Bioinformatics and Database Tools for Glycans

by

Eric Zachary Berry

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Masters of Engineering in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2004
[June 2004]

Author ..[illegible]..
Department of Electrical Engineering and Computer Science
May 18, 2004

Certified by ......[illegible]..
Ram Sasisekharan
Professor
Thesis Supervisor

Accepted by .....[illegible]..........
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Bioinformatics and Database Tools for Glycans

by

Eric Zachary Berry

Submitted to the Department of Electrical Engineering and Computer Science
on May 18, 2004, in partial fulfillment of the
requirements for the degree of
Masters of Engineering in Computer Science and Engineering

Abstract

Recent advances in biology have afforded scientists with the knowledge that polysaccharides play an active role in modulating cellular activities. Glycosaminoglycans (GAGs) are one such family of polysaccharides that play a very important role in regulating the functions of numerous important signaling molecules and enzymes in the cell. Developing bioinformatics tools has been integral to advancing genomics and proteomics. While these tools have been well-developed to store and process sequence and structure information for proteins and DNA, they are very poorly developed for polysaccharides. Glycan structures pose special problems because of their tremendous information density per fundamental unit, their often-branched structures, and the complicated nature of their building blocks. The GlycoBank, an online database of known GAG structures and functions, has been developed to overcome many of these difficulties by developing a common notation for researchers to describe GAG sequences, a common repository to view known structure-function relationships, and the complex tools and searches needed to facilitate their work. This thesis focuses on the development of GlycoBank. In addition, a large, NIGMS-funded consortium, the Consortium for Functional Glycomics, is a larger database that also aims to store polysaccharide structure-function information of a broader collection of polysaccharides. The ideas and concepts implemented in devloping GlycoBank were instrumental in developing databases and bioinfortmatics tools for the Consortium for Functional Glycomics,

Thesis Supervisor: Ram Sasisekharan
Title: Professor

Acknowledgments

This thesis never would have been possible without the guidance and support of Rahul Raman, David Berry, Ganesh Venkatemeran and Ram Sasisekharan. In addition, Howard Chou provided numerous insights and contributed tremendously to the GlycoBank implementation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The successful sequencing and analysis of several genomes, combined with the increased use and potential of microarray technology, has begun to truly reveal the action of proteins within the cell. Modern research increasingly highlights the importance of protein-protein and protein-polysaccharide interactions for regulating enzymatic activity *in vivo*. Contrary to its newfound importance, the nascent field of glycomics, studying the role of polysaccharides as agents in cellular signaling and enzymatic modulation, has not experienced rapid growth. The analogous fields of genomics and proteomics, however, have matured rapidly.

The development and implementation of an online repository for known polysaccharide structure-function data using a single, unified nomenclature is required in order to capture the complexities of carbohydrate structures. Because carbohydrates possess such tremendous variability in their composition, a very powerful and extensible system is required to capture this information and allow for future modifications. Furthermore, to fully comprehend the complex structure-function relationships of polysaccharides, as well as the manner in which they interact with proteins, specialized computational frameworks must be developed in conjunction with the nomenclature. Because no truly standard notation presently exists, it is very difficult to provide analysis on the corpus of existing knowledge as specialized parsers are required for the different formats used by each research group.

GlycoBank has been created to address these needs. It utilizes the above frameworks and is also complemented by several bioinformatics tools that supplement them with additional computational environments. The repository and its associated tools will provide a large informatics framework to accelerate discovery of novel structure-function relationships of polysaccharides. A recent, large-scale initiative funded by the National Institute for General Medical Sciences (NIGMS), The Consortium for Functional Glycomics, has been created with the similar goal of studying and disseminating polysaccharide structures and functions. Many of the techniques, algorithms, and data structures developed in creating the repository, tools and searches were further utilized by applying them on a larger scale to the Consortium for Functional Glycomics.

1.1 Background and Significance

This project focused primarily on heparin sulfate glycosaminoglycans (HSGAGs), a major class of glycosaminoglycans (GAG). HSGAGs are highly active in modulating cell growth [1, 5]. They regulate fibroblast growth factors (FGF) as crucial agents ensuring proper expression[7]. In particular, researchers have demonstrated that HSGAGs modulate the FGF1/HS/FGFR1 signaling complex [10]. HSGAGs have also been shown as crucial components involved in cellular signaling pathways. [2]

Polysaccharides like HSGAGs predominantly reside in the cell surface and extracellular environment and as constituents of proteoglycan complexes. HSGAG sequences vary in length, usually extending between 20 and 100 disaccharide units, and have no branched linkages. The basic HSGAG disaccharide repeat unit consists of two monosaccharides, glucosamine and uronic acid. This unit can vary at five different positions, unlike the one position of variability found in amino acids. A schematic representation of the HSGAG disaccharide is shown in figure 1-1. Because of the variable positions, the fundamental units can assume up to 48 distinct building blocks, excluding enzymatic alterations. This information density far exceeds that of DNA or proteins and there are too many basic structures to use one alphanumeric character per disaccharide unit.

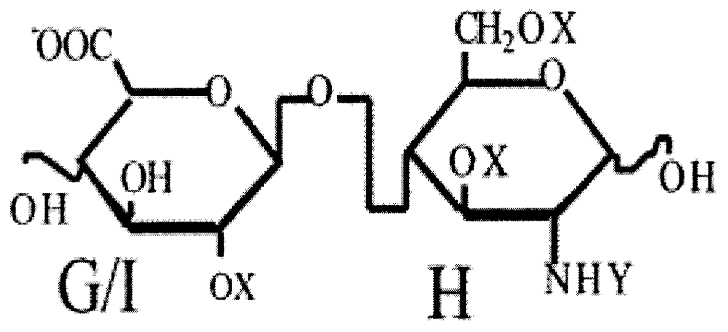


Figure 1-1: Schematic representation of the HSGAG dissacharide repeat unit. Each position labelled with an "X" has two possible states - sulfated or not. A "Y" indicates three possibilities - sulfation, acetylation or free amine. The uronic acid can be either glucoronic or iduronic acid.

The tremendous quantity of information and the nature of the variabilities in the disaccharide repeat unit require special consideration and make the study of HSGAGs more complicated than their protein or nucleic acid counterparts. The number of variabilities and the nature by which they vary from one another must be maintained when creating a unified notation. Furthermore, all decisions made on HSGAG notation must account for the need to perform extensive sequence analysis.

1.2 Specific Aims

To address the challenges of storing polysaccharide structures and developing the tools to interface with this information, this thesis has focused on developing the GlycoBank. The nature of the information required to store the HSGAG class of polysaccharides is discussed in chapter 2. Data entry, storage and curation and data models follow in chapter 3. The implementation and architecture of searches is discussed in chapter 4 and that of bioinformatics tools in chapter 5. The manner in which work for GlycoBank was adapted to the Consortium for Functional Glycomics, a large-scale National Institute of General Medical Science (NIGMS) effort with a more broad goal, is discussed in chapter 6.

1.2.1 Central Database (Chapter 3)

The effective dissemination of the accumulated efforts of disparate researchers requires a centralized data store using a single, standard notation. Otherwise, it proves more difficult to enforce the usage of a single nomenclature. The notation used by GlycoBank must be concise and simple enough that it promotes usage, yet it must also retain enough flexibility and power that it can represent all forms of polysaccharides. The development of this nomenclature must precede database efforts.

The database itself has to use the nomenclature that was developed for HSGAGs, which will be discussed in chapter 2. It must also maximize availability of the data and extract insightful relationships ascertained from sequences it contains. The internal storage format used by GlycoBank for sequence information will use a format other than the sequence nomenclature described in chapter 2 if it proves more efficient. All web interfaces must present only the standard notation. Also, like most internet-accessible databases, GlycoBank must be developed with the heuristic that it reads, specifically searches, far more frequently than it adds new data.

The sequences will be added to the database by researchers through a web-based interface that must appeal to their diverse backgrounds. HSGAGs are entered as three distinct classes: deterministic sequences, probabilistic sequences and mixtures. Each requires a user-interface that is sufficiently powerful to accommodate all possible variations yet retains enough simplicity that it is approachable to non-computer specialists. The data model can be arbitrarily complicated and even require substantial overhead to populate as long as this is translucent to the user.

1.2.2 Complex Polysaccharide Property Searches (Chapter 4)

After researchers populate the database with sequences, and the relevant structure-function information, they require a powerful and flexible means for searching. Users of internet databases have grown accustomed to searching based on a large number of parameters with speed and accuracy. Internet users typically demand results within twenty seconds and make exceptions only when it is obvious why their search takes longer, like BLAST. GlycoBank must therefore perform even the most complicated searches as quickly as possible.

The searches available through GlycoBank include chemical motif, mass, NMR readings, chemical composition and enzyme digests. These searches, especially the enzyme digest, require tremendous amounts of computation. Furthermore, as the datasets grow, the complexity of these searches will render an unoptimized search impractical. Special care must be taken to optimize these searches during their implementation.

1.2.3 Polysaccharide Tools (Chapter 5)

The data that exists in the database presents researchers with a useful common storage for existing structure-function data. GlycoBank, however, also aims to augment existing knowledge beyond the use of searches. A variety of bioinformatics tools operate on the nomenclature developed for the HSGAGs and the information in the database. These computational tools include facilities to allow HSGAG sequencing and multiple-sequence motif-aligners with noise tolerance. While tools of this nature for HSGAGs do not presently exist outside of GlycoBank, optimizations are required to make these extremely computationally difficult jobs complete quickly enough that they are useful to the user.

1.2.4 Consortium for Functional Glycomics (Chapter 6)

The Consortium for Functional Glycomics is a large-scale initiative which also focuses on furthering increasing structure-function knowledge of polysaccharides. Much of the software written for GlycoBank and the practices developed in creating the website can be usefully applied to the Consortium. Ideas pertaining to data security, searches, tools, web-site navigation and user access have all been successfully applied to the consortium.

Chapter 2

HSGAG Sequences

The information density of HSGAG sequences requires the development of a nomenclature to concisely represent the makeup of each fundamental unit. The HSGAG sequence notation must indicate the value of the variability at each position. Furthermore, enzymes modify the reducing or non-reducing end of the HSGAG sequences in a number of distinct manners. The structural changes in the monosaccharides caused by the effects of any such reaction must be represented in the nomenclature as well. The notation must also account for the sequences observed in nature that deviate from the theoretical compositions assumed. Because research on HSGAGs has not concluded that the set of known modifications and structures is complete and new ones continue to be found, the proposed nomenclature must also allow for arbitrary additions without requiring extensive changes in the software that has already been developed.

2.1 Property Encoded Nomeclature

The Property Encoded Nomenclature (PEN) has evolved substantially through the course of this project. Initial projections, made before the GlycoBank project began, foresaw thirty-two distinct disaccharide structures [9]. The ease with which standard hexadecimal notation represented these possibilities resulted in an elegant PEN based on base-16. As the number of theoretical structures increased to forty-eight, the need to maintain backwards compatibility with the old sequence notation resulted in a system in which the hexadecimal system continued as the core, but modifiers were added to account for the new units. In its present incarnation, the PEN can represent all forty-eight possible structures, which are described in table A.1, and 8 different monosaccharide modifications. Depending on the length of the sequence and the position of the disaccharides in the sequence, a single HSGAG repeat unit can assume well over 200 distinct flavors.

This initial PEN required a human-readable notation. This notation also proved useful for computational work on the sequences. As the complexities grew and number of modifiers increased, it became clear that no notation could be both human-readable and conducive to computational studies of the sequences. Because people had already grown accustomed to the PEN, a separate optimized internal representation was developed and will be discussed below. The term human-readable notation will hitherto be used interchangeably with PEN. The computer-based notation shall be referred to as the binary notation.

An arbitrary chemical structure does not necessarily correspond to only PEN sequence. Any given PEN, however, describes only one chemical structure. Because enzymatic modifications may eliminate a position of variability, many different sequences may have led to the chemical structure. This could only be resolved by having a completely different nomenclature for every enzyme-affected disaccharide. This solution is unreasonable, so the many-to-one compromise was allowed.

The PEN itself follows the general formula of a set of required brackets enclosing the base disaccharide unit, followed by any modifications in parenthesis. The modifications must be comma-delimited if there is more than one. The PEN follows, left-to-right, describing the sequence towards its reducing end. Therefore, [D(du,man)][7(man)][-4][D(anh1)] describes the sequence with the disaccharide backbone of D, 7, -4, D. The first D, found at the non-reducing end of the GAG oligosaccharide, has a delta 4,5 unsaturated uronic acid (du) instead of the normal uronic acid, and a mannose instead of the glucoronic acid. A complete listing of the sequence modifications and where they may happen in the sequence is found in table A.2.

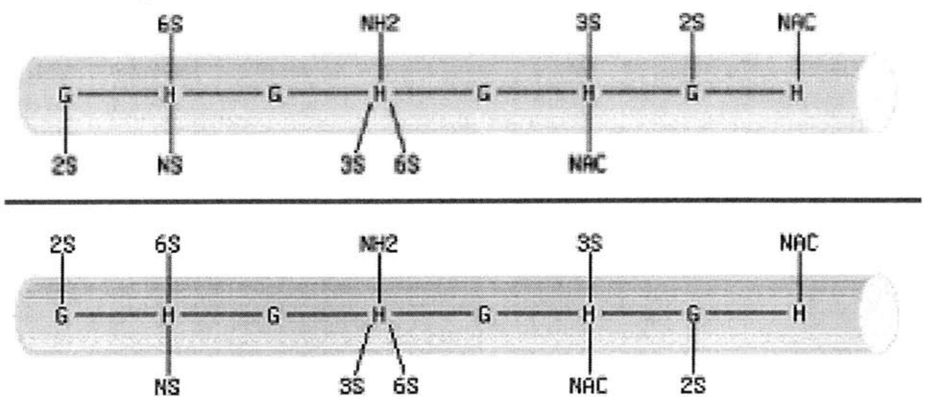


Figure 2-1: Schematic representation of the two conformations of the HSGAG sequence [-D][-7n][-2][-8].

As described above, certain enzymatic modifications nullify the difference in chemical structure between two otherwise distinct disaccharides. For example, [D] ordinarily differs from [-D] because the former has iduronic acid while the latter, glucoronic acid. [D(du)], however, is an isomer with [-D(du)] because both types uronic acids have become delta 4,5 unsaturated uronic acid. While this may cause confusion, it is allowed because the differences in notation imply knowledge about the sequence from which this modified sequence was derived. Distinct PEN sequences without modifications, however, are never isomers. Chemically identical HSGAG sequences may be found using the binary notation.

Every HSGAG sequence has two possible conformations in space. Each U-H and H-U bond has a regular spin that remains constant between all monosaccharides. The orientation of the first U-H bond, however, may assume one of two positions. Therefore, every otherwise unique PEN sequence actually describes two distinct chemical structures. The two conformations for the sequence [-D][-7n][-2][-8] are illustrated schematically in figure 2-1. The rotational component, however, is often unknown or unnecessary. This information is also not conveyed by chemical formulas. Both rotational components of the structure are also included as part of the binary notation.

2.2 Binary Notation

The binary notation consists of several 32-bit integers, where each bit denotes a specified property of the sequence. Using bitwise "and" operations, an arbitrary feature set of an HSGAG can be trivially extracted. The binary notation consists of two distinct groups of integers which combine to describe the complete chemical makeup. The first set describes the chemical motif of the sequence while the second denotes the enzymatic modifications on a disaccharide basis, as well as all the rotational information.

The chemical motif data includes all the information pertaining to the 5 positions of variability: I/G, 2, 6, 3 and N. Every position has two variabilities, and can be represented with one bit, except for the N position. This has three possibilities and so it requires two bits to represent all possible states. An additional bit distinguishes a disaccharide represented by all 0's from trailing space in the sequence word. The seven required bits per disaccharide allow for up to four disaccharides per 32-bit block. The remaining four bits are presently unused. The second word includes all the modifiers as well as the orientation information. Each piece of information is always found in a certain position for every sequence. This modification word stores one bit for every disaccharide in the sequence and indicates whether the glucosamine is a mannose contracted ring. All the other modifications listed in table A.2 can occur only at the beginning or the end of the sequence and therefore require only one bit per modification word. To store the orientation, 2 bits are required per disaccharide. The relative orientation between the 2-sulfation to the 6-sulfation is stored.

2.3 Future Modifications

During the development of GlycoBank, researchers have made several discoveries about the structure of HSGAG sequences, many of which initially caused irreparable inconsistencies with the data structures used. The PEN and binary notation have therefore both been designed to handle nearly any reasonable modification to the standard HSGAG structure. The PEN is handled through a specialized parser that inspects each disaccharide for its syntactic validity and can extract the feature set for use within the server software. This parser can easily be modified to handle future rules. The binary data scheme is even more flexible. Originally designed to maximize the computational efficiency, this architecture has seen a compromise in the speed at which it performs various tasks. In return, however, the binary notation will not need substantial modifications at any point in the future. This compromise came in the form of the second word containing the enzymatic modifications. Originally, all information was stored in a more compressed form, using only one word per 4 disaccharides. Extracting the full chemical makeup of a sequence now requires at least twice as many operations.

Chapter 3

GlycoBank Data Entry

GlycoBank is foremost a collection of the known structure-function relationships in HSGAG sequences. The rest of the application depends on the information collected by the data entry and stored in the database. The entry process has been refined numerous times as knowledge about linear polysaccharides develops. In its present iteration, the data entry interfaces and the data structures to which they feed have been designed to allow for simple and modular upgrades should new structure types or models be discovered. While this sacrifices performance, the optimizations made for data-heavy operations, as discussed below, will render this minor overhead insignificant. Presently, the data entry has been completed only for HSGAGs. As with most web-based applications, all data entry tools were developed under the assumption that the database would be written rarely and read frequently, so no optimizations were made to increase the speed of data entry. Rather, they were all focused on maximizing the availability of the data once it is in the database.

3.1 HSGAG Data Entry

HSGAGs may be entered into the GlycoBank as either sequences or mixtures. Both data types have associated references to literature, biological and chemical classifications and images that yield pertinent information. The different classes of HSGAGs are stored separately because of the vastly different nature of the data the two types of entries produce in the database. The consistency between the different classes is that every HSGAG must have at least one primary reference and classification in order to be viewable by the public.

3.1.1 Sequences

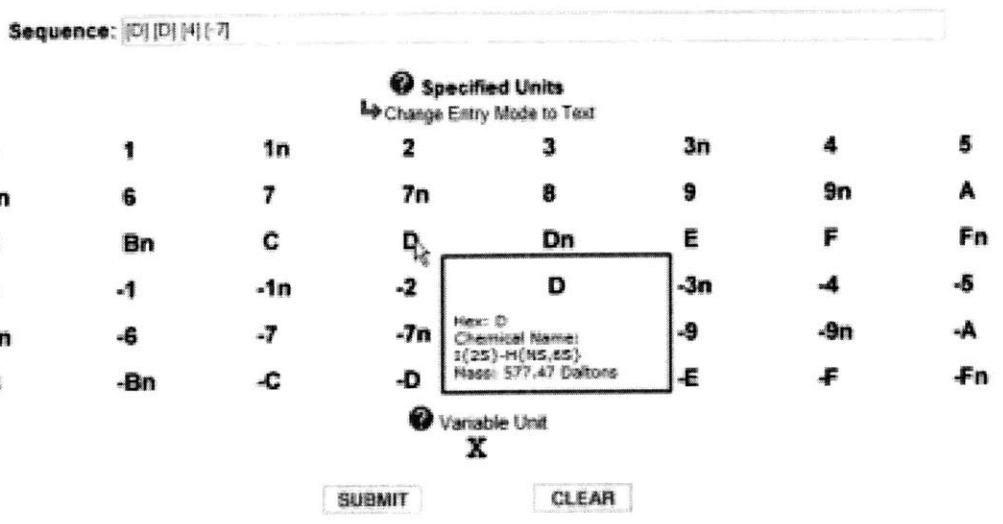
The HSGAG data entry accepts both completely determined sequences as well as those whose sequencing has produced uncertainties at various positions of variability. The former sequence type has been labeled as deterministic, while the latter is referred to as probabilistic. Both sequence classes describe a single HSGAG sequence with varying degrees of precision. When entered from the internet, both classes of sequence are initially entered from the same user interface to clarify their relationship. If the user indicates variability at any disaccharide position in the sequence, the server initiates the probabilistic entry, otherwise the deterministic sequence entry completes and the user is prompted to enter classification and reference data.

Deterministic Sequences HSGAG sequences in which every position of variability within the polysaccharide has been found with a high degree of confidence are deterministic sequences. The difficulty associated with sequencing HSGAGs to such precision necessarily implies that these entries occur with a relatively high frequency. Users enter deterministic sequences through an intuitive interface in which he clicks on a display that presents an exhaustive list of the 48 possible disaccharide units. A screen capture of this interface is shown in figure 3-1. The GUI displays either the complete chemical name or the PEN of each disaccharide, based on the preference of the user. Furthermore, while the mouse hovers over the name of the disaccharide, a more detailed information box appears in order to reduce ambiguity for those migrating to the PEN system. Clicking on the disaccharide assembles the sequence from its non-reducing end.

Upon submission, the sequence is analyzed by server-side software that analyzes the sequence and presents the user with the option for adding enzymatic modifications where they are possible. Mutually exclusive modifications are presented as radio inputs while the others are selected using checkboxes. The software also checks to ensure that enzymatic modifications that are not inherently mutually exclusive, but may be part of a set of exclusive relationships that cannot be entered for the same disaccharide. This cannot be represented with HTML forms and, without this check, would otherwise produce meaningless results. For example, the user cannot enter both du and monou because the du would be modifying the monosaccharide that has removed from the sequence. In fact, no modification of the first disaccharide can be selected with monou.

Each deterministic sequence can exist only once in the database. Uniqueness is enforced by lexically matching the disaccharide sequence including all the enzymatic modifications. Because the modifications are added to the sequence notation by the server, they are always added in the same order. Thereforem [D(du,man)] will never be confused for [D(man,du)]. While different sequence and enzymatic modifications may produce the same chemical structure, GlycoBank does allow for the same structure to exist multiple times in the database. These enzymatic modifications often describe the sequence as a subsequence of a longer HSGAG. Therefore, distinct subsequences, even if they are isomers, imply additional information that provides heuristics about the nature of the sequence from which they were derived. If there is an exact lexical match while entering an HSGAG, the server informs the user that the sequence has previously been entered. He may then choose to add new references or classifications to the original sequence object. The details regarding access control of this operation will be explained in a subsequent section.

Probablistic Sequences Because HSGAG sequencing frequently results in at least one irreconcilable ambiguity, users often must resort to entering probabilistic sequences[9]. For example, uncertainty in the uronic acid of a one-disaccharide sequence, with knowledge of every other variability in the disaccharide, would yield two equally likely sequences. For example, $[D]$ or $[-D]$ could result from aforementioned ambiguity. If both possibilities are equally likely, they both have a probability of 0.5. Further knowledge of the sequence may bias the probabilities.



Sequence: [D] [D] [4] [-7]

D
Hex: D
Chemical Name:
I(2S)-H(NS,6S)
Mass: 577.47 Daltons

Figure 3-1: Sequence entry user interface for both deterministic and probabilistic sequences

Probabilistic sequences are first entered through the same interface as deterministic sequences. If the user selects at least one disaccharide in the sequence to be a variable position, denoted in the entry screen with $[X]$, the sequence is defined as probabilistic. An eight disaccharide sequence with variabilities in the first and sixth disaccharide, for example, could be entered as the sequence $[X]\,[D]\,[D]\,[4]\,[-7]\,[X]\,[9]\,[1]$. After entering the sequence, the user again enters the enzymatic modifications, which are verified and appended to the sequence nomenclature. Therefore, the above example could become $[X\,(du)]\,[D]\,[D\,(man)]\,[4]\,[-7]\,[X]\,[9]\,[1]$ after adding the modifications. This sequence would be used to as the parent sequence for the probabilistic HSGAG entry. The user must then define each of the five positions of variability in every probabilistic disaccharide. The interface used to specify these positions is shown in figure 3-2. To limit the amount of variability and the number of child sequences, only seven or fewer total variabilities are allowed.

To specify the characteristics at each position of variability, GlycoBank creates a dynamic input screen with a schematic representation of each probabilistic disaccharide. The positions that are completely defined by modification affecting that disaccharide, such as the N position in a disaccharide with mannose contraction, are not editable by the user in this screen. Every other feature of the disaccharide must

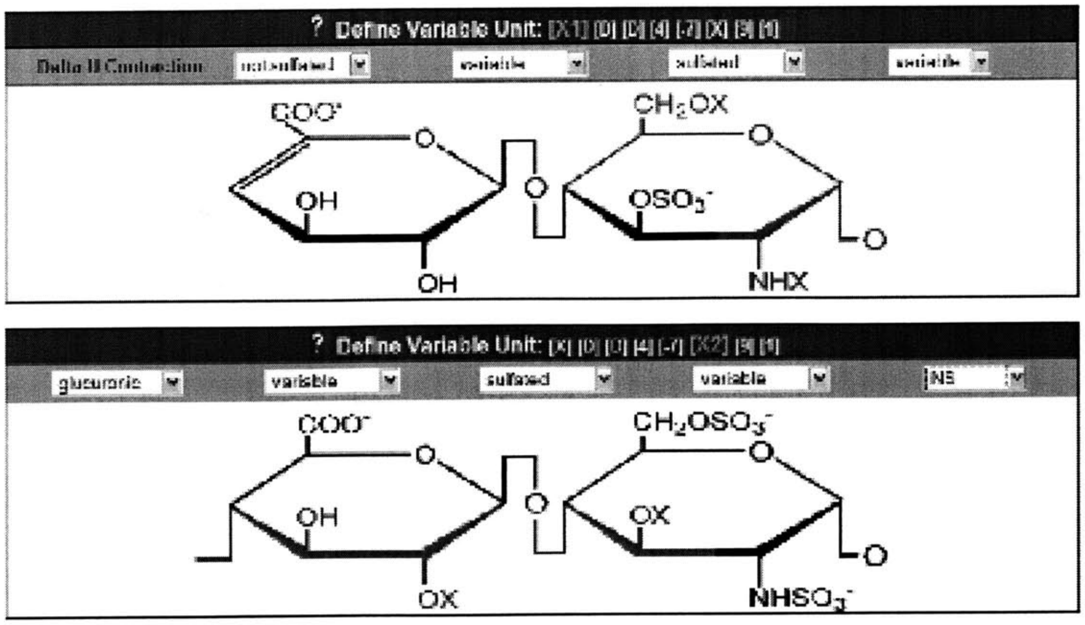


Figure 3-2: Capture from the probabilistic the data entry variability specification screen.

be specified as either variable or defined from a drop-down menu with its appropriate deterministic value. Naturally, each probabilistic disaccharide must have at least one position that does not have a deterministic value.

Upon submission, the software iteratively generates all possible sequences that could be formed based on definitions of the probabilistic disaccharides from the previous step. The user subsequently assigns each sequence with a probability representing the likelihood that it is the actual sequence. If the user has no knowledge of the sequence beyond the variabilities, each sequence is given a probability of 1/n, where n is the number of possible sequences that the probabilistic parent could assume. For two probabilistic sequences to be identical, the parent sequence and probabilities of every child sequence must match. If a user enters a sequence that exactly matches one already in the database, he can add new references or classifications, like with deterministic sequences.

While probabilistic sequences store numerous sequences per parent sequence, they refer to only one actual sequence and therefore are not considered mixtures. Each child sequence represents one possible reality of the parent sequence along with a probability that it is the correct sequence. Searches can therefore ignore the low-probability children of probabilistic sequences without missing important data in the database.

3.1.2 Mixtures

Beyond being unable to fully specify every position in an HSGAG sequence, researchers often cannot separate the constituents of a mixture of HSGAGs. The sequence information of the mixture, therefore, is often unavailable [8]. Using a variety of experimental methods, including capillary electrophoresis, mass spectrometry, nuclear magnetic resonance analysis, and enzyme degradation profiling, researchers ascertain information about the members of the mixture.

To enter a mixture, the user must select which experiments he has performed from a menu of known experiments. GlycoBank iteratively presents him with a series of specialized input screens to accept the data specific to each experiment. The specific data entry screen for an HSGAG mixture that has been analyzed with capillary electrophoresis, for example, displays a form to accept the relative abundance of each disaccharide, all of which have ambiguity in the uronic acid. Users may upload up to one image for every experiment performed to supplement the numerical data entered. Regardless of the results of each experiment, every new mixture must automatically be considered unique because further experiments might have revealed that the two otherwise identical mixtures actually contained different constituent sequences. Every mixture must also have associated references and classifications before being viewable by the public. The associations are stored in the same table as the sequences. Thus, all references and classifications can easily be associated with both classes of HSGAG objects.

Each experiment is considered a child of the mixture as a whole. Several mixtures have constituent sequences that are identifiable and can be sequenced either deterministically or probabilistically. For this type of mixture, the user must enter the sequence as a regular HSGAG sequence. The known sequence may then be associated with the mixture through a mapping table and this sequence may then be considered a child of the mixture.

3.1.3 Users

GlycoBank will only accept data from registered users who have logged in before performing the data entry. This measure ensures that all information has a verified source. If further clarification of the data is required, the appropriate questions can be directed to the knowledgeable party. To register with GlycoBank, users must enter their personal information, including their e-mail address, institution, and full name, in an online application. The application is then forwarded to GlycoBank administrators who may approve or reject the user. After the administrators approve the user, the database activates the user account and sends a response to the supplied e-mail. After confirming that the e-mail address is valid, the user may begin to enter data.

Users of GlycoBank must perform all data entry through an encrypted communication channel to ensure that sensitive data remains private. Because all users must log in to access the secure, data-entry pages, and because a password is also sent during login, the user login is therefore also encrypted. While the GlycoBank utilizes the session manager that is included in mod_php to maintain data on the server, the system has been constructed such that a malicious eavesdropper on the network that obtains this insecure session cookie cannot circumvent the security. An additional secure cookie is sent to the client machine during login. This secure cookie contains a large, randomly-generated number. Every time the user attempts to connect through the secure channel, this authentication number must be sent as well. According to their specifications, secure cookies are never sent through unencrypted communications channels by the client browser, so it is not vulnerable to packet sniffing. Php sessions can be used on a per-user basis to view past searches performed by that user, for example. A compromised insecure cookie will not lead to a comprise of the security guaranteed to the users of GlycoBank, namely that all data entry and editing is performed only by trusted sources.

The administrators responsible for approving a user can also change the class of that user to administrators. Similarly, curators promote other users to have curation privileges, and the programming team alone can give other users access to the otherwise hidden API documentation. Users that are members of multiple user classes can promote users to any of the classes of which they members. It is currently assumed that administrators and curators will not act maliciously. There is thus no straightforward method to revoke the privileges of a user, should such an action be required. If it eventually proves necessary, this can be performed on a case-by-case basis by the IT team. Sensitive information about users, such as passwords, is encrypted in the database and is therefore not viewable even by the IT team that manages the database.

3.1.4 Literature References and Biological/Chemical Classifications

Every HSGAG in the database, regardless whether it is a mixture of a sequence, has a key that serves as the key for everything associated with the object. The key for deterministic sequences maps to a single sequence in the object table while probabilistic and mixture entries map to more complicated data structures. Each ID in the object table also maps to the references and classifications associated with it to provide additional data about the sequence. These keys allow this additional sequence metadata to be seamlessly integrated with the sequence regardless of its nature.

Literature references and biological/chemical classifications are each considered their own objects. A reference is entered into the database and is given a unique identification number which serves as its key. This key is generally not viewable by users of GlycoBank, but an HSGAG object may then be associated with the reference through a mapping table pointing to this key. This allows several sequences to point to the same reference and one sequence to use several references. Classifications follow the same model. For convenience, users can add a new reference to an existing HSGAG in one step. This process automatically creates the reference object and then adds the object-sequence relationship to the mapping table. A similar convenience has been implemented so that users can enter classifications in one step.

Images can also be directly uploaded to the database. Every image must be associated with an existing reference. Each reference can have any number of images. The images themselves, however, can only be associated with one reference. Thus, multiple copies of the same image might exist in the database. The difficulty in checking for identical image matches and the complexities that would necessarily propagate into the data model to represent copies of images with dynamic permissions led to this compromise.

By directly mapping the reference and classification keys to the HSGAG key, similar sequences can be found by performing a simple search through the mapping table for those HSGAGs that share a significant number of references or classifications. Similarly, noteworthy references may be found by finding sequences that shares a substantial number of references with a target HSGAG and returning the references to those sequences that are not shared with the target HSGAG.

Each HSGAG is also assigned a primary reference and classification from the references and classifications to which it maps. These primary entries default to the first chronological association made by the user that originally entered the HSGAG. This user may then select any reference and classification associated with the sequence to be the primary reference and classification.

3.2 Curation

To ensure that the GlycoBank contains only verified HSGAG, reference and classification information, all entries must be approved by human curators. Each entry has a curator-approved bit, which always defaults to "false" upon entry. An object is not publicly visible until this attribute becomes "true".

Curators are appointed by the existing curators of the GlycoBank and are generally prominent members of the linear glycosaminoglycans research community. When a user enters any object, a sequence, mixture, reference or classification, it is initially private. The final screen of the data entry process displays a selection where the user can indicate that the entered sequence will be publicly viewable within two weeks or less. This screen is presented only if the HSGAG has both a primary reference and a primary classification. The user may return at later time and indicate that the sequence has become, or within two weeks, will become publicly viewable if it was not at the time of the initial data entry. Again, this requires that the sequence have both a primary reference and a primary classification. Once this user-approved attribute is set, the HSGAG immediately enters the pool of objects to be curated.

When any curator next logs in, he will be presented with an exhaustive list of the curatable objects. A system in which curators would be notified by e-mail of pending curation duties has been considered and may be implemented in the future based on feedback from actual curators. Based on the class of sequence that the curator chooses to approve, he is presented with appropriate input screens. In each screen, the curator can choose to approve or reject the object. Approval by the curator automatically sets the curator-approved bit to "true". Rejection does not delete the object. Rather, the user sees the rejection and a note explaining the rationale behind the curator choosing to reject it. The user may opt to edit the object and resubmit it. Otherwise, the object remains in the database but is no longer viewable or usable in any manner. The data remains to ensure that the same invalid data is not entered again.

When a curator approves a sequence or a mixture, the primary reference and classification are also approved automatically. The curator may choose to approve all the publicly viewable references and classifications associated with the sequence, or may individually approve and reject them. After the HSGAG enters the public domain, the curator must approve each new reference or classification that is subsequently associated with it. Each reference and classification requires its own curation only once, even if it is associated with multiple HSGAG objects because the object itself becomes approved, not the mapping.

3.3 Access Control

Sequences, mixtures, references and classifications often exist in the database well before they are released publicly. For example, researchers may choose to enter a probabilistic sequence and its reference before the publication has actually been released in order to ensure that the data is available when the data appears publicly. The material is therefore copyrighted by the publisher and may not be freely distributed. After the release date, information about the sequence may be cited in the database. Before that date, however, care must be taken to ensure that this sequence is not viewable by anyone except the author.

Each object, including every parent HSGAG, reference and classification, has a publicly-viewable attribute. As with the curator-approved bit, an object is viewable by all users only after this bit has been set to "true". The user that entered the object retains view and editing privileges until the user-approved bit described above is set by the user himself. At this point, the user may not edit the entry again unless it is rejected by the curator. When entering any HSGAG, the user is given the opportunity to set the user-approved attribute to false. When this bit is set to false, only the user that entered the sequence, when logged in, can view the sequence. The publicly-viewable bit is automatically set to true two weeks after a curator approved any object. This gives the user a two-week window between when it becomes submitted for curation and when it enters the public domain.

Each object - every parent sequence, reference and classification - has a publicly-viewable attribute. As with the curator-approved, the object can only be viewed by all users when this is set to "true". The user that entered the object retains view and edit privileges until the user-approved bit described above is set. At this point, the user may not edit the entry again unless it is rejected by the curator. When entering any HSGAG, the user is given the opportunity to set this user-approved attribute to false. When this bit is set to false, only the user that entered the sequence, when logged in, can view the sequence and it remains editable. The publicly-viewable bit is automatically set to true two weeks after a curator approves any object.

3.4 Data Storage

The most important aspect of GlycoBank database is the data storage. It has been designed to minimize the latency of searches and tools and to maximize the availability of the data. This optimization often creates a large up-front cost. To handle the potential delay during data entry, every sequence or mixture in the database has a "ready-bit" that indicates, when set to 1, that all necessary actions to introduce the HSGAG into the database have completed. After the sequence or mixture entry has completed from the perspective of the user, the server spawns a process that runs in the background and independently completes the data entry process. This process finally sets this bit to 1 when it has finished. Actions such as search hashing and the compositional analysis of the sequence are performed by this background process as they often take several seconds or more of processor time. Until this ready attribute has been set, all data-driven applications effectively cannot see the new sequence.

3.4.1 Sequences and Mixtures

The sequences table in the GlycoBank database includes all the administrative information about any HSGAG sequence. Similarly, a mixtures table retains this information about HSGAG mixtures. This information includes the molecule class, the user that entered the sequence, the date of its entry, its curation and other access-control bits, and the ready bit. All of this information can be joined to any query and thus can be used to easily filter out sequences that are not viewable based on the above criteria. The same SQL statement may be used through all searches to add this functionality and thus the access control is simply a modular element that is added to all queries.

3.4.2 Linkage Matrix

Defining characteristics of an HSGAG polysaccharide sequence are often obtained by examining its disaccharide linkage profile. By comparing the linkage profile of any sequence against the known linkages found in the linkage matrices, one can develop heuristics about the frequency of that sequence occurring in nature and the likelihood of it occurring in a probabilistic sequence. Sequencing can also be biased towards the known linkages, as will be described in the Tools chapter.

Linkages have been separated into two distinct varieties, one-dimensional and two-dimensional. One-dimensional linkages are the linkages between neighboring disaccharides. Two-dimensional linkage are the linkages from one disaccharide to its neighbor and then on to that disaccharide's neighbor. The two-dimensional linkages, therefore constitute three different disaccharides and have a substantially greater number of possibilities.

Both one- and two-dimensional linkages are stored for all deterministic and probabilistic HSGAG sequences. In addition, both left- and right- linkages are stored to speed various types of queries. When a sequence is entered into the GlycoBank, each direct disaccharide linkage is obtained, omitting any enzymatic modifications. In the future, alternate linkage profiles will be developed that include this information. The deterministic HSGAG sequence $[-D\,(du)]\,[D\,(man)]\,[4]\,[-7]$ has a right linkage profile of:

$[-D]\,,[D] = 1$

$[D]\,,[4] = 1$

$[4]\,,[-7] = 1$

The left linkage profile is computed in a similar fashion, but uses the opposite linkages. Because there are only 2034 possible left and right linkages, without modifiers, both one-dimensional linkage tables are stored directly in database tables with an enumeration for every possibility. The value of every linkage pair corresponds to the number of times that linkage has occurred in the sequences in GlycoBank. Each reference that is added to a sequence also adds another occurrence of every linkage in the HSGAG. Probabilistic sequences count every linkage in every child sequence. Each probabilistic linkage counts not as 1, like in deterministic sequences, but as the probability of the child sequence in which it was found. Each of the references for each deterministic sequence increment each linkage in that sequence by a total of 1. References added to probabilistic parent sequences increment every linkage of all the child sequences by the probability of the child sequence. While this method does not precisely indicate the frequency of linkages in nature, it gives a reasonable estimate of the linkages that occur frequently and which are rare, if they occur at all.

Constructing and populating the two-dimensional linkage matrix proves somewhat more difficult. Analysis of the known HSGAG sequences showed that only a fraction of the possible two-dimensional linkages were ever used. It would therefore be very inefficient to construct a single table with enough columns to contain all 110,592 two dimensional linkages. A compromise was constructed such that each right- and left-link table for two-dimensional linkages is named with the starting position of the linkage. The table then describes the 1-dimensional linkage from that initial disaccharide. The tables are dynamically created when as each new one is needed. This provides for more flexible searching and improved performance. The population of these tables follows the same rules outlined for one-dimensional linkages.

3.4.3 Composition Profile

Every HSGAG sequence in the database, both probabilistic and deterministic, also stores an extensive composition profile. When the sequence is entered, the background data entry process automatically performs a compositional analysis of the sequence or sequences. Several searches have been built to interrogate this table and so it has been constructed to maximize the efficiency of these searches.

The mass is computed and stored by analyzing the structure of each disaccharide. The glucoronic and iduronic profile is stored, along with the sulfate, acetate and free amine counts. The polysaccharide sequence is analyzed in terms of the monosaccharide content and its various enzymatic modifications.

The pattern of sulfation in the sequence is also stored in the composition profile. The sulfation pattern contains only the number of sulfates positions in each disaccharide, not their positions. Therefore, the sulfation pattern stored as 1-2-0-1 describes a four-disaccharide sequence in which the first and fourth disaccharide have one sulfation, the second has two, and the third, zero. This sulfation pattern can be found in a very large number of different polysaccharide sequences. The composition profile also stores the results of several different experiments that are simulated against the sequence. For example, a simulated capillary electrophoresis experiment stores its results as a vector in the same table.

3.4.4 Search Hashing

Many complex searches require substantial computational overhead and the more complicated joins may need to search through several tables multiple times. To minimize the latency of such searches, a cache of frequently used searches has been implemented in the GlycoBank database. Because the searches have been cached, each new entry added to the database must clear the cache of all stored searches. The searches must then be re-executed to populate the cache tables with the newest sequence information. The details of this step will also be explained in the chapter on searches. This is the final step of the data entry process because it requires that all other computed information about the sequence already exist.

Chapter 4

Searching

The search features are among the most important tools for the users of GlycoBank. By allowing the user to quickly and easily manipulate sequences and to search based on readily available and easy-to-enter parameters, he can obtain several new insights about the structure-function relationships of linear sugars. As with the GlycoBank data entry, searches have only been implemented for the HSGAG class of polysaccharides. To ensure the integration of access control, every HSGAG search automatically joins the results with the sequences and/or mixtures table discussed in the Data Entry chapter, to filter for sequences that are viewable.

4.1 Search Parameters

All searches connect to the GlycoBank server through an insecure channel. Because user authentication requires encryption in order to send the secure user cookie, searches are performed without authentication. To preserve the access control of sequences, those that are not publicly viewable are never returned by searches, even by those users that entered them. Users can view their private sequences in contexts other than searches. Using a secure connection for all searches would provide a tremendous overhead to the server as encryption requires additional computation and bandwidth. Searches can be cached for all users if access control is automatically set to the lowest level of access for all users. Otherwise, caching would be a very difficult undertaking, as described in the Consortium for Functional Glycomics chapter.

All searches provide the user with the opportunity to filter low probability sequences out of the results, as those probabilistic sequences most likely do not actually exist. The cutoff for low probability sequences is presently those with a probability of less than 30 percent, but can change as more sequences are added to the database and a better metric for low probability can be determined.

4.2 Light and Advanced

The light search acts as the general search engine for all the data stored in GlycoBank. Users can use traditional Boolean modifiers such as "AND", "OR", and "NOT" to yield more precise matches. The software on the server parses these using the order of logical hierarchy in which NOT precedes AND, which, in turn, precedes OR.

Searches performed by the light search often present several disparate search terms, such as "Sasisekharan OR FGF-2". In this case, one of the terms derives from the author field of the reference table while another which is generally found in the classification table. The light search must therefore return those sequences that are either in a paper authored by Sasisekharan or those that interact with FGF-2. Those sequences that match both must be given a higher score than one that fits only one of these characteristics. Therefore, every search term must be scored independently. Furthermore, every search must cover an extremely large amount of data as nearly every field in both the reference and classification tables is searchable.

To facilitate the light search as it performs many searches on the same tables, full text indexing is performed on the searchable fields during data entry. The indexing occurs on a per-word basis. Therefore, searches need only inspect the index to return the valid sequences. The light search also has a regular expression engine that recognizes the particular formats of sequence IDs and will not consult the text index when it detects any such ID. In this case, the molecule page of the sequence is displayed instead of the search results page.

The advanced search, despite its name, is actually easier for the software to process than the light search. The input screen consists of several text inputs with drop-down menus in which the user can select the field in which this term is to be searched. The user may also select Boolean relations between the various search terms. With the advanced search, the text indexing may not be used because the text index does not store the columns in which each instance of the word was found, only the row. Instead, only those fields that have been indicated in the search page need to be checked. Unlike the light search, no text parsing must be conducted on the search terms because the users enter the Boolean relations in radios in the form rather than as part of the search.

4.3 Sequence

The sequence search allows the user to enter a sequence of disaccharide units using an interface that is very similar to that which is used during data entry for deterministic sequences, shown in figure 3-1. The enzymatic modifications that are selected by the user during the subsequent screen, however, do not necessarily apply to the disaccharides that are listed in the sequence. For example, the user can enter the sequence [7][-4][D], and indicate a du in the first position. He can choose whether this du should correspond to the [7] at the non-reducing end of the target sequence, or whether it can apply to any disaccharide that is at the beginning of the sequence. In this latter case, the sequence [C(du)][0][1][2][D][7][-4][D] would match the search parameters.

HSGAG sequences are stored in two distinct manners in the database. The sequences of probabilistic child sequences are stored with the deterministic sequences so that any fully specified HSGAG in this table may be found by a single-column search through this table. The child sequences and deterministic sequences are stored both with and without the enzymatic modifications in the text of the sequence. Another table has been created in the GlycoBank database in which a key has been created for each different type of modification that is known. A table maps each sequence with the modifications and the disaccharide position in the sequence where they occur. Therefore, the full sequence of [D(du,man)][7(man)][-4][D(anh1)] can be obtained from the sequence [D][7][-4][D] and its mappings to the enzyme modification relationship table.

As described above, if the user selects that the du modifies the [7] in the target sequence, [7][-4][D], the database can perform a string search on the column containing the sequences with their enzymatic modifications in the text representation for the string "[7(du])[-7][D]". This search runs relatively quickly and requires only joins with the probability and permissions tables for the result filters described above. If, however, the search requires that the du modify the first disaccharide of the sequence regardless of whether that disaccharide is the beginning of the target sequence, the server performs a completely different search. The database searches through the column of the sequences table without the enzymatic modifications for the string "[7][-4][D]". After the sequences that contain this string have been found, they are joined with the enzymatic modifications mapping table to search for the sequences have a du in the first position. The sequences that match both criteria are then joined with the probability and permissions tables. The optimization for the first type of search has been implemented because it is assumed that this type of search will be conducted far more frequently than its more complicated counterpart. As the database grows, this optimization may serve to reduce the load on the server.

4.4 Motif

The sequence searches described above allow users to perform simple searches based on the PEN disaccharide notation developed for GlycoBank. Users often do not require that every position of variability in the search target be defined, but rather search only for motifs within a sequence. The sequence search does not provide the facility for any variability in the search target. The motif search, however, allows the user to search the sequences in GlycoBank for a motif of arbitrary length and complexity in linear time using the algorithm described later in this section.

To search based on a chemical motif, the user must first select length of this motif in disaccharides. No hard limit on the number of disaccharides allowed in the motif has been imposed. Counterintuitively, longer sequences actually tend to reduce the load on the server because the SQL query can easily filter out those sequences that are shorter than the length of the motif and thus eliminate a preponderance of the sequences in the sequence table before they are inspected for this complicated motif. After selecting the length of the motif, the user is presented with a screen that is very similar to the variability specification table from the probabilistic sequence entry, shown in figure 3-2. Each position of every disaccharide in the motif must be denoted as being variable, or else the required characteristic must be specified. Enzymatic modifications can be added just as in the probabilistic variability specification. Because the variable positions can lead to a large number of valid disaccharides at the position of variability, performing sequence searches of every permutation would require an exponential number of different sequence searches. Instead, the motif searches use the binary notation discussed previously to perform the motif search in only one pass, which itself is faster than just one sequence search.

Every bit in the binary sequence representation indicates a specific property. This property is consistent for every sequence. The first two binary bits, for example, represent the N-position of the first disaccharide. "00" indicates free amine, "01" acetylation, and "10" sulfation. The next bit represents the 3-position, where "1" means that the position has sulfation and "0", means that it is not sulfated. The 6-position follows the 3-position and uses the same format for storage. The numbers above represent the states of the bits themselves, and not a text representation. The binary representation table stores only integers. A simplified search example concerned only with the 3-, 6- and the N-positions follows:

Supposing a user wants to find all the sequences that have sulfation at the 3-position, no sulfation at the 6-position, and the state of the N-position does not matter. The search software constructs an "interest mask" for the four bits of this abridged motif of `1100`. The two zeros represent the N-position, the position to the left is the 3-position, and left-most 1 represents the 6-position. The "output target" becomes `0100` to indicate that valid sequences have sulfation at the 3-position, no sulfation at the 6-position, and because the output of the N-position has been zeroed by the interest-mask, the results of these two bits are always zero. The search then scans every position of the binary representation of a polysaccharide using a bitwise AND of the sequence against the interest mask and comparing the result to the output target. This is an integer operation which can be conducted extremely quickly on modern processors. The same logic is used to join the enzymatic modifications and 2-dimensional rotational properties with the sequences, if they were included in the search.

Because the above example was an extreme simplification, it does not reveal the power of the motif search and ignores the complexities of the actual implementation. Motifs of five disaccharides in which the user has two or three variabilities per disaccharide would require well over one hundred different sequence searches - string comparisons that are substantially more computationally expensive than arithmetic operations. The motif search, meanwhile, would require only one search. While the motif does have to be checked against every possible position in every disaccharide, the comparisons are integer operations which can run at or faster than the clock speed of modern CPUs. The variabilities are all accounted for by the interest mask and output target and thus the search time is dependent only on the number of sequences in the database and their average length. Two sample SQL statements of more complicated searches are shown in Appendix B.

The motif search must also gauge the affinity of sequences to a particular motif. Sequences that have more matches to a particular motif must therefore be given a higher score. Adding this functionality in the queries shown in Appendix B would add undue computational difficulty to the search. A subsequent query on only the matching motifs is used to score each sequence and thus return them in order of affinity.

4.5 Mass

The mass search presents the user with input boxes for a target mass and the allowed tolerance. This mass search itself simply inspects the table of HSGAG sequences and finds all those sequences whose masses fall within the given tolerance of the target mass. The results are then ordered by absolute value of the difference between their mass and the target mass. Because the mass of sequences is automatically calculated during data entry, the mass search only requires two floating point operations for every sequence in the database. Modern CPUs can interleave floating point add and subtract operations at the speed of the processor, so this search completely extremely quickly.

Beyond this straightforward search functionality, the mass search allows users of GlycoBank to find the sequence compositions that match their mass range, the range of masses that is allowed by the mass/tolerance pair. Generic sequences consist of the chemical composition of sequences, including all enzymatic modifications, and do not contain information about the ordering of the sulfations and other properties. Therefore, a tetrasaccharide sequence with three sulfations in the first disaccharide and none in the second matches has the same generic sequence as one with one sulfation in the first disaccharide and two in the second. To calculate these generic sequences, GlycoBank initially finds the range of sequence lengths that could fall into the tolerance. Generally, tolerances are less than 10 Daltons, which, for shorter sequences, allows for only one or two different disaccharides in length. As the sequences grow in length, this rule degenerates.

Logic has been included to ensure that impossible combinations of modifications and other sequence properties are not presented as valid results. For example, certain enzymatic modifications reduce the number of possible sulfations or acetylations that can be in that sequence. Optimizations that define the interaction of sulfations and acetylations, as well as others sequence properties, have been included to reduce the search space for valid masses. This search presents a list of results that link directly to the chemical composition search through the search API. Therefore, one can find all the generic compositions that match a certain mass and use this information to directly view the existing sequences in the database that match these generic compositions.

4.6 Chemical Composition

The chemical composition search allows the user to explore the HSGAGs in the database through a number of different lenses. First, the user can search through these sequences based on their chemical properties. For example, a user can find all the sequences in the database that have a length of 10 disaccharides and where 20 positions within the polysaccharide are sulfated. A search for these parameters can be considered in three distinct manners: as an exact match, a search looking for only sequences that precisely match 10 disaccharides in length and 20 positions of with sulfation; as a near-match, any sequence with a length of about 10 disaccharides and roughly 20 sulfations; or as a ratio match, a search for any sequence with same ratio of sulfations per disaccharide as the target sequence. Even with this simple search, there are at least 3 different types of searches that can be returned. The user can choose between these three options, but all searches default to "near match" searches because this behavior is found on most major bioinformatics web applications.

The three chemical composition searches described above can search through a number of properties. The user essentially enters a vector of numbers through a web interface that represent the various searchable properties. These properties include mass, the number of iduronic or glucoronic acids in the uronic positions of the sequence, the number of sulfations, free amines and acetylations, the number of each type of disaccharide and monosaccharide, and the length of the disaccharide.

When performing the exact match search, the database scans through each field and returns only those that have the precise match for each position in the composition vector. This requires only checking for integer matches. This search is used primarily with the generic mass search where exact matches are implied by the results display of the generic sequence composition.

Near-match searches use the square of the differences between the target values and those found in the database. Presently, the value that is squared is simply the difference between the value in the database and the target value, regardless of the nature of the property or its magnitude. In the future, these properties may be modified to allow users to specify the relative importance of the fields and to weight the results based on their magnitude.

Ratio searches require that the user specify at least two fields. The field with the greatest magnitude is chosen as the denominator and every number in the vector then becomes the ratio of its value divided by this denominator instead of searching for original integers. The square of the difference of these ratios between every entry in the database and the target ratio is added and is used as the search score.

Ratio and near-match searches require that the database does not use indexing, because mathematical operations must be performed on every value. Databases do not index on the results of arithmetic operations. The searches, therefore, are relatively computationally expensive. Every value must be checked for every search.

4.7 Enzyme Digest

For users attempting to sequence an HSGAG, GlycoBank provides a facility to find the known sequences that fit the maldi profile resulting from the enzyme digest. This can narrow the search space tremendously for the user. The database provides this facility through a search screen in which users enter the properties of the enzyme that they used as well as the mass peaks that resulted from the maldi readings of the enzyme digest.

The enzyme digest search assumes that only one sequence is represented by the mass peaks of the maldi profile. Because the peaks can come from different fragmentations of the same sequence, the only *a priori* constraint that can be placed on the mass of the sequences in a search is that it must be at least as large as the mass of the largest peak, and the sum of the smalles two peaks listed, if three or more are listed. All of the remaining sequences must undergo computational enzyme digest during the search.

Every enzyme can be characterized with both a motif of polysaccharides that represents its binding site and the specific action it takes upon this domain in the sequence. The specifics of digesting a sequence are explained in more detail in the Sequencing section of the Tools chapter. Basically, every sequence is digested into every possible set of fragments based on the properties of the enzyme. If a set of subsets of the possible fragmentations of a sequence completely covers all the masses in the profile and this set of fragmentations has no masses that are not in the profile, the sequence is a valid match.

The ranking of this search is arbitrary because a sequence can either match the enzyme profile experiment or it can not match it. Sequences cannot do this with any more intensity than any other sequence. Therefore, the ordering of the sequences is presently determined by their sequence ID. In the future, this may be changed to reflect the sequence scores of all the matching sequences. Therefore, the sequence that is more likely to occur in nature will be returned first and those that appear very rarely are returned last. Sequence score is described in more detail in the Sequence Scoring section of the Tools chapter.

4.8 Search Caching

Modern databases like PostgreSQL optimize their internal data structures only on the data that is stored in the tables. For example, indices can be created on the length of a polysaccharide because this is simply a number that is stored in the length column, so the database can store an optimized table that points to all the instances of each value. All queries requiring a sequence of a certain length need only consult this index to return the proper sequence ids. To query which sequences have twice as many positions of sulfation as their length, however, the indices no longer prove useful because every length field and every sulfation field must be consulted individually and compared to one another and to the desired ratio. Therefore, all searches must check every entry in the columns of interest. Every such advanced search cannot be optimized using databases alone.

As the database grows and more users simultaneously perform complicated searches on the database, a search taking several second to complete will yield unacceptable performance. To address this issue, the more computationally difficult searches are stored so that they do not need to be performed every time a user enters the same query. The enzyme digest search, the most computationally difficult search in GlycoBank, experiences the greatest performance increase from search caching.

Every time a user performs an enzyme (or any cached) search, all the characteristics of the search are stored as a hash such that each hash string corresponds to only one type of enzyme search. The counter for this particular hash string is incremented in the database. The hash for that search will be added if it did not already exist. The database therefore keeps track of the number of times each different difficult search has been performed.

When searches reach a minimum threshold, they become treated as cached searches. The threshold is adjustable and will depend on the usage of the database, the architecture of the server, and the performance of the application with increased traffic. Cached searches store the hash string and counter, and also save a "fresh" bit along with a vector of the IDs in the database that result from the particular search, in the order in which they are returned, and the code necessary to re-perform that particular search. To maintain the freshness of the search cache, every time a new sequence enters the public domain, the search cache must be refreshed.

When a new sequence enters the public domain, the "fresh" bit of every search stored in the search caching table is reset to 0. The code for entering a sequence into the public domain also triggers a background process that refreshes the cache table. Conducted in the order of the frequency with which the cached searches have been entered, the process will perform each search again and include all the new sequences in the database. When the vector of valid IDs is updated, the fresh bit is once again set to 1. The assumption behind this method of search caching is that sequences will be added to the public domain infrequently and in batches. If, as GlycoBank evolves, this is seen to not be the case, a new method of search caching will be adopted.

Every time a search is performed, the cache must be checked. This requires creating the search hash and checking it against the search cache table. If the search exists in the cache, the fresh bit must be 1 for the values in the cache to be used. Otherwise, the search must be performed independently. Searches cannot wait for the caching to complete if the fresh bit is 0, as it may take several minutes before the current search has been completed by the background process.

4.9 Molecule Pages - Display Information

The results page of every GlycoBank search includes links to the molecule pages of every matching polysaccharide sequence. The molecule page is meant to provide the user with more detailed information about the sequence and to possibly lead to more insights about the composition or function of the molecule. This page, therefore, must reveal all possible information about the molecule in as intuitive a fashion as possible.

Beyond simply stating the characteristics of the sequence, including its name, mass, and ID number, the GlycoBank molecule pages reveal every publicly viewable reference and classification that is associated with the molecule. The references frequently have links to the article if it has been published online. Beyond simply presenting these relationships, however, a facility for searching on the common words or phrases in the references and classifications associate with the sequence is being developed. Therefore, if several of the references have "FGF-2" in the title, it follows that this molecule interacts somehow with FGF-2. Therefore, the facility to automatically link this molecule page to a search for every molecule that also deals with FGF-2 would prove very useful to the user.

The molecule page must also display similar insights about the chemical composition of the sequence. This page therefore has several graphs to indicate various properties of the HSGAG. The disaccharide composition graph that is included in every molecule page is a histogram of the different disaccharide units that are in the sequence. The user may then use this information to search through the composition search for sequences with identical compositions. Further enhancements include a line graph that displays the sulfation pattern through the sequence. This again leads to a search to reveal all the sequences that match this profile.

Perhaps most importantly, the molecule page also presents the user with a table with information about the one-dimensional linkage profile of the sequence. This resembles a discrete Ramachandran plot where the axes are the left- and right-disaccharides in the linkage instead of the chi- and phi-angles. The linkages that occur most frequently in the database are colored more vibrantly than those that do not occur at all. The linkages in the sequence appear as diamonds in the colored square. By placing the mouse over any position in linkage plot, an informational window will appear that reveals the linkage, the frequency that it occurs in the database, and the frequency that it occurs in the sequence. This information can easily be used to determine if the sequence fits the expected linkage profiles or is otherwise deviant. Presently, this information is not searchable, but will be when the linkage search has been completed.

Chapter 5

Tools

5.1 Sequencing

The difficulty associated with sequencing HSGAGs remains among the most substantial hurdles preventing more widespread understanding of this class of polysaccharide. Because no method currently exists to deterministically sequence every position of variability in a polysaccharide with complete precision, researchers must conduct numerous experiments followed by painstaking computations for several hours. Often, this sort of effort is unfeasible. GlycoBank offers tools to automate the sequencing process and thus promote the active sequencing of HSGAGs.

5.1.1 Capillary Electrophoresis

Most HSGAG sequencing begins with a procedure known as capillary electrophoresis (CE) [6].This experiment produces a mapping of each disaccharide to its frequency in the target sequence. CE cannot resolve the uronic acid positions, because so both [D] and [-D] appear as [D(du)] in the CE results. Using the mappings of the disaccharide to frequency, the sample space of all possible sequences that can be the target sequence can be generated. This exhaustive list requires that every ordering of disaccharides found by CE be considered. Furthermore, every distribution of disaccharide modifications must be considered. This results in computation requiring both space and time of order $\Theta(c^n)$, where n is a property depending on the number of number of disaccharides used and their distribution within the twenty-four disaccharides that are found in the CE experiments. Furthermore, the constant, c, includes a factor of $\Theta(2^n)$ where n is the total number of disaccharides used. This figure derives from the process of creating every possible distribution of iduronic and glucoronic acids for every sequence to compensate for their absence in the CE results.

Presently, enzymatic modifications such as mannose are not supported in HSGAG sequenceing, and they further increase the time and space requirements. For example, adding mannose alone increases the above factor of c to $\Theta(4^n)$. Creating this exhaustive list is a necessary second step when not using an NMR experiment because every possible sequence must be inspected to determine whether it accurately matches the MALDI profile. All optimizations of this process are therefore extremely important, and avoiding this task where possible increases the speed of the sequencing job immensely.

5.1.2 Linkage Matrix Bias

One optimization to reduce the number of possible sequences is biasing their generation against only known disaccharide-disaccharide linkages. As discussed in previous chapters, this information is stored in the GlycoBank database. Because all known one-dimensional linkages are readily available and are small enough to store entirely in memory, this information is easily used to reduce the number of possible sequences. Specifically, the possible sequences are generated using a recursive process in which each of the different unused disaccharides that remain from the CE experiment are appended onto a growing chain, thus branching the chain with a factor based on the number of different types of disaccharides of remaining. By ensuring that every linkage is based on one existing in the database, the branching factor decreases significantly. This procedure often results in a substantial reduction in the number of possible sequences, and may often provide further computational savings. Furthermore, this process does not bias only to sequences that exist in the database, but provides a more flexible system that allows for novel ordering of existing linkages.

5.1.3 Nuclear Magnetic Resonance

Nuclear magnetic resonance (NMR) experiments can be performed by researchers to substantially decrease the space of possible sequences [3]. This single experiment often makes otherwise intractable sequencing jobs possible. The results from the NMR experiment reveals the number of each variety of monosaccharide in the sequence as well as all the H-I linkages found in the sequence. By employing only the number of monosaccharides found in the NMR experiment, the $\Theta(2^n)$ factor described above can be substantially reduced. The order that results from this optimization relies on the distribution of iduronic and glucoronic acids in the NMR results. The assignment of iduronic and glucoronic acids becomes a pigeon-problem as the number of each type of monosaccharide is known, one needs only to consider all the distributions of n iduronic acids and m glucoronic acids, which is (n+m) choose n. The worse-cast optimization drops this factor from $\Theta(2^n)$ to $\Theta\left(\frac{n!}{(n/2)!n!}\right)$. Furthermore, the use of NMR can often completely resolve the uronic acid of the first monosaccharide in the sequence, which is completely unattainable using only the enzyme digest method.

The H-I linkages from the NMR experiment also provide a tremendous benefit when creating the list of possible sequences. In addition to using only the remaining disaccharides to extend the sequence, the linkage information can be used to decrease the branching factor. By passing a list of the different types of valid linkages and their frequency along with the number of each type of valid disaccharides remaining, only those disaccharides that, when appended to the growing sequence chains, add a valid linkage, will be added to the list of possible sequences. If none of the disaccharides produce valid linkages, the branch is terminated. The valid linkages can be dynamically updated to reflect the linkages that have already been used in the sequence and thus further the lower the branching factor. The memory issues resulting from numerous recursive function calls with long argument lists has been handled with specialized hashing functions and depth-first function calls.

Unfortunately, the NMR experiment itself frequently produces imperfect results. Specifically, it often cannot accurately discriminate the sulfation of the 2-position in linkages and the sulfation of the 6-position in monosaccharides. The software that performs the sequencing with NMR has been designed to handle these ambiguities when they have been specified by the user.

5.1.4 Enzyme Digest

Enzyme digests are used as the final type of experiment to narrow the list of valid sequences to convergence, when possible. HSGAG digest experiments frequently use one of a predetermined set of enzymes, which includes Heparinase 1, 2 or 3 and nitrous oxide. Each of these enzymes acts in a specific fashion that can be characterized in terms of the motifs of their binding domain and the specific region that is cleaved. Users must always enter the enzyme that they used in the enzyme digest and may select from the above enzymes or enter the specifications of an arbitrary enzyme. The enzymes added by users can be stored to see if certain ones are used frequently enough to be added to the list of predetermined enzymes.

After selecting an enzyme, or entering their own, the user must enter the MALDI results of the enzyme digest. The interface presently allows for up to ten distinct mass peaks, but this number may be easily increased on the future depending on the demand. Users can enter an arbitrary number of masses in any order but must enter all of the visible peaks in order to obtain reasonable results. To complete the enzyme digest, the server must then perform a lengthy series of calculations.

Before the enzyme digest processing may begin, the generation of the list of possible sequences, either from the joint NMR and CE results, or from the CE results alone, must have completed. The GlycoBank software then iteratively scans through every possible sequence on this list. Using an optimized algorithm similar to the motif search, described in the Search chapter, every motif that matches the binding site of the enzyme used in this experiment is found for every sequence. Different enzymes act on the sequences in unique fashions. Some operate specifically by starting at the non-reducing end and working progressively towards the reducing end of the sequence as a function of time, while other enzymes cleave randomly wherever they find valid binding sites. The software thus finds all the binding sites and chooses which different possible sets of cleavages are possible based on the enzyme. For example, suppose the hypothetical sequence, [1][2][3][4][5][6], can be cleaved by an enzyme between disaccharides 1 and 2 (site A), 3 and 4 (site B) and 5 and 6 (site C). An enzyme that proceeds from the non-reducing end to the reducing end (a sequential enzyme) would allow cleavage profiles (locations of where the enzyme cleaved the sequence) of only no cleavage, A, AB or ABC. Non-sequential enzymes allow the profiles of no cleavage, A, B, C, AB, AC, BC or ABC. This selection is automatically generated.

After generating all of the possible cleavage profiles, every profile must be inspected and the cleavages must actually be simulated. The hypothetical sequence discussed above, if being cleaved by a sequential enzyme, would produce the fragment groups shown below:

no cleavage: [1][2][3][4][5][6] (2888.42 Da)

A: [1] (435.37 Da) [*2][3][4][5][6] (2453.05 Da)

AB: [1] (435.37 Da) [*2][3] (956.80 Da) [*4][5][6] (1496.25 Da)

ABC: [1] (435.37 Da) [*2][3] (956.80 Da) [*4][5] (956.80 Da) [*6] (539.45 Da)

The software must generate each cleavage group iteratively and calculate the masses of the resultant fragments, and subsequently compare them against all the valid masses. If the user had entered mass peaks of 2888.42, 2453.05 and 1000.00, the program would mark the 2888.42 as having been found, for example, because it occurred in a fragment group that contained that valid mass and no invalid masses. The software scans through every possible fragment group, marking off the valid sets of masses as it finds them and stopping either when every mass peak reported has been found or it reached the end of the possible fragment groups. If the sequence does not match all of the reported mass peaks, it is considered invalid and thus removed from the list of possible sequences. If every mass of all the cleavage profiles is valid, but not every valid mass has been found, the sequence is also not valid.

After every sequence in the valid sequence list has been inspected, a new, truncated valid sequence list is created which contains only those sequences from the initial list that have also been shown to match the enzyme digest profile. Subsequent iterations of this experiment, using different enzymes and different mass peaks, can often reduce this list to a small number of possible sequences. Because the iduronic and glucoronic acids weigh the same amount, it is impossible to deterministically discern the leading monosaccharide using only enzyme digests. Therefore, at least two sequences will always be returned if there any valid sequences unless the user has used NMR to determine this position.

5.1.5 Web Interface

Given that GlycoBank does not operate with an advanced computing facility, the traditional website paradigm of submitting a query and receiving the results immediately must be replaced with one more adept at handling the asynchronous and high-latency tasks involved in HSGAG sequencing. The web interface has thus been designed around most efficiently directing the user to completing the data entry for the experiments performed in the sequencing job. The first task in any HSGAG sequencing is invariably to enter the capillary electrophoresis data. After completing this, the user is directed to a page that reveals their HSGAG sequencing ID, which can be used to retrieve the results when the entire sequencing job has completed, to receive its status while processing, or to enter additional sequencing tasks to the job. The screen displaying the sequencing ID also presents the user with an inquiry as to whether he plans to enter NMR data or if he will proceed directly to the enzyme digest portion of the sequencing. In the case where NMR is used, the user indicates which ambiguities are present in their particular NMR method and he will be presented with an interface that reflects the various linkages and monosaccharides that can be determined with their experiment. Once the NMR data has been submitted, a job to process this information is placed on the sequencing queue, which is described below.

The enzyme digest screen either follows the CE or NMR entry screen, depending on whether the user performed an NMR experiment. The researcher is then prompted for the mass peaks and the various properties of the enzyme that was used. After completing the enzyme digest data entry, he may proceed to enter as many enzyme digests as he has performed. At any point, however, the user may query the progress of the various computational tasks pertaining to this sequencing job. Even after all the enzyme digests have completed, the user may enter subsequent experiments if the results are inconclusive. The server also maintains the list of the valid sequences with the sequencing ID. The user may also view details of each task involved in the sequencing job including how many valid sequences there were at the beginning and the end of the task. Furthermore, if he selects an individual sequence from this list, GlycoBank will create a visual representation of the sequence, the binding sites of the enzyme on the sequence, and fragmentation profiles that produced the valid masses. This representation can be created for every valid sequence and every enzyme that has completed processing for this sequencing job.

5.1.6 Sequencing Queue

The various processes described above grow exponentially in the computation time required based on the length of the target sequence. Furthermore, completely sequencing an HSGAG often requires several iterations of enzyme digest experiments. Each enzyme digestion experiment may take up to several minutes to complete, even when running optimized code as the sole process on a powerful server. Because the web interface attempts to have the user enter all the available data, the sequencing tool must be able to process asynchronous data entry and the subsequent results generation. The tasks required for every sequencing ID are therefore entered into a sequencing queue that is used to direct the back-end processing.

The sequencing queue contains the status of every task that is entered when sequencing an HSGAG - whether that task is presently being executed, has completed execution, or has yet to begin. Each job in the queue points to a file in the local file system which stores all the necessary details, including the ID of the sequencing task, the nature of the task, the parameters associated with it, and a pointer to the list of valid sequences. Capillary electrophoresis experiments themselves will not generate an exhaustive list of the sequences once the data has been entered, because often NMR results will be entered shortly thereafter. This can be used in conjunction with the CE data to more effectively produce this list. If the user indicates that they will not be entering NMR results for this experiment, the CE may then begin execution. Otherwise, this step waits for the NMR data. A background process continually monitors the state of the sequencing queue and allows up to three tasks to execute from it simultaneously. Each sequencing ID can only execute one of its tasks at a time to ensure the coherency of the valid sequence list and to simplify the tracking of the various jobs related to the ID. Using the sequence queue, any number of jobs may be entered for an arbitrary sequencing ID without losing the coherency of the valid sequence list. In addition, if all the jobs of a sequencing task complete without producing a satisfactory answer, the user may add subsequent experiments at any point and the queue will automatically process this experiment with the current valid sequences list.

5.1.7 Future Plans

HSGAG sequencing, despite all the complexity already in place, stands to benefit from a number of possible enhancements. Most significantly, sequencing on GlycoBank cannot presently handle all the enzymatic modifications. The current implementation only recognizes delta U, and mannose when is occurs at the reducing end. In addition, certain portions of the sequencing code, presently written in PHP, are iterated thousands of times. Future optimizations may include rewriting the same algorithm from scratch in a substantially faster language like C and with fewer accesses to memory.

While performing enzyme degradation, many experiments do not appreciably reduce the number of valid sequences. By adding a mechanism to scan through possible degradation profiles and find an optimal enzyme to use, researchers may save substantial time and effort. No work towards this end has been conducted thus far. Machine learning for the sequence generation using NMR data has already been studied at Bell Labs and will hopefully be implemented at some point in the future. Finally, users often have a number of sequences that they wish to sequence simultaneously. This process proves tremendously more complicated and computationally intensive. While it may prove impossible to completely sequence several different HSGAGs concurrently using the current architecture, a framework may hopefully be developed to at least provide heuristics about the nature of the HSGAG mixture.

5.2 Motif Finder

The Motif Finder tool on GlycoBank attempts to determine, from a collection of sequences that are known to interact with a common enzyme, the binding domain of that enzyme. This requires knowledge of the deterministic sequence of as many HSGAGs as possible in this collection, but it requires no knowledge about the nature of the enzyme.

An exhaustive search of every possible motif and then determining its frequency among all the constituent sequences in the collection is computationally intractable. Motifs are therefore found using the Gibbs sampling algorithm for noisy sequences [4]. This is a non-deterministic algorithm that could, in theory, converge to the incorrect motif, but it is extremely unlikely. The Gibbs sampling algorithm works by selecting one of the n sequences in the collection at random and choosing a random motif, equal in length to the shortest sequence in the collection. This motif is then matched against every position in every other sequence in the collection. The alignment with this motif is scored against each position. Based on the scores that were just calculated, another random motif is selected from a randomly chosen sequence. This new motif is then scored against every position in every sequence. This process is repeated many times. The sampling algorithm generally converges to the optimal motif, but because the sampling algorithm operates using a random-number generator, it may never converge. A more rigorous description of this algorithm and its limitations can be found in the Lawrence 1993 paper.

Because HSGAGs are unlike DNA or protein sequences in that every position has a number of different characteristics, each of which affect the motif, many of the simple sequence alignment strategies do not work without major modification. The motif finder for HSGAG sequences, while utilizing the theories employed in simple sequence alignment, actually requires numerous parallel alignments to create a matrix of characteristics. The user can select different parameters to grade the importance of the HSGAG characteristics to adjust the scores discussed above. For example, the iduronic/glucoronic characteristic of the uronic acid is frequently the most important characteristic of any motif. Therefore, this defaults with a score of 100, compared to 5 to 10 for the various positions of sulfation. The sampling algorithm eventually returns a motif which, using the scoring metric, is optimally matched to every sequence. Using a consensus of the motif against all the sequences, the positions that do not have consensus are described as variable. The motif is also shortened to maximize its score. The default settings for the scoring were based on a crude analysis of existing motifs. Future work is required to fine tune these numbers. The motif finder returns the two two-dimensional motifs that were found in the sequences and provides an interface to search the database for this motif.

5.3 Sequence Score

Scoring HSGAG sequences proves extremely useful when sequencing HSGAGs and faced with the reality that certain ambiguities cannot be resolved with current experimental methods. By providing a quantitative score of the characteristics of the sequence, a collection of sequences may hopefully be narrowed to only a few likely possibilities.

Only a few methods have currently been employed to create a sequence score. Every sequence is degenerated into its linkage profile, which is compared against the known linkage profiles. The higher frequency linkages are ranked more highly, while less frequently occurring linkages negatively affect the score. The two-dimensional linkage profile is also analyzed and incorporated into the score. The number of sulfations, acetlyations and free amines are also factored into the equation. This is then assembled into a score using a hand-written formula. In the future, the patterns of sulfations, the different motifs present in the sequence, and many other factors will also be included.

The process of scoring sequences inherently requires that numerous HSGAGs be entered into the database as a training set. Because GlycoBank has been under active development, and has not been released to the public, it still has a relatively small data set. The development of the sequence scoring has therefore been limited. The scoring method may be further refined with a training set acting on an SVM. This will train the weights of the various components of a formula that will be used to generate the sequence score. Furthermore, machine learning will be employed to help distinguish the chemical properties are useful for scoring sequences from those that are unimportant.

Chapter 6

Consortium for Functional Glycomics

The Consortium for Functional Glycomics (CFG) is a large-scale, collaborative initiative funded by the National Institutes for General Medical Sciences "Glue Grant" which includes several disparate research groups. The scope of the CFG is substantially broader than that of GlycoBank. While developing GlycoBank, however, many architectural design decisions and algorithms proved useful in the design of the web application run by the Consortium.

6.1 Data Entry

The Consortium requires a large number of different interfaces for data entry as the different labs and research groups across the country need to enter their own, unique type of data. The a substantial portion of the work done by the CFG bioinformatics team focuses on the need to process and store so much information. Substantial effort was expended creating an intuitive interface for the GlycoBank data entry, so many of the insights from the development of GlycoBank were then applied to the data entry portions of the CFG to create more intuitive and standardized interfaces across their numerous input screens.

6.1.1 Security

Entering data into the CFG database necessarily propagates that information through several different tables as foreign keys and through other additional constraints. Other users can subsequently use that information in their own experiments and aggregate their own data with it. Because all data in the CFG database is assumed, by the researchers that use it, to be scientifically sound, great care has been taken to ensure that only authentic users can enter data. The system implemented for the Consortium builds upon the methods that were used in GlycoBank.

Every login is necessarily encrypted through a secure socket layer. Therefore, all username/password pairs are not sent through plaintext. As with GlycoBank, if somehow a user manages to send an unencrypted username and password, it is summarily rejected by the server. The validity of plaintext password transmissions is never checked. The method used for ensuring user authenticity is identical to that employed by GlycoBank except that authenticating value is stored in a database in the Consortium. An unencrypted session cookie is sent to the client with an encrypted verification cookie. Only when both of these are sent, which happens only through encrypted connections, and the verification cookie matches the value in the database, can the user access data entry pages.

The use of this authentication process is described in more detail in the previous chapter discussing data entry in GlycoBank. This security system also ensures that all the data entered into the database has an authentic, verifiable source. In addition, all data editing for the CFG database requires the same level of authentication. This level of security does allow for a hacker to view a private piece of information by eavesdropping on the network. The CFG has a constraint, however, that no piece of data may remain private for more than six weeks. The complexity associated with sending all private information over secure connection and public information in plaintext, when combined with the limitation of the privacy guaranteed by the Consortium resulted in this compromise.

6.1.2 Insecure Uploads

CFG users often require direct uploads of data to the database in the form of images or large Excel spreadsheets. Uploads can reach well over 20 MB in size. Because encryption dramatically slows down the rate of data transfer and many CFG users already have slow connections, the compromise was reached that all data uploads be unencrypted. In order to upload data, however, users still require a valid session. Therefore, arbitrary users cannot enter data despite the elimination of the encrypted authentication process from the upload.

The protocol for uploads does allow a malicious user to intercept the session ID. Using this session ID, the user may upload an arbitrary file to the database. This action does require an advanced attacker, and because all raw files are stores in their original form without propagating very far through the database, these uploads can be easily deleted if spurious data is ever found. The files are also contained in the database, so viruses cannot be placed on the Consortium server in a method that would allow them to interfere with its operation.

6.2 Searching

The CFG, due to its highly collaborative nature, has many different research groups contributing copious amounts of information. Each research group also demanded the ability to search their data and the data of the other research cores. Each different search displays distinct data and searches a specific group of tables. Implementing separate searches for each of these cores would have required tremendous effort and would have led to inconsistency within the CFG application.

While developing GlycoBank, an extensible results engine was developed to ensure that each results page had a similar look and feel, and to simplify the process of writing new searches. This engine was eventually abandoned as PHP templates were employed, which further simplified the process for a project with the scope of GlycoBank. The algorithms and design developed in this first results engine proved to be very compatible with the architecture and demands of the CFG application.

6.2.1 Implementation

Because the CFG stores substantially more information than GlycoBank, in well over 100 tables, a more robust search tool was required than what could be obtained by simply recoding the results engine that was developed for GlycoBank. Furthermore, the demands of the Consortium members required that searches be easily sorted, filterable, and that the views of the data be as extensible as possible.

The goal, therefore, was to create a search tool that simplified the backend interface to each table and then maximized the availability of the data for the CFG users once it was retrieved. Searching in the CFG consists of a number of different static methods, which simply construct the appropriate queries on the types of relevant tables in the CFG database and then reformulate the results to the standard required by the results display tool. The simplicity and uniformity of these methods allowed over twenty searches to be constructed in only a few days. Every search method was designed to return results in the same fashion, with certain pieces of metadata required to specify the layout. The result engine therefore accepts the search results, stores them in the user session in the CFG server, and then formats the appropriate entries for display according to the layout information and the portion of the result set that is viewed on the current results page. Any new search that returns data in this format and provides the required metadata can also use this results page. Furthermore, the same search method may be used for a number of different searches on the same data. By providing an extensible framework to return the same results with variable links, the same searches can be used for returning both viewable and editable objects, and for linking to several other pages as well.

The results page itself has been built to process the generic data passed by the search methods. It can display an arbitrary number of results per page, resort the data based on any of the fields, and apply a filter to the data. These operations require as few new searches as possible in order to minimize unnecessary calls to the database. The results engine itself, therefore, handles nearly all of these operations.

6.2.2 Future Work

Several searches, including the mouse search, yield thousands of results whenever a complete list is requested. Because the results engine stores the entire result set in memory on the server for each user that performs this query, many simultaneous searches can quickly overwhelm the CFG server and will dramatically affect its performance. Because the searches account for the access control for each unique user, it is generally not possible to share result sets between users. The large mouse data set, however, contains mostly public data so optimizations may be made to use one complete dataset and store those results that users cannot see separately for each user.

A proposed project involves creating a caching process to periodically monitor the mouse data set for any modifications. This single cache can then be used with a modified results engine to exclude any results in the cache that the user cannot access. This means that there will only need to be one copy of the large data sets in memory, regardless of the number of searches being concurrently performed. Because each user cannot see far fewer mice than he can see, this substantially reduces the amount of memory required by large searches.

6.3 Molecule Pages

The CFG also aims to collect and distribute information about all the major glycan molecules or those that interact with glycans, how they have been studied, and their role in various pathways. The display of this information requires specialized molecule pages, like those found in SwissProt or PDB. Because of the large number of relevant molecules, the bioinformatics staff at the CFG could not afford to populate and curate each individual molecule page. To address this issue, a web spider was developed to maximize the amount of data available for any given molecule using only SwissProt and UniGene IDs.

6.3.1 Molecule Page Spider

The molecule page spider was developed primarily to maximize the reliability of data retrieval. Beginning with the SwissProt page, numerous regular expressions were developed to parse through the page and extract as much information as possible and also glean any links to other bioinformatics database that would include as much of the required data as possible. Regular expressions and methods to parse these pages for the information that they have on specific molecules were created for the following different websites: SwissProt (raw and nice), LocusLink (locus page and search page), ScanSite, nucleotide and protein Blast, SourceDB, the DBJ, and the Queen Mary University IUBMB site.

The molecule page spider was carefully developed to avoid requesting too many pages from any single website. Starting with a list of desired information about a certain molecule, the spider crawls to each page and parses it. The information that has been found is marked as complete, and the spider uses the ID numbers it has found from this page to search through other pages that might contain the missing information. This process completes when the spider has found as much information as possible. Because this inherently relies on other web applications being active, the spider automatically throws an error if any group of websites that have all the known resources for a certain piece of information are inactive.

6.3.2 Automated Molecule Page Generation

The researchers that provide the information about the molecules to be represented in the molecule pages do not have access to the software running on the CFG servers, only to the web applications. For security reasons, the software that populates the CFG database for molecule page information is not available through the internet. Researchers, therefore, must rely on the Bioinformatics team of the Consortium to accurately enter their information. The CFG members only need to send a Microsoft Excel spreadsheet containing the information about the molecules that they want to enter into the database. This is submitted by a Bioinformatics team member to the software which automatically parses this spreadsheet and adds those molecules that do not exist. An extensive process of reconciling a previously-existing molecule with a new one has been created which preserves as much information as possible while still allowing the information from the new molecule to supplement the entry.

Chapter 7

Conclusion

7.1 Internet Availability

GlycoBank has been in a constant state of development. After undergoing many major revisions and incorporating the suggestions made by experts in the field, the application has seen numerous stages of testing for usability and bioinformatics capabilities. It is now nearing its first public release.

7.2 GlycoBank

7.2.1 Implementation

GlycoBank has been re-implemented three different times using a wide array of technologies. The first incarnation utilized simple Perl scripts and a MySQL database. The development was very rapid with the decisions that were made about the software architecture. Flexibility and power became an issue as the applications scope grew well beyond its initial projections.

To address these increasingly apparent shortcomings, the decision was made to try to reuse as much as possible from the initial implementation, but use a far more flexible system design to allow the application more room to develop. An XML-based architecture was chosen which allowed GlycoBank to be created in any computing environment as long as the output adhered to the various XML schema defined beforehand. A parser was then used to format the XML with HTML for a consistent presentation.

Ironically, this resulted in a design that was practically difficult to implement because of all the flexibility. This problem was noted early in the development process and it was quickly abandoned in favor of the current architecture which runs primarily on PHP and PostgreSQL. The system uses an advanced template engine to ensure a standard interface and also allows other programming environments to be used. The problems encountered with choosing a correct model for implementation were not inherently problems that could have been avoided. The shortcomings of the Perl language itself, its lack of a truly object-oriented structure and the difficulty understanding its code, led to its eventually being abandoned in favor of the more powerful and structurally sound PHP language. PostgreSQL was selected over MySQL because of the advanced database functionality that it allows, like full-text indexing, that are impossible or very difficult in MySQL.

7.2.2 Data Entry

The data entry portions of GlycoBank were also constantly revised. The initial implementation called for a one-to-one mapping of the references and classifications to sequences. As the data model grew in complexity, the requirements in how a user could enter data inherently grew more complicated as well. Effort was expended understanding how similar databases guided users through their complicated data structure without exposing them to the complicated backend relationships. It was noted that many successful systems employ non-interactive navigations during data entry that inform the user of their progress along the way. When the user understands what is required of them, they more readily accept those requirements. Therefore, the design of GlycoBank centered on informing the user which aspects of the data entry were incomplete and what was required to correct them. The users were not expected to understand any of the mappings between data.

7.2.3 Search

Searching in GlycoBank also presented several major hurdles. The initial Perl implementations simply queried the database and presented every result every time a search was performed. As the database grew, it became increasingly obvious that this would not scale. The XML implementation actually allowed for a very powerful display of search results because searches could be stored as results on the file system without needing to reside in memory for the duration of every user session. The development of the initial results engine again created a large degree of complexity which was abandoned in favor of the current template system. The actual caching of the searches and the search results proved to be a difficult process to engineer and has not been perfected.

7.2.4 Tools

Before implementing the tools on GlycoBank, numerous studies were conducted both about how to do the sequencing itself and about the time that it requires. The tools, therefore, despite being the most theoretically difficult aspect of GlycoBank, were the most easily developed. As mentioned previously, the sequence score tool is far from being completed because too little information is in the database to effectively learn the characteristic that make for a high-scoring sequence. The actual implementation of this tool will probably require expert knowledge of HSGAGs and several hundred structures.

7.3 Consortium for Functional Glycomics

The Consortium for Functional Glycomics web application is a much larger project than GlycoBank. The team of informatics professionals contributing to this project is several times the size of that for GlycoBank. The CFG also caters to a group of individuals who demand certain characteristics of their website, rather than offering them novel tools to which they never previously had access. The strategies employed during implementation therefore differ tremendously between the two groups.

Many practices learned while working for the Consortium, however, will prove very useful to the continued development of GlycoBank. For example, the deployment of simultaneous versions of the web application allowed for users to migrate seamlessly to an upgraded version of the application. This revision created required major changes in the server-side implementation that would have otherwise required days of server down-time to implement.

Many of the conclusions developed while creating GlycoBank also proved irrelevant or inaccurate for the users of the Consortium. A specific example was the implementation of probabilistic sequences. GlycoBank employed a parent-child strategy to represent every possibility of a probabilistic sequence under the heading of a single sequence with variability at known positions. When presented for Consortium researchers, this model was rejected in favor of the system that stores each probabilistic sequence independently, even if they derive from one parent sequence.

Appendix A

Tables

Table A.1: PEN Representation for the 48 Fundamental Disaccharide Units

PEN	Dissacharide	PEN	Dissacharide	PEN	Dissacharide
0	I-H_{Nac}	1	I-H_{NS}	1n	I-H_{NH_2}
2	I-$H_{Nac,3S}$	3	I-$H_{NS,3S}$	3n	I-$H_{NH_2,3S}$
4	I-$H_{Nac,6S}$	5	I-$H_{NS,6S}$	5n	I-$H_{NH_2,6S}$
6	I-$H_{Nac,3S,6S}$	7	I-$H_{NS,3S,6S}$	7n	I-$H_{NH_2,3S,6S}$
8	I_{2S}-H_{Nac}	9	I_{2S}-H_{NS}	9n	I_{2S}-H_{NH_2}
A	I_{2S}-$H_{Nac,3S}$	B	I_{2S}-$H_{NS,3S}$	Bn	I_{2S}-$H_{NH_2,3S}$
C	I_{2S}-$H_{Nac,6S}$	D	I_{2S}-$H_{NS,6S}$	Dn	I_{2S}-$H_{NH_2,6S}$
E	I_{2S}-$H_{Nac,3S,6S}$	F	I_{2S}-$H_{NS,3S,6S}$	Fn	I_{2S}-$H_{NH_2,3S,6S}$
-0	G-H_{Nac}	-1	G-H_{NS}	-1n	G-H_{NH_2}
-2	G-$H_{Nac,3S}$	-3	G-$H_{NS,3S}$	-3n	G-$H_{NH_2,3S}$
-4	G-$H_{Nac,6S}$	-5	G-$H_{NS,6S}$	-5n	G-$H_{NH_2,6S}$
-6	G-$H_{Nac,3S,6S}$	-7	G-$H_{NS,3S,6S}$	-7n	G-$H_{NH_2,3S,6S}$
-8	G_{2S}-H_{Nac}	-9	G_{2S}-H_{NS}	-9n	G_{2S}-H_{NH_2}
-A	G_{2S}-$H_{Nac,3S}$	-B	G_{2S}-$H_{NS,3S}$	-Bn	G_{2S}-$H_{NH_2,3S}$
-C	G_{2S}-$H_{Nac,6S}$	-D	G_{2S}-$H_{NS,6S}$	-Dn	G_{2S}-$H_{NH_2,6S}$
-E	G_{2S}-$H_{Nac,3S,6S}$	-F	G_{2S}-$H_{NS,3S,6S}$	-Fn	G_{2S}-$H_{NH_2,3S,6S}$

Table A.2: PEN modifiers

Modifier	Positions Found	Modifies	Description
du	Non-reducing end only	Uronic acid	Delta 4,5 unsaturated uronic acid. This uronic acid is generated by beta eliminative cleavage of the hexosamine-uronic acid linkage by enzymes or chemical methods.
man	Any	Glucosamine	Mannose contracted 5-carbon ring instead replaces glucosamine
anh	Reducing end only	Glucosamine	1,6-Anyhydro. This normally arises from $\pm$D or $\pm$5.
anhm	Reducing end only	Glucosamine	1,6-Anyhydro with mannosamine. This normally arises from $\pm$D or $\pm$5. When following another dissacharide, the I-H bond is not susceptible to heparinase cleavage.
monou	Non-reducing end only	Uronic acid	Dissacharide cleaved to become a monosacharide. The uronic acid has been removed and the glucosamine acts as it normally does in the sequence.
monoh	Reducing end only	Glucosamine	Dissacharide cleaved to become a monosacharide. The glucosamine has been removed and the uronic acid acts as it normally does in the sequence.
gal	Non-reducing end only	Uronic acid	Uronic acid replaced with galacturonic acid.
mana	Reduing end only	Glucosamine	Glucosamine repalced wtih mannosamine.

Appendix B

Selected SQL

The SQL code shown in this secton is meant to illustrate the specifics of various commands. The code in the database is often optimized through the use inner joins and indices. For clarity, the code below is written using only outter joins and no SQL optimizations.

B.1 Motif Search

B.1.1 Simple Example

```
SELECT
 a.hsgag_id
FROM
 hsgag_binary_sequences a, sequence b, hsgag_sequences c
WHERE
 (binary_sequence & 116) = 96 AND man1 = 0 AND
 b.public_domain=1 AND c.probability > 0.3;
```

Searches for any disaccharide with glucoronic acid, and the 2- and 3- positions not sulfated which does not have mannose. Because the database stores the same sequence frameshifted by to every possibile binary conformation, the one query will automatically scan for this motif in every position. This query also ensures that the sequence is in the public domain and is not a low probability sequence.

B.1.2 More Complicated Example

```
SELECT
 a.hsgag_id
FROM
 (SELECT
   hsgag_id, word, offset
  FROM
   hsgag_binary_sequences
  WHERE
   (binary_sequence & 245185140) = 202911840 AND
   man1 = 0 AND man2 = 0 AND man3 = 0 AND man4 = 0) a,
 (SELECT
   hsgag_id, word, offset
  FROM
   hsgag_binary_sequences
  WHERE
   (binary_sequence & 245185140) = 202911840 AND
   man1 = 0 AND man2 = 0 AND man3 = 0 AND man4 = 0) b,
 sequence c, hsgag_sequence d
WHERE
 b.word = a.word + 1 AND a.offset = b.offset AND
 c.public_domain=1 AND d.probability > 0.3
```

This search for eight consecutive disaccharides without mannose that have glucoronic acid, and no sulfation at the 2- or 3-positions. If this search were conducted using all possible sequence searches instead, it would require 429,981,696 different searches instead of one, relatively simple query.

B.2 Chemical Search

B.2.1 Near Match

```
SELECT
 hsgag_id
FROM
 hsgag_sequences
WHERE
 (abs(length-10)^2) + (abs(sulfate_count-20)^2) +
 (abs(acetate_count-4)^2) < 5
ORDER BY
 (abs(length-10)^2) + (abs(sulfate_count-20)^2) +
 (abs(acetate_count-4)^2);
```

This searches for sequences with 10 disaccharides, 20 positions of sulfation and 4 acetylated N-positions. The tolerance is 5. In the future, weights may be added to certain fields that are of greater importance.

B.2.2 Ratio Match

```
SELECT
 hsgag_id
FROM
 hsgag_sequences
WHERE
 (abs((length_count/sulfate_count)-(10/20))^2) +
 (abs((acetate_count/sulfate_count)-(4/20))^2) < 1
ORDER BY
 (abs((length_count/sulfate_count)-(10/20))^2) +
 (abs((acetate_count/sulfate_count)-(4/20))^2)
```

This search is the ratio version of the near match search shown above. The sulfation component of the search has been removed as it would have simply been

```
abs(sulfate_count/sulfate_count-1)^2
```

. This produces 0 every time. The tolerance here is 1, although this may be adjusted based on the number of parameters used.

Bibliography

[1] D. Berry and C.-P. Kwan et. al. Distinct heparan sulfate glycosaminoglycans are responsible for mediating fibroblast growth factor-2 biological activity through different fibroblast growth factor receptors. *FASEB J.*, 15(8):1422–1424, 2001.

[2] R. C. Binari and B.E. Stavely et. al. Genetic evidence that heparin-like glycosaminoglycan are involved in signalling. *Development*, 124(13):2623–2632, 1997.

[3] M. Guerrini and R. Raman et al. A novel computational approach to integrate NMR spectroscopy and capillary electrophoresis for structure assignment of heparin and heparan sulfate oligosaccharides. *Glycobiology*, 12(11):713–719, 2002.

[4] C. E. Lawrence and S. F. Altschul et al. Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignments. *Science*, 262:208–214, 1993.

[5] R. Padera and G. Venkataraman et al. FGF-2/fibroblast growth factor receptor/heparin-like glycosaminoglycan interactions: a compensation model for FGF-2 signalling. *FASEB J.*, 13(13):1677–1687, 1999.

[6] A. Rhomberg and S. Ernst et al. Mass spectrometric and capillary electrophoretic investigation of the enzymatic degradation of heparin-like glycosaminoglycans. *PNAS*, 95(8):4176–4181, 1998.

[7] R. Sasisekharan, S. Ernst, and G. Venkataraman. On the regulation of fibroblast growth factor activity by heparin-like glycosaminoglycans. *Angiogenesis*, 1(1):45–54, 1997.

[8] Z. Shriver, R. Raman, and et. al. Sequencing of 3-O sulfate containing heparin decasaccharides with a partial anithrombin III binding site. *PNAS*, 97(19):10359–10364, 2000.

[9] G Venkataraman, Z. Shriver, R. Raman, and R. Sasisekharan. Sequencing Complex Polysaccharides. *Science*, 286(3439):537–542, 1999.

[10] Z. L. Wu and L. Zhang et al. The involvement of heparan sulfate (HS) in the FGF1/HS/FGFR1 Signalling Complex. *J. Biol. Chem.*, 278(19):17121–17129, 2003.