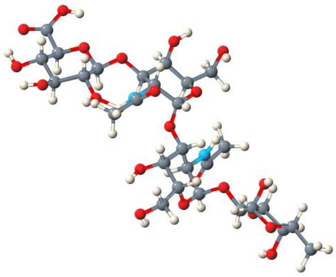
Thesis for the Degree of Master of Science (20p)

Prediction of Oligosaccharide 3D Structure using Genetic Algorithm Search Methods



Abraham Nahmany and Francesco Strino

Department of Mathematical Statistics/Bioinformatics
Chalmers University of Technology

Department of Medical Biochemistry/Structural Chemistry
Göteborg University

Department of Computing Science
Chalmers University of Technology

Göteborg, January 2004

Abstract

Prediction of oligosaccharide 3D structure using genetic algorithm search methods

Abraham Nahmany and Francesco Strino

Supervisor: Per-Georg Nyholm

Examiner: Graham J.L. Kemp

In predicting oligosaccharide conformations, achieving a good trade-off between good sampling of the conformational space and computation time is a major problem. For example, sampling a pentasaccharide with eight torsion angles in 15° steps around each rotatable bond results in over 100 thousand million conformations, and it would take years to calculate the energy of these on a PC in order to find the conformation with the lowest energy. Genetic algorithms (GAs) have been shown to achieve a good trade-off in similar applications [1], so the aim of this project is to investigate whether GAs can be used successfully in modeling oligosaccharides. In the present study, we have implemented a system called GLYGAL that can perform conformational searches using several different GA methods. The searches are performed in the torsion angle conformational space and energy calculations are performed using the MM3 [2] force field method. Tests on several oligosaccharide structures, such as the blood group related oligosaccharide, the O-specific oligosaccharide of the *Shigella dysenteriae* type 2 and 4 were performed showing very good results. The results obtained using different GAs implemented in GLYGAL are compared with each other, and with results obtained from experimental method such as NMR, as well as computational methods such as MM3 filtered systematic search.

Key words: Genetic algorithms, oligosaccharide, molecular mechanics, Shigella dysenteriae, Schistosomiasis

Acknowledgements

This project would not have been done without the help and support of several persons who have contributed their ideas, their valuable time and a lot of patience. Thanks go to all contributors, especially to our supervisor **Per-Georg Nyholm**, our examiner **Graham J.L Kemp** and our colleague **Jimmy Rosen**.

Abraham Nahmany and Francesco Strino
January 2004

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Saccharides linked to proteins and lipids cover a large fraction of the surface area of most cells. Many of these saccharides are involved in specific recognition processes. To understand their biological function it is necessary to have information about their 3D structure. Oligosaccharide 3D structure is also important in vaccine development. For example Schistosomiasis, also known as bilharzia, is a disease caused by the *Schistosoma* parasite. The life cycle of the parasite involves two hosts: fresh water snail and human. It has been shown that the parasite has developed mechanisms to resist the immune system of both hosts. In the various stages of development, oligosaccharides at the surface of the parasite are suggested to be involved in these resistance processes [3], and these oligosaccharides may be good targets for vaccine development.

NMR and X-Ray crystallography are two experimental techniques for determining oligosaccharide 3D structures. Both of these techniques are expensive and time consuming. An alternative approach for oligosaccharide conformational analysis is to search the space of possible conformations to find favorable low energy conformations. One major problem using computational methods is achieving a good trade-off between good sampling of the conformational space and computational time.

Genetic algorithms (GAs) have been shown to achieve a good trade-off in similar applications, so the aim of this project is to investigate whether GAs can be used successfully in modeling oligosaccharides. In the present study we have implemented a system called GLYGAL that can perform conformational searches using several different GA methods. The searches are performed in the torsion angle conformational space and energy calculations are performed using the MM3 force field method.

The nature of the problem dealt with in this project provided us with several challenges. Challenges of a computing science nature which included software development and the analysis of GAs performance for oligosaccharide modeling, as well as of a biological/biochemical nature which dealt with the analysis of the results we got from running the program on different structures.

In chapter 2 we give background information about basic carbohydrate biochemistry, genetic algorithms overview and a short review on related work. In chapter 3 we give the principles of modeling oligosaccharide using genetic algorithms and we describe the program and algorithms that were developed in this project. In chapter 4 we present the results achieved from running tests on several oligosaccharide structures using our software. Here again we give results for the computer scientist and results that may interest the biologist/chemist. We then present the conclusions we could draw after analyzing the results, evaluation of the software created in the project and other tools used. Some future work will also be presented in the last chapter. In appendix A we present a prototypical genetic algorithm, appendix B illustrates the life cycle of the *Schistosomiasis* disease and appendices C and D presents the user manual and the maintenance manual for the GLYGAL software developed in this project.

Chapter 2

Background

Understanding the basics of carbohydrate conformation as well as knowing the basics of genetic algorithms is a prior requirement for understanding the work done in this project. The last section of this chapter describes work related on saccharides conformation analysis.

2.1 Genetic algorithms - an overview

Genetic Algorithms (GAs) are adaptive heuristic search algorithms based on the evolutionary ideas of natural selection and genetics. GAs are part of what is called Evolutionary Computing that were first introduced in the 1960s by Rechenberg in his work “Evolutionary Strategies”. GAs were introduced in the mid 1970s by John Holland that together with his students and colleagues have developed GAs further. The schema theorem of Holland [4] can characterize mathematically the evolution over time of the population within a GA.

The search for an appropriate solution begins with an initial population of solutions (individuals). Individuals of the current population give rise to the next generation population by means of operations such as random mutation and crossover, which are patterned after processes in biological evolution. At each step the individuals are evaluated using a given measure of fitness, with the most fit individuals selected probabilistically as seeds for producing the next generation [5, 6]. In appendix A we present a prototypical genetic algorithm.

GAs have been applied successfully to a variety of learning tasks and to other optimization problems. The popularity of GAs can be motivated by number of reasons:

- Evolution is known to be a successful method for adaptation within a biological systems.
- GAs can search very complex search spaces.
- GAs are relatively easy to code and parallelize.

In Section 2.1 we describe in more details the basics in genetic algorithm search methods.

2.1.1 Biological background

The ideas behind GA search methods is to mimic the biological processes known from genetics and evolution. The chromosomes existing in all living cells consists of genes that encodes proteins. In the recombination (crossover) process the chromosomes from the parents will merge to create a new chromosome of the offspring. When copying the chromosomes from the parents to the offspring a change (mutation) in one or more genes can occur. The success of an offspring in “surviving” would be the fitness of this organism.

2.1.2 Encoding

When a problem is to be solved using a GA the first question that comes up is how to encode the chromosomes. In the original GA and in many applications the favorable encoding is a binary encoding where a chromosome is represented as a bit string. This representation will allow a straightforward application of operators, such as mutation and crossover, on the chromosomes. Another encoding strategy which can be seen as more ‘natural’ for some applications uses a so called value encoding. In this case the chromosomes will consist of an array of values. These values can be integers, real-numbers or even characters. This encoding method, although good for some specific properties, requires special crossover and mutation operators to be designed.

2.1.3 GA operators

The generation of successors in a GA is determined by a set of operators that recombine and mutate selected members of the current population. The most common ones are *mutation* and *crossover* operators.

Mutation

The mutation operator works on a single parent represented as a bit string and produces a single offspring. This operator chooses a single bit at random and changes its value. For example given the parent represented by the bit string 1000110011 and by choosing randomly the fourth bit from the left to be changes we get the following offspring 100**1**110011.

Crossover

The crossover operator produces two offspring from two parents, by exchanging selected bits between the parents. Figure 2.1 illustrates the single-point crossover operator.

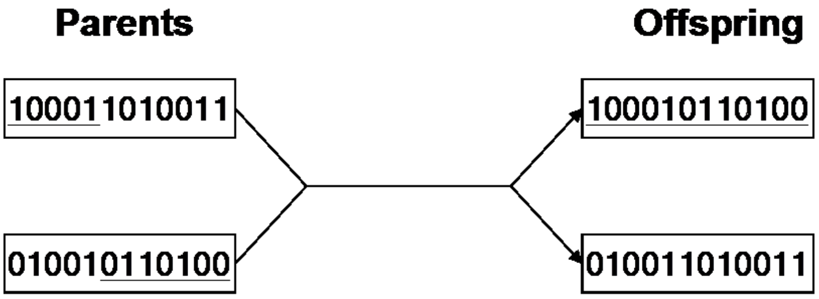


Figure 2.1: Single-point crossover. The parents to the left and the offspring to the right. The underlined parts in the parents are the bits selected to construct the first offspring whereas the non-underlined parts construct the second offspring.

In a similar way one can apply more than one crossover point. Some extensions of the crossover operator can be applied to more than two parents and generate more than two offspring.

2.1.4 Fitness function and selection

Natural selection, in its most general form, means the differential survival of individuals: some individuals live and some die. For this to happen there must be a population of individuals that are capable of reproduction. Natural selection prunes this population according to the criterion of survivability (fitness). The fitness function defines this criterion for ranking individuals and for probabilistically selecting them for inclusion population of the next generation.

In the standard GA the probability that an individual will be selected is given by the ratio of its fitness to the fitness of other members of the population. This method is called *roulette wheel* selection. Other selection methods have been suggested such as rank selection and steady-state selection.

2.1.5 Variants of GAs

The standard GA described above can be extended or changed to create a slightly different genetic algorithm. We describe here the three variants that have been implemented in the GLYGAL system described in chapter 3.

Parallel GA

In the parallel GA, which is an extension of the standard GA, several populations will evolve in parallel usually on different processors with individuals migrating between the different populations. This simulation provides resembles "real life" evolutionary processes. Using several different populations one can avoid the so called crowding problem where some individual that is more highly fit than others in the population quickly reproduces and takes over a large fraction of the population. That isolation can be a requisite for evolution. This was pointed out by Darwin in his report about the evolution of finches in the Galapagos islands. By allowing some migration between the different populations, successful genes -"new blood"- can be spread into the populations.

The algorithm itself changes only slightly with comparison to the standard GA. Here, each population will run on one node using the standard GA search with the standard mutation and crossover operators. The different thing happening here is that "best" individuals will "migrate" between the populations. This migration operation can be done in different ways, and the most common way is that a fixed number of "best" individuals from each population will move to another population every fixed number of generations.

Evolutionary programming

Evolutionary programming is another variant of the standard GA. Here, the population evolve using only the mutation operator. This method simulates asexual reproduction. In this algorithm one parent will mutate to create one offspring. The best individuals from the parents and the offspring are chosen to create the next generation.

Lamarckian GA

As we saw above most of the GA method simulate Darwinian evolution and Mendelian genetics. This is shown in the left side of Figure 2.2 where an individual is evolving from one generation to the next. Another model of evolution, Lamarckian evolution, is based on the theories proposed by the nineteenth century scientist Jean Batiste de Lamarck [7]. His theory suggests that experiences of a single individual could directly affect the genetic makeup of their offspring. One example is the giraffe neck. According to Lamarck, since the giraffe had to stretch its neck to reach the higher branches for the leafs its neck got longer and this feature was inherited to its offspring.

Although the Lamarckian theory is not accepted as a model of biological evolution, it has been shown to improve the effectiveness of the computerized GA [11, 12]. Figure 2.2 illustrates the difference between the Lamarckian and the standard Darwinian approaches. The idea behind the Lamarckian approach is that an individual will “improve” during its lifetime by finding the nearest local minimum.

The Lamarckian GA is similar to the standard GA with one basic difference. The geometry of an individual after local minimization and evaluation will be directly used by the offspring.

The Lamarckian GA search method can be parallelized in a similar way to the standard GA. Thus several populations are running a Lamarckian GA search in parallel with individuals migrating between the populations.

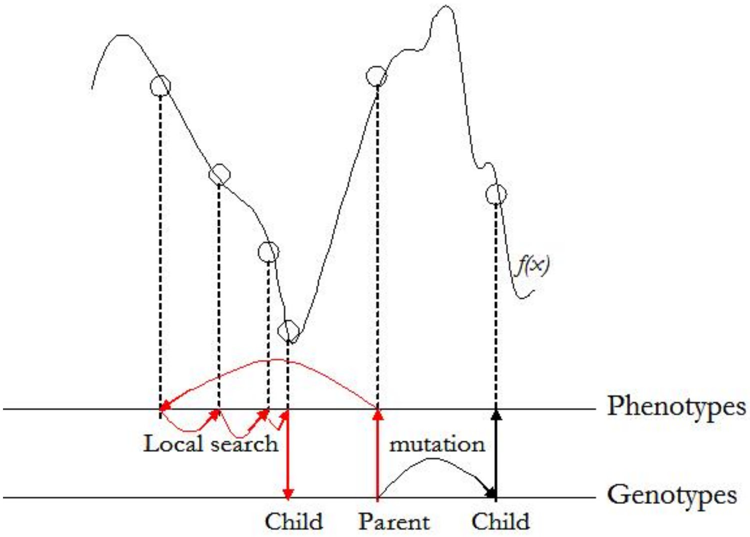


Figure 2.2: The difference between Lamarckian and Darwinian GA approaches. The space of the genotype and phenotypes are represented by the upper and lower horizontal lines respectively. The fitness function is f(x). On the right-hand side we see the behavior of the Darwinian GA when applying the mutation operator on a parent's genotype. On the left-hand side we see the behavior of the Lamarckian approach where a local search is performed on the phenotype space. Adapted from [12].

2.2 Basics in carbohydrate conformation

Carbohydrates constitute one of the most abundant types of biomolecules occurring widely in all living matter and, as the name suggests, are mainly composed by carbon, hydrogen and oxygen atoms. They function as structural or protective materials and as an energy store. In addition, carbohydrates perform much broader biological roles [9]. For example, they appear to be essential in the process of infection by certain pathogenic species; they specify human blood group types and are intimately involved in the immunochemistry of blood; they determine cell-cell recognition; they function as receptors in the antigen-stimulated lymphocyte antibody immune response, and they may have an important role in cancer pathology. The role of carbohydrates in biological processes is to a large extent determined by their conformational properties [10].

2.2.1 Monosaccharide as building blocks

Saccharides are composed of several units, called monosaccharides, that cannot be divided by hydrolization. There are many different monosaccharides in nature, depending on the position of some residue groups in the chain and their orientation. Each monosaccharide can assume different conformations.

Monosaccharides can assume either a linear or a ring form (Figure 2.3), the latter being more common and stable. In this work we concentrate on ring conformations, as chain conformation increase significantly the complexity of the search space since these have greater flexibility and more rotatable bonds to consider.

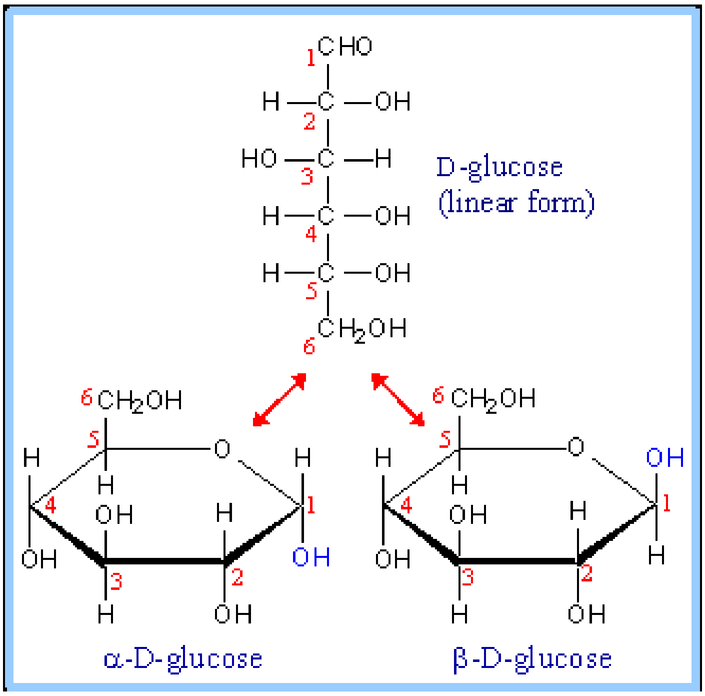


Figure 2.3: An example of the linear and ring forms of glucose. In the upper part linear form and in the lower part two presentations of a ring form. Figure taken from [8].

2.2.2 Oligosaccharides

Oligosaccharides are chain carbohydrates connecting up to 20 monosaccharide units. They can be either linear or branched chains.

Two monosaccharides can be connected together through glycosidic linkages, in which a so called bridge oxygen connects two carbon atoms belonging to the different rings.

Such link can have a considerable flexibility whereas the rings themselves are very rigid. Thus to describe the overall conformation of an oligosaccharide it is usually sufficient to use the torsion angles of the C1-O and the O-CX bonds of the glycosidic linkage.

Several ways of defining torsion angles are suggested in the literature. The most common one used is known as the *light atoms* definition. In most cases the linkage is defined by the torsion angle ϕ, identified by H1-C1-O-CX' and ψ, identified by C1-O-CX'-HX' illustrated in the upper part of Figure 2.4. In some cases additional bonds are involved in the linkage and more torsion angles needs to be considered. For example, in order to describe the 1→6 linkage shown in lower part of Figure 2.4, we need to define additional torsion angle χ (or ω), identified by O-C6'-C5'-H5'. In this case, the definition of ϕ and ψ are changed to H1-C1-O-C6 and C1-O-C6'-C5' respectively.

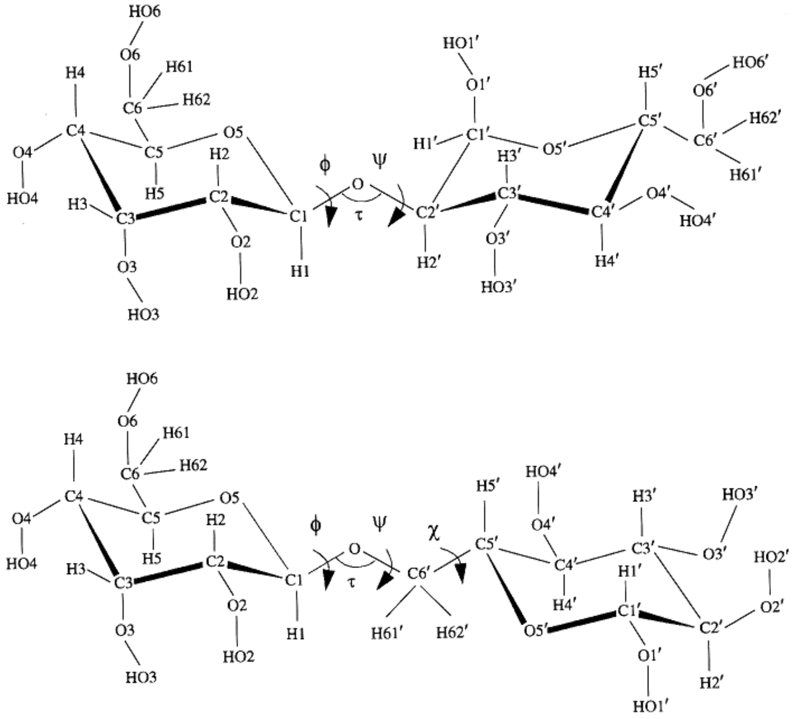


Figure 2.4: Definition of the glycosidic linkages. In most cases, ϕ and ψ are defined as H1-C1-O-CX' and C1-O-CX'-HX' where x=1,2,3 or 4 (upper part). In the case of 1→6 linkage ϕ,ψ and χ are defined as H1-C1-O-C6, C1-O-C6'-C5'and O-C6'-C5'-H5' (lower part). Figure adapted from [10].

2.3 Oligosaccharide conformational analysis

2.3.1 Experimental techniques

NMR and X-ray crystallography are two experimental techniques for determining oligosaccharide 3D structures. Both of these techniques are expensive and time consuming. Crystallizing saccharides with more than two-three residues is very difficult due to their flexibility and their hydrogen bonding properties. Determining oligosaccharide structure by NMR is difficult due to the small number of relevant signals in the spectra. This is why most NMR work on oligosaccharides is combined with 3D structure predictions using computational methods.

2.3.2 Computational methods

An alternative approach for oligosaccharide conformational analysis is the use of computational methods to search the space of possible conformations to find favorable low energy conformations.

One very popular approach of evaluation is the use of force field methods to calculate the conformational energies. The underlying assumption is that a molecule has its native conformation in the state of lowest conformational energy. There are several different methods for calculating molecular energy. The force field methods define the energy of the molecule as a sum of terms, concerning the ideal geometry of the molecule. One typical potential energy function of the force field, is shown in formula 2.1, consists of terms considering bond length, bond angles, van der Waals interactions etc. E_{bl} stands for the deviation from the ideal bond length, E_{ba} for the bond angles, E_{tor} for the torsion angles, $[E_{oop}]$ stands for the out-of plane deformations of planar systems, E_{VdW} for the Van der Waal interactions, E_{el} for the electrostatic interactions, $[E_{hb}]$ for the hydrogen bonding and $[E_{sol}]$ for the solvation energy. The terms in brackets are only formal in certain force fields.

$$E = E_{bl} + E_{ba} + E_{tor} + [E_{oop}] + E_{VdW} + E_{el} + [E_{hb}] + [E_{sol}] \tag{2.1}$$

Related work

Some computational methods using different force field calculation has been proposed for conformational analysis. The Glycan program, developed by Nyholm *et al.* [13], based on extensions of the truncated force field HSEA calculations, considers mainly van der Waals interactions resulting in rough estimates of the conformational energy. These estimates, though rough, are calculated at high speed and can be useful as a starting point for conformational studies of large systems.

Another method shown to be successful mainly on small structures is based on a filtered systematic search using MM3 force field calculation [14]. The search starts with a minimized conformation obtained by Glycan followed by a systematic ϕ/ψ search using MM3 for energy calculations and local minimization. The structure is investigated in fragments of disaccharides and trisaccharides where energy maps with 15° step size are generated for the favorable ϕ/ψ angles. The structure is then constructed and minimized as whole using a filter based on the favorable angles found in the systematic search. This method provides a good overview of the ϕ/ψ search space of each linkage and is useful especially for branched saccharides. However, the method suffers from some disadvantages. Pre and post-processing work is needed to be able to perform the search. Another problem with the filtered systematic search is the fact that it requires a large amount of CPU time. Performing such a search on a pentasaccharide structure demands several hundreds of hours of CPU time.

The goal of this application is not only to find the single optimal conformation of an oligosaccharide based on a force field fitness function but also to be able to find a set (if exists) of different favored conformations.

Why then use genetic algorithms? One reason would be that GAs have been shown to give good results in similar applications [1]. In this chapter we reviewed the main features of GAs. Using that knowledge combined with the understanding of the characteristics of the problem of modeling oligosaccharide it seems like GAs would be a suitable tool.

Chapter 3

Materials and methods

In chapter 2 we saw that there are still problems in obtaining the 3D structure of oligosaccharides using experimental techniques like NMR spectrometry and X-Ray crystallography as well as with existing computational methods. In an attempt to overcome some of these difficulties we have used genetic algorithm search methods. We will here give the basic principles for doing that and present the GLYGAL software developed during this project.

3.1 Modeling oligosaccharide using genetic algorithms

The basic ideas of modeling oligosaccharide using genetic algorithms are:

- Initial population of randomly generated conformations of an oligosaccharide.
- Evaluation using a fitness function based on molecular mechanics calculations using the MM3 force field and selection using a roulette wheel selection method.
- Standard genetic operators like mutation and crossover to generate offspring.
- Termination criteria satisfied either after a fixed number of generation or when no improvement has occurred during several generations.

3.1.1 Initial population

In the case of oligosaccharides we need a randomly generated initial population of conformations each representing a possible solution of the problem. For every application of a genetic algorithm one has to decide on a representation for the "genes". In the case of oligosaccharide modeling, a hybrid approach is used [15]. In this approach the genetic algorithm operates on numbers, not bit strings as in the original genetic algorithm. The representation of oligosaccharide using the hybrid approach can be done in several ways including Cartesian coordinates and torsion angle representation.

In the Cartesian coordinates representation the 3-dimensional coordinates of all the atoms in an oligosaccharide are recorded. This representation though easily converted to and from the 3-dimensional conformation of the oligosaccharide has the disadvantage that a mutation operator would be difficult to perform. This difficulty comes from the fact that the mutation operator might create "irrelevant" conformations where some atoms lie too far apart or collide. These conformations will have to be filtered resulting in an unnecessary extra CPU time.

Another representation presented in Figure 3.1 is using torsion angles. This representation is more natural for conformational studies in which the genetic operators work on torsion angles. For the hybrid representation here we use a torsion angle vector. Each glycosidic linkage of an oligosaccharide contains two torsion angles phi (ϕ) and psi (ψ) (in some special cases a third angle chi (χ) or omega (ω) can be found). The torsion angle vector will be a vector containing the ϕ and ψ angles.

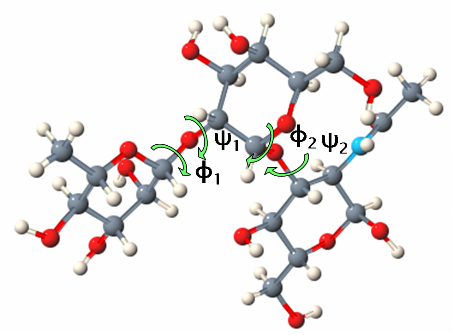


Figure 3.1: A hybrid representation of an oligosaccharide using a torsion angle vector. The torsion angle vector in this case will be $[(\phi_1, \psi_1), (\phi_2, \psi_2)]$.

3.1.2 Genetic algorithm operators

The above hybrid representation using a torsion angle vector makes it an easy task to apply the GA operators.

Mutation

The mutation operator now simply changes the value of one or several torsion angles in the vector randomly. For example, the glycosidic torsion angles of the structure in Figure 3.2a are defined in the vector $[(\phi_1, \psi_1), (\phi_2, \psi_2), (\phi_3, \psi_3)]$. We now apply the mutation operator choosing randomly to mutate one angle, in this case ϕ_2. We get the vector $[(\phi_1, \psi_1), (\phi_2^{'}, \psi_2), (\phi_3, \psi_3)]$ representing the offspring structure in Figure 3.2b.

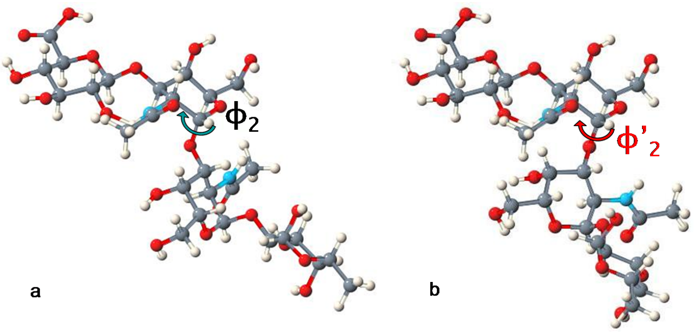


Figure 3.2: Mutation operator. a) The structure before mutation with the values $[(-24, -19), (-19, -27), (-42, 30)]$ b) The structure after mutation (offspring) with the values $[(-24, -19), (\mathbf{25}, -27), (-42, 30)]$.

Crossover

In the crossover operator we will select two structures to be the parent structures. We now choose one crossover point randomly and apply the crossover operations resulting in two offspring. For example, let the first parent be the vector $[(38, 19), (-25, -25), (-41, 29)]$ representing the structure in Figure 3.3a and the second parent be the vector $[(39, 25), (-19, -26), (-47, -11)]$ representing the structure in Figure 3.3b. Let the crossover point be between the first and second glycosidic bonds. The result will be the two following offspring $[(38, 19), (-19, -26), (-47, -11)]$ shown in Figure 3.3c and $[(39, 25), (-25, -25), (-41, 29)]$ shown in Figure 3.3d.

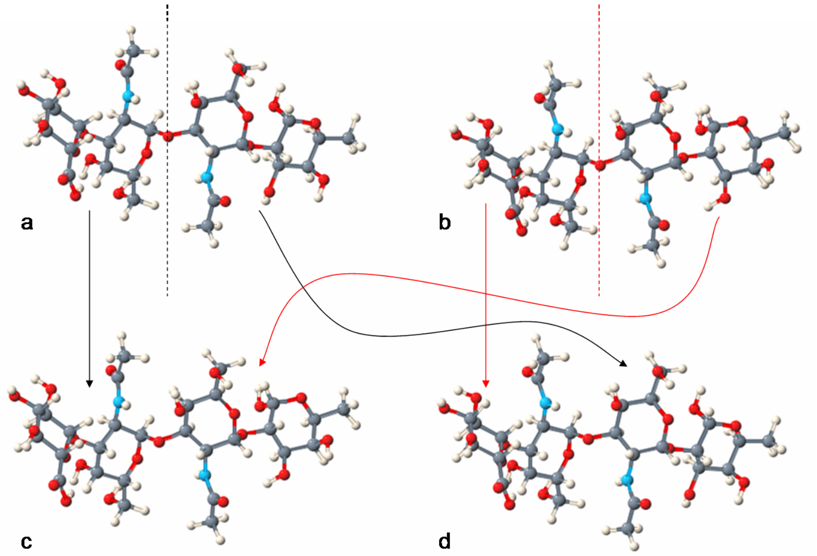


Figure 3.3: Single-point crossover operator. a and b illustrates the parent structures whereas c and d represents the offspring structures. The dotted line on the parents shows the crossover point. The black dotted arrows coming from the parents structures a and b showing the parts of the parents that construct the offspring c. In a similar way the red dotted arrows are for the construction of offspring d.

The crossover operator can be modified in different ways. One such modification could be choosing more than one crossover points and thus the possibility of generating more than two offspring. Another modification could be to choose more than two parents.

3.1.3 Fitness function and selection

As described in chapter 2, one important feature of genetic algorithm methods is selection. How do we select the individuals of a population on which we will apply the genetic operators? We will have to, in some way, set a value for each individual in the population. We use a so called fitness function that will evaluate the individuals. In our case the individuals in the population are oligosaccharide structures. How do we evaluate their “fitness”? What is a “good” structure? and what is a “bad” one?

Force field methods are widely used for predicting favored conformations. Also in the case of modeling oligosaccharides using GA search and force field methods seems to be a good choice.

The MM3 program is one of the best force fields available today for saccharides. Energy minima calculations done using MM3 have been shown to fit well with experimental data on disaccharide structures obtained by X-ray crystallography.

We chose to base our fitness function on energy calculations using the MM3 force field program. Using the *roulette wheel* selection, the fitness of a structure is calculated using the scoring function presented in formula 3.1 where E_{min} is the energy of the best structure and $E_{max} = Min(worststructure, E_{min} + threshold)$.

$$\begin{aligned} S_{best-structure} &= 10 \\ S_{worst-structure} &= 1 \\ S_i &= 1 + (10-1)\frac{E_i - E_{min}}{E_{max} - E_{min}} \end{aligned} \tag{3.1}$$

Constructing the next generation

The construction of the next generation in a GA is fairly simple after the structures were evaluated using the fitness function and the GA operators were applied on the present generation according to the selection criteria. One would normally use the mutation operator on 5 percent of the population where structures with “better” fitness have a higher probability of being selected. The mutated structures will be copied to the next generation. We would usually use the crossover operator on 85 percent of the population where the selection of the parents can be decided in different ways. One way is to choosing at most one of the parents that has “good” fitness. In that way one can prevent the so called *crowding* problem where many copies of only “good” structures will be generated and the population will converge too fast. The remaining 10 percent of the next generation are the best 10 percent structures from present generation copied directly.

The structures in the newly constructed generation are now evaluated using the fitness function and the process will continue until termination criteria will be achieved.

3.2 The GLYGAL program

The major goal of this project was to turn all the above theory into easy to use software for oligosaccharide conformational search using GAs and MM3 force field for energy calculations.

The GLYGAL software developed in this project is the fulfillment of this goal. The program implements three different GAs for the purpose of oligosaccharide conformational search. The algorithms implemented are standard GA, parallel GA and an evolutionary programming algorithm as well as one simple Monte Carlo implementation used mainly for comparison. All the GAs implemented can be used with local minimization (e.g. Lamarckian GA) performed by the MM3 program. Some default GA parameters, such as population size, number of iterations etc. are suggested by to the user and those can easily be set using the GLYGAL graphical user interface (GUI). Some snapshots of GLYGAL can be seen in the GLYGAL user manual presented in Appendix C.

Given that a structure was chosen for a conformational search and the GA search algorithm was also chosen. A file containing a representation of the structure is used as a template file. A simplified series of steps describing the conformational search would look like this:

1. The torsion angles of the structure to be adjusted in the search are identified.
2. The template structure file is copied as many times as the size of the first population of the GA.
3. The copied structures are modified randomly by adjusting torsion angles to create the first randomly generated population of the GA.
4. The files are then sent to the MM3 program for evaluation and local minimization and the results are sent back to GLYGAL.
5. The selection process in GLYGAL picks the structures to be manipulated by the genetic operators.
6. Genetic operators, such as mutation and crossover, are performed on the structures and the next generation is created.
7. Evaluation of the structures is done and termination criteria are checked. If not fulfilled resume process from step 4.

In this section we describe the algorithms developed in GLYGAL and the environment GLYGAL is working in. More detailed information about the use and development of the software is given in appendices C and D.

3.2.1 The "wrapper" program SugaRun

As mentioned, GLYGAL uses the MM3 force field program for the energy evaluation and local minimization. The SugaRun program, developed by Jimmy Rosen, is a "wrapper" program used by GLYGAL and mainly takes care of the connection with the MM3 program. The MM3 program requires a .mm3 file as input. Apart from the Cartesian representation of the molecule and atom connectivities this file, also has some parameter lines. SugarRun is also responsible for the distribution of "jobs" generated by GLYGAL to the different nodes in the cluster. Each "job" is a request generated by GLYGAL for energy calculation from MM3 and performs energy evaluation and minimization on one or more structures.

The cluster used is a Linux cluster of eight nodes with dual 2200+ AMD processors on each node. GLYGAL can submit jobs and get the results from the server in parallel. GLYGAL can also resubmit jobs that generated malformed results and restart the server when it is not responding.

3.2.2 Identifying rotatable bonds

One of the main problems with the computational methods existing for oligosaccharide modeling is the need of preprocessing work, both manual and computational. In the filtered systematic search described in chapter 2, manual and computational preprocessing is needed before one can start the search.

One such problem was identifying the torsion angles. To be able to adjust the torsion angles one needs to identify them first. Until now one would need to locate manually the torsion angles using a 3D molecular viewer. In GLYGAL we solved this problem by developing an algorithm for identifying the torsion angles automatically.

This algorithm creates a connection matrix containing the torsion angles of all the glycosidic linkages and a ring vector containing all the torsion angles needed to identify the position of all atoms within a certain ring. The outline of the algorithm is:

1) identify the groups of connected carbon atoms and finds the oxygens that are connected to two carbon atoms (Figure 3.4a).

2) An oxygen connected to two carbons belonging to the same carbon group identifies a ring. The rings are numbered as shown in Figure 3.4b.

The procedure is as follows:

- The carbon atom which is not connected to any other carbons is numbered carbon 1. If both carbons are connected to other carbons (e.g. furanoses), the program assigns the label 1 to one of those two carbons.
- The chain that links the first carbon to the ring oxygen is found using a backtracking approach, in order to take into account possible residue substitutions in the ring. Once the chain is identified, the carbon atoms belonging to this chain are numbered in ascending order.
- Adds the carbon 6 and possible other carbons connected.

3) the oxygens connecting carbons belonging to two different rings identify the glycosidic linkages (Figure 3.4c). The torsion angles defining the bond are then identified according to the light atom definition and added to the connection matrix.

4) the carbon atoms in the ring and the atoms in the bonds are marked as determined. While the position of an atom directly connected to those is determined, the position of an atom whose "distance" is two bonds or more, depends on the values of one or more torsion angles.

The residue chains are thus parsed according to the minimum number of bonds needed to connect all atoms to the root (normally one of the ring carbon atoms). In the example in Figure 3.4d, this value is equal to three for the linkage C5-C6: C5 (determined), C6, O6 and H6 (furthest edge). It is interesting to note that in case of glycosidic linkages 1 → 6 the carbon 6 will be determined by the bond and the maximum chain length will be just two C6 (determined), O6 and H6 (furthest edge). In NAc group the value assumed is four for the CX-N connection, three for the N-C connection, etc. GLYGAL selects all the bonds where the minimum number of required bonds is greater than a specified threshold (the default value is two and includes every possible torsion angle) and adds a torsion angle to the ring vector.

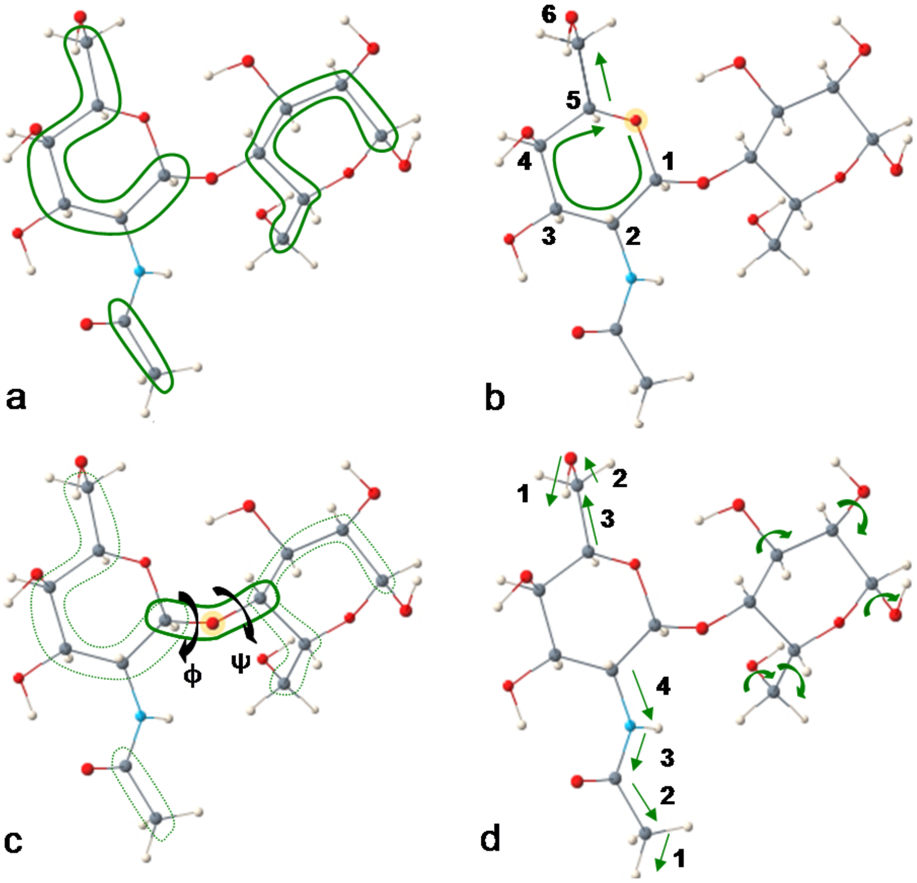


Figure 3.4: Analysis of the structure β-D-GalpNAc-(1→4)-D-Glc. **a)** Identification of the carbon groups. **b)** Numbering of the carbon atoms of the Gal. **c)** Identification the bond torsions in the glycosidic linkage. **d)** Identification of the torsion angle in the the rings. On the left side, the minimum number of bonds needed to connect all the atoms of the subtree are shown for the NAc group and for the sixth carbon. On the right side, the torsion angles needed to determine the position of all the Glc atoms are shown.

After the connection matrix and the ring vector have been created, a “Glyco-structure” vector is created as follows:

Initialization: Find a ring that is connected to just one other sugar ring.

Recursive step:

- add the bonds to the vector from the branch that has more rings to the one that has less.
- if there is one or more branches, add bond bonuses to the bond genes for each branch. The number of branches is simply defined as $max(0, numberOfConnectedRings - 2)$.

- calls the recursive function on the linked rings if they have not yet been processed, from the branch that has less rings to the one that has more.

This method keeps related genes as close as possible in the “genome”. The bonds in a branching point, which are the most important genes for the search, are grouped together tightly with the branching ring, as these elements interact significantly. The choice to add the smaller sub-branch first keeps the rings as close as possible from the branching point.

3.2.3 Graphical user interface

One important task when creating the software was providing a user-friendly GUI. The GUI was created mainly using the Java swing components is used for convenient parameters setting as well as for molecule visualizing purposes with the Jmol molecular viewer [16]. More information about the use of the GUI is given in the user manual presented in Appendix C.

Chapter 4

Results

In this chapter we present the results obtained in this project. We first present a summary of the performance of GLYGAL in comparison to the MM3 filtered systematic search. We then give the results obtained concerning some of the more biologically significant oligosaccharide structures we investigated using GLYGAL.

4.1 Program performance

One of the main problems in predicting oligosaccharide conformations with the help of computational methods is finding a good trade-off between sampling of the conformational space and computational time. In systematic search for example, one would have to cover the search space in a more thorough way having to pay a high price in the run time of the search. For example, sampling a pentasaccharide with eight torsion angles in 15° steps around each rotatable bond results in over 100 thousand million conformations. It would take years to calculate the energy of these conformations on a PC in order to find the conformation with the lowest energy. On the other hand one can ignore certain features of the oligosaccharide energy in the force field, when using a truncated force field as in Glycan, to achieve a faster result but with a less accurate value calculated of the molecular energy giving a less accurate prediction.

In this section we present a summary of the performance of GLYGAL. This summary is mainly a comparison with the filtered systematic search in terms of search speed. The filtered systematic search is a good comparison measurement since it also uses the MM3 program for molecular energy calculations.

4.1.1 Performance summary

What we were interested in investigating, apart from the performance of GLYGAL in comparison to other methods, was the behavior of different GAs implemented within GLYGAL. Is there any advantages using a certain GA for a certain structure size? To answer this question conclusively one would need to perform very extensive testing using different GAs on different sizes of structures. This type of testing was not possible in the time frame of this project so the results, or let us say observations, we present here are based on the testing we performed combined with a gut feeling we acquired during this time.

We have looked at the performance of the three GAs: standard GA, parallel GA and evolutionary programming. These were tested on different oligosaccharide structures having 2-7 residues, linear or branched. The performance is measured by the number of structures needed to be sampled in order to find the best conformation.

We observed the following:

- almost no difference between the different GAs when investigating disaccharides. On average we need to sample about 100 structures.

- for larger structures, linear or branched, the parallel GA is preferable. We would need to sample fewer structures using a parallel GA.

- the number of structure we need to sample for a branched oligosacchride is larger than the number needed for a linear structure when both have the same number of residues.

- using any GA we could observe that an increase of the number of structures in the initial population would not necessarily result in a faster convergence to the minima.

In Table 4.1 we summarize the performance of GLYGAL compared to the filtered systematic search. The comparison is done on the number of structures needed to be sampled in order to get the best conformation. Observe that the number of structures needed to be sampled by GLYGAL is an average number whereas the numbers used in the filtered systematic search are constant. For example, we need to sample 576 structures with the filtered systematic search for a disaccharide and only 100 (on average) using GLYGAL.

Search method	Saccharide type	Sampled structures
Filtered systematic search	Disaccharide	576
	Trisaccharide	150000
	Tetrasccharide	too many
	Pentasaccharide	too many
Genetic algorithms	Disaccharide	100
	Trisaccharide	1000
	Tetrasccharide	5000
	Pentasaccharide	10000

Table 4.1: The number of structures needed to be sampled to find best conformation, using MM3 filtered systematic search (FSS) versus genetic algorithm search.

4.2 Results of biological importance

In this section we will concentrate on some examples with a biological/chemical significance. In the first part will presents the results on the blood group A related oligosaccharide structure and compare these to the experimental and computational data available. In the second part we investigate fragments of the O-specific oligosaccharide found on the *Shigella dysenteriae* type 2 and 4 for which computational data is available. The third part we present our results on the oligosaccharide found on the surface of the cercariae of the *Schistosoma mansoni* where no published data is available.

4.2.1 Blood group and related oligosaccharide

The ABO blood group system was first discovered by Landsteiner in 1901. The ABO(H) type blood system has been shown to be specified by carbohydrate present on human red blood cells. This system consists of three antigens H, A and B and is highly important in blood transfusions. More antigens, some protein dependent and some carbohydrate dependent, have been identified since.

The medical importance of blood groups and the related oligosaccharides have led many researchers to study the preferred conformation of these molecules using computational methods as well as experimental methods [17, 18, 19].

The existence of experimental as well as computational data on the ABO(H) antigens was a good reason to start our tests on these oligosaccharide structures. We chose to perform the tests on probably the most investigated one, the blood group A antigen.

The A antigen is a trisaccharide with the sequence $\alpha - L - Fuc - (1 \rightarrow 2) - [\alpha - D - GalNAc - (1 \rightarrow 3)] - \beta - D - Gal - 1$. We performed tests using different GAs with different sets of the GA parameters to see that the most favorable conformations we got were coherent with the results in the literature [17, 18]. The favorable conformation is shown in Figure 4.1. The ϕ, ψ for the linkage $\alpha - L - Fuc - (1 \rightarrow 2) - Gal$ is ≈ 28,-50° and for the $\alpha - GalNAc - (1 \rightarrow 3) - Gal$ linkage is ≈ -40,-44° .

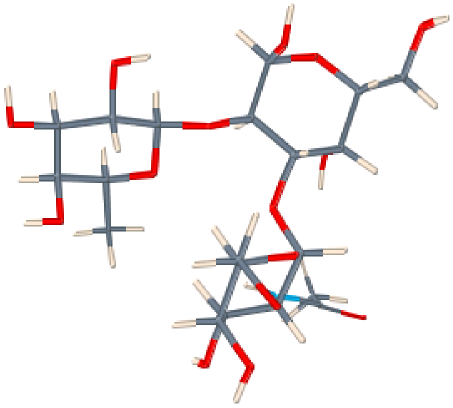


Figure 4.1: A antigen trisaccharide favorable conformation.

4.2.2 *Shigella dysenteriae*

Shigellosis is an infectious disease caused by a group of bacteria called *Shigella*. It is endemic in many developing countries with approximately 170 million cases in the world. It is estimated that around one million people die each year from Shigella infections. *Shigella dysenteriae* (Sd), which accounts for around 5 percent of all cases of shigellosis, has been classified in 10 different serotypes going from 1 — the most severe one — to 10. Investigating the 3 dimensional structures of the O-specific oligosaccharide of *Shigella* bacteria is of a great interest for the development of glycoconjugate vaccines [20].

The O-specific oligosaccharide 3D structures of Sd1 has been the subject of experimental as well as computational studies [13, 21]. The O-specific oligosaccharides of Sd2 and Sd4 were, among other structures, the subject of a computational study within our structural biochemistry group [14]. Using filtered systematic search Rosen *et al.* could predict favorable conformations of those O-specific oligosaccharides.

Here we present the results, obtained using the GLYGAL program, concerning these structures and compare them with the results obtained by the filtered systematic search.

Shigella dysenteriae type 2

Rosen *et al.* investigated the O-specific oligosaccharide of Sd2. They started by investigating one repeating unit of the polysaccharide. The sequence of the repeating unit is illustrated in Figure 4.2. This pentasaccharide was first divided into fragments of disaccharides. A systematic search on the ϕ/ψ space with 15° step size using the MM3 force field was performed on each of the disaccharides. The results were energy maps providing a nice overview of the ϕ/ψ search space of each of the disaccharides. The energy maps for the different disaccharides are presented in the left column of Figure 4.3. The energy maps were then used as filters for the systematic search performed on the trisaccharide and partially on a tetrasaccharide on the branching point for favorable ϕ/ψ angles. The new energy maps for the trisaccharide at the branching point are shown on the right column of Figure 4.3. They could then construct the whole structure manually using the minimized fragments to get the favorable conformation shown in Figure 4.4. In total the filtered systematic search had to sample approximately 150000 structures for this prediction.

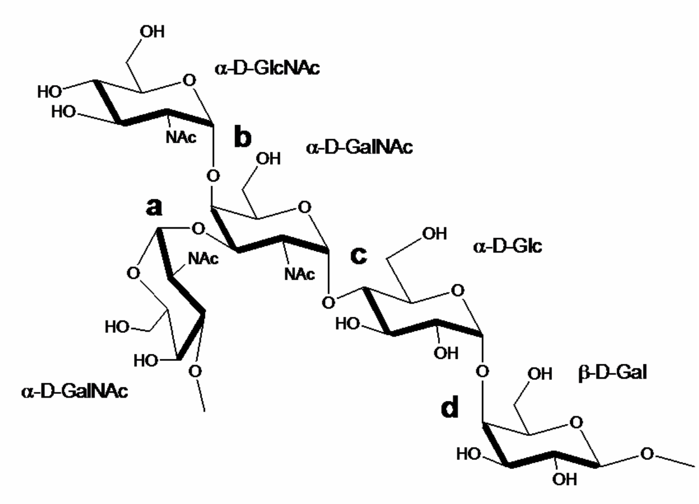


Figure 4.2: One repeating unit of the O-specific polysaccharide of *Shigella dysenteriae* type 2. a, b, c and d represents the different linkages.

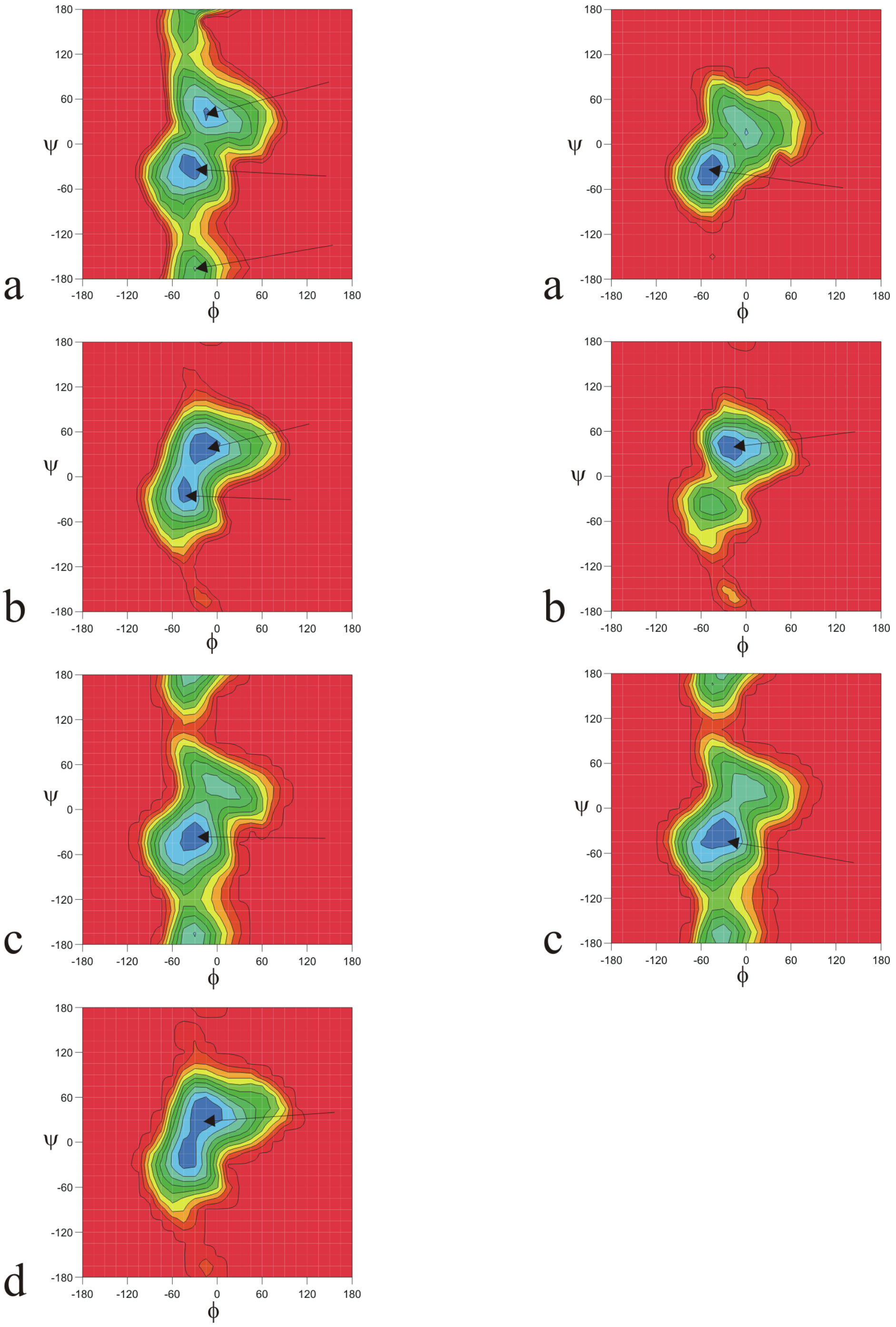


Figure 4.3: MM3 Adiabatic energy maps generated by filtered systematic search for the different disaccharide moieties (left column) as well as for the trisaccharide at the branching point (right column) of the repeating unit of the Sd2 O-antigen. The arrows mark the minima found using the GLYGAL program. The enumeration a to d refers to the linkages in Figure 4.2.

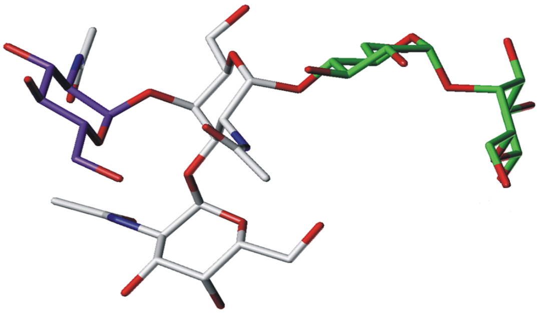


Figure 4.4: The 3D structure of the repeating unit of the O-specific polysaccharide of sd2 predicted by the filtered systematic search (Rosen *et al., in press*).

For comparison purposes we used the GLYGAL program with a similar search scheme to the search scheme suggested by Rosen *et al.*(in press). We first divided the repeating unit oligosaccharide to fragments of disaccharides and used GLYGAL to search for favorable conformations. The GA runs showed excellent convergence to the minima found in the systematic search. The arrows on the left column of Figure 4.3 mark the minima found using GLYGAL. For the searches we used a Lamarckian GA with 10 structures in the initial population. On average 10 generations were needed to complete the search and find the minima. The searches were performed using the cluster in a couple of minutes each.

We then tested the trisaccharide and tetrasaccharide on the branching point and obtained excellent results. Here again, all minima found using systematic search, were found by the GA (marked with arrows in right column of Figure 4.3). In this case a Lamarckian parallel GA was used with 4 initial populations each with 40 initial structures. After only 10 generations over 70 percent of the populations converged to the low energy minima found in the systematic search.

The complete repeating unit pentasaccharide was not tested using the filtered systematic search due to the huge CPU time needed for this kind of search. Using GLYGAL, on the other hand, we could conduct a search on the repeating unit as a whole and the result showed a very good fit to the structure constructed by Rosen *et al.* (see Figure 4.5). Using the GLYGAL program we sampled approximately 8000 structures for the prediction.

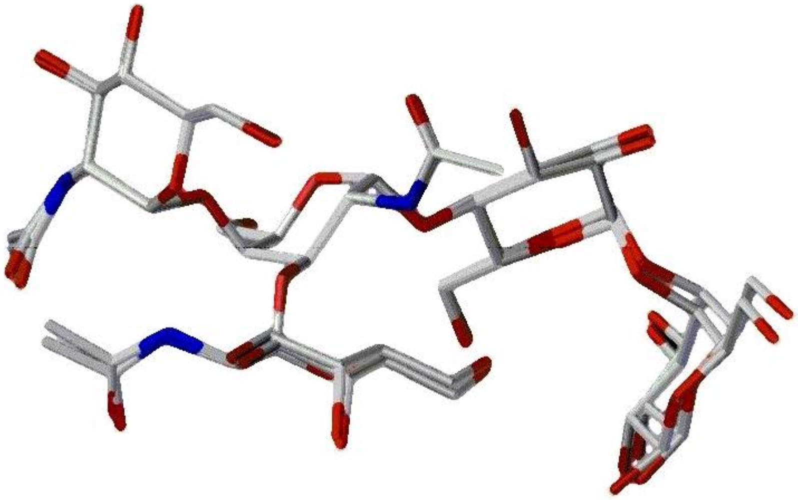


Figure 4.5: Superimposition of the 3D structure of the Sd2 repeating unit obtained from GLYGAL and the filtered systematic search.

Shigella dysenteriae type 4

The O-specific oligosaccharide of the Sd4 was investigated in a similar way to the O-specific oligosaccharide of the Sd2. Here again we used the filtered systematic search for comparison. Figure 4.6 illustrates Sd4 MM3 adiabatic energy maps generated by the filtered systematic search. The arrows represent the minima found using the GLYGAL program. We observe that GLYGAL detected the minima for the linkage $\alpha - D - GlcNAc - (1-3) - \alpha - D - GlcNAc$ shifting from one favorable minima when investigating the disaccharide (energy map A in lower row) to another minima when investigating the trisaccharide (energy map A in upper row) on the branching point. The minima found when investigating the trisaccharide is the favorable minima even when investigating the structure as whole using GLYGAL. We observe also that GLYGAL detected the change in minima for the linkage $\alpha - L - Fuc - (1-4) - \beta - D - GlcNAc$ (energy map B in lower and upper rows). Investigating the trisaccharide at the branching point three distinct minima were found using GLYGAL.

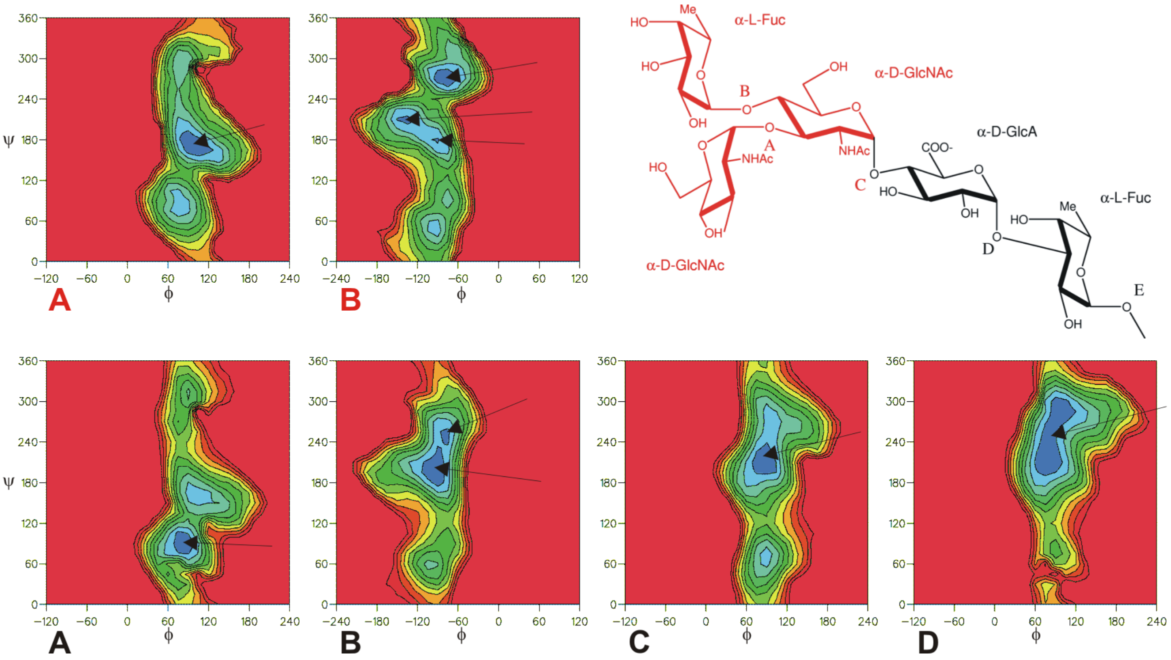


Figure 4.6: MM3 Adiabatic energy maps generated by filtered systematic search for the different disaccharide moieties (lower row) as well as for the trisaccharide at the branching point of the repeating unit of the Sd4 O-antigen. The arrows mark the minima found using the GLYGAL program.

4.2.3 *Schistosoma mansoni*

As mentioned in the introduction chapter Schistosomiasis is a parasitic disease, caused by blood flukes of the genus *Schistosoma* affecting 200 million people worldwide. There are number of species of *Schistosoma* that can affect humans, but most humans are affected by one of the three following species: *Schistosoma mansoni*, *Schistosoma haematobium* and *Schistosoma japonicum*.

The life cycle of the parasite, presented fully in Appendix B, involves two hosts: fresh water snail and human. After the intermediate host (snail) is infected the cercariae are released from the snails. The cercariae can penetrate the human body through the skin and the human is infected. The fact that the parasites remain in both hosts for some time, suggests that they have developed mechanisms to avoid the immune systems of the hosts.

In this section we present results obtained for a highly specific oligosaccharide structure existing on the surface of the cercariae of the *Schistosoma mansoni*. This structure, among other structures, is proposed to elicit the formalin of protective antibodies [3].

The sequence representation of the oligosaccharide is shown in Figure 4.7 where the hexasaccharide highlighted in red is the subject of our minimization. Within this hexasaccharide there is a tetrasaccharide that has been shown to be highly specific for the parasite [22] and is hence a good target for vaccine development.

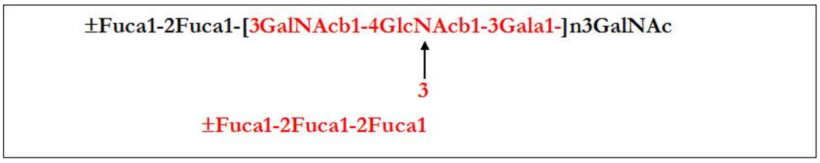


Figure 4.7: Sequence representation of the oligosaccharide existing on the surface of the cercariae of the *Schistosoma mansoni*. The highlighted tetrasaccharide is specific for the parasite i.e. it does not occur in humans.

For the search performed by GLYGAL we used a parallel GA with 4 populations of 60 structures each running for 40 generations which gives approximately 10000 structures that were sampled. The predicted structure is shown in Figure 4.8.

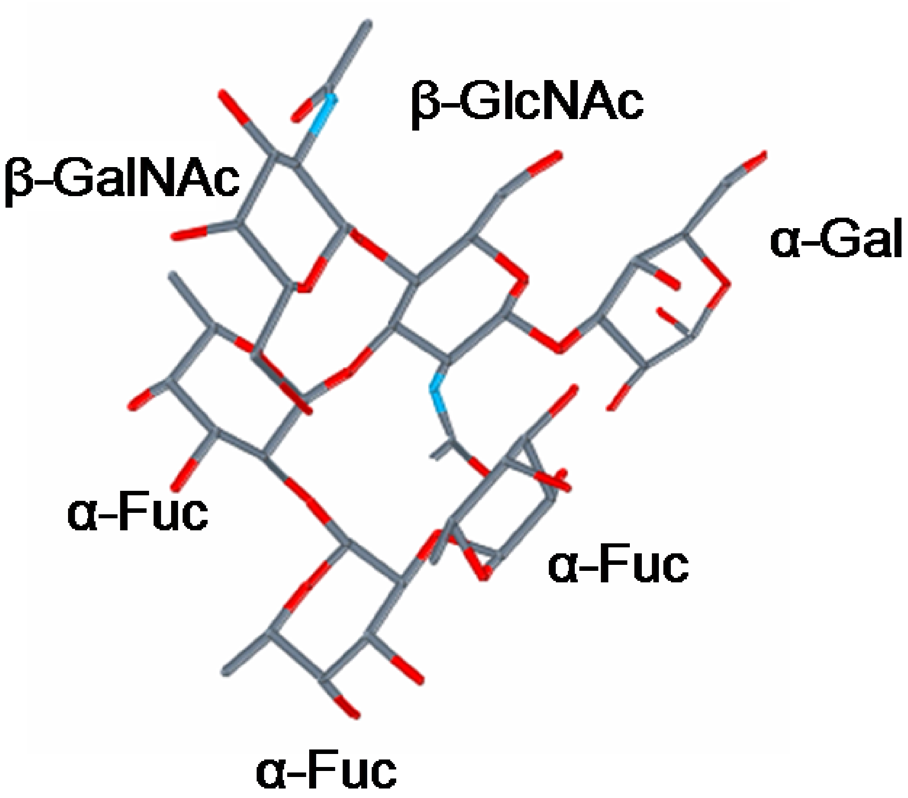


Figure 4.8: The predicted 3D structure of the studied oligosaccharide of *Schistosoma mansoni*.

The structure presented here is a novel model for a biologically important oligosaccharide from *Schistosoma mansoni*. At present, no experimental nor computational data on this structure are available in the literature. It is hoped that this model might give insights into the molecular mechanism of this disease.

Chapter 5

Conclusions

5.1 Achievements and evaluation

The main aim of this project was to investigate the performance of genetic algorithm search methods for predicting oligosaccharide 3D structures based on the MM3 force field for energy calculations. For that we developed GLYGAL, a program coded in JAVA that implements different genetic algorithms. GLYGAL includes an "easy to use" GUI incorporating a three dimensional molecular viewer and uses the SugaRun program for the communication with the MM3 program.

Using GLYGAL we were able to investigate different oligosaccharide structures showing that genetic algorithm search methods, though based on heuristics, are a very good tool for predicting oligosaccharides 3D structures. Our results showed that using GAs is faster since at least fifteen times fewer conformations need to be sampled in comparison to the filtered systematic search.

Identifying torsion angles is now done automatically using an algorithm developed in this work and is implemented in GLYGAL.

The MM3 program used for force field energy calculations and local minimization is probably the most established program existing today for this type of work. On the other hand MM3 is highly sensitive and would crash often when doing large scale tasks. The SugaRun "wrapper" program developed by Jimmy Rosen deals mainly with communication with the MM3 program. It has evolved significantly during the course of this project and is now a stable and reliable program for its task.

Incorporating the 3D molecular viewer Jmol into GLYGAL main desktop was a good design decision taken during this project. For a modeler, being able to look at a structure prediction in three dimensions is a necessity.

5.2 Future work

"Contrary to the outstanding work of art, outstanding theory is susceptible to improvements" K. Popper

The list below includes the main points for future extensions to GLYGAL:

- Start with a sequence. In its current version GLYGAL cannot start a search provided with an oligosaccharide sequence but only with a 3D template representation of the oligosaccharide. Since all the parts needed to complete this task are available only an integration task remains.
- Filters. As in the filtered systematic search, one could even reduce the search space explored by the GA using filters. There are different possible ways of achieving this, one of which is using the potential energy maps generated using systematic search.
- Better statistics for GA parameter setting. An extensive testing of many different structures may enable us to establish better default values for the GA parameters.
- Other fitness functions. Since the MM3 program has been sensitive , we would like to try connecting GLYGAL to other molecular mechanics software, such as the free open source program MM3-TINKER.

Bibliography

[1] S. Schulze-Kremer, *Genetic Algorithms and Protein Folding*, `http://www.techfak.uni-bielefeld.de/bcd/Curric/ProtEn/contents.html`

[2] Allinger's Molecular Mechanics Research Lab homepage `http://europa.chem.uga.edu/`

[3] R.D. Cummings, and A.K. Nyame, (**1996**) *Glycobiology of schistosomiasis.* FASEB J. 10:838-848.

[4] J.H. Holland, (**1975**), *Adaptation in natural and artificial system.* University of Michigan Press (reprinted in 1992 by MIT press, Cambridge, MA).

[5] T.M. Mitchell, (**1997**), *Machine Learning*, chapter 9 pages 249-262.

[6] T.M. Mitchell, (**1996**), *An introduction to genetic algorithms.* MIT press, Cambridge, MA.

[7] J.B. Lamarck, (**1914**), *Zoological Philosophy*, Macmillan, London. Originally published in Paris in 1809.

[8] `http://www.rpi.edu/dept/bcbp/molbiochem/MBWeb/mb1/part2/sugar.htm`

[9] R.A. Dwek, (**1996**), *Glycobiology: Toward understanding the function of Sugars.* Chem. Rev. 96, 683-720.

[10] V.S.R. Rao , P.K. Qasba, P.V. Balaji, and R. Chandrasearan, (**1998**), *Conformation of Carbohydrates*, pages 49-189.

[11] J.J. Grefenstette, (**1991**) *Lamarckian learning in multi-agent environment.* In R. Belew and L. Booker (Eds.). Proceedings of the fourth international conference on genetic algorithms. San Mateo, CA: Morgan Kaufmann.

[12] G.M. Morris, D.S. Goodsell, R.S. Halliday, R. Huey, W.E. Hart, R.K. Belew, and A.J. Olson, (**1998**), *Automated Docking Using A Lamarckian Genetic Algorithm*

and an Empirical Binding Free Energy Function, Computational Chemistry J. 19, 1639-1662.

[13] P-G Nyholm, L.A. Mulard, C.E. Miller, T. Lew, R. Olin, and C.P. Glaudemans , (**2001**), *Conformation of the O-specific polysaccharide of dysenteriae type 1: molecular modeling shows a helical structure with efficient exposure of the antigenic determinant alpha-L-Rhap-(1→2)-alpha-D-Galp*, Glycobiology ; 11:945-55.

[14] J. Rosen , A. Robobi , and P-G Nyholm, (**2002**) *Conformation of the branched O-specific polysaccharide of Shigella dysenteriae type 2: Molecular mechanics calculations show a compact helical structure exposing an epitope which potentially mimics galabiose*, Carbohydrate research.

[15] L. Davis, (**1991**) *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York.

[16] Jmol homepage:
`http://jmol.sourceforge.net/`

[17] M. Biswas, and V.S.R Rao, (**1982**), *Conformational studies on the ABH and Lewis blood group oligosaccharides.* Carbohydr. Polymers, 2, 205-222.

[18] M. Biswas, (**1982**), *Conformational studies on blood group and related oligosaccharides.* Ph. D. thesis, Indian Institute of Science, Bangalore.

[19] A. Imberty, E. Mikros, J. Koca, R. Millicone, R. Oriol, and S. Perez, (**1995**), *Computer-simulation of histo-blood group oligosaccharide - energy maps of all constituting disaccharides and potential energy surfaces of 14 ABH and Lewis carbohydrate antigens.* Glycoconj. J.,12, 331-349.

[20] **WHO** State of the art of new vaccines: research and development.
`http://www.who.int/vaccineresearch/documents/newvaccines/en/index1.html`

[21] B. Coxon, N. Sari, L.A. Mulard, P. Kovc, V. Pozsgay, and C.P.J. Glaudemans, (**1997**) *Investigation by NMR spectroscopy and molecular modeling of the conformations of some modified disaccharide antigens for Shigella dysenteriae type 1*, Journal of Carbohydrate Chemistry, 16, 927-46.

[22] A. van Remoortere, C.H. Hokke, G.J. van Dam, I. van Die, A.M. Deelder, and D.H. van den Eijnden (**1999**) *Various stages of Schistosoma express Lewis^x, LacdiNAc carbohydrate epitops: detection with monoclonal antibodies that are characterized by enzymatically synthesized neoglycoproteins.*

Appendix A

A prototypical genetic algorithm

This prototypical GA was proposed by Tom M. Mitchell [4].

GA (fitness, fitnessThreshold , p, r, m)
  fitness: A function that assigns a score to a solution
  fitnessThreshold:
  p: The number of solutions in the population
  r: The fraction of the population to be replaced by Crossover at each step.
  m: The mutation rate.

- Initialize population: P ← generate p solutions at random
- Evaluate: For each s in P, compute fitness(s)
- While $[maxfitness(s)] <$ fitnessThreshold do

create a new generation, P_n:

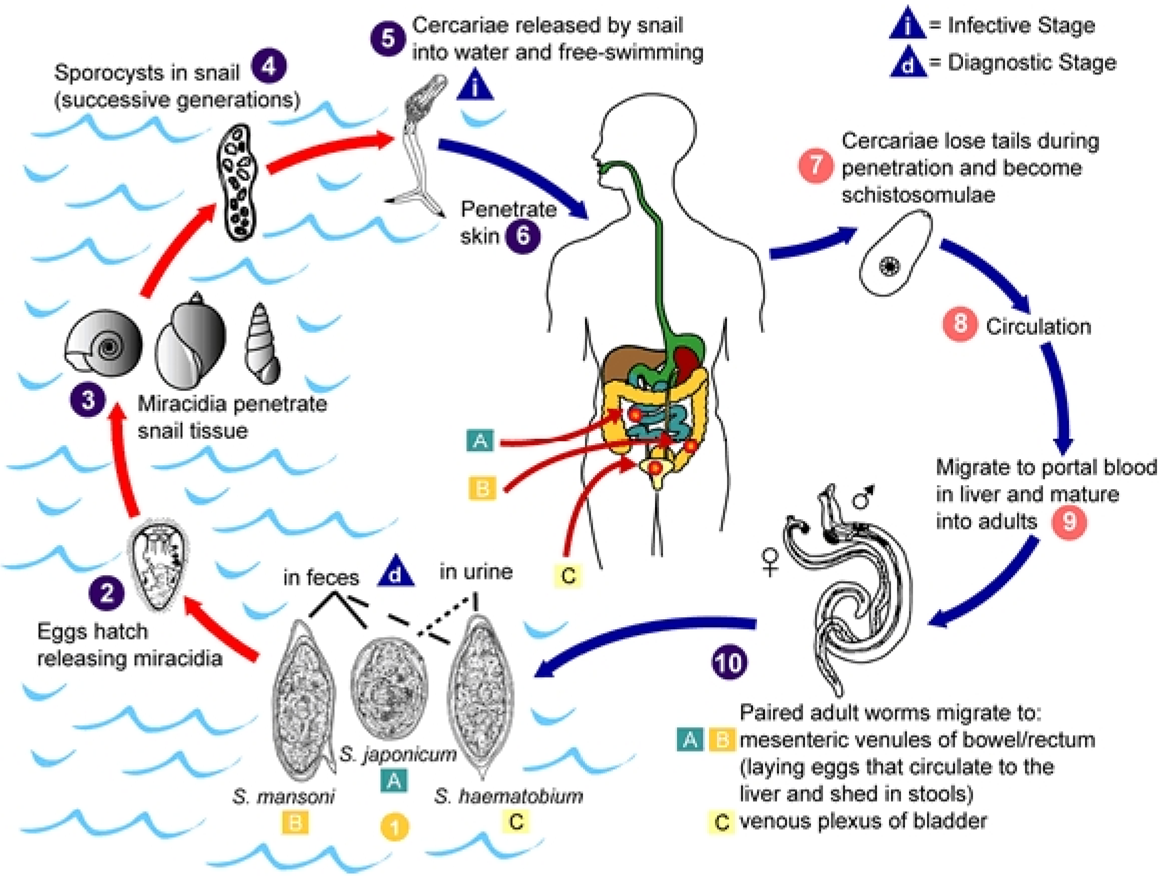
1. select: Probabilistically select (1−r)p members of P to add to P_n. The probability Pr(s_i) of selecting solution s_i from P is given by
$$Pr(s_i) = \frac{fitness(s_i)}{\sum_{j=1}^{p} fitness(s_j)}$$
2. Crossover: Probabilistically select
$$\frac{rp}{2}$$
pairs of solutions from P, according to Pr(s_i) given above. For each pair s_1,s_2, produce two offspring by applying the Crossover operator. Add all offspring to P_n.

3. Mutate: Choose m percent of the members of P_n with uniform probability. For each, invert one randomly selected bit in its representation.
4. Update: P ← P_n.
5. Evaluate: for each s in P, compute fitness(s)

- Return the solution from P that has the best fitness.

Appendix B

Schistosomiasis life cycle



i = Infective Stage
d = Diagnostic Stage
5 Cercariae released by snail into water and free-swimming
i
4 Sporocysts in snail (successive generations)
Penetrate skin 6
7 Cercariae lose tails during penetration and become schistosomulae
8 Circulation
3 Miracidia penetrate snail tissue
A
B
C
Migrate to portal blood in liver and mature into adults 9
♂
♀
2 Eggs hatch releasing miracidia
in feces d in urine
10
S. japonicum
A
S. mansoni
B
1
S. haematobium
C
Paired adult worms migrate to:
A B mesenteric venules of bowel/rectum (laying eggs that circulate to the liver and shed in stools)
C venous plexus of bladder

Figure B.1: The 10 stages in the *Schistosomiasis* life cycle.

Appendix C

GLYGAL user manual

C.1 About GLYGAL

Welcome to GLYGAL - GLYcosidic bonds Genetic ALgorithm. The GLYGAL software, coded in JAVA, implements different genetic algorithm search methods using MM3 force field calculations to predict the 3D structure of oligosaccharides.

GLYGAL in its present form, though can be used as a stand-alone program, is dependent on a "wrapper" program SugaRun created by Jimmy Rosen and, among other things, takes care of the the communication with the MM3 program.

The GLYGAL software was developed by Francesco Strino and Abraham Nahmany as part of a master thesis project in bioinformatics at Chalmers Tekniska Högskola in Göteborg, Sweden.

The project was supervised by Per-Georg Nyholm from the Department of Medical Biochemistry at Göteborg University and Graham J.L. Kemp from the Department of Computing Science at Chalmers Tekniska Högskola.

Version 1.1 January 2004.

C.2 Executing the program

C.2.1 Important facts about GLYGAL

1. GLYGAL is coded in JAVA using the NetBeans IDE 3.5.1 editor which is free software created by SUN. For more information about JAVA and NetBeans please check SUNs homepage on http://java.sun.com
2. GLYGAL runs on a Linux CSOL HOBORG cluster which is *gentoo* based developed by Jimmy Rosen and is using several computers (since written in JAVA it can by easily work on any other platform including UNIX, windows or Mac).
3. The program uses the "wrapper" program SugaRun developed by Jimmy Rosen for communication with the MM3 program and the cluster distribution work. These programs are necessary for the function of GLYGAL but using a different molecular energy evaluator GLYGAL could easily do without the MM3 program.

C.2.2 Program execution

The execution of GLYGAL is very simple when the glygal.jar file is available. One should keep in mind that GLYGAL in its present form is dependent on two programs mentioned above. The calculations were carried out on the Linux cluster running at the structural chemistry group at the Department of Medical Biochemistry at Göteborg University.

For execution type: *java -jar glygal.jar [options]* in a shell window from the glygal.jar file directory.

If no option are specified, a GLYGAL main desktop is created. Otherwise the first argument should be the molecule template file. More options are available from the command line in the form of flags:

-s to start the search directly.

-o [directory name] to set the output directory.

-p [parameter file] to set the parameter file.

-c to start without showing the GLYGAL desktop on the screen. This methods supports only .mm3 input files.

C.3 Using GLYGAL in four simple steps

C.3.1 Step one: Getting started

After executing the program (see Section C.2.2 “Program execution”) GLYGAL main desktop window shown in Figure C.1 will appear:

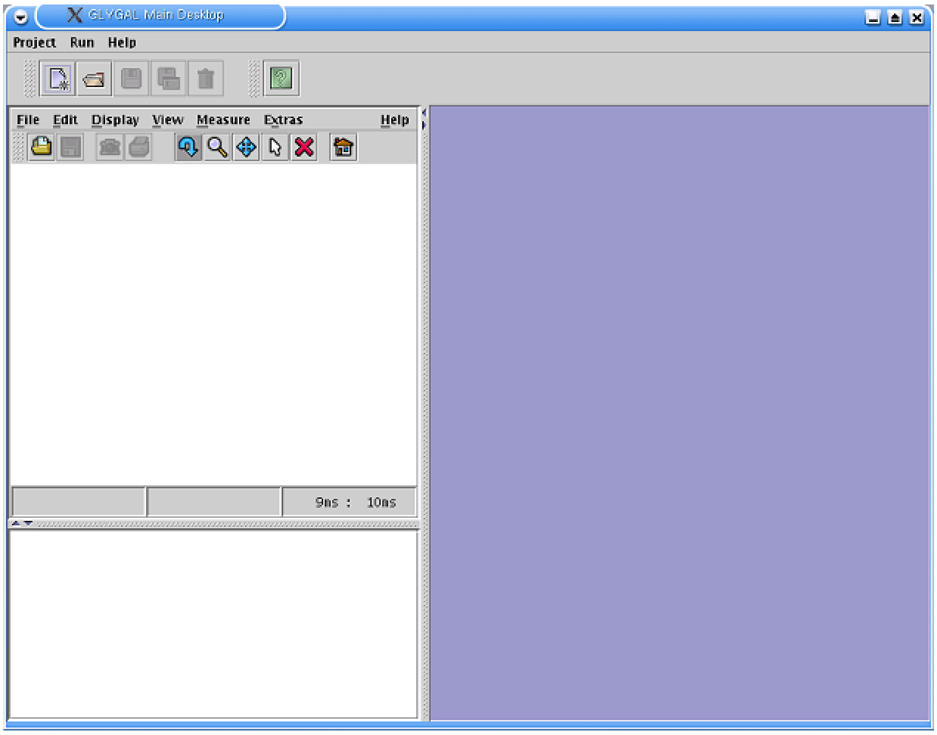


Figure C.1: GLYGAL main desktop.

To create a new project press the new project button from the button toolbar (alternatively use the new item in the Project menu). A project refers to a search performed on a structure and will include all files that are generated as a result of the search. (see more about program output in section C.7: Program output).

To open an existing project press the open project from file button from the toolbar (alt. use the open item in the Project menu). An existing project includes the output files of a search.

C.3.2 Step two: Creating a new project

When choosing to create a new project the dialog window shown in Figure C.2 will show:

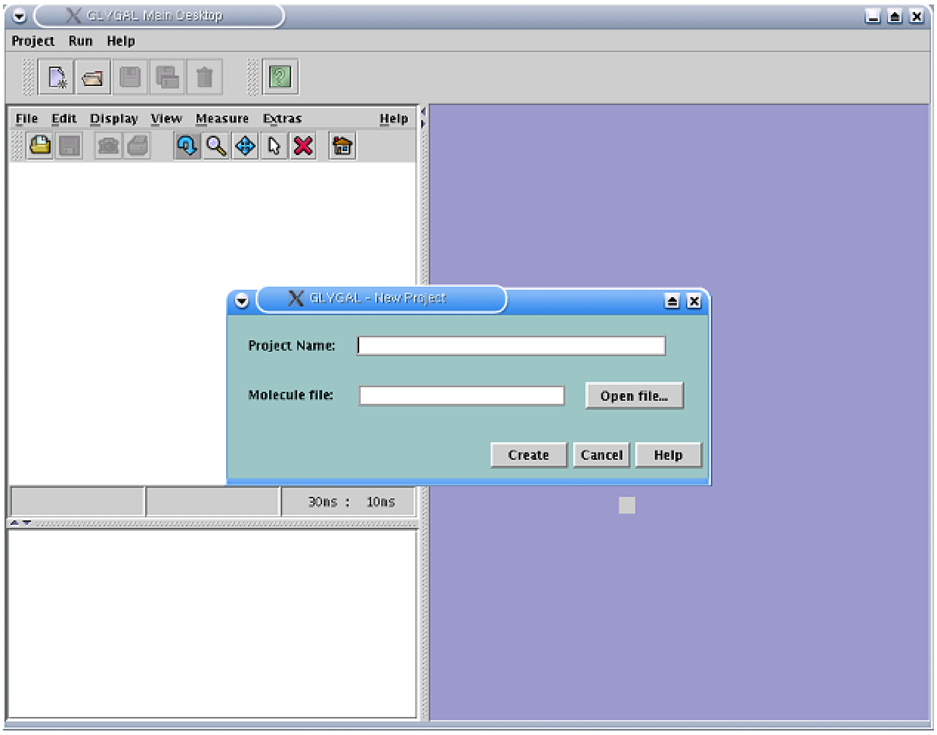
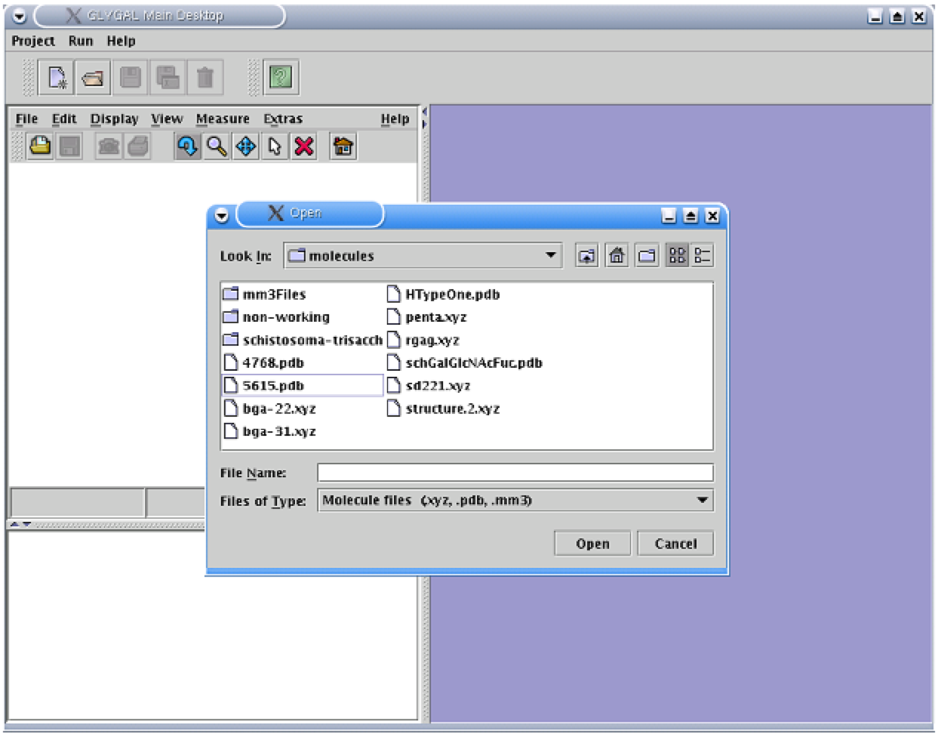


Figure C.2: Create a new project dialog window shown on top of the main desktop.

Press the “Open file” button to open a molecule file (see Section C.4 “GLYGAL - molecule files”). When “Open file” button was pressed the dialog frame shown in Figure C.3 will appear:



Look In: molecules

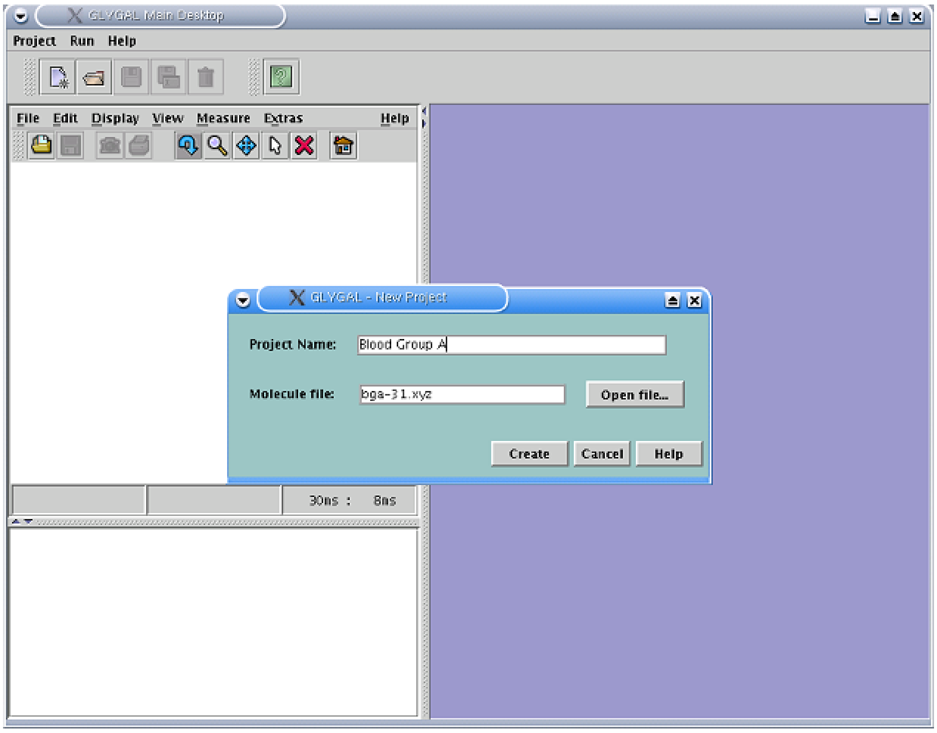
mm3Files
non-working
schistosoma-trisacch
4768.pdb
5615.pdb
bga-22.xyz
bga-31.xyz
HTypeOne.pdb
penta.xyz
rgag.xyz
schGalGlcNAcFuc.pdb
sd221.xyz
structure.2.xyz

File Name:

Files of Type: Molecule files (.xyz, .pdb, .mm3)

Figure C.3: Open a molecule file dialog frame.

Choose a molecule file and press “Open”. A project name will be suggested by the program in the “Project name” text field. To choose another project name type in a new project name in the “Project name” text field as shown in Figure C.4.



Project Name: Blood Group A

Molecule file: bga-31.xyz

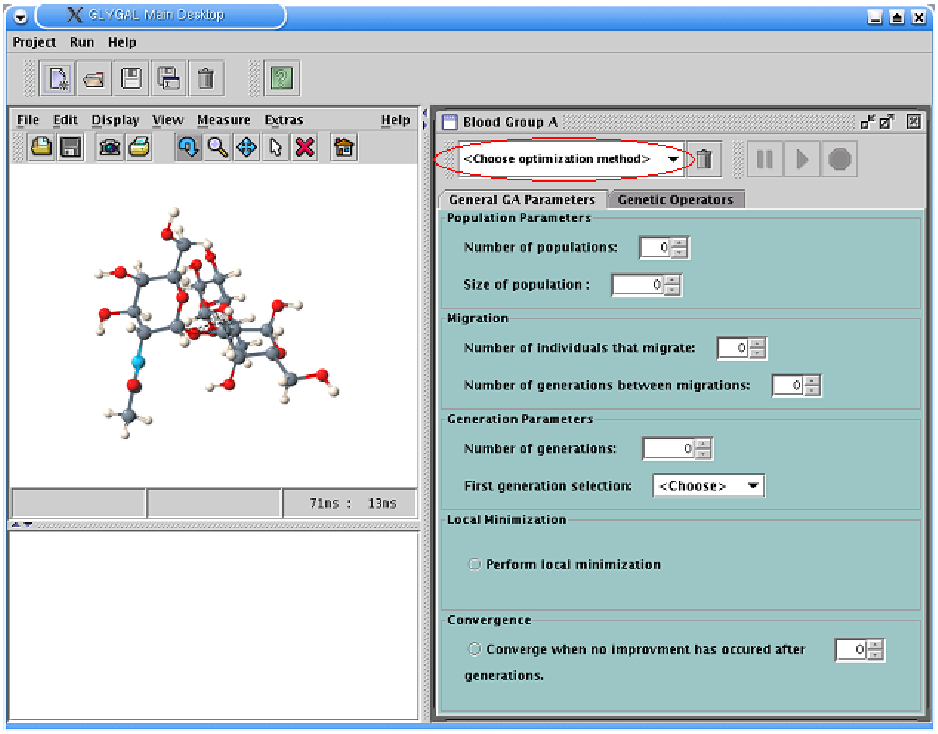
Figure C.4: New project dialog window after choosing a project name.

Press the “Create” button to create the new project.

C.3.3 Step three: Choosing a search method and setting the parameters

When a new project is created a three dimensional representation of the molecule to be optimized appears on the upper left side of the main desktop (see also Section C.6 “Visualizing the molecule”).

To choose an optimization method use the “Choose optimization method” combobox marked in Figure C.5 (see also Section C.5 “GLYGAL - Search Methods”).



Population Parameters

Number of populations: 0

Size of population : 0

Migration

Number of individuals that migrate: 0

Number of generations between migrations: 0

Generation Parameters

Number of generations: 0

First generation selection: <Choose>

Local Minimization

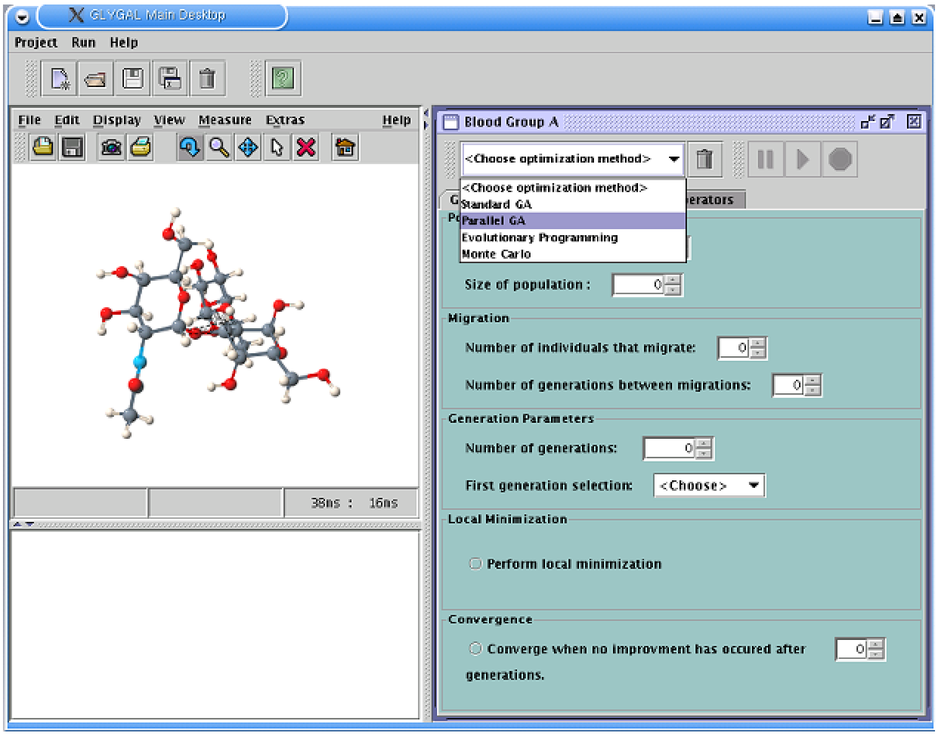
○ Perform local minimization

Convergence

○ Converge when no improvment has occured after 0 generations.

Figure C.5: GLYGAL main desktop. The red circle mark the “Choose optimization method” combobox.

For example, choosing a parallel GA as the optimization method as demonstrated in Figure C.6:



<Choose optimization method>

<Choose optimization method>
Standard GA
Parallel GA
Evolutionary Programming
Monte Carlo

Size of population : 0

Migration

Number of individuals that migrate: 0

Number of generations between migrations: 0

Generation Parameters

Number of generations: 0

First generation selection: <Choose>

Local Minimization

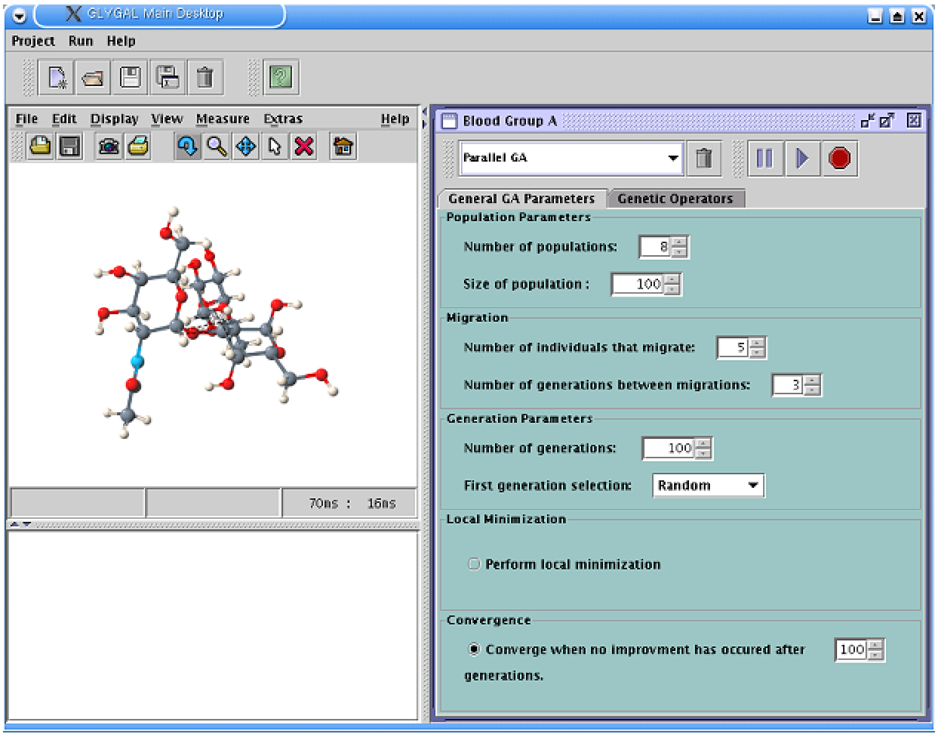
○ Perform local minimization

Convergence

○ Converge when no improvment has occured after 0 generations.

Figure C.6: GLYGAL main desktop. Choosing parallel GA as the optimization method in the "Choose optimization method" combobox.

When the optimization method is chosen, a set of default values for the parameters in the panels general GA parameters and genetic operators are shown. This is illustrated in Figure C.7 Those parameters can easily be changed by the user (see Section C.5 “GLYGAL - Search Methods”). The “Run”, “Stop” and “Pause” buttons will also be activated as shown in Figure.



Number of populations: 8

Size of population : 100

Number of individuals that migrate: 5

Number of generations between migrations: 3

Number of generations: 100

First generation selection: Random

○ Perform local minimization

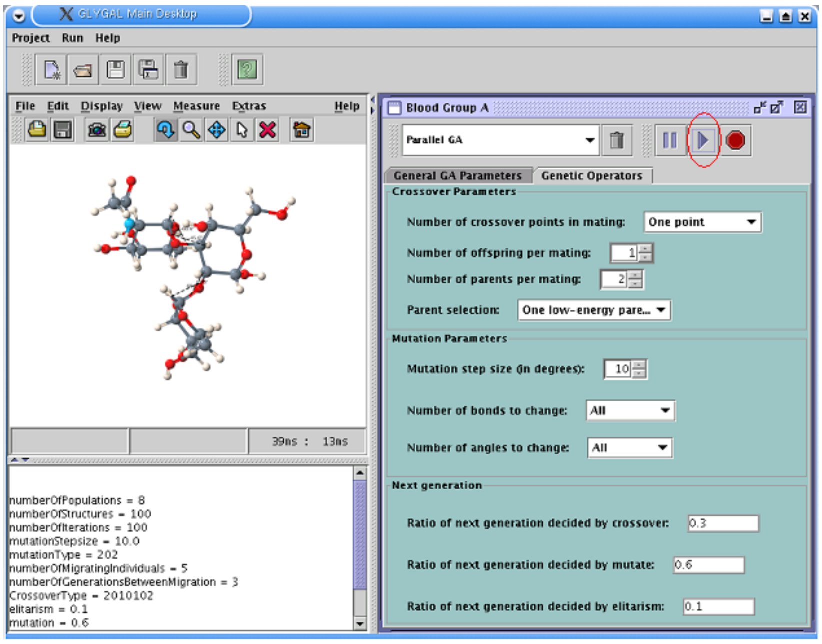
● Converge when no improvment has occured after 100 generations.

Figure C.7: GLYGAL main desktop after choosing the optimization method.

C.3.4 Step four: Running a search

To run search simply press the “Run” button marked in the figure bellow. To kill/pause the running program press the “Kill”/“Pause” buttons respectively.

Output information about the parameters chosen and basic information about the structures (e.g. angle values and energy value) will be written in the textbox in the lower left side of the main desktop as shown in Figure C.8. More output files with information about the structures evaluated by the program, the parameters, the results etc. will be generated by GLYGAL (see Section C.7 “GLYGAL - Output”).



Crossover Parameters

Number of crossover points in mating: One point

Number of offspring per mating: 1

Number of parents per mating: 2

Parent selection: One low-energy pare...

Mutation Parameters

Mutation step size (in degrees): 10

Number of bonds to change: All

Number of angles to change: All

Next generation

Ratio of next generation decided by crossover: 0.3

Ratio of next generation decided by mutate: 0.6

Ratio of next generation decided by elitarism: 0.1

```
numberOfPopulations = 8
numberOfStructures = 100
numberOfIterations = 100
mutationStepsize = 10.0
mutationType = 202
numberOfMigratingIndividuals = 5
numberOfGenerationsBetweenMigration = 3
CrossoverType = 2010102
elitarism = 0.1
mutation = 0.6
```

Figure C.8: GLYGAL main desktop. The red circle mark the "Run" button. On the left bottom corner - output of GLYGAL.

While the program is running the structure shown in the molecule viewer will always be updated with the best structure found (see also "GLYGAL - Visualizing the Molecule"). When the search is completed a message box will appear asking the user if to perform the search from the point the last search ended or to finish. To continue the search press "Continue", to finish press "Finish" in the message box.

C.4 GLYGAL - molecule files

GLYGAL can read in three different molecule file types: .pdb, .mm3 and .xyz.

C.5 GLYGAL - search methods

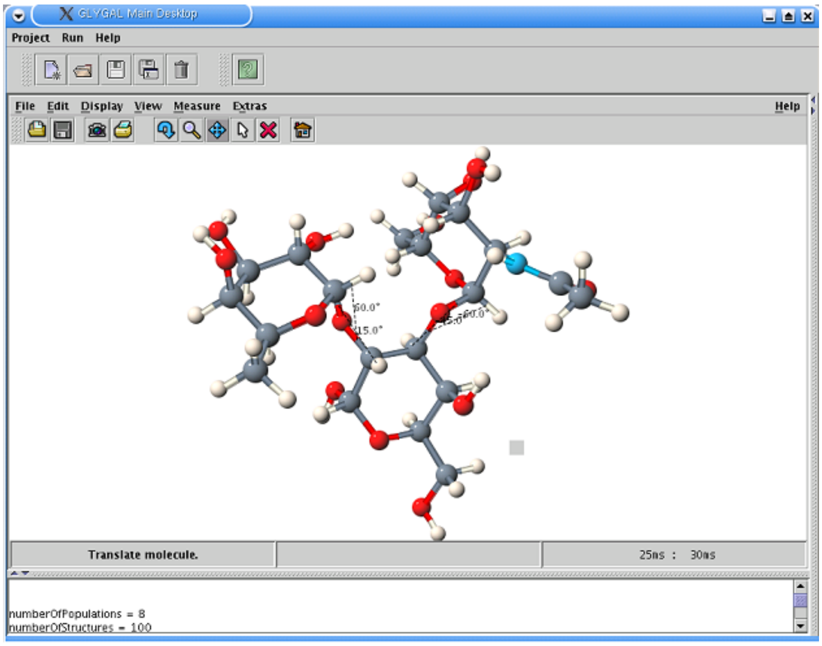
Four search methods are available in GLYGAL. Three of methods standard GA, parallel GA, evolutionary programming are genetic algorithm methods. The fourth method, a simple Monte Carlo search is also implemented mainly to allow comparison with the other methods.

C.5.1 Setting the parameters

When the optimization method is chosen by the user a set of default parameters suggested by the creators of the program are suggested. Changing the parameters is done in a simple way using the parameters setting panels on the right side of the main desktop.

C.6 Visualizing the molecule

To be able to visualize the best molecule found, during and after the search, we incorporated the three dimensional molecule viewer Jmol in GLYGAL's main desktop. Using the option within the Jmol program we can label and view the torsion angles we rotate. To learn more about Jmol c see Jmol help facility. In the Figure C.9 we see a trisaccharide where the four torsion angles values in the two glycosidic linkages are shown.



Translate molecule.

25ms : 30ms

```
numberOfPopulations = 8
numberOfStructures = 100
```

Figure C.9: Jmol 3D viewer.

C.7 Program output

The GLYGAL program has a simple printout on the bottom-left side of the main desktop. This printout presents the parameters chosen by the user and the torsion angle vectors with the corresponding molecular energy.

Except for the simple printout on the main desktop GLYGAL generates a set of output files. When a project is created, say with the name “testproject”, and a search was performed, a “testproject” directory will be created (in case such project already exists a directory with the name “testproject[1]” will be created and so on). Under this directory we find parameter files, the results on the search in the shape of subdirectories for each generation and .mm3 files of all structures calculated. In the project folder a search.txt is created containing all the desktop printouts and including the parameters used for the search.

The best non-redundant structures are stored in the directory “interesting”.

Appendix D

Maintenance manual

This appendix contains some basic information about the architecture of the program. More information can be found in the Javadoc documentation included in the code.

D.1 Development environment

The code and the user interface have been written using NetBeans 3.5.1, a free integrated environment for Java Developers. More information are available on the NetBeans website at http://www.netbeans.org/.

D.2 Files and directory structure

As SugaRun works only on the cluster in medkem, GLYGAL is installed only in the computers belonging to this net. The positions of all the directories needed for the execution of the program are included in the file ∼/Glygal/globalParameterFiles.par. A static reference to this file is included in the class GUI.GlygalMainDesktop. This reference can be changed and the jar file recompiled if the location of this file needs to be changed.

This files contains the location of the server directory, which must also contain a file containing periodic table data and the two files mm3.org and para.org, which are needed for MM3 calculations. It is also possible to specify the directories where the parameter and molecule files are located and the directory where the results will be saved.

D.3 Packages

The program is divided into two main packages: GLYGAL, which contains the main system, and GUI, which contains the graphical user interface.

D.3.1 GLYGAL

The package GLYGAL contains the main class GLYGAL, which handles the command line input of the program. All the other classes are grouped into sub-packages:

- GA: contains all the classes needed for the GA algorithm and the Monte Carlo search. Subclasses of Controller control the execution of a search job. Subclasses of Gene represent the components of the GlycoStructure vector which is manipulated by the operations included in the class GeneticOperators.
- energy: contains the interface with MM3 and some “toy” scoring functions for energy evaluation.
- filter: contains tools for filtering the most interesting conformations. The Picker class allows to choose elements in an array according to a scoring function specified by a subclass of PickerScoringFunction.
- param: reads, writes, modifies and groups conveniently the parameters needed in the program.
- util: contains the torsion angle identifier, an utility to convert MM3 files into xyz format and an utility to create energy maps for test purposes.

D.3.2 GUI

The package GUI contains all the classes composing the user interface. Its function is to generate inputs for the GLYGAL package and to show graphically the results. GLYGALMainDesktop is the main class of the interface and is executable.

D.3.3 External packages included in GLYGAL

- sugarun: developed by Jimmy Rosen, handles the molecule manipulations and the interface with MM3 and the cluster.

- jmol: the open-source molecule viewer and editor JMol 8.0. Jmol is written in Java and provides means of viewing 3D molecular models produced by various software packages (ACES II, ADF, GAMESS, PC GAMESS, Gaussian 9x, XYZ, PDB, and CML). More information can be found in the Jmol Web site at http://jmol.sourceforge.net/.

D.4 Interaction between the packages

The package GLYGAL is independent from the GUI, which is used only to initialize the parameters and visualize the results in a convenient way. The package Jmol is just used by the GUI package. Similarly, the package SugaRun is required only by the GLYGAL.energy package and by the utility that converts MM3 files into XYZ files. The MM3/cluster interface is implemented by the class MM3Evaluator and the related classes (ServerWrapper, JobThread and JobGroup). The substitution of MM3Evaluator with a different energy evaluator could make GLYGAL portable.

The interactions between the packages are shown in Figure D.1, where each arrow mean a dependency between two packages:

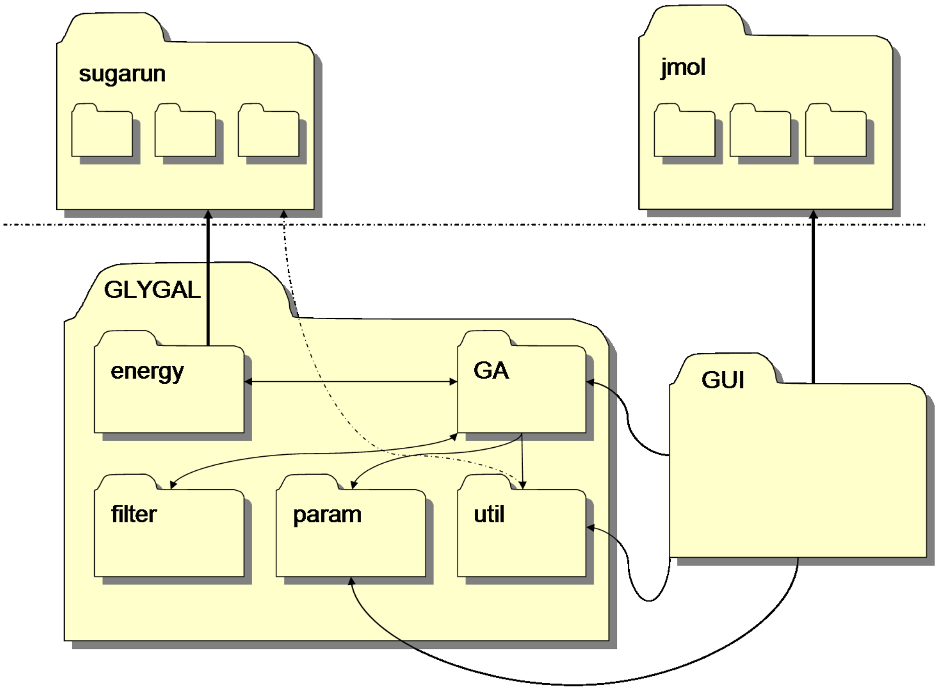


Figure D.1: Schematic representation of the interactions between the packages.

GLYGAL sub-packages and the GUI package play well defined roles in the dataflow of the program. The GUI initially collect informations of what the users wants the the program to do. This information is passed to GLYGAL and processed in order to initialize the search controller in the GA package. The search controller now interacts with the filter and the energy packages. Eventually it returns the results to the GUI. The dataflow is shown in Figure D.2:

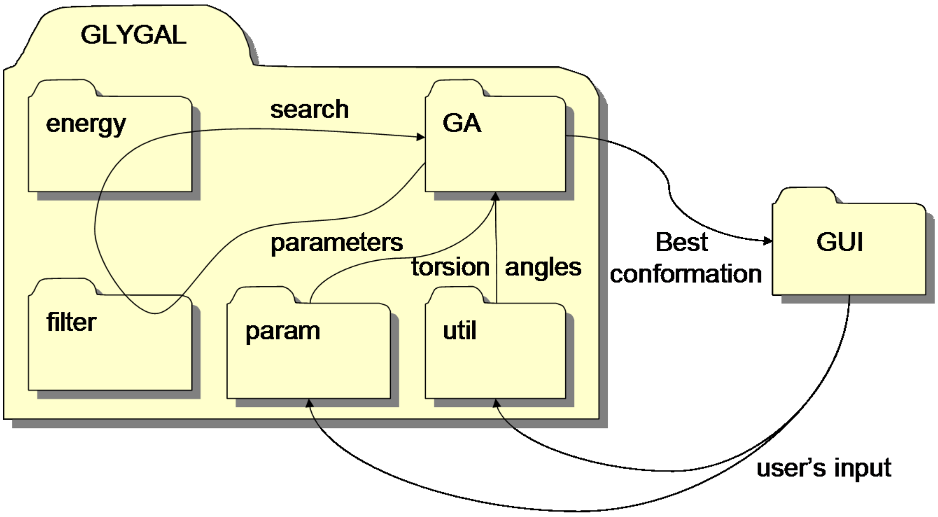


Figure D.2: Schematic representation of the program dataflow.