even slight differences between the setup of superficially identical computers will cause unexpected behavior from the program. This problem is exaggerated because most student software is written for the UNIX OS because the operating system itself has the character of a student project. Different versions (or flavors) of UNIX may appear to be the same, but the small real difference may be so difficult to track down that it is a waste of time for the average users. Most users are also very intolerant of unexpected behavior (such as the program flashing unintelligibly worded error messages), and most student authors are not interested in supplying detailed support for their software (ascribing all errors to problems between the chair and the keyboard rather than their software). Distributing and maintaining a piece of software that is easily used and installed requires the dedication of a many people, ranging from small groups to huge corporate entities.

The CGI coupled with cheap, standardized World Wide Web browsers has alleviated almost all of the difficulties associated with amateur software.

Rather than sending code out to a user to install on the user's own computer (which must be of the same type as the program was developed upon), the program is written so that it can accept input using CGI and produce output that is compatible with HTML. The program can run on the computer it was developed and debugged on, even though a user's input and output was performed on a remote computer of any type or configuration. Therefore, in order for the authors to use the program themselves, they will eliminate all of the bugs that other users will see, rather than relying on the authors' altruism to remove bugs that they may not be able to properly reproduce on their own systems. HTML is a fairly simple page formatting language, making it easy for authors writing programs to produce elegantly formatted output pages quickly. The combination of the complicated nature of the tools for producing visually pleasing output in any OS and the almost magical ability to use the program from any place on earth (at least theoretically) makes CGI a very attractive interface.

The prevalence of "amateur" software is a great strength of network-based computing for scientists. The software tends to address problems associated with the cutting edge of scientific research and it need not address a problem that has any commercial motivation. If the piece of software is truly useful, then the student will obtain gratifying positive reactions from people all over the world and be encouraged to improve the program and to remove as many of the bugs that existed in the initial release as possible.

The prevalence of "amateur" software is also a tremendous weakness of network-based computing for scientists. The programmers rarely have any input from the nonprogrammers that actually use the software, and they have no particular reason to listen to any customer feedback that they receive. Consequently, the user interfaces that allow access to CGI programs tend to be an afterthought rather than the result of considered design. Consequently, the interface tends to be ideal for debugging a program (which is of great concern to the developer) rather than use of a program by an inexperienced user [11]. Documentation for the program is usually some combination of incomprehensible and incomplete, mainly because there is no conceivable justification for having a student or postdoctoral fellow spending time writing program documentation when the person could be working on the software.

The following passage is a good example of the curious type of documentation that currently exists for CGI programs:

> EXPECT
>
> The statistical significance threshold for reporting matches against database sequences; the default value is 10, such that 10 matches are expected to be found merely by chance, according to the stochastic model of Karlin and Altschul (1990). If the statistical significance ascribed to a match is greater than the EXPECT threshold, the match will not be re-

> ported. Lower EXPECT thresholds are more stringent, leading to fewer chance matches being reported. Fractional values are acceptable. (See parameter E in the BLAST Manual).

This clearly incomprehensible passage was taken from the online documentation for BLAST (Basic Local Alignment Search Tool) and it explains the most important numerical parameter for performing a search with the program. An inappropriate selection for the EXPECT parameter will result in a search that will not find sequence similarities that exist, depending on the value of other parameters and the length of the sequence fragment used to perform the search. It is not our intent to single out this piece of text out as an example of bad program documentation; it is a typical example. BLAST is probably the most used piece of biopolymer sequence searching software, and it is used every day by hundreds of molecular biologists who depend on the accuracy of the results obtained from its searches. With this said, almost no one who uses the program really knows what values to use for BLAST's parameters to optimize their particular search. Typically, they use the default parameters and hope for the best. Note: We have been unable to find the reference to "parameter E" in any of the documents associated with BLAST, either through HTTP- or FTP-accessible documents.

An underappreciated feature of professional software is "version-control"; that is, different versions of the software are released at long time intervals and the particular version of the software is easily determined by the user. When the output of a program is reported in a publication, it is possible to write down the version number of the software used, so that it can be examined in the future in light of any bugs that are subsequently discovered in that program. CGI programs almost never carry any version information with them, and they are changed often without warning by the developer. Therefore, there is no definitive way to determine whether a known bug has affected your calculation or not. Even when a CGI program does report a version number, it is difficult to determine with certainty whether that version number is correct or whether changing the version number was forgotten during the last build process.

B. Link Lists

Hypertext pages that are lists of other hypertext pages on a particular subject are the simplest type of network-accessible resources. These pages allow users to rapidly obtain information about a particular subject, relying on the judgment and diligence of a particular list's author. Many lists exist, although the most comprehensive one currently available is maintained by Kermit Murray at Emory University [36]. This list is so extensive that it has its own "search engine"—a CGI program that searches through all of the links available to find the ones that most closely match some search phrase.

V. THE FUTURE—DISTRIBUTED COMPUTING

The model for a desktop computer operating system that was current in 1995 was an easy-to-use, easy-to-configure graphical user interface that allowed users to run single copies of locally held applications. That is, the only data storage/data retrieval operations that an application would participate in would be using its own mass storage devices (magnetic disks, CD-ROM, etc.) or through a network that made remote disks behave as though they were local disks. All computer applications were written to take advantage of that model.

The 1997 model for a desktop computer operating system is completely different. It is now thought that the operating system on any desktop computer should expect to interact with an IP network in a nearly transparent manner. The physical location of disks (i.e., their location on a particular LAN configuration) will no longer be of much consequence. In fact, applications will interact with servers, rather than at the hardware level of dealing with actual devices. Using common, standard methods for addressing these servers, it will be possible to break monolithic computer applications up into much smaller parts, referred to as "objects," which can be assembled into a wide variety of software built from a common set of components. This approach is currently used in software development, but all the applications are meant to run under one operating system. In the new scheme, all programs built of these objects should be compatible with all computers, resulting in a great time saving for the development of applications. These software objects will be available to your local computer in a variety of ways: Some will be on a local hard disk, while others will be downloaded to your computer and used as needed.

Several different software and hardware development groups are vying for the lead in this new style of operating system. An operating system independent computer language called Java is the most favored candidate for producing these objects, while a scheme called JavaBeans will allow Java objects to communicate with each other in a straightforward manner. Java objects (also called "applets") are currently available and they run inside of the current generation of World Wide Web browser software. JavaBeans has just been proposed and it is not yet fully implemented in widely available platforms.

It is anticipated, however, that the combination of IP-network-aware operating systems, simple client–server interactions, and applications built of reusable and easily available objects will revolutionize data-intensive undertakings, such as bioinformatics [37]. Traditional bioinformatics, with centralized databases and large expensive computers, will probably be replaced with a large number of specialized servers that are much less expensive and that can be tuned for specific tasks. Applications that can take advantage of this new diversity of options and opportunities will hopefully make it possible to finally get an overall view of the vexing problems in protein chemistry, such as predicting

folding, selective proteolysis, and understanding the role of sequence divergence in molecular evolution.

REFERENCES

Note: All of the universal resource locators (URLs) listed here were tested and operational in March 1997.

1. D Fenyö, W Zhang, BT Chait, RC Beavis. Internet-based analytical chemistry resources: a model project. Anal Chem 68:721A–726A, 1996.
2. R Elmasri, SB Navathe. Fundamentals of Database Systems, 2nd ed. Menlo Park, CA: Addison-Wesley, 1994, pp 611–700.
3. URL: http://expasy.hcuge.ch/sprot/sp-docu.html.
4. URL: http://www.gdb.org/Dan/proteins/pirusersdoc.html.
5. URL: http://www.ebi.ac.uk/ebi_docs/embl_db/embl_db.html.
6. URL: http://www.ncbi.nlm.nih.gov/Web/Genbank/index.html.
7. URL: http://www.biochem.ucl.ac.uk/bsm/dbbrowser/OWL/OWL.html.
8. URL: http://www.ncbi.nlm.nih.gov/dbEST/index.html.
9. WJ Henzel, TM Billeci, JT Stultz, SC Wong, C Grimley, C Watanbe. Identifying proteins from two-dimensional gels by molecular mass searching of peptide fragments in protein sequence databases. Proc Natl Acad Sci USA 90:5011–5015, 1993.
10. M Mann, P Højrup, P Roepstorff. Use of mass spectrometric molecular weight information to identify proteins in sequence databases. Biol Mass Spectrom 22:388–392, 1993.
11. DDJ Pappin, P Højrup, AJ Bleasby. Rapid identification of proteins by peptide-mass fingerprinting. Current Biol 3:327–332, 1993.
12. JR Yates III, S Speichner, PR Griffin, T Hunkapiller. Peptide mass maps: a highly informative approach to protein identification. Anal Biochem 214:397–408, 1993.
13. P James, M Quadroni, E Carafoli, G Gonnet, Protein identification by mass profile fingerprinting. Biochem Biophys Res Commun 195:58–64, 1993.
14. HH Rasmussen, E Mortz, M Mann, P Roepstorff, JE Celis. Identification of transformation sensitive proteins recorded in human two-dimensional gel protein databases by mass spectrometric peptide mapping alone and in combination with microsequencing, Electrophoresis 15:406–416, 1994.
15. M Mann, M Wilm. Error-tolerant identification of peptides in sequence databases by peptide sequence tag. Anal Chem 66:4390–4399, 1994.
16. JR Yates III, JK Eng, AL McCormack. Mining genomes: correlating tandem mass spectra of modified and unmodified peptides to sequences in nucleotide databases. Anal Chem 67:3202–3210, 1995.
17. SD Patterson. Matrix-assisted laser-desorption/ionization mass spectrometric approaches to the identification of gel-separated proteins in the 5–50 pmol range. Electrophoresis 16:1104–1114, 1995.
18. SJ Cordell, MR Wilkins, A Cerpa-Poljak, AA Gooley, M Duncan, KL Williams, I Humprey-Smith. Cross-species identification of proteins separated by two-dimen-

sional gel electromphoresis using matrix-assisted laser desorption/ionization time-of-flight mass spectrometry. Electrophoresis 16:438–443, 1995.
19. KR Clauser, SC Hall, DM Smith, JW Webb, LE Andrews, HM Tran, LB Epstein, AL Burlingame. Rapid mass spectrometric peptide sequencing and mass matching for characterization of human melanoma proteins isolated by two-dimensional PAGE. Proc Natl Acad Sci USA 92:5072–5076, 1995.
20. A Shevchenko, ON Jensen, AV Podtelejnikov, F Sagliocco, O Vorm, P Mortensen, H Boucherie, M Mann. Linking genome and proteome by mass spectrometry—Large-scale identification of yeast proteins from two dimensional gels. Proc Natl Acad Sci USA 93:14440–14445, 1996.
21. URL: http://prowl.rockefeller.edu/PROWL/pepfrag.html.
22. URL: http://prowl.rockefeller.edu/cgi-bin/ProFound.
23. URL: http://rafael.ucsf.edu/MS-Fit.html.
24. URL: http://rafael.ucsf.edu/mstag.html.
25. URL: http://www.mann.embl-heidelberg.de/Services/PeptideSearch/FR_PeptideSearchForm.html.
26. URL: http://www.mann.embl-heidelberg.de/Services/PeptideSearch/FR_PeptidePatternForm.html.
27. URL: http://cbrg.inf.ethz.ch/subsection3_1_3.html.
28. URL: http://gserv1.dl.ac.uk/SEQNET/mowse.html.
29. W Hioki. Telecommunications, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1995, pp 330–374.
30. C Hunt. TCP/IP Network Administration. Sebastopol, CA: O'Reilly and Associates, Inc., 1992, pp 6–49.
31. S Garfinkel, D Weise, S Strassmann. The UNIX-Haters Handbook. Indianapolis, IN: IDG Books Worldwide, 1994, pp 43–61.
32. URL: http://members.aol.com/rmcdjcamp/faq.htm.
33. URL: http://wwwchem.uwimona.edu.jm:1104/software/jcampdx.html.
34. URL: http://www.proteometrics.com/software/paws.htm.
35. URL: http://www.proteometrics.com/software/mz.htm.
36. URL: http://tswww.cc.emory.edu/~kmurray/mslist.html.
37. A Eccleston. Java needs to put bio into bioinformatics. Nature Biotech 15:315, 1997.