Ludwig Krippahl

Integrating Protein Structural Information

Dissertação apresentada para obtenção de Grau de Doutor em Bioquímica, Bioquímica Estrutural, pela Universidade Nova de Lisboa, Faculdade de Ciências e Tecnologia.

LISBOA
2003

To Lena, Bruno, and Vasco

Acknowledgements

Anytime you see a turtle up on top of a fence post, you know he had some help.

Alex Haley, paraphrased.

My work on molecular modelling began eight years ago, as a naïve undergraduate student of a bioinorganics course, when I proposed to write a protein docking program instead of the course's regular assignment, which was a simple experimental protocol. Fortunately, the course lecturer was both generous and understanding, and let me off with a much simpler task. Little did I know that eight years later I would still be working with Professor José Moura on the assignment I had initially proposed. What I achieved these seven years I owe to the chance he gave me and to his constant support, orientation, and friendship.

It was also at that time that I began working with Doctor Nuno Palma, and most of my work on BiGGER and Chemera was under his supervision. We worked extraordinarily well together, and our complementary views on most problems led me much farther than I could ever have gone otherwise. Working with Doctor Palma was a wonderful experience, and I could not have developed the docking algorithms without his supervision, his knowledge of molecular modelling, and his skill at finding the right tests and test cases to show us the problems along the way.

Professor Pedro Barahona was the lecturer on the constraint programming course in the artificial intelligence master programme. Both the subject and the way it was presented aroused my interest in constraint programming, and led me to the application of these techniques to protein structure and interaction. The PSICO algorithm was developed under the guidance, and with the teachings, of Professor Barahona, and much of the credit should go to his orientation, experience, and interest in the problem of determining protein structures.

A lot of work goes into implementing an algorithm. In this book I describe the algorithm itself, but gloss over the long labour of transforming an idea into a working application. In this I have to thank Marco Correia for his precious assistance and keen eye for my inevitable programming errors. His work on profiling and testing PSICO was invaluable in implementing the algorithm and, especially, in producing the dynamic link library that will allow the integration of this solver in other applications.

I am thankful to my friends, family, and teachers for their contribution, direct or indirect, to this work. I thank Professor Victor Teodoro for showing me the beauty of numerical methods; Professor John Wampler for a memorable orientation, both for the experience of staying in his lab and for the ideas he planted and which grew into much of BiGGER and Chemera; Doctor Graham Pettigrew for a our long and fruitful collaboration, and for many delightful conversations; Doctor Sofia Pauleta and Patrícia Raleiras for their patience in bearing with the constant changes of the docking algorithms; Doctor Xavier Morelli, for his work with docking and constraints, a major factor in the development of the constraint processing systems in BiGGER; Doctor Brian Goodfellow and Doctor Anjos Macedo for their precious help with the NMR techniques; Professor Isabel Moura for her support; Doctors Francoise Guerlesquin, Antonio Rosatto, Tom Dietterich, Xin Wang, Michael Trosset, Jorge Cruz, Francisco Azevedo and Paula Amaral for fruitful discussions and their help.

I am most grateful to my family for their support, especially all those times my work on this project kept me from the duties (and pleasures) of fatherhood.

Sumário

O tema principal deste trabalho é a aplicação de técnicas de programação por restrições e outras técnicas de inteligência artificial à modelação de estrutura e interacção de proteínas, com o objectivo de melhor combinar dados experimentais com métodos de previsão estrutural.

A primeira parte desta dissertação introduz os temas principais de estrutura de proteínas e programação por restrições, resume as técnicas mais recentes de modelação de estruturas e complexos de proteínas, descreve o contexto em que se inserem as técnicas descritas nas partes subsequentes, e delineia o ponto fulcral da tese: a integração de dados experimentais na modelação.

O primeiro capitulo, Protein Structure, introduz o leitor às noções básicas de estrutura de amino ácidos, cadeias proteicas, e enrolamento e interacção de proteínas. Estes são conceitos importantes para compreender o trabalho descrito nas partes dois e três..

O segundo capitulo, Protein Modelling, dá uma visão breve das técnicas experimentais e de previsão teórica usadas para criar modelos de estruturas proteicas. Este capítulo dá o contexto onde se insere o trabalho descrito nas partes dois e três, mas não é essencial para a compreensão dos algoritmos apresentados.

O terceiro capítulo, Constraint Programming, delineia os conceitos principais desta técnica de programação. A compreensão de métodos de modelação de variáveis, noção de consistência e programação, e de métodos de pesquisa ajudará o leitor interessado nos detalhes dos algoritmos descritos na segunda parte desta dissertação.

O quarto capítulo, Integrating Structural Information, resume a tese aqui proposta, os objectivos deste trabalho, e dá uma ideia de como os algoritmos desenvolvidos podem contribuir para a modelação de estruturas de proteínas. O objective principal é obter um sistema flexível e em evolução continua para a integração de dados experimentais e previsões teóricas.

A segunda parte descreve os algoritmos desenvolvidos, que são a principal contribuição original deste trabalho. Esta parte é especialmente dedicada a leitores interessados em confirmar os resultados, em melhorar os métodos propostos, ou em integrar estes algoritmos em outros programas. Os aspectos bioquímicos são descritos apenas brevemente e só quando estritamente necessário, visto ser esta parte dedicada principalmente aos algoritmos e ao código.

O capitulo cinco, The PSICO Algorithm, descreve as componentes principais deste agoritmo para previsão e determinação de estruturas de proteínas. Estas incluem a modelação de domínios e variáveis, restrições binárias de distância, restrições sobre grupos rígidos e ângulos de torção, e detecção de sobreposições de átomos. Este capitulo descreve também como os diferentes métodos de propagação são integrados e as heurísticas usadas para guiar a pesquisa de soluções.

O sexto capitulo, The BiGGER Algorithm, descreve o algoritmo de modelação de complexos de proteínas. Detalha o filtro geométrico e a pesquisa geométrica de modelos candidatos, e o algoritmo de avaliação que estima a viabilidade destes modelos. O capitulo seis descreve também a integração de restrições experimentais na fase de pesquisa e eliminação, usando técnicas de programação por restrições..

O capítulo sete, Algorithms in Chemera, descreve um conjunto de algoritmos auxiliares para visualizar estruturas e propriedades de proteínas. Alguns dos algoritmos apresentados neste capítulo tais como os de agrupamento ou avaliação de simetria de complexos são um complemento ao algoritmo BiGGER, permitindo um processamento adicional dos modelos gerados pelo algoritmo de previsão de complexos.

A terceira parte desta dissertação apresenta os resultados experimentais usados para testar e parameterizar os algoritmos, bem como exemplos de aplicações práticas a casos reais, e é parte da contribuição original deste trabalho.

O capítulo oito foca os testes do algoritmo PSICO, usando principalmente dados simulados. Este capítulo delineia também alguns problemas que só poderão ser adequadamente resolvidos quando o algoritmo começar a ser aplicado a casos reais.

O capítulo nove descreve a parameterização da fase de pesquisa e triagem geométrica do algoritmo BiGGER, bem como o trabalho mais recente no aperfeiçoamento das funções de avaliação dos modelos gerados.

O capítulo dez é uma selecção de aplicações práticas de ambos os algoritmos BiGGER e PSICO. A maioria destas aplicações contou com a colaboração de vários grupos de investigação que forneceram os dados experimentais. Todos os exemplos referem-se ao algoritmo BiGGER e a previsão de complexos proteicos, por ser este o algoritmo mais maduro, já em fase avançada de utilização. Duas excepções são a Dynamic Link Library do algoritmo PSICO, concebida para permitir a integração fácil deste algoritmo em qualquer programa em ambiente Windows™, e a aplicação

deste algoritmo a problemas gerais de escala multi-dimensional (Multidimensional Scaling), que usam parte dos mecanismos de propagação e pesquisa do algoritmo PSICO para produzir valores iniciais para algoritmos de optimização por pesquisa local.

O último capítulo da dissertação dá uma visão algo pessoal dos objectivos atingidos e de direcções futuras deste trabalho.

Abstract

The central theme of this work is the application of constraint programming and other artificial intelligence techniques to protein structure problems, with the goal of better combining experimental data with structure prediction methods.

Part one of the dissertation introduces the main subjects of protein structure and constraint programming, summarises the state of the art in the modelling of protein structures and complexes, sets the context for the techniques described later on, and outlines the main points of the thesis: the integration of experimental data in modelling.

The first chapter, Protein Structure, introduces the reader to the basic notions of amino acid structure, protein chains, and protein folding and interaction. These are important concepts to understand the work described in parts two and three.

Chapter two, Protein Modelling, gives a brief overview of experimental and theoretical techniques to model protein structures. The information in this chapter provides the context of the investigations described in parts two and three, but is not essential to understanding the methods developed.

Chapter three, Constraint Programming, outlines the main concepts of this programming technique. Understanding variable modelling, the notions of consistency and propagation, and search methods should greatly help the reader interested in the details of the algorithms, as described in part two of this book.

The fourth chapter, Integrating Structural Information, is a summary of the thesis proposed here. This chapter is an overview of the objectives of this work, and gives an idea of how the algorithms developed here could help in modelling protein structures. The main goal is to provide a flexible and continuously evolving framework for the integration of structural information from a diversity of experimental techniques and theoretical predictions.

Part two describes the algorithms developed, which make up the main original contribution of this work. This part is aimed especially at developers interested in the details of the algorithms, in replicating the results, in improving the method or in integrating them in other applications. Biochemical aspects are dealt with briefly and as necessary, and the emphasis is on the algorithms and the code.

Chapter five, The PSICO Algorithm, describes the main components of this algorithm for predicting and determining protein structure. These include the modelling of the variable domains, binary distance constraints, n-ary group constraints, torsion angle constraints and global atom overlap inconsistency detection. The chapter also describes how the different propagation methods are integrated, and the heuristics used to guide the search for a solution.

Chapter six, The BiGGER Algorithm, describes the algorithm for modelling protein interactions. It details the geometric search and filtering of likely candidates, and the scoring algorithms to estimate the likelihood of each retained model being an accurate representation of the real complex. Chapter six also describes the integrating experimental data in the search stage of the algorithm, using constraint programming techniques.

Chapter seven, Algorithms in Chemera, describes a set of auxiliary algorithms for the visualisation of protein structures and properties. Some of these algorithms, such as clustering, symmetry score evaluations and scoring of docking constraints are a complement to the BiGGER algorithm, serving to further process the models generated by the docking algorithm.

The third part of this dissertation presents the experiments and results that validate the methods, and describes their application. It is part of the original contribution of this work, describing the experiments to test and parameterize the algorithms, and the practical applications of these algorithms.

Chapter eight focuses on the testing of PSICO, using mostly simulated data. This chapter also outlines several issues with this algorithm that can only be properly resolved using experimental data, something not yet done at this stage.

Chapter nine addresses the parameterization of the geometric search and filtering stages of the BiGGER algorithm, and the more recent and ongoing work on the scoring functions that evaluate the candidate models.

Chapter ten describes practical applications of BiGGER and PSICO. Most of the work described in this chapter was done in collaboration with several research groups, who provided the experimental data. Also, the majority are applications of BiGGER, the more mature algorithm. Two exceptions are the PSICO Dynamic Link Library, an application extension that can provide the PSICO algorithm to any program in the Windows™ environment, and the application to

Multidimensional Scaling, which uses part of the PSICO propagation and search mechanisms to provide initial values for dissimilarity searches in this field.

The final chapter, chapter eleven, is a more personal view on the achievements of this work and on possible directions for future research.

Contents

INTRODUCTION **1**

1 Protein Structure 3
- 1.1 Amino Acids and Primary Structure 3
- 1.2 Secondary Structure 5
- 1.3 Folding 5
- 1.4 Protein Interactions 6

2 Protein Modelling 8
- 2.1 Historical Overview 8
- 2.2 Experimental Techniques 9
- 2.3 Structure Determination from NMR Data 11
- 2.4 Structure Prediction 11
- 2.5 Protein Docking 12

3 Constraint Programming 14
- 3.1 Variables and Constraints 15
- 3.2 Consistency, Support, and Propagation 15
- 3.3 Searching for a Solution 16
- 3.4 Heuristics 17
- 3.5 Applications to Bioinformatics 18

4 Integrating Structural Information 19
- 4.1 Processing NMR Data and Predicting Protein Structure. 21
- 4.2 Modelling Protein Interactions with Prediction and Experimental Data 21

ALGORITHMS **25**

5 The PSICO Algorithm 27
- 5.1 Modelling the Variables 28
- 5.2 Distance Constraints 31
- 5.3 Rigid Group Constraints 34
- 5.4 Torsion Angle Constraints 43
- 5.5 Atomic Overlap Global Constraint 44
- 5.6 Enforcing Consistency 46
- 5.7 Enumeration, Heuristics, and Backtracking 48
- 5.8 Local Search Optimisation 50
- 5.9 Performance 51

6 The BiGGER Algorithm 57
- 6.1 Sampling the Search Space 58
- 6.2 The Geometric Filter 59
- 6.3 Soft Docking 62
- 6.4 The Side Chain Filter 63
- 6.5 Constrained Docking 64
- 6.6 Optimising the Geometric Filter Algorithm 67
- 6.7 Scoring 69

7 Algorithms in Chemera 71
- 7.1 Structure Comparison 71
- 7.2 Clustering 72
- 7.3 Complex Symmetry Score 74
- 7.4 Evaluating Docking Constraints 75
- 7.5 Electric Properties 75

TOOLS, APPLICATIONS, AND RESULTS **81**

8 PSICO: Protein Structure 83

8.1 Binary Propagation 83
8.2 Group Propagation 85
8.3 Local Search Refinement 89
8.4 Unresolved Issues 90
9 BiGGER: Protein Docking 92
9.1 Chemera and BiGGER Version 2.0 92
9.2 The SPIN-PP Dataset 98
9.3 Rotation Search and Geometric Filtering 101
9.4 The new chain contact filter 106
9.5 Soft-Docking 110
9.6 Unresolved Issues 111
10 Applications 113
10.1 Software Tools 113
10.2 Electron Transfer between Aldehyde Oxydoreductase and Flavodoxin 114
10.3 Electron Transfer Complexes between Cytochrome c550 and Cytochrome c Peroxidase 118
10.4 Electron Transfer Complex between Cytochrome c553 and Ferredoxin 120
10.5 Electron Transfer Complex between Ferredoxin and Ferredoxin NADP+ Reductase 121
10.6 Electron Transfer Complex between Cytochrome b5 and Cytochrome C 123
10.7 The CAPRI Experiments 123
10.8 Combining PSICO with Multidimensional Scaling 129
11 Concluding Remarks 135
References 137
APPENDIXES **145**
Appendix I: PSICO Dynamic Link Library 147
AI. 1. Initialisation, Finalisation, and Problem Selection functions 147
AI. 2. Atom Handling Functions 148
AI. 3. Enumeration and Backtracking Functions 149
AI. 4. Propagation and Domain Handling Functions 150
AI. 5. Group Handling Functions 150
AI. 6. Optimisation Functions 151
AI. 7. Debugging Functions 151
AI. 8. Other Functions 151
Appendix II: Tables 153
Glossary 158
Index 161

Table of Figures

Figure 1-1 A short segment of four amino acids. The backbone is outlined in gray, and the atoms are represented in different colours (nitrogen in blue, carbon in white, oxygen in grey, hydrogen atoms are not shown). The backbone atoms are outlined in grey, with each N-C-C sequence belonging to one amino acid (from top to bottom). Each amino acid has a unique side chain. 4

Figure 1-2 Example of two domains in the desufloferrodoxin monomer (Coelho and others 1997), in dark blue and in orange. Different domains are not only distinct structural motifs, but can also have different and independent functions. 6

Figure 1-3 The structure of the photosynthetic reaction center from Rhodopseudomonas viridis (Deisenhofer and otherrs 1984). Different protein monomers are shown in different colors. The whole protein consists of four different protein sub-units plus a large number of non-protein co-factors. 7

Figure 5-1 Excluding atom A from region X requires a simple alteration in the domain of A if the coordinates of A are modelled together, but the separate x and y domains, represented by the bars, are not changed by removing the region X from domain A. 29

Figure 5-2 The domain of an atom (top left) is represented using a simple cuboid block (top right), the Good region, plus A set of zero or more additional blocks represents the no-Good region, from which the atom is excluded (bottom). 30

Figure 5-3 Modelling distance constraints as cubic regions. On the left a sphere defined by the Euclidean distance to a point- On the centre, the cube modelling an In Constraint, with an edge of twice the distance value. On the right, the cube modelling an Out Constraint, with an edge of twice the distance value divided by the square root of three. 32

Figure 5-4 Propagation of an In Constraint. The left panel shows the Good Regions of the two atom domains. The centre panel shows the neighbourhood region of atom B, and the right panel shows the new domain of atom A, which was restricted to the intersection with the neighbourhood of the atom B. 33

Figure 5-5 Propagating the Out Constraint. The left panel shows the Good Regions of two overlapping domains. The central panel shows the exclusion region of atom B. The right panel shows the intersection of the exclusion region with the good region of atom A, which is added to the No-Good Regions of atom A. 34

Figure 5-6 The three atoms A, B, and C, form a rigid group (dark orange circles), and each atom is restricted to a rectangular domain. The left panel shows the limits for a horizontal translation, and the right panel shows the limits for a vertical translation (light orange circles with dashed line). The dashed boxes on the right panel indicate the accessible regions for each atom with the group in this orientation. 36

Figure 5-7 The position of an atom relative to the center of the group as a function of the rotation angle ψ can be expressed as a sine function with amplitude A and phase α'. The position of the center relative to the atom is a similar curve, but with the phase shifted by 180º (α), giving the sine line shown on the right. 37

Figure 5-8 Each atom constrains the translation of the group as a function of the domain of the atom and rotation of the group. This figure shows such constraints for two atoms (red and blue) along one coordinate axis for one rotation. The grey areas indicate the angles for which

the two atom positions are compatible and the corresponding constraints on the translation of the whole group in this orientation. 38

Figure 5-9 The thick lines show the lines obtained by adding δ (top line) and subtracting δ (bottom line) to the line defined by the centre of the coordinates interval. The top panels show three orientations of the rectangle defined by the coordinate intervals. A, B, C, and D are the extreme points of the rectangle, and thin lines in the lower diagram show their trajectories as a function of the φ rotation parameter. 42

Figure 5-10 Two rigid groups, A and B, joined by a rotatable bond (1). Note that the two atoms on both ends of bond 1 belong to both groups, since rotation around this axis does not change the position of these atoms relative to the atoms in A or B. There are other bonds and other groups in the figure, but not all are displayed, for clarity. Atoms are coloured by element: C white, N blue, O red. Hydrogen atoms are not displayed. 44

Figure 5-11 Calculating the minimum volume occupied by B (domain shown as a dashed box) in the domain of A. Atom B is considered to be as far as possible from A (white circle), and occupies the region shown in orange. The red box shows the minimum volume B occupies in A. 45

Figure 5-12 Evolution of the enumeration cycle. From left to right, after one iteration, 15, 30 and 300. The last panel shows only the backbone for clarity. 49

Figure 5-13 The torsion angle model. The panel on the left shows the data structure as a tree, with atom groups for nodes and dihedral angles for arcs. The panel on the left shows the respective protein structure. 50

Figure 5-14 Piecewise linear upper and lower bounds (blue and orange, respectively) for a sine line. See text for how to calculate the points. 54

Figure 6-1 Using Boolean operations to determine the surface and core regions. Panel A shows the initial grid. A copy of this grid is shifted one cell to the right (B) and combined with the original using an XOR operation (C). Panel D shows the total (using OR operation) of all XOR combinations. Panel E shows the trimmed surface grid, and F the core grid. Filled squares represent grid cells with a value of one. 61

Figure 6-2 The surface and core grids (blue and orange, respectively) for the HIV protease structure (Wlodawer and others 1989). The three panels show sections of the grid in three orthogonal planes. 61

Figure 6-3 Converting a binary grid into a segment array. The binary grid represented by blue squares blue and dashed squares (one and zero, respectively) is converted into an array of segment lists. The fourth segment list contains two segments, (1,2) and (4,6), to represent the two disjoint segments on the fourth line of the grid. 62

Figure 6-4 Difference in the grids when using hard (left panel) or soft docking (right panel). Grey cubes represent the core grids, blue cubes the surface grids. Note that the surface grids are identical, but the core grid in the soft docking option is empty in all cells spanned by the side chain of the aspartate shown in the centre of each panel. 63

Figure 6-5 Generating the displacement domain in one dimension. The left panel shows the generation of the neighbourhood of radius R of group B. The panel on the right shows the allowed displacements for each atom, and the final displacement domain for an N of 2. See constraint 6.1 and text for more details. 65

Figure 6-6 Computing the domains for Constraint 6.1 from the regions where each atom respects the distance constraint. Step one shows the allowed regions for two atoms. Step two marks

the entry and exit points on the displacement array. Step three runs through the displacement array from left to right, adding the value on each cell to the value on the accumulator. The region where both atoms respect the distance constraint is shown in green. 67

Figure 6-7 Calculating the minimum overlap, for the surface contact score. The probe grids (core in grey, surface in light blue) and the target grids (core in grey, surface in dark blue). The vertical alignment represents a given displacement value for the vertical axis. The surface cell totals for the surface grids are in the two columns of numbers adjacent to the grids. The rightmost column shows the minimum value for each row. The total of 11 means that no model obtained by searching the horizontal displacement can have an overlap score greater than 11 for this vertical displacement. 68

Figure 7-1 Desulforedoxin (orange) fitted to the desulforedoxin-like domain in desulfoferrodoxin (blue). Side chains are shown as thin lines, and the backbone as thick lines. The structures used were 1DXG (Archer and others 1995) and 1DFX (Coelho and others 1996, 1997). 72

Figure 7-2 Two clusters of models for the docking simulation of Aldehyde Oxidoreductase (Rebelo and others 2001), in green, complexed with an homology model for flavodoxin. 73

Figure 7-3 High and low symmetry complexes (left and right panels, respectively), from a docking simulation to reconstruct the desulfoferrodoxin dimer (PDB code 1DFX). Each monomer is shown in a different color. 75

Figure 7-4 Three representations of the electrostatic properties of cytochrome C-522. The left panel shows the individual atomic charges (red for negative, blue for positive), the central panel shows the total charges for each amino acid residue, and the right panel shows the dipole moment vector of this protein. 76

Figure 7-5 Shows three representations of the electrostatic field surrounding the cytochrome C552 monomer from Pseudomonas nautical. The left panel shows the field in a plane, with the colour indicating the intensity of the field (red for negative, blue for positive). The centre panel shows isopotential lines at $\pm$0.1Kcal/mol (solid) and $\pm$0.01Kcal/mol (dotted). The right panel shows a three-dimensional representation of the $\pm$0.1 Kcal/mol isopotential surfaces. 79

Figure 7-6 Electron tunnelling within and between two dimers of desulfoferrodoxin (Coelho and others 1996, 1997). The red colour indicates the probability of electron transfer from the iron atom at centre A, suggesting a possible pathway from atom A through B and to C. 80

Figure 8-1 RMSD from the target structure and computation times for the 183 test structures. Computation times are scaled to a 1GHz PC. The red and green circles indicate the structures shown on Figure 8-2. See Table 2, page 150. 84

Figure 8-2 Two structures illustrating the performance of binary distance constraint propagation in PSICO. The left panel shows the target structure (in blue) for a DNA binding protein, PDB code 1FJL (Wilson and others 1995) and the PSICO model in red. The right panel shows the target structure of a 1,3-Beta-Glucanase (Varghese and others 1994), PDB code 1GHS, in blue and the PSICO model in red. 85

Figure 8-3 Propagation times and final domain volumes for arc-consistency in binary distance constraints and group propagation for a randomly generated group. Times are in seconds for a PII at 300Mz running Windows. Volumes are arbitrary units. 86

Figure 8-4 Propagation times and final domain volumes for arc-consistency in binary distance constraints and group propagation for a spiral generated group, simulating an α-Helix structure. Times are in seconds for a PII at 300Mz running Windows. Volumes are arbitrary units. 87

Figure 8-5.Percentage of detected failures as a function of the maximum displacements of the atom coordinates. The figure shows three different group sizes (5, 10, and 20), comparing the binary arc-consistency algorithm implemented in PSICO with the group propagation described here. 89

Figure 8-6 PSICO model for desulforedoxin using experimental NOESY data (Goodfellow and others 1998), shown in red, compared with the crystallographic structure of desulforedoxin, in blue (PDB code 1DXG, Archer and others 1995). 90

Figure 9-1 Ranking of the highest scoring model with less than 4Å deviation from the target complex for the 20 complexes in the test set. 98

Figure 9-2 Shows monomers D (red) and F(green) of Bovine F1-ATPase (PDB code 1BMF). The remainder of the protein is shown in grey. It is more likely that the placement of these two monomers is determined by interactions with other chains in the protein than by a mutual affinity. 100

Figure 9-3 Shows the variation in the fraction of interfaces modelled with success as a function of the added radius parameter, both with and without hydrogen atoms. An interface was successfully modelled if there was at least one model at less than 3Å RMSD from the known structure within the 10 highest scoring models. Only the surface contact score was used in this test, and the chains were placed in the correct relative orientation. 103

Figure 9-4 The left panel shows the percentage of complexes with at least one accurate model in the highest scoring 30 models for a single orientation, as a function of the rotation angle of one partner rotated. The right panel shows the estimated probability of finding an accurate model for each angular step value. Note that there are three different criteria for an accurate model: 3 Å, 4Å, and 5Å. 105

Figure 9-5 Effect of the angular step on the probability of finding and retaining at least one correct model. See equation (9.3). 106

Figure 9-6 Shows the average t values for the differences between the average contacts of correct and incorrect complexes, for the three different contact vector dimensions and as a function of the contact radius modifier. The green circle shows the selected parameters (V28, with +1Å of contact radius modifier). The yellow circle shows the parameters for the previous version of the filter, using V15 and approximately +1Å of contact radius modifier. See Table 8 on page 153. 110

Figure 10-1 The electrostatic field profile of MOP at 0M and 0.5M ionic strength. Dotted lines indicate negative potentials, full lines positive potentials. The isopotential lines are drawn at 1 and 0.1 Kcal/mol for the 0M ionic strength, and at 0.1, 0.01, and 0.001 Kcal/mol for 0.5M ionic strength. The Fe2S2 cluster is shown at the centre of each figure. 116

Figure 10-2A comparison of the three flavodoxin-MOP dockings. For each case the best ten models (according to the electrostatic score) are represented by the FMN residue relative to the MOP monomer. 117

Figure 10-3 Comparison of the relative positions of the prosthetic groups for CO Dehydrogenase, Xanthine Oxidase and the best model for the MOP-Flavodoxin complex. The FMN group of flavodoxin is shown at the bottom, aligned with the FAD groups of CO Dehydrogenase and Xanthine Oxidase. The Fe2S2 clusters are shown at the center of each protein. 118

Figure 10-4 Families of docking models for cytochrome c550 and cytochrome c peroxidase. One peroxidase monomer is shown in grey, and the other monomer is shown in green in the front view. The yellow circles represent the placement of cytochrome c550 in different models. The

two hemes on the peroxidase monomer are indicated by the letters E (electron transfer heme) and P (peroxidatic heme), and their atoms shown in purple. The highlighted family of solutions near the electron transfer heme contains several likely models for an electron transfer complex. Figure adapted from Pettigrew, Pauleta and others 2003. 119

Figure 10-5 The top row shows the positions of cytochrome c553 in the best models (circles) relative to ferredoxin (shown in a backbone representation). The bottom row shows the same models but displaying the backbone of cytochrome c553 and the position of the ferredoxin as coloured circles. Red circles indicate a higher score, and columns A and B represent the best 1000 models according to the global score (column A) and the contact score from NMR data (column B). Column C shows the best models according to each score, and shows in red those models that score highest in both. 120

Figure 10-6 The highest scoring solutions for the Synechocystis FNR-Fd complex, using three different scores. From left to right: NMR titration data (blue), site-directed mutagenesis data (green), and electron transfer (red). 122

Figure 10-7 The two models for the Synechocystis FNR-Fd complex. The ferredoxin is shown in blue, on top, and the FNR in green. 123

Figure 10-8 Summary of the results for all CAPRI participants. The histograms represent all models submitted, evaluated by the fraction of correct residues at the interface. The red bars indicate the best models of the group with the lowest average, the green bars the best models of the group with the highest average, and the blue bars the best BiGGER models. 125

Figure 10-9 Models for target 4 (left panel) and target 5 (right panel). The backbone of the crystallographic structure is shown in thick lines and the backbone of the docking models in thin lines. The largest partner, on the left on each panel, is in the same position for all models. 127

Figure 10-10 Models for target 6 (left panel) and target 7 (right panel). The backbone of the crystallographic structure is shown in thick lines and the backbone of the docking models in thin lines. The largest partner, on the left on each panel, is in the same position for all models. Note that the probe molecule for CAPRI target 7 was truncated before docking, and the docking models show only the fragment used for docking. 128

Figure 10-11 Desulforedoxin Monomer (Archer and others 1995). 260 Atoms, 5951 Constraint pairs. The RMSD from the crystallographic structure was 2.1Å for PSICO and 0.02 Å for MDS. The average constraint violations were, respectively, 0.82 Å and 0.04 Å 132

Figure 10-12 Trypsin Inhibitor 448 Atoms, 11613 Constraint pairs. The RMSD from the crystallographic structure was 2.8Å for PSICO and 0.02 Å for MDS. The average constraint violations were, respectively, 0.73 Å and 0.05 Å 132

Figure 10-13 Mutant P53 Anti-Oncogene (McCoy and others 1997): 514 Atoms, 12938 Constraint pairs. The RMSD from the crystallographic structure was 2.5Å for PSICO and 0.02 Å for MDS. The average constraint violations were, respectively, 0.68 Å and 0.05 Å 132

Figure 10-14 Phosphotransferase (Jia and others 1993): 639 Atoms, 17206 Constraint pairs. The RMSD from the crystallographic structure was 2.8Å for PSICO and 0.02 Å for MDS. The average constraint violations were, respectively, 0.67 Å and 0.01 Å 133

Figure 10-15 Barstar Mutant (Buckle and others 1994) 693 Atoms, 18996 Constraint pairs. The RMSD from the crystallographic structure was 4.1Å for PSICO and 0.02 Å for MDS. The average constraint violations were, respectively, 0.76 Å and 0.01 Å. 133

PART

1

Introduction

In This Part:

Chapter 1

Protein Structure

Chapter 2

Protein Modelling

Chapter 3

Constraint Programming

Chapter 4

Integrating Structural Information

1 Protein Structure

> One unmoving that is swifter than Mind, That the Gods reach not, for It progresses ever in front. That, standing, passes beyond others as they run. (…) That moves and That moves not; That is far and the same is near; That is within all this and That also is outside all this.
>
> Isha Upanishad, 4-5, translated by Sri Aurobindo, in "Arya" August 1914.

There are proteins in all living beings. They mediate all life processes, from growth to thought. They catalyse chemical reactions and are involved in the movements of our muscles and in the structure of our bodies. There are proteins in and around all organisms and everywhere there is life. The writers of the Vedas were unaware of this, having written the Upanishads some five thousand years ago, but even though they meant something else, theirs is an uncanny description of the role and importance of proteins.

Proteins have fascinated chemists for centuries, for solidifying when heated, unlike other substances. In 1777, Pierre Maquer called them albuminous substances, from albumin (egg white, rich in protein) with which they shared this property. Gerardus Mulder coined the term "protein", in 1839, when he proposed that all these substances were apparently multimers of a molecule with the chemical formula of $C_{400}H_{620}N_{100}O_{120}$ (Mulder 1839). He was wrong in his determination of the basic molecule that formed the albuminous substances, but the name was accepted.

This chapter outlines the chemical composition and structural features of proteins. It focuses mainly on aspects important for the work presented in this book and provides only a basic description, so the author recommends additional sources, such as (Darby and Creighton 1993; Branden and Tooze 1999) for an introduction to protein structure.

1.1 Amino Acids and Primary Structure

Proteins consist of amino acid residues bound together in long chains. The process by which proteins are synthesised in organisms is beyond the scope of this book, and the author refers the interested reader to a biochemistry textbook (for example, Stryer 1989). For our purposes, the important aspects are the structure and interaction of amino acid residues.

Figure 1-1 shows a segment of an amino acid chain. The repeating sequence of Nitrogen, Carbon, Carbon atoms is the backbone, or main chain. Branching out from the main chain we can see the amino acid side chains, which are unique to each amino acid.

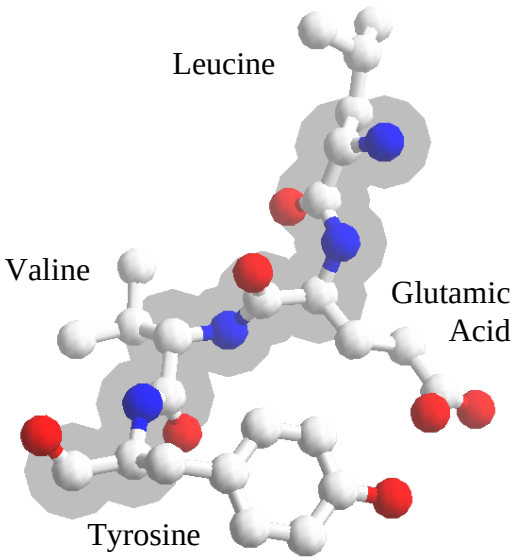


Figure 1-1 A short segment of four amino acid residues. The backbone is outlined in gray, and the atoms are represented in different colours (nitrogen in blue, carbon in white, oxygen in red, hydrogen atoms are not shown). The backbone atoms are outlined in grey, with each N-C-C sequence belonging to one amino acid residue (from top to bottom). Each amino acid residue has a unique side chain.

The amino acid side chains have important properties that affect both the structure and the function of the protein. Their interaction with solvent is one example. The interaction of some side chain groups with water molecules is as favourable as the interaction between water molecules, or even more favourable, and this makes the side chains hydrophilic. For other amino acids, the interaction with the water molecules is not as favourable, suffering an entropy penalisation for forcing the surrounding water molecules to reorganise.

Charge distribution is another important factor, as electrostatic attraction and repulsion is often crucial for complex formation, not only to keep the proteins together, but also to guide them to the correct configuration. It is also important in the folding of the protein, though not as important as hydrophobicity or hydrogen bonding for this. Hydrogen bonding, the sharing between two atoms of a loosely bonded hydrogen atom, is the last major factor for protein folding and complex formation. It stabilizes the main motifs of protein structure, and effectively binds proteins together in protein complexes.

1.2 Secondary Structure

Protein chains form several characteristic local structures due to hydrogen bonds between main chain atoms of different amino acids. The peptide bond between two amino acid residues locks the carboxylic oxygen and C' carbon of one residue and the amidic nitrogen and hydrogen of the other in a plane, and does not allow these atoms to rotate out of the plane. However, the bonds linking the alpha carbon to the C' Carbon and the main chain Nitrogen allow for a range of rotations. The angles of rotation around these bonds, known as phi and psi, respectively, determine the secondary structure and, ultimately, the folding of the protein.

One example is the alpha-helix, in which the main chain forms a spiral, with the phi and psi angles ranging from -60º to -50º. The alpha helix has 3.6 amino acids per turn, and hydrogen bonds between the oxygen of one residue n and the nitrogen of residue n+4 gives it great stability. Less common variants are the pi helix, in which the hydrogen bonds are formed between n and n+5, or the 310 helix, with hydrogen bonds to residue n+3.

Another important example is the beta strand. This local structure consists of usually 5 to 10 residues in a fully extended configuration (phi and psi angles close to 180º). Beta strands can bind together to form beta sheets, also stabilized by hydrogen bonds between the hydrogen of the amino groups and the oxygen of the carboxyl groups.

Secondary structures can combine to form structural motifs, or supersecondary structures. These are very important for structural modelling for several reasons. One reason is predictability; from sequence data alone it is possible to predict secondary structures and motifs with a 70% accuracy (Frishman and Argos 1997; Baldi and Brunak 1998; Wang and others 1999). Another reason is the conservation of secondary structure within protein families, both because of its strong correlation with local amino acid sequences and because evolution seems to favour the conservation of local structures in general (for example, Gille and others 2000). Finally, secondary structure elements are often easier to identify from experimental data, which even allows automated assignment procedures in multidimensional NMR spectroscopy (Bailey-Kellogg and others 2000).

1.3 Folding

Each protein chain folds over itself to form a three-dimensional structure, called the tertiary structure. This structure determines the properties of the protein and the mechanisms by which one

protein interacts with other molecules. Knowing the tertiary structure is crucial for understanding and manipulating the reaction pathways in living organisms.

Often, a protein chain forms regions with separate hydrophobic cores. These regions are called domains, and can operate independently, as if they were several different proteins kept together by peptide linkers. Figure 1-2 shows two domains of desulfoferrodoxin.

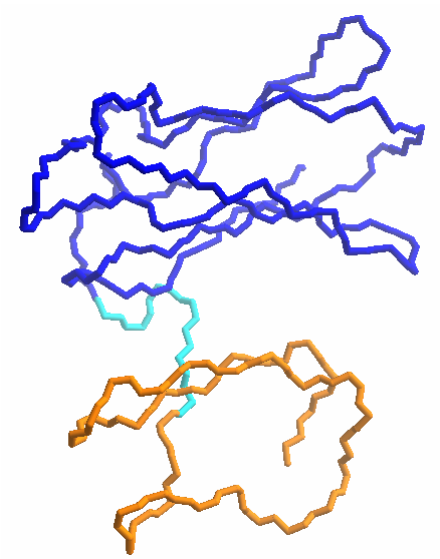


Figure 1-2 Example of two domains in the desufloferrodoxin monomer (Coelho and others 1997), in dark blue and in orange. Different domains are not only distinct structural motifs, but can also have different and independent functions.

The semi-independence of protein domains makes these structural features the natural unit of protein structure and function. When comparing different proteins, or even protein systems, it is often necessary to take into account the modular nature of protein chains.

1.4 Protein Interactions

Most, if not all, proteins interact with other proteins. These interactions range from transient contacts, in which the partners meet briefly to perform some chemical reaction, to life-long partnerships, with each protein chain specifically adapted to be permanently associated to its partners. The structure of such assemblies of protein chains is the quaternary structure.

Protein interactions play a central role in all living organisms, thus the importance of understanding the mechanisms and structure of protein complexes. As an example, Figure 1-3 shows the structure of a bacterial photosynthetic reaction centre (Deisenhofer and otherrs 1984). The 1988 Nobel prize in Chemistry was awarded to Johann Deisenhofer, Robert Huber, and Hartmut Michel for the determination of this structure.

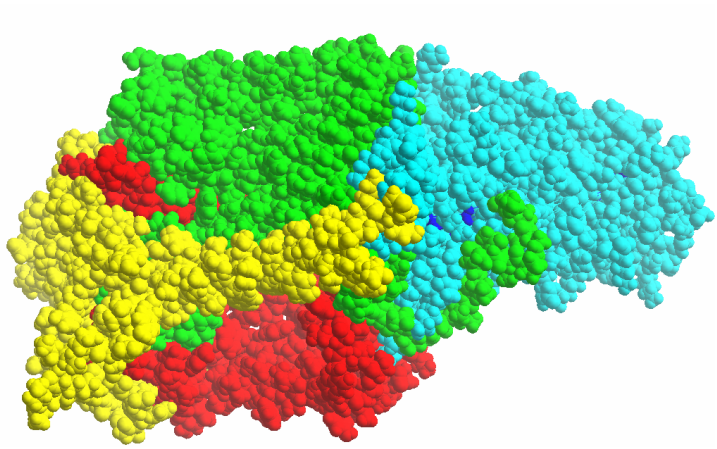


Figure 1-3 The structure of the photosynthetic reaction center from Rhodopseudomonas viridis (Deisenhofer and otherrs 1984). Different protein monomers are shown in different colors. The whole protein consists of four different protein sub-units plus a large number of non-protein co-factors.

2 Protein Modelling

> There's a reason physicists are so successful with what they do, and that is they study the hydrogen atom and the helium ion and then they stop.
>
> Richard Feynman

Quantum mechanics is a powerful tool to understand the dynamics of sub-atomic particles and simple atoms and molecules. However, our capacity to analyse the details of molecular systems decreases as complexity increases, and proteins are perhaps the most complex molecules known. Though smaller than DNA and some other polymers, their versatility and the diverse roles they play in chemical reactions make it very hard to predict and model all their features in detail.

This chapter focuses on the modelling of one aspect of proteins: their structure. By protein structure one usually means an average three-dimensional configuration of the atoms in the protein, sometimes with a cursory indication of the mobility of each part of the protein chain. There are other important aspects, from folding to function, which are the subjects of much research, but fall outside the scope of this work. After an historical overview of protein structure modelling, this chapter addresses the most important experimental techniques for determining protein structures, and some prediction methods currently available. The emphasis is on Nuclear Magnetic Resonance (NMR) techniques and docking, for being the more relevant to the work presented in this book.

2.1 Historical Overview

The first technique that elucidated the structure of a protein was X-Ray crystallography, and it is still the most important technique in the number of structures generated every year. Successful crystallization of proteins dates at least to 1840, when Hunfield crystallized earthworm haemoglobin. In 1913 Sir L. Bragg solved structure of NaCl, and in 1932 Asturby and Street reported the X-Ray diffraction pattern of human hair and other protein fibers (Asturby and Street 1932). Pauling and Corey (1951a; 1951b) publish the structures for the α-helix and β-sheet in 1951, and between 1954 and 1960, Perutz and Kendrew determined the structure of myoglobin (Bragg and Perutz 1954; Kendrew and others 1960).

Meanwhile, in 1945, Felix Bloch and Edward Purcell independently observed the magnetic resonance of hydrogen nuclei, initiating the field of NMR spectroscopy and winning them the 1952

Nobel Prize in physics. The techniques to manipulate relaxation times (Bloemberg and others 1948) established NMR spectroscopy as a tool to explore molecular structures. In 1965, Richard Ernst and Weston Anderson used a radiofrequency pulse that excited all the spins in the sample, and then applied a Fourier transform to the free induction decay results to obtain the frequency spectrum. This technique allowed all frequencies to be scanned simultaneously with a single pulse, greatly increasing the sampling rate for NMR spectroscopy. In the same year, James Cooley and John Tukey published the fast Fourier transform algorithm (Cooley and Tukey 1965), and this combination of techniques gave rise to Fourier-transform NMR and high resolution NMR spectroscopy. The first protein structures determined by NMR spectroscopy would appear two decades later. Richard Ernst won the 1991 Nobel Prize in chemistry for his contribution to high resolution NMR spectroscopy, and in 2002 this award was attributed to Kurt Wüthrich for the development of NMR spectroscopy techniques for the determination of protein structures.

As of July 2002, there were 20922 protein structures deposited on the Protein Data Bank, 18118 determined by X-Ray crystallography and 2804 by NMR spectroscopy, and the number of protein structures is growing exponentially.

2.2 Experimental Techniques

The two main techniques for determining protein structure, at present, are X-Ray crystallography and NMR spectroscopy. X-Ray crystallography is based on the diffraction of X-Rays by protein crystals. The ordered placement of the molecules in the crystal produces a diffraction pattern that contains information on the structure of the individual molecules. Many diffraction patterns are recorded at different orientations of the crystal (depending on the crystal symmetry), from which it is possible to calculate the electron density distribution that is generating the diffraction patterns and then fit the protein chain to this three-dimensional electron density map.

One major problem is that the Fourier transform converting the reflection intensity data into a three-dimensional electron density map requires information on both the magnitude and phase of the reflections, but the diffraction pattern only provides the intensity of the reflections. This is known as the phase problem, and can be solved either by computation techniques (especially in smaller molecules) or by experimental techniques like Multiple Isomorphous Replacement or Multiple Anomalous Dispersion Phasing. Another major problem is often obtaining a crystal with the necessary quality for structure determination.

Nuclear Magnetic Resonance (NMR) spectroscopy is based on a different principle. Some atomic nuclei are magnetic dipoles, the most common example being the hydrogen nucleus (some nuclei are magnetic quadrupoles, but this section will focus on dipoles, without loss of generality). In a magnetic field, the magnitude of the component of the nuclear magnetic dipole vector that is parallel to the magnetic field is quantized, and can only take two values, either parallel or anti-parallel to the magnetic field.

The energy difference between these two orientations results in a net absorption of radiation by these nuclei, because relaxation mechanisms allow the nuclei to switch between the two states without emission, and thus dissipate the energy absorbed when changing from a lower energy state to a higher energy state. With the typical magnets used in NMR, the radiation is in the range of radio frequencies, at several hundred mega Hertz.

Interactions with electrons and other nuclei affect both the magnetic field at each nucleus and the relaxation mechanisms by which the nucleus dissipates the energy absorbed. Thus, the environment around each nucleus influences both the resonance frequency and the intensity of the absorption. Not only do the NMR signals distinguish between nuclei in different environments – for example, hydrogen nuclei in different parts of the protein – but also identify interactions between one nucleus being irradiated and other nuclei whose signals are being affected. With these techniques, NMR spectroscopy can provide data on molecular bonds, orientations, and distances between atoms.

The major problem in NMR spectroscopy is the assignment of the resonance frequencies to the corresponding atoms. This can be a difficult process depending on the protein, and is often done iteratively with structural calculation to help guide and correct the assignments.

There are other techniques, such as electron microscopy or Infra-Red spectroscopy, among others, that can provide structural information. Two examples are the use of X-Ray scattering data to improve homology modelling (Zheng and Doniach 2002), and high-resolution atomic force microscopy to model protein structures (Todd and others 2003). However, X-Ray crystallography and NMR spectroscopy account for the vast majority of structural data on proteins at present.

2.3 Structure Determination from NMR Data

Currently, the standard method to calculate protein structures from NMR data is an energy minimization using a torsion angle model of the protein and including in the energy function penalties for the violation of experimental constraints on dihedral angles and atomic distances.

The torsion angle model of a protein represents the three-dimensional structure as a function of the rotation angles around some of the molecular bonds. This model implicitly takes into account the differences in flexibility in different parts of the chain. See section 5.8 for an illustration and more details on the torsion angle model.

Available software packages such as DYANA (Guntert and others 1997; Hermann and others 2002), FANTOM, CNS (Brünger and others 1998), and X-Plor all use simulated annealing methods, often in conjunction with molecular dynamics or gradient methods such as Newton-Rhapson or conjugated gradient minimization (see section 5.8). X-Plor allows the option of using a distance geometry method, but it seems torsion angle minimization methods are more efficient at the moment (Guntert and others 1997, but see section 10.8 and Trosset 2000 for recent developments in the distance geometry approach).

So far, the use of constraint propagation techniques in NMR seems to have been restricted to the assignment problem. AUTOASSIGN is an example of this application (Zimmerman and others 1994).

2.4 Structure Prediction

Homology modelling is the most widely used and successful technique to predict protein structures. Proteins with high sequence homology, especially in closely related organisms, often have similar structures (Clothia and Lesk, 1986), which makes it possible to predict the structure of one protein from the structures of homologous proteins, if such structures are known.

Other prediction methods involve the use of neural networks or hidden Markov models to learn secondary structure folds from known structures (Baldi and Brunak 1998; Wang and others 1999) or evolutionary computation techniques to determine minimal energy folds (Fogel and Corne 2003). These are mostly successful only on small peptide chains or for secondary structure prediction.

Protein structure prediction involves a complex mixture of techniques, and a detailed analysis of these methods is outside the scope of this introduction. The important point for the work presented here is that prediction techniques can provide candidate structures that can be confronted with experimental data. Section 5.3 details the algorithm for this integration of experimental data with predictions of secondary structure, domains, or even of the whole protein structure.

2.5 Protein Docking

Protein interactions play a crucial role in all bio systems, and knowing the structure of a protein complex is an essential step in understanding the interaction mechanism. Modelling plays an important part in this process because of the difficulties in determining the structure of protein complexes by experimental techniques.

In many cases proteins interact only temporarily; they must come together to perform some task and then go their separate ways once again. This makes it very difficult to co-crystallize the partners to obtain a crystallographic structure of the complex, and the large size of these complexes often precludes structure determination by NMR. This makes theoretical prediction a useful approach in these cases. Algorithms to predict such complexes are known as docking algorithms, and this prediction as protein docking.

There is a considerable variety of docking algorithms being used at present. Examples of some approaches from the CAPRI experiment (Janin 2002) are: ICM, which uses detailed energy calculations and Brownian movement simulations to mimic the approach of the docking partners (Fernandez-Recio and others 2002); BUDDA, using geometric hashing techniques to quickly match surface patches (Schneidman-Duhovny and others 2003); GAPDOCK, using genetic algorithms to optimize contact energy (Gardiner and others 2001); HEX, which uses spherical harmonics to match complementary surfaces (Ritchie and Kemp 2000).

However, the majority are based on the Fast Fourier Transform method first developed by Katchalski-Katzir and others (1992), which calculates cubic matrix correlations to find the best placements of the two docking partners. Examples are ZDOCK (Chen and Weng 2002) and MolFit (Katchalski-Katzir and others 1992).

These are just a sample from the participants in the CAPRI experiment (Janin 2002), but a representative sample, since this is the most important assessment program for docking predictions.

3 Constraint Programming

> When you have eliminated the impossible, whatever remains, however improbable, must be the truth.
>
> Sir Arthur Conan Doyle, "The Blanched Soldier"

The ideas behind Constraint Programming (CP) originated in the decades of 1960 and 1970. Two seminal works in this area were Sketchpad (Sutherland 1963), an application to manipulate geometric figures, and the Waltz labelling algorithm, for labelling surfaces of an image representing a three dimensional scene (Waltz 1975). These applications and their successors introduced the fundamental concepts of propagation and consistency – in essence, the science of eliminating the impossible.

An important shift came in the 1980's with the realisation that logic programming, and declarative programming in general, was an example of CP (Gallaire 1985; Jaffer and Lassez 1987). In all these cases problems are declarations of variables, variable domains, and constraints on these variables; the Constraint Satisfaction Problem (CSP). This led to a common identification of CP with logic programming, and declarative languages are currently the preferred medium for CP. Nevertheless, it is also appropriate to use imperative languages, especially in specific problems where efficiency and integration with other applications are important, as is the case in the work presented on part 2.

This chapter outlines the main concepts in CP and is meant as a brief introduction for the reader unfamiliar with CP. These concepts are important for the discussion of the algorithms in chapters 5 and 6. Some of the material in this chapter is based on a review article by Roman Bartak (Bartak 1999) and on Bartak's Foundations of Constraint Satisfaction site, which, for their availability on line (at least at the time of writing) are good starting points for an initiation on this subject.

This chapter also provides a theoretical context for the algorithms described in part 2, especially in chapter 5. As such, the material in this chapter focuses mostly on those techniques relevant to the implementation of PSICO and is not meant to be a representative overview of constraint programming.

3.1 Variables and Constraints

Variables are the unknowns in the problem, and may be anything from simple True/False variables to real numbers. Constraints are logical relations involving one or more variables. For example "x<5", "At least one of Dumbo or Snoopy is a mammal", "There can be at most one class in each classroom at any one time", "$r_1<|p_1;p_2|<r_2$".

The solution to a CSP is an attribution of values to all variables such that all constraints are satisfied. In this, CP is similar to other methods for solving problems involving variables, such as linear programming or simulated annealing. What distinguishes the CP approach is that, along with variables and value attributions, CP also relies on variable domains. These are the sets of possible attributions for each value, in other words, the attributions that are not known to violate constraints. Typically, CP approach consists mainly in reducing the domains and not so much in manipulating value attributions, as is the norm in other approaches.

3.2 Consistency, Support, and Propagation

Consistency is a fundamental notion in constraint programming. Broadly, an attribution of values to a set of variables is consistent if it respects all constraints over all variables in the set. In this sense, consistency is the goal of CP, for a consistent attribution of values is the solution to the CSP.

More generally, we can widen the meaning of consistency to apply to variable domains instead of only to single values. A domain is consistent with other domains if none of the values in the domain necessarily violates a constraint, which means that, for each value in the domain, there is at least one value in each of the other domains that would make the whole set of attributions consistent.

The downside is that enforcing consistency, in this sense, is as difficult as solving the problem, and thus not useful in achieving that goal. It is more useful to use weaker notions of consistency that are restricted to sub-sets of variables, which can be enforced with less cost, and which can be used to reduce the number of possible value attributions.

The simplest case is node-consistency, in which consistency is enforced by considering each variable (and corresponding domain) in isolation, and considering only those constraints that involve that variable alone. For example, if a constraint specifies that some variable must be an odd number, we can remove from its domain all numbers that are not odd.

Arc-consistency, the most used level of consistency, enforces consistency over all pairs of variables, considering all constraints over two variables. There are several standard algorithms for enforcing arc-consistency (Mackworth 1977; Bessiere 1994; Hentenryk and others 1992; Bessiere and others 1999). This level of consistency seems to be the best compromise between effectiveness and computational cost, and has inspired considerable research efforts.

Path-consistency considers triplets of variables, while k-consistency is the general case considering k variables in simultaneous. Apart from special cases, enforcing higher levels of consistency is too expensive to be of practical use. One exception is the case of global constraints, in which the properties of a constraint can be used to improve propagation. An example, in this work, is the group propagation algorithm presented in sections 5.3 and 5.4.

Section 5.2 describes how arc consistency is enforced on modified distance constraints. Although enforcing arc consistency on non-linear constraints in continuous domains can be computationally demanding, a modification of the distance constraints simplifies the propagation and results in an efficient algorithm.

3.3 Searching for a Solution

Since it is generally not feasible to enforce complete consistency, it is necessary to search for sets of attributions that respect all constraints. This is a combinatorial problem, for potentially all combinations of attributions must be tested before either finding a solution to the CSP or proving that no solution exists.

The simplest method of systematic search is generation and test. A candidate solution is generated by attributing values to all variables, and this solution is then tested to determine if it respects all constraints. Obviously, this approach is inefficient, as it will tend to waste most of the time on attributions that do not respect the constraints.

A better method for a systematic search is to build the set of attributions incrementally. Instead of assigning values to all variables, the variables are instantiated one at a time. At each instantiation it is possible to determine inconsistencies or propagate constraints and thus eliminate values to test. The search can be imagined progressing through a tree, where each node represents a choice of possible instantiations, and where constraint propagation allows the pruning of branches from the search tree.

Whenever an inconsistency is detected, it is necessary to try alternative routes through the search tree. This is the backtracking algorithm, named for the imaginary movement backwards through the search tree to explore another branch. Though always more efficient than the generation and test algorithm, backtracking has some weaknesses. A major problem is that, by simply undoing the attribution that resulted in a detectable inconsistency, backtracking loses the information on the conflict of attributions that caused the inconsistency.

Backjumping is an alternative to backtracking that addresses this weakness by jumping backwards through the search tree up to the attribution responsible for the inconsistency detected. This is a significant improvement over backtracking alone without constraint propagation, but since constraint propagation eliminates inconsistent nodes before they are visited, in this case backjumping may bring no benefits.

Local search methods are a broad and widely used category of incomplete search methods. The fundamental idea in local search is to generate a new candidate by a slight modifications of the current attribution. Tabu search, genetic algorithms and simulated annealing are some examples of this approach.

3.4 Heuristics

An important aspect of searching though a tree of possible attributions is the order in which the nodes are examined. Ideally, if one can ensure that the correct attributions are the first to be tested, an existing solution is found immediately on the first set of attributions. In practice, however, this ideal situation is very rare, and it is necessary to consider other approaches. Given the fallibility of most heuristics in a realistic situation, both the ordering of variables and of the values to attribute are important.

Variable ordering can have a great impact on search efficiency (Gent and others 1996), and a common approach is to follow the fail first principle: "The best search order is the one that minimizes the expected length or depth of each branch" (Haralick and Elliott, 1980). This can be achieved by choosing first the variable with the smallest domain. However, it is important to note that branch depth is only part of what determines the number of nodes explored, and thus the search time. The branching of the search tree is also a factor, and these considerations led to the variable ordering heuristic described in section 5.7 for PSICO. A combination of first fail, round

robin enumeration, and terminating the search when the domains are sufficiently reduced is a compromise between the branching factor and the length of the branches in the search tree.

Value ordering is also very important, but value ordering heuristics are often problem dependent, and it is hard to find a general principle that can guide this choice. The closest thing to such a principle is to choose the value that has the least impact on other choices. The heuristic described in section 5.7, though specific to the problem of determining a protein structure, is based on this concept. By splitting the domains so as to place each atom as far apart from the others as possible, this value ordering heuristic delays conflicts and allows enumeration to proceed farther before inconsistency is detected. Though in apparent contradiction to the first fail approach of the variable ordering heuristic, this ordering of values is crucial to the success of the algorithm, because the size of the search tree prevents the recovery from any but the most trivial conflicts.

3.5 Applications to Bioinformatics

Constraint programming is not a widely used technique in bioinformatics, but some interesting applications show it to be a promising approach in this field. Two recent examples are the determination of mutations to replace an amino acid by selenocysteine (Backofen and others, 2002), choice of side chain rotamers (Swain and Kemp, 2001). Other related techniques are also gaining ground in biochemical applications, such as integer programming for solving the phase problem in X-ray crystallography (Lunin and others, 2002), but the bioinformatics field is still dominated by soft computing approaches like genetic algorithms, neural networks and hidden Markov models (Fogel and Corne 2003, Wang and others 1999, Baldi and Buinak 1998).

Backofen (1998, 2001; Backofen and others 1999, 2002) applied constraint programming to protein folding, but only using the simple lattice models of protein structure, which represents each amino acid as an occupied cell in a cubic grid. Though potentially interesting from a theoretical and computational perspective, this approach did not yet show a practical application.

The work presented here is thus an innovative application of constraint programming techniques to protein structure and interaction problems using real data.

4 Integrating Structural Information

> There are two things in the painter, the eye and the mind; each of them should aid the other.
>
> Paul Cezanne

Molecular biology changed in 1953 with the publication of a structural model of DNA by Watson and Crick (Watson and Crick 1953). Aside from its importance in this field, this model illustrates the combination of theoretical considerations and empirical data that is necessary for modelling macromolecular structures. Watson and Crick were aware of the X-Ray diffraction patterns of DNA crystals that indicated DNA formed a helical structure, and of the near-unitary ratios of adenine to thymine and guanine to cytosine always found in DNA. These data had direct implications for DNA structure. In addition, there were theoretical considerations on atomic radii, bond angles and distances, hydrogen bonding and, especially, the ability of DNA to be copied in living organisms. In short, the relevant information was diverse and broadly divided into two categories: observation data on the structure to model and other information on its purpose, similar molecules, and general chemistry.

Modelling protein structures is a difficult task due to the complexity of protein folding and interaction, and there are two major approaches to this problem: predictive methods that build models by applying general rules, and experimental methods based on comprehensive data on the system to model.

Many predictive methods are based on detailed simulations of the physics of protein folding and interaction. Examples are the pioneering work of the Kollman research group in protein folding simulations (Duan and Kollman 1998), distributed folding applications such as FOLDING@HOME (Stanford 2000; Zagrovic and others 2001), or protein folding and protein-ligand docking methods like ICM (Abgayan and others 1997; MolSoft 2003). Other predictive methods rely on identifying patterns in the data available and generalizing this information using learning algorithms or other forms of statistical analysis. This approach widely used in secondary structure prediction (Frishman and Argos 1997; Baldi and Brunak 1998; Wang and others 1999; Fogel and Corne 2003), but also for ternary structure (Simons and others 1997); in protein docking algorithms like AUTODOCK (Morris and others 1998); and in established methods like homology modelling and threading.

Predictive methods show great potential. As computational power and structural knowledge improves, these methods may eventually predict protein folding and interaction as accurately as they now predict the structure and activity of smaller ligands. However, currently these methods are not sufficiently reliable to solve the problem by themselves. In practice, modelling protein structure and interactions always requires a complement of experimental data to make up for the shortcomings of the theoretical prediction algorithms.

Experimental methods for protein structure determination are well established. X-ray crystallography and Nuclear Magnetic Resonance (NMR) provide most of the data, and there are established algorithms for processing this information. The recent development in this area owes more to technological innovations providing new techniques for obtaining data than to improvements in the algorithms, especially in comparison with predictive methods.

Though protein structure is a major research area, and both predictive and experimental approaches are the focus of much effort, the interface between these two areas is somewhat unexplored. The goal of the research into predictive algorithms seems to be to make predictions as general and as independent from experimental data as possible, often at great computation cost. However, researchers working with specific structural problems seldom wish to model the structures without experimental data, but rather using all available information. Computation costs and the discrepancy between what these algorithms are designed to do and what the potential users would like them to do restrict the application of this promising approach.

The case of algorithms to process experimental data is the opposite. Their aim is to process homogeneous data, typically focusing on one experimental technique, and not to provide the necessary flexibility to take advantage of the diversity of data in most real applications.

The major goal of the work presented here is to bridge this gap between theoretical prediction and processing of experimental data. The two problems addressed in this book illustrate the two approaches, with protein docking simulation being an example of a theoretical prediction (chapter 6), and protein structure determination by NMR a problem of processing experimental data (chapter 5).

4.1 Processing NMR Data and Predicting Protein Structure.

Acquiring and processing structural NMR data is time consuming and demanding. There are methods for semi-automated assignment (Bailey-Kellogg and others 2000; Herrmann and others 2002; Szyperski and others 2003), but these invariably depend on isotopic labelling and multi-dimensional spectra, which greatly increases the cost and complexity of producing the protein. In addition, these methods require multiple spectra with redundant information, also increasing the cost of data acquisition. In general, the problem of attributing resonance frequencies to atoms is a difficult one to solve.

It is especially at this stage that additional information can be brought to bear. The goal is to determine the structure of the protein based on the experimental data, but predictive methods may help by providing candidate structures that can help decipher the NMR data. Furthermore, it may be that the NMR data is only partial – for example, if paramagnetic centres prevent the acquisition of data in one region of the protein – in which case predictive methods may fill in the gaps.

It is possible, in theory, to do this with current techniques. We could generate additional constraints from each candidate structure, and combine them with the data available. However, currently available applications for processing NMR data are not meant to do this, which raises additional problems, such as ensuring the generation of an adequate set of constraints.

PSICO (Processing Structural Information with Constraint programming and Optimisation, see chapter 5) was conceived from the start as a general geometric constraint solver that can process, in simultaneous, structural information from different sources, such as NMR spectroscopy, secondary structure prediction, or homology modelling, and it can be used to check the consistency of constraint sets, and thus evaluate the theoretical assumptions. This efficient combination of diverse sources of information can simplify the assignment and speed up the determination of protein structures.

4.2 Modelling Protein Interactions with Prediction and Experimental Data

Another example is the study of protein interactions. One way of determining how two proteins interact is to crystallize them together and solve the structure, but this is a difficult task. Alternatively, there are several docking programs available that can try to predict the right configuration of the complex (see section 2.5). The major problem with docking predictions is that

even the best systems currently available have a very low success rate. According to the CAPRI evaluators:

"[…] the performance of current procedures taken globally is by no means reliable enough to allow their use as a routine tool. For many real-world problems, as in the case of targets T01–T06, only around 10% of the submitted predictions are of acceptable quality or better. Only when detailed structural information is available on a related complex in which the interaction mode is conserved, as for Target 07, do we see a higher success rate of 30% correct predictions."

The situation is thus that neither experimental data nor theoretical predictions alone can reliably elucidate the structure of protein complexes. The integration of the two approaches is one possible solution to this problem.

Most docking programs allow the user to restrict the model generated by specifying the contact surface, but the data seldom allows us to be so specific in protein-protein docking. More often, we have data on important residues, such as given by site-directed mutagenesis or NMR titration experiments. These data indicate which residues of the protein affect, or are affected by, docking, but do not necessarily reveal a contact surface, for some residues may show effects even if outside the contact region.

HADDOCK (Dominguez and others 2003) is an exception, for it uses NMR or mutagenesis data explicitly, in a similar manner used in the BiGGER contact score (Morelli, Czjzek and others 2000; Morelli, Dolla and others 2000; Morelli, Palma and others 2001; Banci and others 2003; also see section 7.4). The HADDOCK approach is different in that it tries to minimize a set of distance constraints instead of maximizing contacts. This probably makes it easier to take advantage of the minimization algorithms implemented with the force field used by HADDOCK (the CNS force field), but can lead to problems if the data contains perturbations that are not caused by close contacts. By minimizing a set of incorrect distance constraints along with the correct ones, the algorithm is likely to systematically miss the correct structure.

The constraint programming techniques used in BiGGER (section 6.5) are more resistance to noisy data because they can isolate sets of potential contacts that are feasible, instead of trying to minimise the distances for all candidate contacts, including the infeasible ones.

Hopefully, the algorithms and results described in parts two and three of this dissertation will support the main thesis that the best course for protein structure and interaction modelling is

through the integration of experimental data and theoretical prediction, in algorithms conceived from the start to make the eye and the mind work together.

PART

2

Algorithms

In This Part:

Chapter 5
The PSICO Algorithm

Chapter 6
The BiGGER Algorithm

Chapter 7
Algorithms in Chemera

5 The PSICO Algorithm

> Seven houses contain seven cats. Each cat kills seven mice. Each mouse had eaten seven ears of grain. Each ear of grain would have produced seven hekats of wheat. What is the total of all of these?
>
> Rhind papyrus, Egypt, circa 1850 BC

Combinatorial problems have fascinated mathematicians since ancient times for their exponential complexity. A hekat is an ancient Egyptian measure of volume of approximately 5 litres, so the solution is: seven houses, 49 cats, 553 mice, 3871 ears of grain, and nearly seventy tons of wheat. The determination of protein structures from geometric constraints is one such problem, for the number of possible arrangements grows exponentially with the number or atoms. This chapter describes the PSICO algorithm, and how PSICO counters the exponential growth of combinations by using constraint programming techniques to prune arrangements that cannot satisfy the constraints.

The initial motivation for PSICO was the determination of protein structures by Nuclear Overhauser Enhancement Spectroscopy (NOESY), a Nuclear Magnetic Resonance (NMR) technique by which it is possible to obtain a set of distances between atoms in a molecule. The original idea was to solve the specific problem of finding the ternary structure of a protein given a set of distances between some of its atoms (see section 1.3).

However, this problem is not as specific as it first appeared, nor its scope as narrow. The sequence of the protein – its primary structure – gives us considerable information about its structure at a small scale, for the covalent bonds that keep the molecule together specify the relative positions of small groups of atoms throughout the protein. The secondary structure of the protein also provide information on groups of atoms, and can be determined by experiments or predicted more easily than the ternary structure. Finally, as the result of a long process of evolution, proteins share similarities with other proteins, sometimes in very different kinds of organism.

With the conception of new methods to process different sources of structural information, PSICO grew from a simple solver of binary distance constraints (Krippahl 1999; Krippahl and Barahona 1999, 2001) into a flexible algorithm with a wide potential for applications, from testing homology models to solving structural NMR constraints (Krippahl and Barahona 2003).

The material in this chapter is divided into nine sections. The first section (5.1) explains the domain representations that store the possible the positions of the atoms. Section 5.2 explains how binary distance constraints affect these domains, and section 5.3 the propagation of rigid group constraints. Section 5.4 extends rigid group propagation to include two groups connected by a torsion angle. Section 5.5 explains how PSICO can detect if groups of atoms are being confined into a volume too small to hold them, which results in forbidden configurations with atomic overlaps.

Section 5.6 describes how the constraint propagation algorithms in the previous sections are brought together to impose consistency on the system. Section 5.7 describes the enumeration and backtracking techniques used to search the possibilities that consistency could not rule out, and section 5.8 explains the local search algorithm that refines the final solution. Section 5.9 deals with several performance issues, mostly affecting only the calculation time.

Chapter 8 describes performance tests on PSICO, and section 10.8 describes the combination of PSICO with a local search algorithm based on Multi-Dimensional Scaling and dissimilarity matrixes, with a potentially more general applicability beyond the field of protein structure.

PSICO is also available as a Dynamic Link Library (DLL), which is described inAppendix I: PSICO Dynamic Link Library, page 147.

5.1 Modelling the Variables

The typical approach to protein structural problems using molecular dynamics or optimisation techniques is to find the coordinate values by solving systems of differential equations (Hansson and others 2002). Though the position variables are formally vectors, each x, y, and z coordinate is, in practice, an independent value, related to the other components of the vector solely by the functions to minimise or the equations of motion. Some methods convert the Cartesian coordinates to other systems to take advantage of some particular restrictions. Torsion angle dynamics is an example of this approximation, by modelling the coordinates of the atoms as a function of rotation angles around selected bonds (Abe and others 1984; Guntert and others 1997). Still, the transformation is only at the level of the model, while the solver, in practice, works with all the variables as independent scalars.

The constraint programming approach to geometrical or structural problems is very different. In many cases, the problem is modelled in a finite domain, where each variable can have only one of

a finite set of possible values (Rodošek 2001; Bakofen 1998, 2001). A more general approach to geometrical problems focuses on the restrictions to the six degrees of freedom (rotation and translation) for the motions of parts of the system under study (Kramer 1992; Bhansali and others 1996). The solution found for PSICO was inspired in the latter system, though applied to the special case of an atom, which has only three degrees of freedom.

First, let us look at the problem of modelling each coordinate of each atom as a different variable. Using this approach, each x, y and z coordinate will have its own domain that is reduced independently of the others, and any relation between the coordinate domains of one atom must be assured by the propagation of constraints. Figure 5-1 shows illustrates, in two dimensions, the reduction of domain A by excluding the atom from region B. If modelled as separate variables, arc-consistency cannot reduce neither the x nor the y domain (represented as horizontal and vertical bars, respectively), because all values have support even after removing region B.

However, if the domain is the set of (x,y) pairs that are allowed (or triplets, in the three-dimensional case), then it is possible to represent the exclusion of B from A to give A'. This is a natural extension of the idea of restricting freedom of motion (Kramer 1992; Bhansali and others 1996).

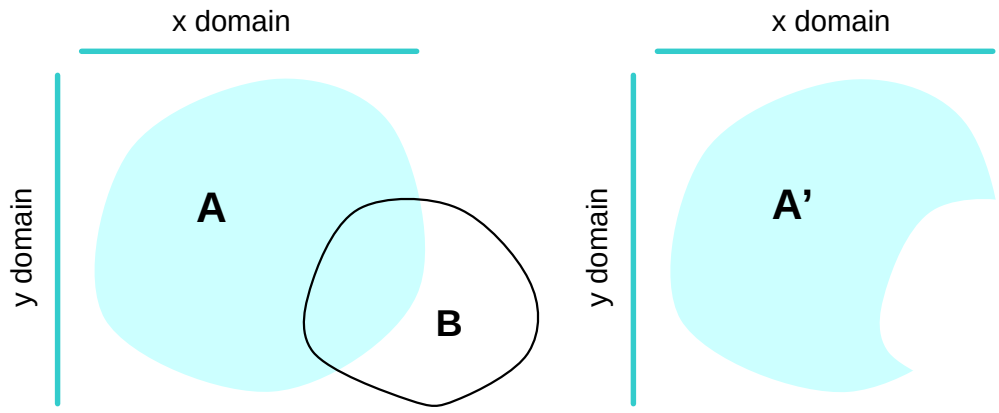


Figure 5-1 Excluding atom A from region X requires a simple alteration in the domain of A if the coordinates of A are modelled together, but the separate x and y domains, represented by the bars, are not changed by removing the region X from domain A.

The problem with this representation is its complexity, for it requires much more information to specify an arbitrary three-dimensional shape than an independent domain for each one-dimensional x, y and z coordinate.

The solution is to simplify the representation, using a set of cuboid blocks to represent each domain. One block represents the Good Region, which is the volume where the centre of the atom can be, and an additional set of zero or more blocks represent the No-Good regions, a set of non-overlapping volumes contained in the Good Region from which the atom is excluded. All block faces are orthogonal to the coordinate axes, so all blocks can be defined by two points marking the lower and upper bounds of the block coordinates in each axis.

Figure 5-2 illustrates this representation. A domain with an arbitrary shape (represented by the blue spheres) can be simplified with a set of blocks. The second panel of the figure shows the Good region; a single block that includes all points in the original domain. A set of No-Good regions improves the approximation by representing volumes inside the Good block that do not belong to the domain represented.

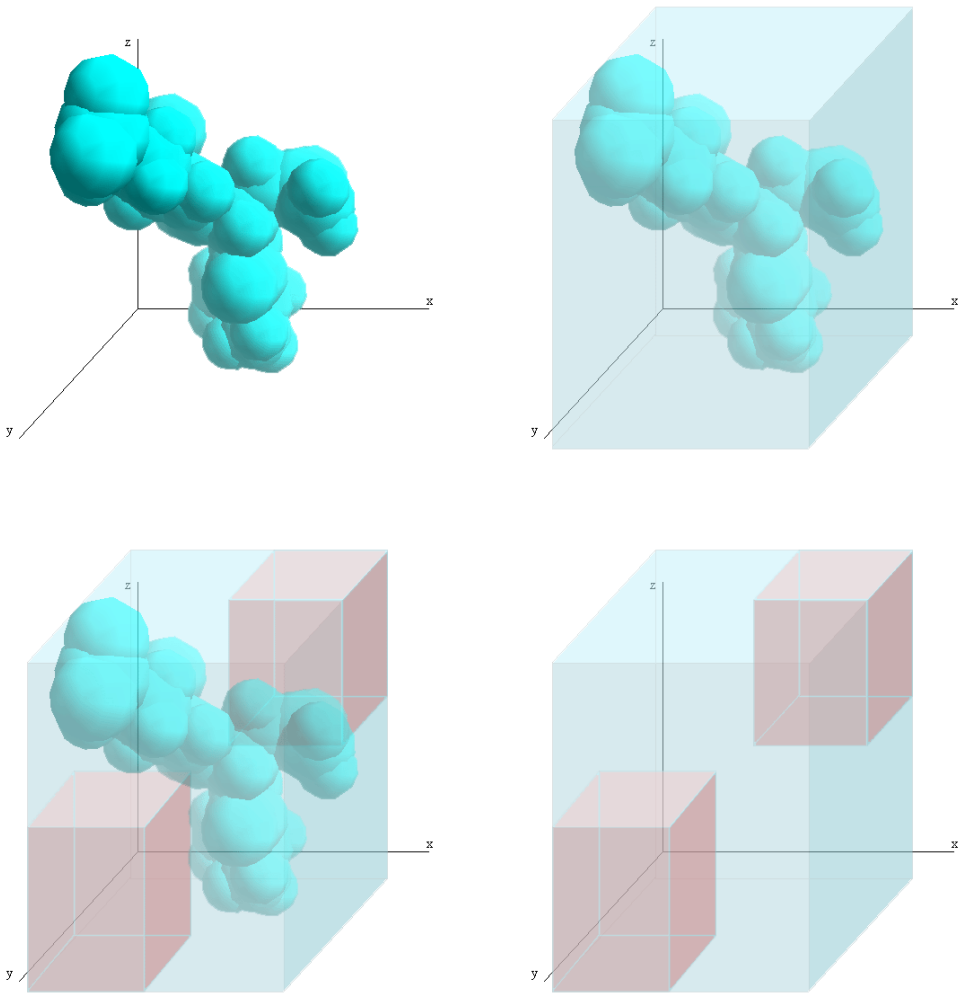


Figure 5-2 The domain of an atom (top left) is represented using a simple cuboid block (top right), the Good region, plus A set of zero or more additional blocks represents the no-Good region, from which the atom is excluded (bottom).

The shape of these blocks and their orientation relative to the coordinate axes simplifies the calculation of intersections or differences of domains. This is a useful property for constraint propagation, as shown in the next sections.

Algorithm 5-1 ensures that the No-Good blocks do not overlap by decomposing intersecting blocks and eliminating the intersections. Algorithm 5-1 is not efficient, but its efficiency is not an important factor in practice, because No-Good blocks tend to appear in significant numbers only in the final stages of the computation.

```
Input: Block, OldBlocks
Output:NewBlocks
For n=1 to length of OldBlocks do
  If OldBlocks[n] contains Block then NewBlocks=OldBlocks; terminate
  Else If OldBlocks[n] intersects Block then
     Add to NewBlocks a set of blocks defining the region not intersecting Block
  Else Add OldBlocks[n] to NewBlocks
End For
```

Algorithm 5-1 inserts a new no-Good block (Block) into a set of no-Good blocks (OldBlocks) ensuring that the blocks in the final set (NewBlocks) do not overlap.

The purpose of Algorithm 5-1 is to simplify the calculus of the total No-Good volume, by preventing No-Good blocks from overlapping. An alternative under consideration is to represent the No-Good region as a Constructive Solid Geometry (CSG) tree (Foley and others 1996) to represent the No-Good region. Using this system, the set of No-Goods would be represented by a tree containing a cuboid block in each node and a set operation (union, intersection or difference) in each arc. CSG trees are widely used in ray-tracing algorithms and computer graphics in general, so there are several methods available for calculating the volume of the shape represented by the CSG tree.

5.2 Distance Constraints

Distance constraints in three dimensions have the general form:

$$d_a \leq \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \leq d_b$$

Meaning that the distance between points (x_1, y_1, z_1) and (z_2, y_2, z_2) must be between the values of d_a and d_b. It is useful to divide this into two different constraints, because, as we shall see, they are propagated in different ways:

Out Constraint: $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \geq d_a$

In Constraint: $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \leq d_b$

The Out Constraint specifies that one point must be outside the neighbourhood of distance d_a of the other, while the In Constraint specifies it must be inside the neighbourhood d_b of the other point.

The problem with this formulation is that these equations define spherical volumes, which do not fit well with the block representation of the domains. Furthermore, the combination of spherical volumes — especially intersections — tends to increase the complexity of the representation. The solution was to reformulate the constraints to use a distance metric of first order, and so define cuboid regions which can be easily combined with the domain representation described in section 5.1:

Out constraint: $|x_1 - x_2| > \frac{d_a}{\sqrt{3}} \vee |y_1 - y_2| > \frac{d_a}{\sqrt{3}} \vee |z_1 - z_2| > \frac{d_a}{\sqrt{3}}$ (5.1)

In Constraint: $\max(|x_1 - x_2|, |y_1 - y_2|, |z_1 - z_2|) \leq d_b$ (5.2)

Figure 5-3 illustrates this reformulation.

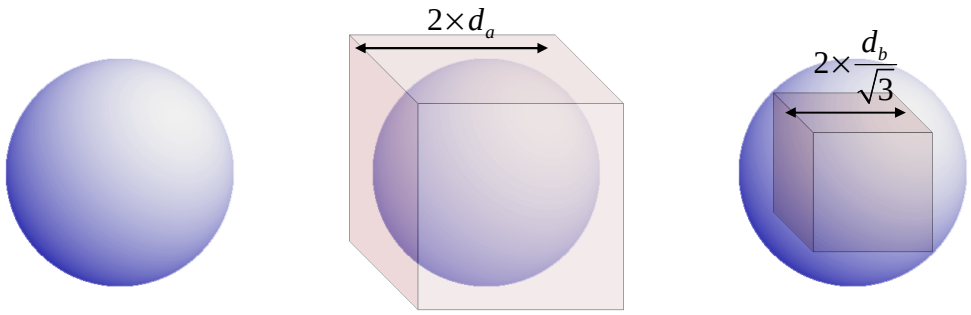


Figure 5-3 Modelling distance constraints as cubic regions. On the left a sphere defined by the Euclidean distance to a point- On the centre, the cube modelling an In Constraint, with an edge of twice the distance value. On the right, the cube modelling an Out Constraint, with an edge of twice the distance value divided by the square root of three.

To propagate In Constraints it is only necessary to intersect the Good Region of the domain of one atom with the neighbourhood of the Good Region of the domain of the other atom, this neighbourhood being determined by the Good Region and the distance value for the constraint. This restricts the domain of the first atom to those points within the required distance of any point

in the domain of the second, thus eliminating regions where it is impossible to respect the In Constraint. Figure 5-4 illustrates this process.

After reducing the domain of an atom by the propagation of an In Constraint, all No-Good blocks in that domain are intersected with the Good Region, to ensure it remains contained in the latter. Algorithm 5-2 outlines the propagation of an In Constraint.

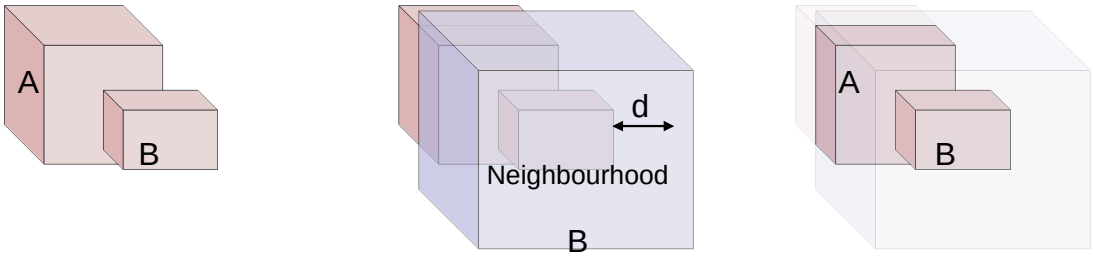


Figure 5-4 Propagation of an In Constraint. The left panel shows the Good Regions of the two atom domains. The centre panel shows the neighbourhood region of atom B, and the right panel shows the new domain of atom A, which was restricted to the intersection with the neighbourhood of the atom B.

```
Input: Atom1, Atom2, Distance
Block=Neighbourhood(Atom1.Good,Distance)
Block=Intersect(Atom2.Good,Block)
Atom2.Good=Block
Intersect Atom2.NoGoods with Atom2.Good
```

Algorithm 5-2 InConstraint. Propagating the In Constraint. The Neighbourhood function returns a copy of the block in the first parameter extended in all directions by the distance value in the second parameter. See Figure 5-4.

To propagate an Out Constraint, it is necessary to eliminate from the domain of one atom those points that cannot be at least as distant as the constraint requires from any point in the domain of the other atom. Thus, the propagation of Out Constraints intersects the exclusion region of one atom with the Good region of the other, and then adds the result to the set of No-Good Regions of the other atom. This exclusion region is the set of points with a distance from all atoms in the domain no larger than that specified by the constraint. Figure 5-5 illustrates this propagation.

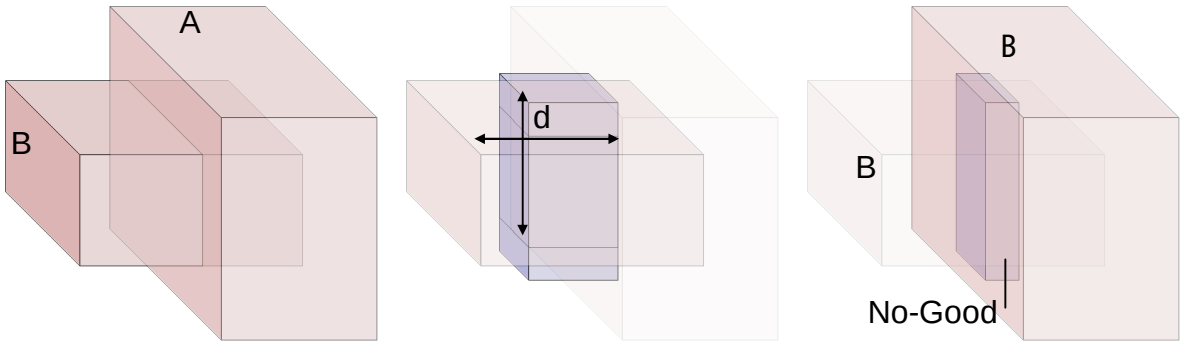


Figure 5-5 Propagating the Out Constraint. The left panel shows the Good Regions of two overlapping domains. The central panel shows the exclusion region of atom B. The right panel shows the intersection of the exclusion region with the good region of atom A, which is added to the No-Good Regions of atom A.

To calculate the low coordinates of the exclusion region (blue, in Figure 5-5) we subtract the distance value from the high coordinates of the original, and add it to the low coordinates of the original to obtain the high coordinates of the exclusion region. We can imagine that we are turning the original block inside out by moving the extremities over the given distance value. If the distance value is less than half the length of the original block in any dimension then this constraint does not generate an exclusion region. Algorithm 5-3 outlines the calculation of the exclusion region and propagation of the Out Constraint.

```
Input: Atom1, Atom2, Distance
For j=1 to 3 do
  Block.HighCoordinates[j]=Atom1.Good.LowCoordinates[j]+Distance
  Block.LowCoordinates[j]=Atom1.Good.HighCoordinates[j]-Distance
If Block.HighCoordinates>Block.LowCoordinates then
  Block=Intersect(Atom2.Good,Block)
  InsertNoGood(Block,Atom2.NoGoods)
```

Algorithm 5-3 OutConstraint. Propagating an Out Constraint. After the exclusion region is calculated, Block is intersected with the Good region of the domain of Atom2, and the result inserted into the set of No Good block of Atom2 using Algorithm 5-1. See Figure 5-5.

Algorithm 5-1, called as InsertNoGood in Algorithm 5-3, ensures that the No-Good blocks do not overlap, to make it possible to determine when the No-Good region fills all the Good Region by simply adding the volumes of all No-Good blocks.

5.3 Rigid Group Constraints

Chemical bonds restrict groups of atoms, making them effectively rigid groups, with a fixed geometry. This is not to say the groups of atoms are rigid. At the small scale of inter-atomic interactions, where quantum phenomena prevail, the very notion is somewhat questionable.

Furthermore, rapid vibrations and changes in the length and angle of covalent bonds are an intrinsic part of their nature, and some freedom of motion is an important factor in the thermodynamics of protein folding and interaction (Searle and others 1992).

However, these considerations are implicit in the idea of protein structure, as we acknowledge that this abstraction represents an average structure, and not a single real molecule. We can consider rigid groups of atoms in somewhat the same sense that the plastic models for benzene rings and other groups are rigid: the rigid model is an accurate model, even if reality is more dynamic.

In this sense, molecular bonds, bond angles, some chemical groups, and even the secondary structure of the protein provide rigid motifs. Sometimes it can even be reasonable to assume a fixed shape for some part of the molecule, either from experimental data (such as knowing the geometry of ligands of a prosthetic group) or from other sources (such as homology modelling).

The motivation to implement a generic solver for rigid group constraints — a constraint that specifies that a group of atoms has a fixed geometry, with each atom keeping its position relative to the others in the group – was to provide a framework to integrate this knowledge with the constraint propagation system described in section 5.2.

The group is a single entity, with three rotation and three translation degrees of freedom. By determining the possible placements of the group allowed by the domains of all atoms in the group, it is possible to restrict the domains of these atoms to those regions allowed by the possible placements of the group.

In essence, the group propagation algorithm consists of three nested loops that run through all orientations of the rigid group. For each orientation, the algorithm determines the domain boundaries of each atom from the limits that all domains impose on the translational freedom of the group, as shown in sub-section 5.3.1. Without loss of generality, we can consider the three rotation loops to correspond to rotations around the x, y, and z axes, from outer to inner loop. The letters χ, φ, and ψ will designate angles of these rotations.

Though it is useful to imagine the algorithm as three nested loops of rotations around the three coordinate axes, the algorithm actually determines the domain limits analytically as a function of one rotation, as section 2.2 describes. Thus, the last rotation, around the z axis, is not an actual loop, but the analytical determination of the domain limits for the x and y coordinates. The domain limits in the z coordinate, being independent of the last rotation around the z axis, are determined

analytically in the second loop, corresponding to the rotation around the x axis. This procedure is explained in section 5.3.2.

5.3.1 Domain limits for a fixed orientation

Given a fixed orientation, it is simple to determine the limits for the translation of the group. Figure 5-6 shows this for a group of three atoms (A, B, and C).

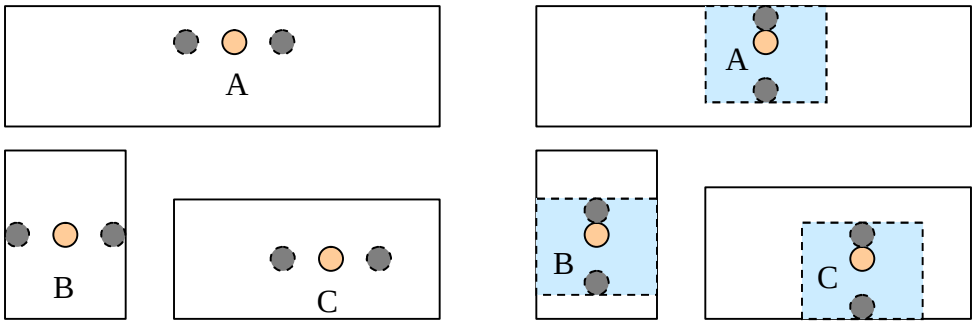


Figure 5-6 The three atoms A, B, and C, form a rigid group (dark orange circles), and each atom is restricted to a rectangular domain. The left panel shows the limits for a horizontal translation, and the right panel shows the limits for a vertical translation (light orange circles with dashed line). The dashed boxes on the right panel indicate the accessible regions for each atom with the group in this orientation.

The range of the translation for the group is simply the intersection of the ranges allowed by all atoms. In this example, the domain boundaries of atom B limit the horizontal displacement (left panel), and the domain of atom A limits the vertical displacement (right panel). For this particular orientation of these three atoms, the domains are restricted to the dashed boxes shown on the right panel.

Denoting by w_c one of the coordinates of the center of the group (x, y or z), by w_j the same coordinate for atom j, and by w_{max} and w_{min} the upper and lower limits, respectively, for that coordinate of a domain (of atom j or of the center c), the limits are related by the following equations:

$$\begin{aligned} w_{\max c} &= Min_{j=1}^{n}(w_{\max j} + (w_c - w_j)) \\ w_{\min c} &= Max_{j=1}^{n}(w_{\min j} + (w_c - w_j)) \end{aligned} \qquad (5.3)$$

The absolute values of w_c and w_j are irrelevant; only the coordinate difference w_c-w_j is important, and is independent of translation. These equations apply to all coordinates x, y, and z. The center point is simply the pivot from which rotation is calculated. It can be the geometric center of the

group, or any other point. The choice has impact only on some performance details (see sub-section 5.9.5), and not on the principles of the algorithm.

5.3.2 Domain limits as a function of a single rotation.

Equation (5.3) assumes a fixed orientation of the group, but we cannot make that assumption, since the group is free to rotate. Without loss of generality, we shall consider the case of the limits in the x and y coordinates as a function of a rotation around the z axis. Hence, the term (w_j-w_c) in equation (5.3) may represent the x-or y-components of the vector from atom j to the centre of the group, or, in other words, the position of the centre relative to atom j.

This vector is a function of the orientation of the group. Denoting by ψ the rotation around the z axis, by A the amplitude of the projection of the vector onto the xy plane (orthogonal to the rotation axis) and by α_j the angle of the vector y_c-y_j at ψ=0, then the terms w_c –w_j for the x and y coordinates are given by:

$$\begin{aligned} x_c - x_j &= A_j \cos(y + a_j) = A_j \sin(y + a_j + p/2) \\ y_c - y_j &= A_j \sin(y + a_j) \end{aligned} \qquad (5.4)$$

Where θ is the rotation around one axis perpendicular to the x coordinate axis, α_j the angle of the vector x_c-x_j at θ=0, and A is the distance of the projection on the plane perpendicular to the rotation axis, as shown in Figure 5-7.

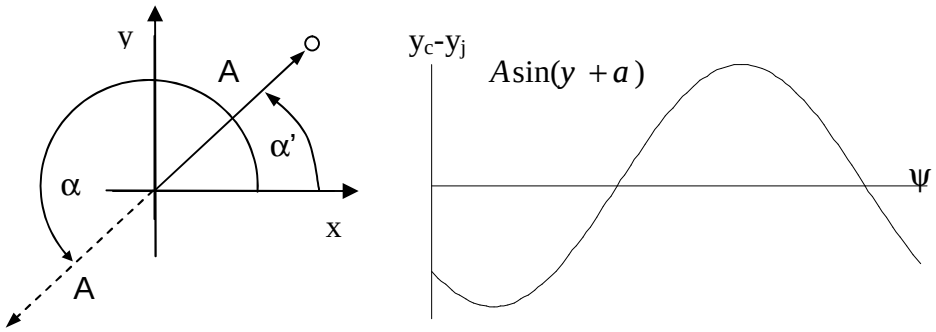


Figure 5-7 The position of an atom relative to the center of the group as a function of the rotation angle y can be expressed as a sine function with amplitude A and phase a'. The position of the center relative to the atom is a similar curve, but with the phase shifted by 180° (a), giving the sine line shown on the right.

Though rotation freedom increases the complexity of the computation, it is necessary because our rigid group constraints only give information on the relative positions of the atoms in the group and not on the orientation or position of the group relative to other atoms or groups.

Let us now recall the algorithm as outlined in the beginning of section 5.3. First, the orientation of the group around the x axis is fixed in an angle we designate χ. Next, the rotation around the y axis is fixed at angle φ. For each (χ;φ) pair, equation (5.4) can be used to describe the x and y coordinates for each atom as a function of the angle ψ, corresponding to the rotation around the z axis. An equation similar to equation (5.4) can also be used to describe the z coordinates of atom j (related to the centre of the group) as a function of the second rotation, φ, around the y axis.

With no loss of generality, we can replace y_c and y_j in equations (5.4) with L_c and L_j to denote, respectively, the domain limits of the center and of atom j in an arbitrary x, y, or z coordinate. We also replace ψ with θ, denoting an arbitrary rotation angle (φ or ψ), and compute the contribution of each atom to the limits on the translation of the center of the group by means of Equation (5.5) below. Equation (5.5) gives us the limits for the translation of the centre as a function of the limit (upper or lower) of the domain of an atom in that direction (x, y, or z) and the orientation of the group around a rotation axis:

$$L_c = A_j \sin(q + a_j) + L_j \qquad (5.5)$$

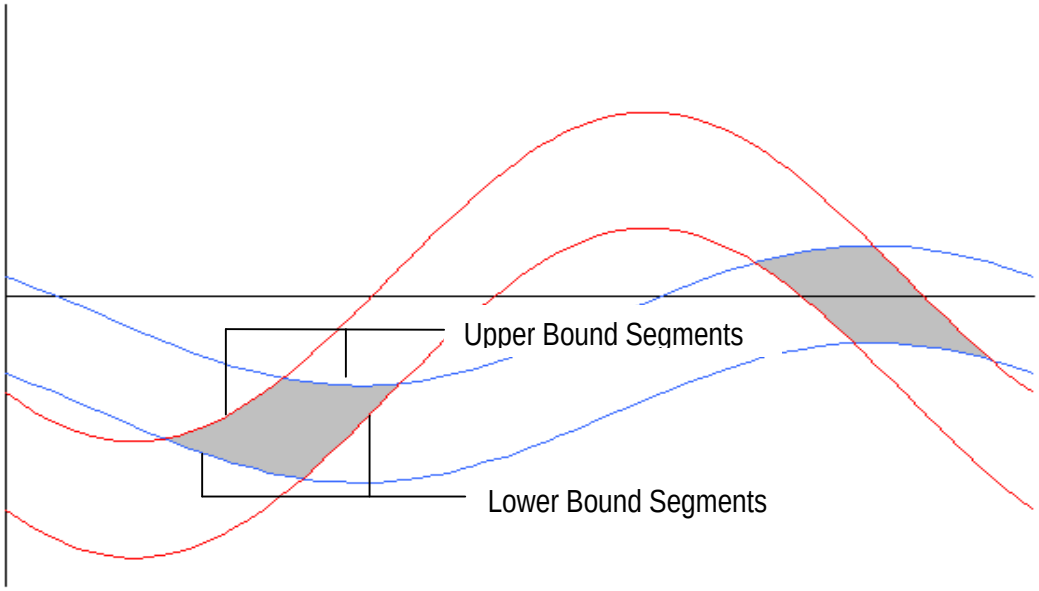


Figure 5-8 Each atom constrains the translation of the group as a function of the domain of the atom and rotation of the group. This figure shows such constraints for two atoms (red and blue) along one coordinate axis for one rotation. The grey areas indicate the angles for which the two atom positions are compatible and the corresponding constraints on the translation of the whole group in this orientation.

Figure 5-8 shows two curves representing the limits on the movement of the centre imposed by two different atoms. As we saw before, the actual limits for the movement of the centre for each

orientation are the most restrictive limits (around the grey areas). To calculate these we must calculate the intersection of the limits imposed by two different atoms.

$$\begin{aligned} &A_i \sin(q + a_i) + L_i = A_j \sin(q + a_j) + L_j \Leftrightarrow \\ &A\sin(q + a) = B_j - B_i \end{aligned} \tag{5.6}$$

The values for A and θ can be calculated using the formulae for the superposition of two sine waves of the same frequency, noting that the sign change for A_j when manipulating equation (5.6) is equivalent to changing the phase θ_j by π:

$$\begin{aligned} \pm A &= \sqrt{A_i^2 + A_j^2 + 2A_iA_j \cos(a_j - a_i + p)} \\ \tan a &= \frac{A_i \sin(a_i) + A_j \sin(a_j + p)}{A_i \cos(a_i) + A_j \cos(a_j + p)} \end{aligned} \tag{5.7}$$

The intersection angles are thus:

$$q = \arcsin\left(\frac{B_j - B_i}{\pm A}\right) - a \tag{5.8}$$

Using equations (5.7) and (5.8), Algorithm 5-4 defines the segments of the lines limiting the upper and lower bounds for the translation of the centre, with each segment being the line between two intersections. The upper limit is formed by the lowest segments of the upper limits of all atoms, and the lower limit by the highest segments of the lower limits of the atoms. The possible region for the centre is thus the intersection of the regions allowed by the limits of each atom, as illustrated in Figure 5-8.

```
Output:Upper and Lower limits of center
Lu=UpperLimit(Atom 1)
Ll=LowerLimit(Atom 1)
For n=2 to number of atoms do
  Lu=LowLine(UpperLimit(Atom n ),Lu)
  Ll=HighLine(LowerLimit(Atom n ),Ld)
PossibleRegion(Ll,Lu)
Return Ll, Lu
```

Algorithm 5-4 CenterLimits; returns the upper and lower boundary segments that limit the displacement of the centre of the group in one direction as a function of the rotation of the group around one axis.

Lu and Ll are lines formed by segments of the form of Equation (5.5). Functions UpperLimit and LowerLimit return the line defined in Equation (5.5), using respectively the upper and lower limit of

the domain of the atom. Functions LowLine and HighLine intersect two lines and return, respectively, the set of the lowest or highest segments between intersections. The PossibleRegion procedure restricts the lines to the ranges of rotation angle where the upper line is greater than or equal to the lower line.

This algorithm returns the upper and lower limits for the displacement of the centre as a function of rotation. These include only the possible regions, where the upper limit is equal to or higher than the lower limit, as shown in Figure 5-8. To calculate the limits for the displacement of each atom, it is necessary to add the function for the position of the atom relative to the centre of the group. In other words, we projected the limits of each atom onto the displacement of the centre, and now project the limits of the centre back onto the atom. This will limit the range of possible positions for the atom using the limits for the centre, and may allow us to reduce the domain of the atom.

As shown in Figure 3, the vectors used defined sine lines that differ only in a phase difference of π. Deriving from Equation (5.5):

$$\begin{aligned} L_c(q) &= A_j \sin(q + a_j) + L_j(q) \Leftrightarrow \\ L_j(q) &= A_j \sin(q + a_j + p) + L_c(q) \end{aligned} \tag{5.9}$$

Equation (5.5) shows that L_c is a function of the rotation parameter. If there were only one atom, L_c would be independent of θ, because we would simply invert the operation in equation (5.5). However, with several atoms, L_j is the limit of the intersection of several possible regions, and can consist of several different segments, which can have different A and α parameters. So the term $A_j \sin(q + a_j + p)$ must be added to each segment of $L_c(\theta)$, and $L_j(\theta)$ is a list with the same number of segments. All these sums are superpositions of sine waves of the same frequency, so we apply equation (5.7).

It is not necessary to do this for all atoms; we can ignore any atom that contributed to $L_c(\theta)$, because this implies the atom reached the limit of its domain, and so its domain cannot be reduced. It is just necessary to keep track of which atom contributed to which segment.

This solves the problem for the rotation around a single axis. However, it takes three axes to describe all possible orientations of the group. If we account for the additional rotations, the A and α terms in equation 3 become trigonometric functions of the other rotation parameters, which makes the system too complex for an efficient analytical solution.

The alternative is to use a discrete representation of the other rotation parameters, and interval algebra to account for the error introduced by this approach.

5.3.3 Full rotation search.

As outlined in the beginning of section 5.3, the algorithm searches the possible rotations by fixing the angle of rotation around the x axis, then for each rotation value it searches the rotations around the y axis, and finally the z axis. Dividing the rotations into finite intervals, each orientation corresponds to an interval of angles, instead of to a single angle, and each coordinate to an interval of values.

As an example, the x coordinate as a function of the rotation φ around the y axis is:

$$x(j)_j = x_j \cos(j) - z_j \sin(j)$$

where x_j and z_j are the x and z coordinates for φ = 0. If we consider the interval $[\varphi_a ; \varphi_b]$, then x will be in the interval $[x_a ; x_b]$, where

$$x_a = Min(x_j \cos(j_a), x_j \cos(j_b)) - Max(z_j \sin(j_a), z_j \sin(j_b))$$
$$x_b = Max(x_j \cos(j_a), x_j \cos(j_b)) - Min(z_j \sin(j_a), z_j \sin(j_b))$$

If x_j and z_j are themselves intervals too, the principle is the same, with the only difference that the Min and Max functions will apply to all the four, following the rules for interval algebra. One thing to note is that the sine and cosine functions must be monotonous in the interval $[\varphi_a; \varphi_b]$. However, this is simple to guarantee by choosing the partition of the rotation so that the step size is a sub-multiple of 90º.

Because of this approach, the equations derived in sub-section 5.3.2 apply to intervals, and not to single coordinate values. Although the coordinates are in three dimensions, we need only consider the projection on the plane perpendicular to the rotation axis that provides the θ angle (θ corresponds to φ for the determination of the z limits as a function of the rotation around the y axis, or to ψ for the determination of the x and y limits as a function of the rotation around z axis). Let us assume, for example, that this rotation is around the z axis, and so the coordinates of one atom are $([x_a;x_b],[y_a;y_b])$. This is a rectangular region, illustrated in Figure 5-9, with the corners labelled A, B, C, and D. As Figure 5-9 shows, these corners have different trajectories when the region is rotated. Now we need to find two lines of the form of equation (5.5) that enclose the trajectories of all points in the region by enclosing the trajectories of the corners. These lines are

the trajectory of the central point plus or minus a slack δ. The slack is half the diagonal of the region, which corresponds to the largest span the region can cover in any orientation:

$$d = \frac{\sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}}{2} \qquad (5.10)$$

Figure 5-9 shows these two lines (thicker black lines). By subtracting δ from the lower limit of the domain of the atom and using the result as the lower limit, and by adding δ to the upper limit and using that result as the upper limit we ensure no values of the domain are lost.

To summarize, the algorithm searches the rotation space starting from the rotation around the x axis (χ), for each interval value of this rotation, the rotation around the y axis (φ), and for each (χ,φ) interval pair, the rotation around the z axis (ψ).

The x and y coordinates of each atom depend on all three rotation parameters, but the z coordinate depends only on the first two (χ and φ, respectively rotations around the x and y axes), being independent of ψ. This suggested Algorithm 5-5.

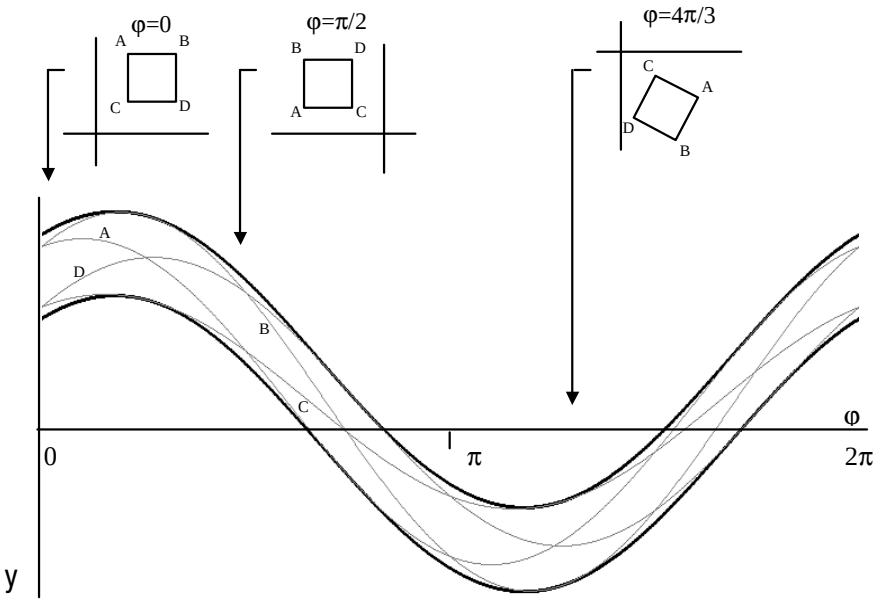


Figure 5-9 The thick lines show the lines obtained by adding d (top line) and subtracting d (bottom line) to the line defined by the centre of the coordinates interval. The top panels show three orientations of the rectangle defined by the coordinate intervals. A, B, C, and D are the extreme points of the rectangle, and thin lines in the lower diagram show their trajectories as a function of the j rotation parameter.

The first rotation, χ, ranges from 0º to 180º; φ and ψ range from 0º to 360º. This is enough to represent all possible orientations of the atom group. For each step of the χ rotation, this algorithm calculates the domain of the centre in the z direction using Algorithm 5-4 for rotation φ and the z limits of all atoms. This will often prune the φ rotation, and only those steps allowed by the z limits of the centre will be considered in the calculation of the x and y limits.

```
Initialize atom range limits
Search χ using discrete intervals. For each step do
  Project atom coordinates and z limits on xz plane
  ZL=CenterLimits for z limits as a function φ
  Search allowed values of φ using discrete intervals. For each step, do:
     Project atom coordinates, x limits and y limits on xy plane
     XL=CenterLimits for x limits as a function of ψ
     YL=CenterLimits for y limits as a function of ψ
     Intersect XL and YL limits to determine allowed ψ values
     For each atom do
        Project XL,YL to atom domain and determine x, y extremes.
        Project φ interval of ZL to atom domain and determine z extremes
     Update atom ranges for each coordinate:
        Upper range limit =Max(new range limit, old range limit)
        Lower range limit=Min(new range limit, old range limit)
```

Algorithm 5-5 RotationSearch. Searches the three rotation axes c, j, and y, calling the CentreLimits algorithm (Algorithm 5-4) to determine the boundaries for the centre as a function at each orientation.

For each (χ, φ) step, we have the centre limits of x and y as a function of ψ and the z limits as a function of φ, of which we only consider the segment referring to the current φ. Projecting these segments onto the atom domains determines the range of each atom in each coordinate. The final domain of each atom is initialised as an empty domain, but after each step it is extended to include the range of coordinates allowed by this orientation of the group. When the rotation search is complete, the final domain of each atom contains all the possible positions in all the possible orientations of the group, as allowed by the initial domains of all atoms in the group.

5.4 Torsion Angle Constraints

A molecular bond that connects two rigid groups restricts the relative movement of the groups, and often this restriction is an important experimental datum. One group is only free to move relative to the other by rotating around the axis defined by the molecular bond connecting the two, and this rotation can be restricted to a narrow range of values. The angle is measured relative to a dihedron

formed by four atoms, and is called a dihedral or torsion angle. . Figure 5-10 illustrates this movement, showing two groups connected by a rotatable bond.

Because of this freedom of movement, we cannot model two connected rigid groups as a single rigid group, but we can model them as an ensemble of rigid groups spanning the conformations allowed by this rotation. In an approach similar to the one used for searching the rotation space of a rigid group, we can search the rotation of one group relative to the other by partitioning the rotation angle into a finite number of steps and using interval algebra to propagate the boundaries on the atomic coordinates. Algorithm 5-6 outlines this procedure.

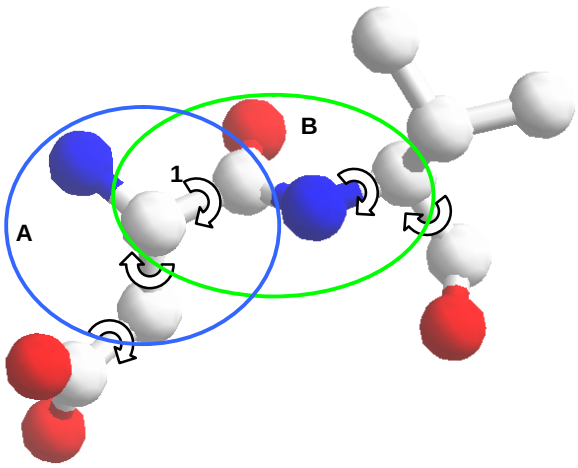


Figure 5-10 Two rigid groups, A and B, joined by a rotatable bond (1). Note that the two atoms on both ends of bond 1 belong to both groups, since rotation around this axis does not change the position of these atoms relative to the atoms in A or B. There are other bonds and other groups in the figure, but not all are displayed, for clarity. Atoms are coloured by element: C white, N blue, O red. Hydrogen atoms are not displayed.

```
Input: Group1, Group 2, Rotation interval
Partition the rotation interval into finite number of steps
Initialize new atom domains
For s=1 to number of steps do
  Generate Group3 by combining Group1 with Group2 in rotation s
  Calculate coordinate intervals for atoms in Group3 that correspond to Group2
  Run RotationSearch without initializing atom domains
```

Algorithm 5-6 TorsionGroup. Searches the rotation around the bond connecting two rigid groups. The RotationSearch procedure is Algorithm 5-5, without the domain initialisation step.

5.5 Atomic Overlap Global Constraint

No two atoms can occupy the same volume at the same time. If we know where each atom is, we can verify this constraint by checking each atom pair for overlap: if the distance between the centres is less than the sum of the two radii, then the constraint is violated. One way that PSICO models this constraint is by propagating Out Constraints between all atom pairs, but these

constraints are on the order of the square of the number of atoms, so the propagation must be limited to limit the impact on performance. See section 5.9 for more details.

However, the most important problem is that we do not know the exact location of each atom. Let us imagine we have N atoms, each with a diameter of one unit, and we must fit them into a box with a width and height of one unit and a length of two units. If we check each atom pair, we will notice no problem, for two atoms fit in the box if we place them correctly. But it is evident that for N>2 not all atoms will fit. The problem we have is similar, because the domains only give us the boxes where the atoms must be in, but not their exact positions. The overlap constraint is thus a global constraint:

The volume spanned by the domains of a group of atoms must be equal to or greater than the total volume occupied by all atoms in the group.

This is a global constraint, because it applies to an arbitrary group of atoms, and the difficulty lies in determining which group of atoms to examine. We cannot expect that all atoms in the problem are restricted to a small enough volume to violate this constraint. The more common situation would be for some atoms, in one region, to be too close together, but if we use the volume of the complete set of atoms we will not notice this. Since the number of possible sub-groups of N atoms is 2^N, it is not feasible to try them all.

The algorithm implemented was a compromise between examining only atom pairs or all possible groups. The check is still restricted to all atom pairs, a total of [N×(N-1)]/2, but for each atom, the algorithm adds the minimum volumes that all other atoms must occupy in the domain of the first atom. Each of these minimum volumes cannot be occupied by more than one atom, because atoms do not overlap, so the sum of the minimum occupied volumes is a lower bound on the total volume the other atoms must occupy in the domain of the first. Figure 5-11 illustrates the calculation of the minimum occupation volume for one atom.

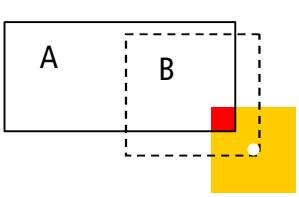


Figure 5-11 Calculating the minimum volume occupied by B (domain shown as a dashed box) in the domain of A. Atom B is considered to be as far as possible from A (white circle), and occupies the region shown in orange. The red box shows the minimum volume B occupies in A.

Calculating the minimum occupation volume is simple. As Figure 5-11 shows, atom B occupies the least volume on the domain of A if it is placed on the corner of the domain of B that is farthest from the domain of A. At most, this only requires measuring the distance from the domain of A to

each of the eight vertexes of the domain of B. Algorithm 5-7 outlines the procedure to calculate the total occupation volumes for all atoms.

```
Input:
  Atoms: array of atoms
For j=1 to number of Atoms do
  Volumes[j]= Atom[j].Good.Volume
for j=1 to number of atoms –1 do
  for k=j+1 to number of atoms do
     Volumes[j]=Volumes[j]-MinOccupancy(k,j)
     if Volumes[j]<0 then Fail
     Volumes[k]=Volumes[k]-MinOccupancy(j,k)
     if Volumes[k]<0 then Fail
```

Algorithm 5-7 Checking atomic overlap. The MinOccupancy function returns the minimum volume that the atom in the first parameter must occupy in the domain of the atom in the second parameter, as shown in Figure 5-11.

5.6 Enforcing Consistency

The previous sections showed the various procedures PSICO uses to propagate and enforce constraints. The algorithm must combine all these to enforce consistency on all atoms and constraints.

Because the distance constraints are the fastest to propagate (5.2), PSICO enforces arc consistency on these constraints (see section 3.2). When the domain of one atom is reduced, this atom is activated and all distance constraints involving that atom are propagated, potentially activating other atoms by reducing their domains. The process is repeated, as outlined in Algorithm 5-8, until no atom remains activated.

After propagating the binary distance constraints to arc consistency, PSICO propagates the group and torsion angle constraints once for each group that was affected by the previous arc consistency round. The sequence is repeated until a fixed point is reached and no atoms remain active.

Finally, the global atomic overlap algorithm checks for a possible failure due to excessive crowding. Algorithm 5-9 describes the complete procedure to enforce consistency. This algorithm is essentially AC-3 (Mackworth 1977), with the added complication of integrating different propagations and consistency checks, with different computation costs, in an efficient manner. More sophisticated Arc-consistency algorithms (Hentenryck and others 1992; Bessiere 1994; Bessiere and others 1999)

require the computation and storage of support sets, which would probably be too costly with the domain representations used in PSICO.

```
Input:
  Atoms: array of atoms
  Active: Boolean array
Repeat
  Select Atoms[n] with smallest domain and Active[n] is True
  Active[n]=False
  for j=1 to Atoms[n].Constraints do
     k=index of atom affected by Atoms[n].Constraints[j]
     InConstraint(Atoms[n], Atoms[k], Atoms[n].Constraints[j].InDistance)
     If Atoms[k] is changed by propagation then Active[k]=True
     OutConstraint(Atoms[n], Atoms[k], Atoms[n].Constraints[j].OutDistance)
     If Volume(Atoms[k].Good)<Volume(Atoms[k].NoGoods) then
        Fail
Until Fail or no Active[n] is True
```

Algorithm 5-8 DistanceArcConsistency. Enforces Arc-Consistency on the network of binary distance constraints. The input is the set of atoms and a Boolean array marking the atoms changed since Arc-Consistency was last enforced. InConstraint is Algorithm 5-2 and OutConstraint is Algorithm 5-3. Note that constraints are propagated using the Good region of the domain, so the OutConstraint propagation does not activate an atom, since it does not affect the Good region of the domain. The cycle continues until no atom is marked as active.

Path-consistency and other higher consistency algorithms would require the manipulation of tuples of values, and a preliminary analysis suggested the added computational cost was significately greater than the benefits of the increase in propagation (Marco Correia, private correspondence).

```
Input:
  Atoms: array of atoms
  Active: array of Boolean
  Groups: array of atom groups
  TorsionAngles:array of connections between groups
repeat
  DistanceArcConsistency(Atoms, Active)
  For all TorsionAngles connecting groups with at least one active atom do
     TorsionGroup(TorsionAngles[n].Group1,TorsionAngles[n].Group2,Interval)
  For all remaining Groups with at least one activa atom do
     RotationSearch (Groups[n])
until Fail or no atom Active
AtomicOverlap(Atoms)
```

Algorithm 5-9 Propagates all constraints to enforce partial consistency. DistanceArcConsistency is Algorithm 5-8, TorsionGroup is Algorithm 5-6, RotationSearch is Algorithm 5-5, and AtomicOverlap Algorithm 5-7.

5.7 Enumeration, Heuristics, and Backtracking

The propagation procedures described in the previous sections can determine the effects of restricting the domain of some atoms on the domains of all atoms in the structure. However, once such reductions are propagated, propagation can no longer help. Since consistency is not complete, a single propagation step will not result in a solution that respects the constraints.

To solve this we need enumeration. In other words, it is necessary to decide which atoms to restrict and how to restrict them before propagation can even begin. For the first choice, PSICO can pick a rigid group and place it anywhere in our coordinate space. This will only determine the position of the protein, but not the structure, since it does not affect the distances between the atoms. After propagating the consequences of this first enumeration, the algorithm must make a choice; it must choose an atom or group, and how to restrict its domain, before it can restart the propagation cycle described in Algorithm 5-9. Since this may be the wrong choice, it must be stored in a choice-point so execution can return to it and correct it if the choice results in an inconsistency. This, in short, is the enumeration and backtracking procedure, as outlined in Algorithm 5-10.

As Algorithm 5-10 shows, there are two decisions to make: which variable to choose, and how to restrict the values allowed for that variable.

The variable selection heuristic currently implemented chooses atoms starting from those with the smallest domains. The chosen atom is then marked as ineligible for enumeration until all other eligible atoms are chosen too. After each round, all atoms with a domain longer than the target length become eligible for enumeration once again.

```
Select a rigid group G and fix it in the coordinate space
Mark as active all atoms in G
Propagate(Atoms, Actives, Groups)
While not Fail and not Done do
  Atom=SelectAtomToEnumerate(Atoms)
  Domain=ChooseDomain(Atom)
  Atoms[Atom].Domain=Domain
  Actives[Atom]=True
  Propagate(Atoms, Actives, Groups)
  If Fail then Backtrack
  Else If all Atoms below the target domain size then Done
  Else StoreChoicePoint
End
```

Algorithm 5-10 Enumeration and Backtracking Propagate is Algorithm 5-9

A round robin selection is necessary because the atom with the smallest domain is likely to be selected again in the next enumeration because enumeration reduces the domain to half, especially if that domain reduction does not propagate to other domains. This would force a depth-first search and take less advantage of the pruning power of propagation by enumerating those atoms that have the least effect.

The value selection heuristic aims to keep atoms as far apart as possible. To this end, it divides the current domain into two parts along its longest dimension, and chooses the one with the smallest total overlap with other atom domains. The objective is to postpone atom collisions as much as possible because these can only be detected if the domains are sufficiently small.

Figure 5-12 shows the progress with the iterations of enumeration and propagation outlined in Algorithm 5-10. The blue structure indicates the target, a monomer of desulforedoxin (PDB code 1DXG, Archer and others 1995). The red structure is the PSICO model, using experimental NOESY data (Goodfellow and others 1998, 2002, 2003). The PSICO models show only those atoms with a sufficiently small domain (less than 5Å to a side).

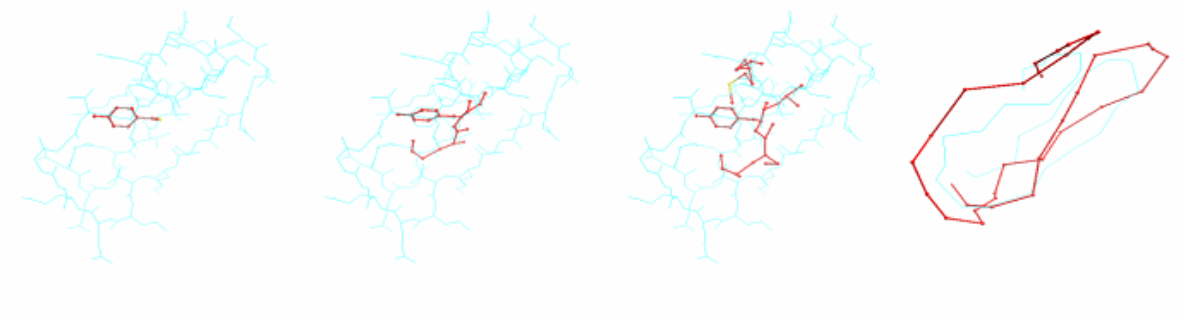


Figure 5-12 Evolution of the enumeration cycle. From left to right, after one iteration, 15, 30 and 300. The last panel shows only the backbone for clarity.

The first iteration, shown on the leftmost panel, fixes the side chain of tyrosine 7, which is the largest rigid group available. By iteration 15 (second panel) parts of the backbone of adjacent residues and some side chain atoms already have small domains. The reduction progresses along the protein chain until iteration 30, when constraints bridging two different segments of the chain start restricting the domains in another region, shown in yellow on the third panel. By iteration 300 (last panel) the backbone is already well-defined, and with an approximately correct folding.

5.8 Local Search Optimisation

After the constraint propagation stage builds an approximate solution, a conjugated gradient minimisation algorithm refines it to reduce remaining constraint violations and atomic clashes. This refinement stage operates on a torsion angle model of the protein. This model is a tree of rigid groups of atoms, connected by rotations around some covalent bonds, and the angles of these rotations determine the structure. Thus the algorithm ensures that bond lengths and angles are correct, since these remain constant, and only those dihedral angles with rotation freedom can change. Figure 5-13 illustrates the data structure and the corresponding molecular structure of the torsion angle model. Note that the nodes in the torsion angle model are similar to the rigid groups between the rotatable bonds, but not exactly the same.

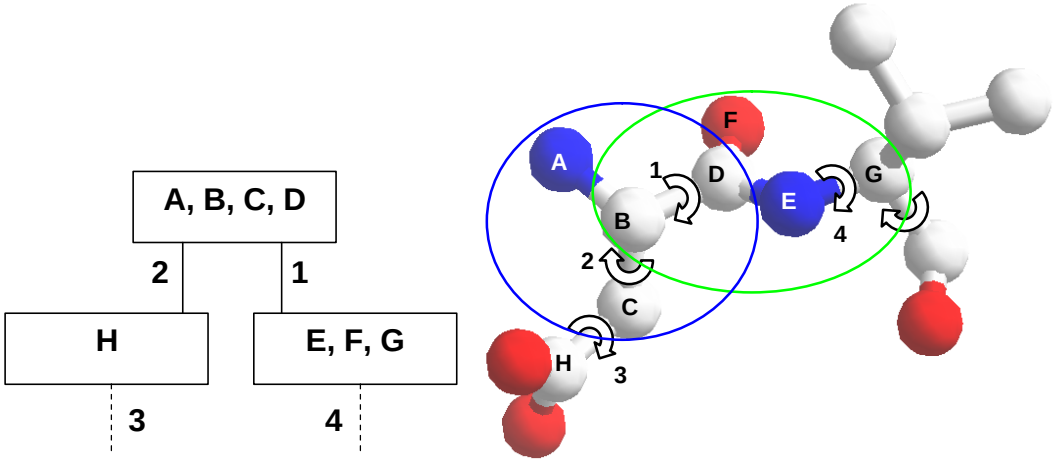


Figure 5-13 The torsion angle model. The panel on the left shows the data structure as a tree, with atom groups for nodes and dihedral angles for arcs. The panel on the right shows the respective protein structure.

The tree shown on the left panel illustrates the data structure, where groups of atoms are connected by nodes which specify their relative orientations.

The minimisation algorithm uses the conjugated gradient method (Press 1994). This is a standard minimisation technique that requires the evaluation of the function to minimise and its first derivative. In this case, the function to minimise is a function of the rotation and translation of the monomers in the protein, vectors R and T on equation (5.11), and the vector of rotation angles describing each protein chain (X in the same equation).

$$f(R,T,X) = \sum_j Viol(j, |A_{j,1}, A_{j,2}|)$$

$$Viol(j,d) = \begin{cases} (d - Min_j)^2, d < Min_j \\ 0, Min_j \le d \le Moax_j \\ (d - Max_j)^2, d > Max_j \end{cases} \quad (5.11)$$

R and T can be ignored for the first monomer, or if the structure contains only one chain, but are important to specify the relative position of the second or subsequent monomers.

The function to minimise is the sum of the constraint violations given by the Viol function, which returns the square of the difference between the interatomic distance and the constraint limit violated, or zero if there is no violation.

Minimising this function is a straightforward procedure. The only complicating factor is the need to calculate the derivatives as a function of the torsion angles in an efficient manner. Abe and others (1984) describes a recurrent method to calculate the derivatives of energy functions with respect to torsion angles. This method was adapted to PSICO with minimal modifications.

Just to outline the method briefly, the idea is to identify which atoms lie within and outside the group affected by each torsion angle. For each angle, the algorithm considers only those atom pairs with one atom affected by the angle and the other unaffected, and ignores distance constraints in which the angle affects either both atoms or none.

5.9 Performance

This section describes some implementation details that are important for the efficiency of the algorithm, but have little or no impact on the procedures or results. Since efficiency is an important issue for the objectives of PSICO, these issues deserve some consideration

5.9.1 Simplified structure model

Approximately half the atoms in a protein are Hydrogen atoms, so it is possible to reduce the number of variables to half by ignoring them during the constraint processing stage. This also reduces the number of constraints, because each Hydrogen atom requires several distance constraints to model the length and angle of its covalent bond.

This is a legitimate simplification because each hydrogen atom can only be covalentely bound to one atom, and its placement is specified by the positions of neighbouring atoms, so they provide no

information on the structure of the molecule. Unfortunately, most experimental distance constraints in protein NMR come from hydrogen resonance spectra. Since hydrogen atoms are not in the model, we have to apply each of these constraints to the atom bound to hydrogen atom, adding the hydrogen bond length to the upper distance limit for the constraint and subtracting the bond length from the lower distance limit.

The same principle applies to pseudoatoms, which are not real atoms but merely placeholders for the geometric centres of groups of atoms that cannot be distinguished in the NMR spectra. For example, the three hydrogen atoms in a CH_3 group may resonate at the same frequency, and the distance constraint found by the Overhauser enhancement of this peak will generate a constraint on the mid-point of the three hydrogen atoms. Some force fields for protein structural NMR (Gunter and others 1997) use an imaginary atom in that place to mark the extremity of the distance constraint. PSICO handles these atoms in the same way as it handles the hydrogen atoms, by shifting the constraint to the nearest atom in the model and loosening the boundaries to account for the displacement.

5.9.2 Global overlap check ignores bonded pairs

The closest atoms in any molecular structure are those that are covalently bound together, with a centre to centre distance of approximately 1-1.5Å. This distance is below the effective precision of the constraint processing stage, which is most efficient when stopping at a domain size of approximately 2.5Å. So bonded atom pairs would not only be the major contributors to detected domain overlaps, they could also mislead the algorithm by detecting a failure in a situation where the algorithm could complete the enumeration to the desired precision.

For these reasons, Algorithm 5-7 ignores any atom covalently bound to the atom under examination.

5.9.3 Out Constraints for overlap are propagated only in enumeration

There is one Out Constraint implicitly connecting each atom pair, because two atoms cannot occupy the same place. This is a large number of constraints: $[N \times (N-1)]/2$, with N the number of atoms, and it is not practical to propagate each of these constraints every time the domain of an atom is changed.

To mitigate this problem, PSICO only propagates these Out Constraints during the enumeration of an atom, and only for the atom being enumerated.

5.9.4 Sine curves and intersections

The trigonometric functions Sine and Cosine are very use in the group propagation algorithm. These are common functions, implemented by the compiler or a mathematical library, that calculate the value of the trigonometric function up to the precision of the floating point representation used. Since this precision is orders of magnitude beyond what is necessary for the applications of PSICO, we can improve performance by trading off precision for speed. The Taylor expansions for the Sine and Cosine functions are:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} \ldots + (-1)^n \frac{x^{2n+1}}{(2n+1)!} + \ldots$$
$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} \ldots + (-1)^n \frac{x^{2n}}{(2n)!} + \ldots$$

The algorithm calculates the power and factorial terms cumulatively and interrupts the iteration cycle when it reaches the desired precision, which is possible because of the alternating sign of the higher order terms. After trigonometric function evaluations, calculating intersections of sine curve segments accounts for most of the computation effort in group and torsion angle constraint propagation. Two quick intersection checks improve performance by ruling out a most cases where segments don't intersect. One check applies to the whole line defined by equation (5.5). From the first equation in (5.7) we can tell that:

$$A \le \left| A_i + A_j \right|$$

From this equation, we know the lines cannot intersect if:

$$\left| \frac{B_i - B_j}{A_i + A_j} \right| \ge 1$$

This verification does not require the calculation of A or α.

The other intersection check uses piecewise linear functions as upper and lower bounds for the sine line. If the regions limited by the linear bounds of two sine lines do not intersect, then the lines cannot intersect either.

To define each linear boundary function and to compare them we need a set of points indicating the extremities of the segments forming the function. Figure 5-14 illustrates the calculation of these points.

P_1 and P_2 are the extreme values of the function. The maximum of the sine function is at $\pi/2$; subtracting the phase α we obtain $P_1(\pi/2-\alpha+2n\pi;L+A)$, where n is an arbitrary integer. The minimum value occurs 180° from the maximum, or $P_2(3/2\ \pi-\alpha+2n\pi;L-A)$. The points P_3 and $P_{3'}$ lie at the zeroes of the sine term of the function: $P_3(\pi-\alpha+2n\pi;L)$, $P_{3'}(2n\pi-\alpha;L)$.

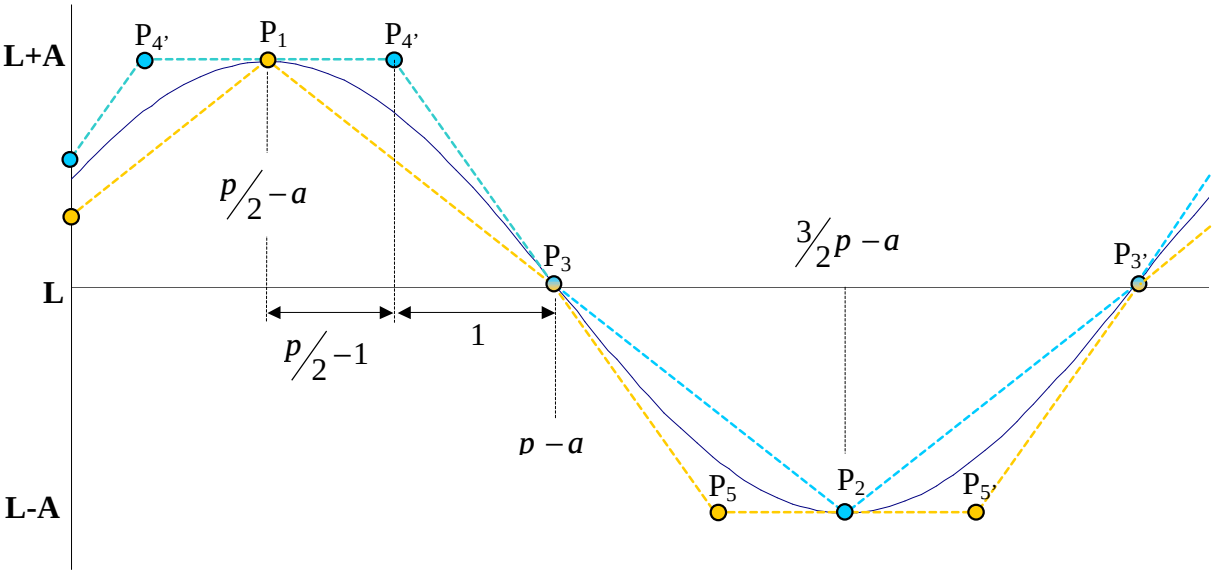


Figure 5-14 Piecewise linear upper and lower bounds (blue and orange, respectively) for a sine line. See text for how to calculate the points.

Points P4 and P4' lie on the intersection of the line y=L+A (the horizontal line crossing the maximum value of the function) and the derivatives of the function at points P_3 and $P_{3'}$ respectively. The coordinates are simple to calculate: the derivative at P_3 is –A, and the ordinate of P_4 is the ordinate of P_3 plus A, so the abscissa of P_4 is equal to the abscissa of P_3 minus one. For P_5 and $P_{5'}$ the reasoning is similar. The following sets of points define the linear boundary functions:

$$UB = \{(1-a; L+A), (p-a-1; L+A), (p-a, L), (\tfrac{3}{2}p-a; L-A), (2p-a; L)\}$$
$$LB = \{(\tfrac{p}{2}-a; L+A), (p-a, L), (p-a+1; L-A), (2p-a-1; L-A), (2p-a; L)\}$$

We can add $2n\pi$ to the abscissa values to extend these functions to all real numbers, but in general only the interval $[0;2\pi]$, or a sub-interval of this, will be relevant. When considering only an interval for the abscissa, we must consider not only the points inside the interval, but also the intersection of the boundary function lines with the extremities of the interval. The two leftmost points (unlabelled) on Figure 5-14 illustrate the result of this simple procedure.

Given two sine line segments A and B, defined in intervals $[A_1;A_2]$ and $[B_1;B_2]$, the algorithm will select all points in the four linear boundary functions for these sine lines (two boundary functions for each segment) with an abscissa value contained in the intersection of $[A_1;A_2]$ with $[B_1;B_2]$. Then it will evaluate the linear boundary functions at all these points plus the two extreme values of the intersection of the two intervals: the maximum of A_1 and B_1, and the minimum of A_2 and B_2. If the sine line segments intersect, there must be at least one value in the abscissas set for which the ordinate of the upper boundary function of each line is at least as large as that of the lower boundary function of the other.

At the intersection point, this is necessarily true, since the sine lines have the same value, but the linear boundary functions are not necessarily evaluated at this point. Nevertheless, if the condition holds anywhere within the interval of the linear segments, it must hold at least at one of the extremities of the interval. Algorithm 5-11 outlines the procedure.

```
Input:
BoundsA, BoundsB: points defining linear boundaries for lines A and B
Low, High: extremities of the interval to check for intersection

Abscissas={Low, High}
Add to Abscissas all points in BoundsA contained in ]Low;High[
and add all points in BoundsB contained in ]Low;High[ and not in Abscissas
Intersect=False
For all Abcissas[n] do
  Intersect=
     (Eval(BoundsA.Up,Abscissas[n])>=Eval(BoundsB.Low,Abscissas[n]) And
     (Eval(BoundsB.Up,Abscissas[n])>=Eval(BoundsA.Low,Abscissas[n])
  If Intersect the Break
EndFor
Return Intersect
```

Algorithm 5-11 Using piecewise linear boundary functions to determine if two sine line segments intersect.

5.9.5 Choice of pivot point

The geometric centre of the group is the ideal rotation pivot (see section 5.3) to minimise the loss in pruning power due to the discrete rotation search on χ and φ (see Algorithm 5-5). The farther away from the pivot, the greater the uncertainty region resulting from the same interval in the rotation angle, and placing the pivot at the centre results in the minimum average uncertainty for all atoms.

However, choosing one atom as a pivot results in one less sine line to process. The atom chosen as a pivot will generate the same limits independent of the orientation of the group, resulting in straight horizontal lines instead of the sine lines seen in Figure 5-8. This is an advantage for small groups, especially those with only three atoms, where the uncertainty due to the discrete rotation search is small and the elimination of one sine line can speed up calculations significantly.

6 The BiGGER Algorithm

> "Can you do addition?" the White Queen asked. "What's one and one and one and one and one and one and one and one and one and one?" "I don't know," said Alice. "I lost count."
>
> Lewis Carroll, Through the Looking Glass

Part of the motivation for a docking algorithm came from experience with protein complex modelling (Palma and others 1994), which made clear the greatest difficulty in predicting the structure of a protein complex: the large number of possibilities to examine. The objective of BiGGER (Bimolecular complex Generation with Global Evaluation and Ranking) is to eliminate a large fraction of these possibilities and retain a manageable set of likely models for the researcher to examine.

For every model generated, there is always the question of how many possible configurations were left unexplored. This concern determined the development of BiGGER as an exhaustive search algorithm instead of a random sampling algorithm like those based on simulated annealing or genetic algorithms (Goodsell and others 1990; Morris and others 1996,1998; Gardiner and others 2001). The starting point for BiGGER was the work of Katchalski-Katzir (Katchalski-Katzir and others 1992) on surface recognition using fast Fourier transform (FFT) techniques to calculate correlation matrixes. There are now several methods derived from this concept (representative examples are: Gabb and others 1997; Ritchie and Kemp 2000; Chen and Weng 2002), and FFT techniques are now a common standard approach to protein docking.

The advantage of FFT is the time complexity for the correlation of two three-dimensional matrixes, which is $O(N^3 \times Log_2 N^3)$, compared to $O(N^6)$ for a real space correlation. If one uses the correlation matrix to determine the configurations of greater complementarity, FFT is clearly the best choice. However, the docking problem is not to compare the three-dimensional shapes, but the surface contacts – a two-dimensional search space. Moreover, the objective is to retain a small proportion of solutions. These suggested that a better approach could be to use a real-space search, instead of a Fourier-space correlation, and use heuristics to reduce the search space. Finally, FFT calculations on large matrixes, for high-resolution docking, require considerably more memory than was readily available in personal computers in 1995, when work on BiGGER begun.

This led us to the Boolean Geometric Interaction Evaluation algorithm (BoGIE), the geometric surface contact filter that is still the core of BiGGER (see section 6.2). Using the bounds on the minimum scores of models to retain, the geometric filter can prune the search space and achieve an average time complexity of approximately $T(n^{2.8})$, which is significantly better than FFT (Palma and others 2000).

A more recent method based on geometric hashing, an algorithm originally developed for computer vision (Lamdan and others 1988; Wolfson and Rigoutsos 1997), may replace the FFT approach for its efficiency in finding shape complementarity (Noussinov and Wolfson 1999). However, it is still not possible to say if this approach is better than the one described in this work, and since, at present, the evaluation of the models retained takes nearly as long as the geometric filtering, the efficiency of this stage is no longer critical.

Though initially successful with the original test case, the then recently determined structure of desulforedoxin (Archer and others 1995), BoGIE was unable to predict complexes that did not have the exceptional shape complementarity of the desulforedoxin dimer. With the introduction of other interaction scores, soft docking, side chain contact filtering, global evaluation of models and, more recently, constrained docking, the algorithm developed into BiGGER (Palma and others 2000; Krippahl and others 2003).

This chapter explains the current implementation of the BIGGER algorithm. Except from the constrained docking methods (section 6.5), the basic algorithm is the same as found in the release version of Chemera and BiGGER (Krippahl and Palma 2001). However, several parameters on the side chain contact filter and on the scoring of the selected models have been adjusted, as discussed in chapter 9.

6.1 Sampling the Search Space

BiGGER generates candidate complexes by an elimination process, which starts from a set of hundreds of billions of possible configurations and ends in a few thousand possible models to be scored and ranked. The main criterion for this elimination is the geometric complementarity between the two molecules.

There are two reasons for this choice. One is that the area of contact is a good predictor of protein binding. Protein interactions are non-covalent, and generally weak, so a large contact surface is

often necessary to keep two proteins together. The exception is Hydrogen bonding, which is a strong binding factor; however, the number of Hydrogen bonds formed between two proteins correlates well with area of contact (Betts and Sternberg 1999). The other reason is that surface contact can be evaluated efficiently, which is essential to process the large set of possibilities in a reasonable period.

BiGGER samples the rotation space by rotating one protein (the probe) relative to the other protein (the target) in small steps, typically of 15º. For each orientation of the probe, the algorithm samples the translation space by moving the probe molecule relative to the target in steps of 1Å.

6.1.1 Using Cylindrical Symmetry

Though BiGGER can only model complexes between two partners, it is possible to extend the algorithm to complexes of more than two proteins if these can be reduced to a single pair by symmetry. This is the case for complexes with cylindrical symmetry. In such complexes, it is enough to dock two adjacent partners and then reconstruct the complex by rotation around the symmetry axis.

The method is to generate a set of axes for rotation in all orientations on the two dimensions of the polar coordinates. For each of these rotation axis, the probe is rotated around the axis by a value of 360º divided by the number of monomers in a complex. For a trimer, for example, this would be 120º rotations. The translation is then restricted to a plane perpendicular to the rotation axis. This procedure ensures that all complexes generated will have a cylindrical symmetry around the axis chosen, and that the order of the symmetry will correspond to the number of monomers in the complex.

Recently implemented, this algorithm was used to predict CAPRI target 10 (Janin 2002, Janin and others 2003), the trimeric form of the TBEV envelope protein. At the time of writing the results have not been disclosed.

6.2 The Geometric Filter

The geometric filter evaluates the atomic overlaps and surface contacts for each of the 10^9 models sampled. For this evaluation, we can imagine each structure to be represented as a three dimensional grid of cubes, indicating surface and core regions. As we will see, this is not the actual implementation, but is an easier way to explain the basic algorithm. The estimator for surface

contact is the overlap of surface cubes of one grid with the overlap of surface cubes of the other. The overlap of core regions indicates atomic collisions, and the geometric filter rejects any model in which core regions overlap. The geometric filter will retain the models with the greatest surface region overlap (the surface contact score).

The number of models retained is predefined, with the default being five thousand; a reduction of six orders of magnitude in the sample size.

The geometric filter operates at each step of the rotation search. The first task is to represent the proteins in a way that makes it easy to calculate surface contacts. BiGGER creates a three-dimensional grid of cubic cells representing each structure. Each cell corresponds to a cubic volume of $1Å^3$ and has a value of 1 if the centre of the cube is occupied by an atom, a value of 0 otherwise. Each atom occupies a spherical volume of radius equal to the atomic Van der Waals radius plus 1Å.

Algorithm 6-1 generates this first grid and, from it, the Surface and Core grids that will be used by the geometric filter.

```
Clear FirstGrid, SurfaceGrid, CoreGrid
For f=1 to number of Atoms do
  Fill FirstGrid cells closer to 1Å+Atoms[f]
For ShiftVector=(-1,-1,-1); (-1,-1,0) … (1,1,1), except (0,0,0), do
  Copy FirstGrid to CopyGrid
  Shift CopyGrid by ShiftVector
  SurfaceGrid=SurfaceGrid OR (CopyGrid XOR FirstGrid)
SurfaceGrid=SurfaceGrid AND FirstGrid
CoreGrid=FirstGrid AND NOT SurfaceGrid
```

Algorithm 6-1 Generating the Surface and Core grids

A surface cell is a cell with a value of one and with at least one neighbour with a value of zero, and the Surface grid represents these cells. Algorithm 6-1 uses Boolean operations to determine if a grid cell has empty neighbours by shifting a copy of the first grid relative to the original, and combining with an inclusive Or (OR) the results of an Exclusive Or (XOR) operation between the original grid and all 26 one-cell displacements of the copy. The XOR operation gives a value of one if a cell is combined with a neighbour with a different value, so combining these results with an OR operation will give a value of one for all cells with at least one different neighbour. After the loop over the 26 displacements, the Surface grid contains a double surface shell, which is then trimmed by an AND operation with the original grid. The Core grid is simply the remainder of the original

grid, calculated by an AND operation between the original and the inverse of the Surface grid (the NOT operation converts zeroes into a ones, and ones into zeroes). Figure 6-1 illustrates this step.

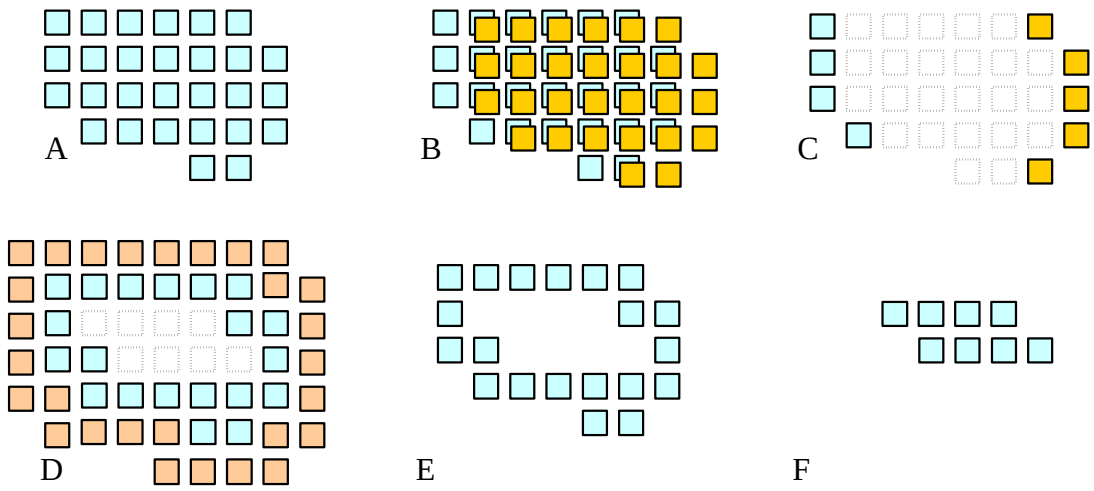


Figure 6-1 Using Boolean operations to determine the surface and core regions. Panel A shows the initial grid. A copy of this grid is shifted one cell to the right (B) and combined with the original using an XOR operation (C). Panel D shows the total (using OR operation) of all XOR combinations. Panel E shows the trimmed surface grid, and F the core grid. Filled squares represent grid cells with a value of one.

Figure 6-2 shows an example of the grids generated for a protein structure, to give an idea of the relative scales of the grid cells and the typical structures used in BiGGER.

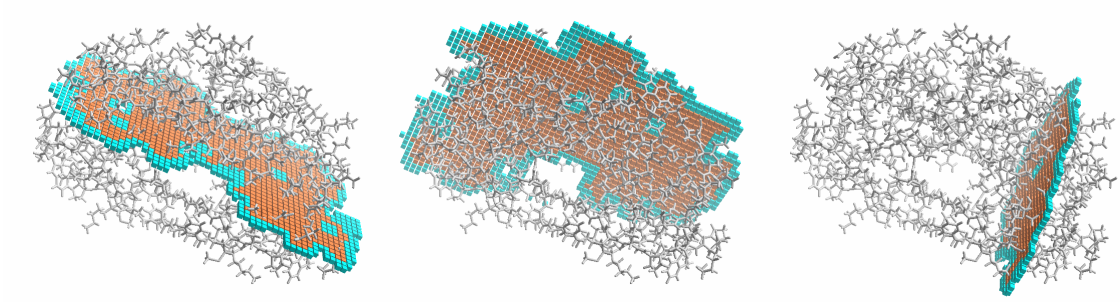


Figure 6-2 The surface and core grids (blue and orange, respectively) for the HIV protease structure (Wlodawer and others 1989). The three panels show sections of the grid in three orthogonal planes.

After Algorithm 6-1 generates the surface and core grids, these are converted from a binary grid representation into a two-dimensional array of grid segment lists. Each grid segment consists of a pair of grid coordinates indicating the low and high coordinate of a contiguous line of ones in the grid. Figure 6-3 illustrates this conversion for a two dimensional example.

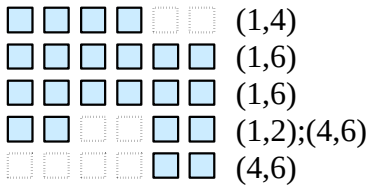


Figure 6-3 Converting a binary grid into a segment array. The binary grid represented by blue squares blue and dashed squares (one and zero, respectively) is converted into an array of segment lists. The fourth segment list contains two segments, (1,2) and (4,6), to represent the two disjoint segments on the fourth line of the grid.

This representation simplifies the calculation of grid contacts, as the number of overlapping cells can be determined from the relative positions of the extremities of the overlapping segments. Similarly, it makes it easier to determine which relative positions of the two docking partners cause the core regions to overlap.

For each orientation of the probe molecule, the geometric filter searches the possible configurations by shifting the two grids of the probe (surface and core) relative to the grids of the target in the x, y, and z directions. This is done in three nested loops. The outer loop searches the z direction. The second loop will search all possible displacements in the y direction for each position in the z direction. Finally, the inner loop will search through the x direction. In the segment array representation, the segments are aligned in the x direction, because it is the inner loop in the x direction that evaluates the overlaps.

Any configuration in which core grids do not overlap is evaluated for surface grid overlap, and the total number of overlapping surface cells is the surface contact score. This model is added to the set of models to retain if the set is not yet filled, or if the score is higher than the lowest scoring model in the set, in which case the new model found replaces the old one.

6.3 Soft Docking

In general, the conformations of surface side chains in an unbound protein structure differ significantly from those of the bound structure, whether because of their mobility or because of uncertainty in the determination of the structure. If this results in excessive overlaps in the correct models for the complex, these will be excluded during geometric filtering.

To solve this problem, BiGGER includes a soft docking option that eliminates the core regions of parts of the more mobile side chains from the grid representation. Figure 6-2 shows the difference in the grids used for hard and soft docking.

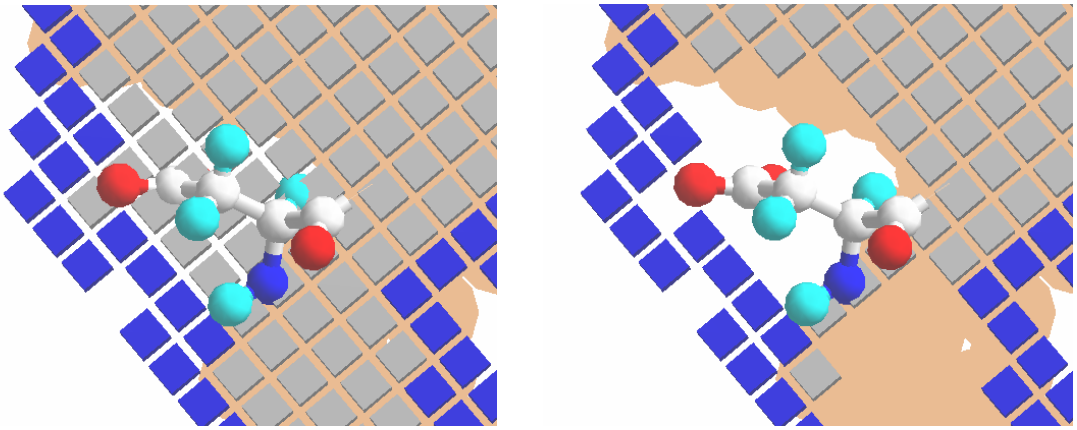


Figure 6-4 Difference in the grids when using hard (left panel) or soft docking (right panel). Grey cubes represent the core grids, blue cubes the surface grids. Note that the surface grids are identical, but the core grid in the soft docking option is empty in all cells spanned by the side chain of the aspartate shown in the centre of each panel.

The surface grid is unchanged, so models have the same surface overlap score in both methods. However, soft docking allows some models that hard docking would rule out due to core grid overlaps. By creating these gaps on the more mobile surface side chains, soft docking makes the geometric filter less sensitive to side chain conformation changes.

The downside of this approach is that it also allows incorrect models that the geometric filter would otherwise eliminate, and these models may drive the correct models out of the list of the best geometric fits. To mitigate this problem, BiGGER also filters any candidate model according to a statistical analysis of side chain contacts.

6.4 The Side Chain Filter

Perhaps the greatest advantage of real space docking algorithms relative to Fourier space methods is the efficiency with which they can use additional criteria to complement the surface contact filtering. While Fourier transform methods must examine all possible contacts for all filters used (Gabb and others 1997), real-space methods can apply the additional filters only to the small subset of models that were selected by surface contact.

The filter implemented in BiGGER is a two dimensional Self Organized Feature Map (Kohonen 1982, in Haykin 1999) with 10x10 neurons. It was trained to classify docking models according to the side chain contacts between 5 disjoint classes of amino acids grouped by structural and chemical similarity (Table 1, page 153).

After training with correct and incorrect models obtained by docking simulations on 88 complexes (sub-section 9.1.3), each neuron was labelled with the number of correct and incorrect examples in its cluster, using the examples on the training set. The response of this neural network classifier is the label of the closest neuron to a given example, and the filter rejects the examples that fall in clusters that contained only incorrect models in the training set.

6.5 Constrained Docking

The method for geometric filtering allows us to impose constraints in a flexible way. Most docking algorithms, with the possible exception of those based on Fourrier transforms, allow the user to restrict the search to specific regions in space. However, experimental data is seldom this specific. Often, amino acids found to be crucial for complex formation (as in site directed mutagenesis) or affected by the other partner's presence (as in NMR titration), are not part of the interface, but are important for other reasons, such as conformation changes.

To use a wide range of experimental data we need a flexible algorithm to constrain the configuration search; an algorithm that can take into account the uncertainty associated with most experimental data. The constrained docking algorithm must be able to interpret any number of constraints of the form:

At least N atoms or residues of set A must be within distance R of atom set S Constraint 6.1

The most important feature is the cardinality constraint, which specifies the number of atoms or residues that must meet the distance requirement. This is what gives the constraint the necessary flexibility to take advantage of most experimental data.

It is simple to test this constraint for a given configuration to evaluate models according to these contact counts or other measures, and section 7.3 shows how Chemera implements these evaluations to score the models BiGGER retains. This section will explain how to apply these constraints to the geometric filter and restrict the search to those configurations the experimental data allows. The advantages of reducing the search space are twofold: more efficient docking and less interference from incorrect configurations that may exclude desirable models from the set of models BiGGER retains.

In section 6.2, we saw that the geometric filter searches the translation space in three nested cycles, spanning the z, y, and x coordinate, from the outer to the inner cycle. To reduce this search

we need an algorithm that can return the allowed values of the displacement in the z direction, the allowed values in the y direction for each allowed value of the z cycle, and for each allowed (y,z) pair, the allowed values of x. In short, the algorithm must generate domains in one dimension.

To understand the algorithm, let us first first note two features of constraint 6.1. One is that it distinguishes the two groups of atoms, for the cardinality constraint only applies to group A (at least N of these atoms must be within distance R of any atom in group B). The other is that the distance R is the same for any atom. This allows us to generate a region around the atoms of group B where the atoms of group A meet the requirements for the distance constraint, and the cardinality constraint becomes a matter of counting the number of atoms of group A in this region. Since we need to define the domains in one dimension at a time, and these domains are sets of integer values corresponding to the displacement of one grid relative to the other in grid coordinates, we can describe the allowed set of displacements as list of segments similar to those encoding the structure grids (see Figure 6-3). Thus, the task is reduced to generating a list of segments representing the displacements for which at least N atoms of group A are inside the segments defining the neighbourhood R of the atoms in group B.

To calculate the displacement values that place the atom of group A inside the neighbourhood of group B we only have to shift the segments defining the neighbourhood of B by the coordinate value of the atom. For example, an atom at coordinate 5 will leave the neighbourhood segment that ends at coordinate value of 11 when the displacement value becomes greater than 11-5=6. Figure 6-5 illustrates this process.

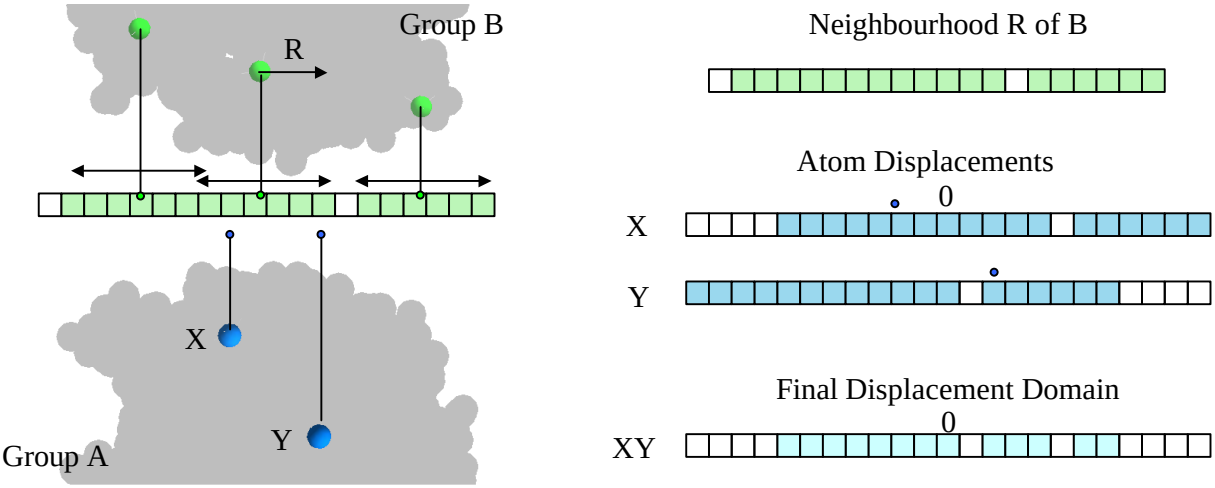


Figure 6-5 Generating the displacement domain in one dimension. The left panel shows the generation of the neighbourhood of radius R of group B. The panel on the right shows the allowed displacements for each atom, and the final displacement domain for an N of 2. See constraint 6.1 and text for more details.

Once we have the displacement segments for all atoms, we must generate the segments describing the region at least N atoms are in the neighbourhood of B (see constraint 6.1). A naïve algorithm would be to count the atoms for each possible displacement value, but the number of operations in this case would be in the order of the number of atoms multiplied by the number of displacements to consider.

The total number of displacements is twice the span of the coordinates of the atoms in group A plus the span of the neighbourhood of B minus two. This is the number of possible grid displacements, from placing the higher atom on group A to the lowest cell on the neighbourhood of B, to placing the lowest atom in the highest cell. Since this can be a large number, cycling through all atoms for all displacements can be inefficient.

However, there is a better alternative. First, we use an array to represent all the displacements. This is an array of integer values, where the index represents the displacement. So cell number five, for example, would correspond to a displacement of five grid cell units. On this array we mark the entry point of each atom by incrementing the array on the cell corresponding to the low coordinate of each displacement segment for each atom, and the exit point by decrementing the cell immediately after the high coordinate for the displacement segment. With the array initialised at zero, if the first segment has the coordinate values (3;8), this means the array will now have a value of one at the cell corresponding to a coordinate value of three, and a value of minus one at the cell corresponding to the coordinate value nine.

After doing this for all atoms, all we need is to run through the displacements array once, from the lowest coordinate index to the highest, and accumulate the values found in the cells. All cells where the accumulator is equal to or greater than N are added to the segments list that defines the domain where constraint 6.1 holds. Figure 6-6 illustrates this computation.

This procedure can also combine any number of constraints of the form of Constraint 6.1. Furthermore, it is not limited to consider a lower limit on the number of instances where the constraint holds, but can be used to specify an exact value, an upper limit, or a range of values too. Though this may not be useful for the case of Constraint 6.1, for there will seldom be a need to specify an upper limit to the number of atomic contacts, it may be useful in combining several such constraints in a way that selects some configurations with specific interface regions while ruling out others. For example, if we suspect that a particular complex has two alternative forms with two

different contact regions, BiGGER can model the two regions as two constraints of the form of Constraint 6.1 and restrict the search to the regions where exactly constraint one holds true.

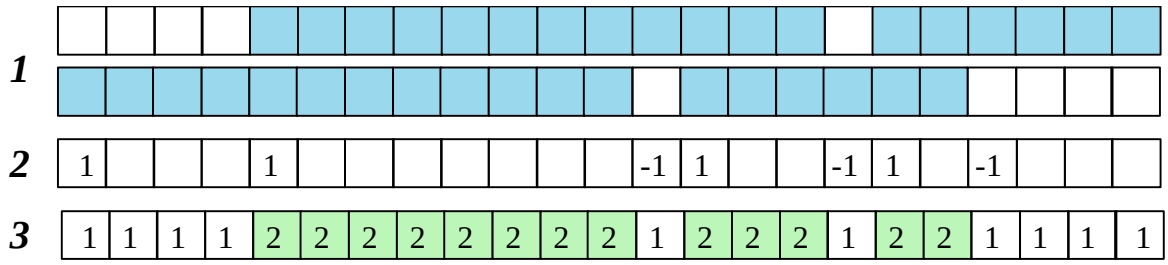


Figure 6-6 Computing the domains for Constraint 6.1 from the regions where each atom respects the distance constraint. Step one shows the allowed regions for two atoms. Step two marks the entry and exit points on the displacement array. Step three runs through the displacement array from left to right, adding the value on each cell to the value on the accumulator. The region where both atoms respect the distance constraint is shown in green.

This flexibility allows BiGGER to integrate a wide range of data and to test different assumptions about the system to model. Algorithm 6-2 outlines the procedure for generating the displacement domain from one constraint of the form of Constraint 6.1.

```
Input
  Allowed: Segment list with the neighbourhood of  Group B
  GroupA: array of coordinate values for Group A
  N: number of atoms in A that must be within R of B
Set length of array Coords to span of Allowed plus 2×Span of GroupA  - 2
Fill array Coords to 0
For each atom j in GroupA do
  Segments=Allowed
  Shift all Segments by the coordinate value GroupA[j]
  For each segment s in Segments do
     Increment Coords[Segments[s].Low]
     Decrement Coords[Segments[s].High]
Accum=0
Domain=empty list of segments
For f=Low(Coords) to High(Coords) do
  Accum=Accum+Coords[f]
  if Accum>=N then add displacement f to Domain
Return Domain
```

Algorithm 6-2 Determines the domain of a displacement variable for which a minimum number of atoms respect a distance constraint. See Constraint 6.1 and text for more details.

6.6 Optimising the Geometric Filter Algorithm

The geometric filter retains only the models with the highest surface overlap score. If we can predict that a region in the translation search space cannot produce models with a higher score

than the lowest scoring model in the set, the algorithm can skip that region. This has no effect on the results, only on the efficiency of the computation.

The first step in the optimisation is to calculate the total number of filled surface grid cells in each slice on the x;y plane as a function of the z coordinate. Searching through the z coordinate, we can use this information to check if the surface overlap can be greater than the minimum value on the set of models to retain. To do this, we compare the number of surface grid cells in each xy slice of the target and compare it with the corresponding slice on the probe, for a given displacement on the z axis. We add the minimum value for each pair of slices, and this gives us an upper bound for the surface overlap score of any model with this z displacement. Figure 6-7 illustrates the procedure in two dimensions, replacing each xy slice with a single line for clarity.

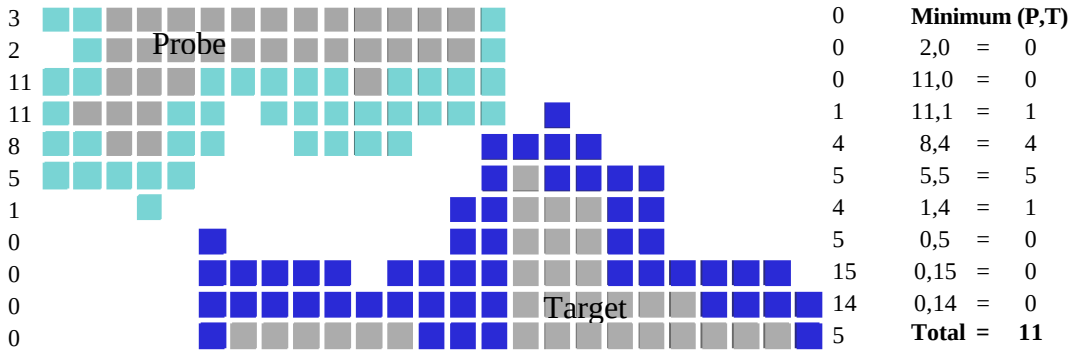


Figure 6-7 Calculating the minimum overlap, for the surface contact score. The probe grids (core in grey, surface in light blue) and the target grids (core in grey, surface in dark blue). The vertical alignment represents a given displacement value for the vertical axis. The surface cell totals for the surface grids are in the two columns of numbers adjacent to the grids. The rightmost column shows the minimum value for each row. The total of 11 means that no model obtained by searching the horizontal displacement can have an overlap score greater than 11 for this vertical displacement.

A similar procedure restricts the search in the y direction for each z displacement value, the only difference being that the values compared are the surface grid counts for the corresponding (y,z) grid rows in the given (y,z) displacement.

To improve efficiency in the early stages of geometric filtering, when the set of models retained contains many low scoring models, we can impose a higher minimum contact score for an acceptable model. Chemera uses a default value of 300 for this parameter, because we can safely predict that the lowest geometric contact score in the models retained will be greater than this, and so this will only speed up the calculations without affecting the results.

6.7 Scoring

Evaluating the selected models is the last task of the docking algorithm. The method evolved from the calculation of electrostatics and solvation parameters, to an aggregate function that was a linear combination of these two plus surface contact and side chain contact statistics, to the present form using a neural network to combine these four scores with a count of contacts between five amino acid groups. This section describes the algorithm as implemented in the release version (Palma and others, 2000). A new scoring method is under development, based on the results described on chapter 9.

6.7.1 Side Chains Contact

The side chains contact score is a measure of affinity of the amino acids at the interface. It is the sum of the frequency of the contact minus the expected frequency if there was no affinity. The frequencies are estimated from a database of high-resolution X-Ray structures (Singh and Thornton 1992), and the contribution of each side chain contact is

$$S_{AB} = \left(\frac{N_{AB}}{N_{AX}} - \frac{N_B}{N_X} \right) \qquad (5.12)$$

where S_{AB} is the contribution of the contact of A with B, N_{AB}/N_{AX} the fraction of AB contacts in all contacts involving A, and N_B/N_X the frequency of amino acid B in the database. For random contacts these fractions have the same expected value. If the AB contact has a higher proportion, this indicates an affinity between A and B, and the score contribution is positive. A lower AB contact proportion indicates an unfavourable interaction and results in a negative score.

6.7.2 Solvent Exclusion

Solvation effects are an important factor in complex formation. To quantify these effects for each model, BiGGER calculates the area of the surface accessible to the solvent that was lost when the complex formed. Each amino acid is divided into several fragments, and each fragment has parameter specifying the solvation energy per surface area unit (Wang, Zhang, Scott 1995; Wang, Zhang, Li and Scott 1995).

The area calculation is simplified, for this interaction score is only one of several that must be calculated for all 5000 models in each docking simulation. For each fragment in one docking partner that is close to other fragments in the other docking partner, the surface variation is equal to the cross section of the other fragment if these fragments are in contact, or zero if the fragments

are at a distance greater than 2.8Å (the diameter of a water molecule). If the distance is between zero and 2.8Å, the surface variation is calculated by a linear interpolation.

6.7.3 Electrostatic Interaction

To estimate the electrostatic interaction between the two docking partners, BiGGER uses the AMBER force field charges (Wang and others 2000; Cornell and others 1995) and a modified Coulomb model:

$$V_e = \sum_{i,j>i} k \frac{q_i q_j}{(r_{ij} + c)^2} \qquad (5.13)$$

where q_i and q_j are the atomic charges, and r_{ij} the distance between atom i and atom j. The factor k is a scaling factor that includes the dielectric contributions and the conversion to Cal/mol, and is 331.98 e^{-2} $Å^2$ Cal mol^{-1}.

One modification is to use a relative dielectric value proportional to the distance between the atoms, which makes the electrostatic potential inversely proportional to the square of the distance. This technique simulates the different dielectrics (water and protein) in solution, and is used in some force fields (Halgren 1996). The other modification is to add a constant to the distance measured. This constant (c) is 1.5Å, and serves to reduce the error from some atomic overlaps due to the soft docking algorithm, and to make the interaction score less sensitive to small differences in position that are below the precision of the docking algorithm.

6.7.4 Global Score

A neural network aggregates the four contact scores (surface contact, given by the geometric filter, side chain contacts, solvation, and electrostatics scores) and 15 additional side chain contact counts, between five categories of amino acids (see Table 1 on page 153 and chapter 9). These make an input vector with 19 values. The network is a fully connected, feed forward, neural network with 19 input neurons and one output neuron, and all neurons have a sigmoidal response.

The global score is calculated from the response of the network, and is the percentage of correct models found in all models in the test set (see sub-section 9.1.3) that elicited a similar or higher response from the network. For example, a value of 9.8 means that 9.8% of the models in the test set with the same or higher output from the network were correct models. A model was considered correct if the RMSD from the known structure was below 4Å.

7 Algorithms in Chemera

"Is that the truth?"

"No, but it's a lot simpler."

Walt Kelly, in Pogo

Much of modelling consists of simplifying reality. In molecular modelling, in particular, it is often necessary to sacrifice realism in order to visualise some aspects. Atoms are spheres, molecular structures are frozen, the solvent is a homogeneous background, and so forth. These representations do no justice to the complex dynamism of real molecules, but a realistic representation of a protein solution would be so complex as to be useless.

The aim of Chemera, like that of other molecular visualisation applications, is to show simplified aspects of molecular properties. Chemera is the interface for PSICO and BiGGER, but is more than just the front end for these two algorithms. Work on the first versions of Chemera (then Cyberzyme) began even before the work on the docking algorithm (Krippahl and Palma 1996), and one of its main purposes has always been to display protein structures and estimate and represent protein properties, in addition to the analysis of docking results. This chapter explains some of these tools and algorithms.

7.1 Structure Comparison

Chemera compares structures by calculating the root mean square deviation (RMSD) of two sets of atoms. The RMSD value is the root of the mean of the squares of the distances between corresponding atoms in the two sets, and the minimum RMSD value, obtained by superposing the two structures, is a standard similarity measure.

There are several closed-form solutions for comparing two sets of points in space (Eggert and others 1997; Horn B. 1987; Horn and others 1988; Walker and others 1991; and see Arun and others 1987 for a comparison). However, Chemera finds the rotation and translation coordinates minimising the RMSD using the conjugated gradient algorithm. The procedure is similar to the structure minimisation described in section 5.8, but without changing the torsion angles. The reason for this implementation in Chemera was expediency, since the algorithm for the minimisation was already implemented.

Figure 7-1 illustrates the results with the structures of desulforedoxin (Archer and others, 1995) and desulfoferrodoxin (Coelho and others 1996, 1997), which contains a domain that is structurally similar to desulforedoxin.

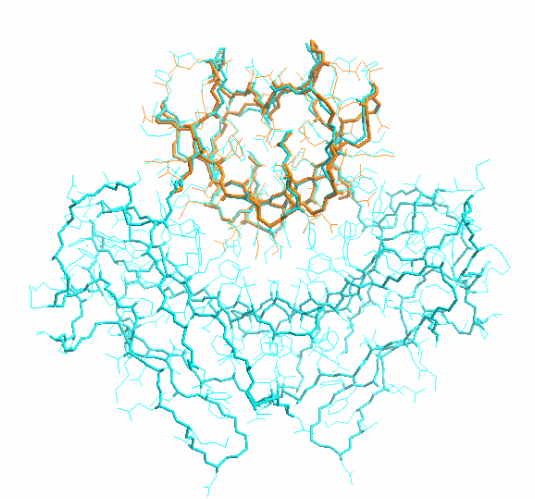


Figure 7-1 Desulforedoxin (orange) fitted to the desulforedoxin-like domain in desulfoferrodoxin (blue). Side chains are shown as thin lines, and the backbone as thick lines. The structures used were 1DXG (Archer and others 1995) and 1DFX (Coelho and others 1996, 1997).

When necessary, Chemera can also determine the best sequence alignment using the Needleman-Wunsch-Sellers algorithm (Needleman and Wunsch 1970; Sellers 1974). The user can also choose to calculate the RMSD on different sets of atoms, such as the α-carbons, backbone atoms, all atoms, and so forth.

7.2 Clustering

The function of the clustering algorithm is to form groups of similar complex models, from the set BiGGER generates. The similarity measure is the RMSD of the α-Carbons of the probe molecule, considering the target fixed in the same position for all models. This obviates the need for minimization, which is useful because the algorithm requires on the order of $n^2/2$ measurements.

The algorithm places in one group any model that is within a given RMSD from at least one element of the group. Algorithm 7-1 outlines this procedure.

```
Input
  Rmsd: threshold RMSD value
  Models: array model structures
For n=1 to Length(Models) do
  Models[n].Group=n
For n=1 to Length (Positions)-1 do
  For j=n+1 to Length (Positions) do
     If ProbeRMSD(Models[n],Models[j])≤Rmsd) then
        OldGroup=Models[j].Group
        For k=1 to Length(Models) do
           If Models[k].Group=OldGroup then
              Models[k].Group=Models[n].Group
        End For k
  End For j
End For n
```

Algorithm 7-1 Clustering similar docking models. Each model is assigned to a group if the probe RMSD to a member of that group is within the threshold limit.

Figure 7-2 illustrates the results, from a docking simulation of aldehyde oxidoreductase (Rebelo and others 2001) to a flavodoxin, from Desulfovibrio gigas. Section 10.2 has more details on this docking simulation.

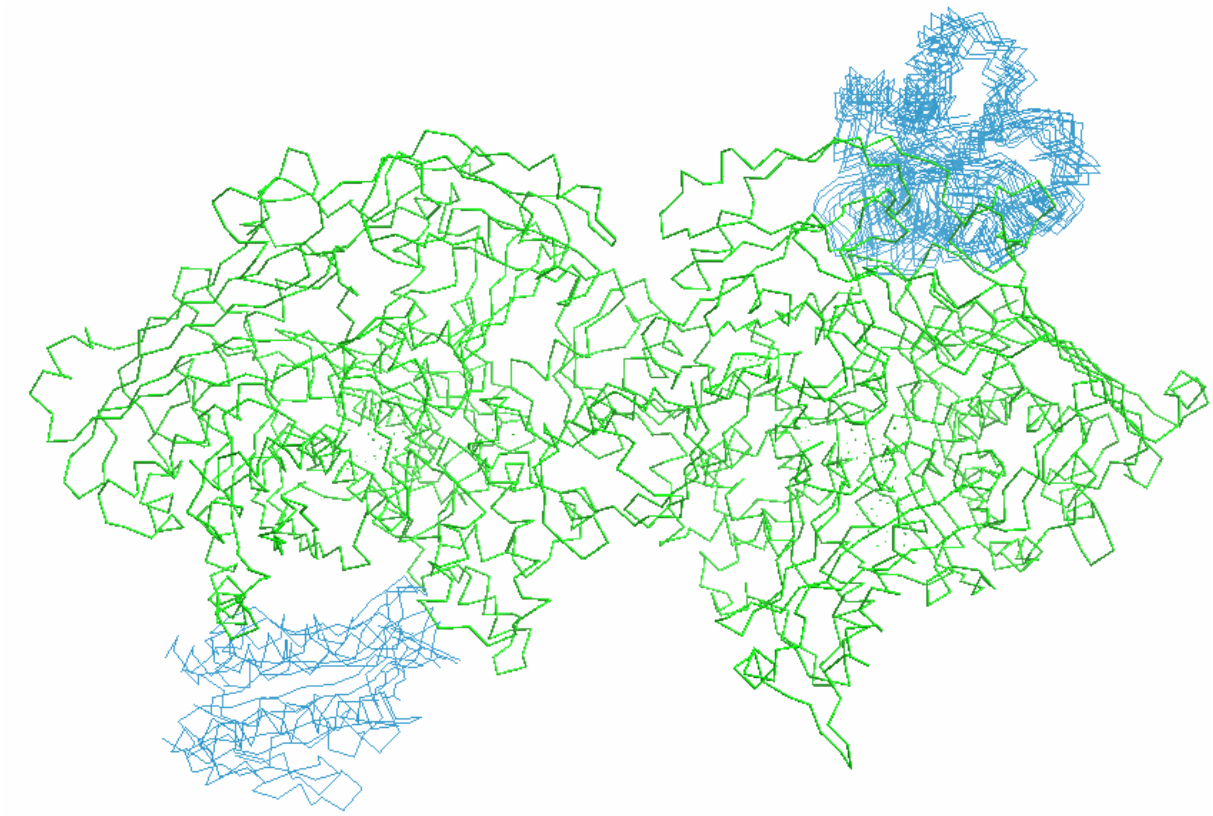


Figure 7-2 Two clusters of models for the docking simulation of Aldehyde Oxidoreductase (Rebelo and others 2001), in green, complexed with an homology model for flavodoxin.

7.3 Complex Symmetry Score

Proteins in living organisms are usually symmetric oligomers, due to evolutionary selection for stability, cooperativity, low solvent accessibility and other factors (Goodsell and Olson, 2000). The stability of these forms differs significantly from case to case, and often the only structure available is for the monomer.

The frequent need to model homodimers led to the implementation of the symmetry score in Chemera. As Algorithm 7-2 illustrates, this is a simple measure of interface symmetry, corresponding to the root means square deviation in the distances of atom pairs. For each atom pair $i_A j_B$ that are in contact, the algorithm adds the square of the difference between the $i_A j_B$ and the distance for the complementary pair $i_B j_A$. For the perfectly symmetrical interface, this difference is zero for all atom pairs, and the score is zero. A higher score means the interface is less symmetric.

At the moment, the algorithm can only handle dimeric structures, but the extension to higher oligomers is trivial, requiring only the identification of the correct atom pairs among the various protein chains.

```
Input
  MonA,MonB: Coordinates of the atoms in the two monomers
  ContactDistance: Threshold value for atomic contacts
Score=0
Count=0;
For i=1 to Length(MonA) do
  For j=i to Length(MonB) do
     If Distance(MonA[i],MonB[j])<ContactDistance then
        Score=Score+Sqr(Distance(MonA[i],MonB[j]-Distance(MonA[j],MonB[i])
        Count=Count+1
        EndIf
     EndFor j
  EndFor i
Score:=Sqrt(Score/Count)
```

Algorithm 7-2 Symmetry score

Figure 7-3 shows two models of a reconstruction of the desulfoferrodoxin dimer (Coelho and others 1996, 1997). The symmetric model has a score of 0.4Å, and deviates from the crystallographic structure by 0.3Å. The asymmetric model has a score of 19Å and deviates 16Å from the crystallographic structure of the dimer.

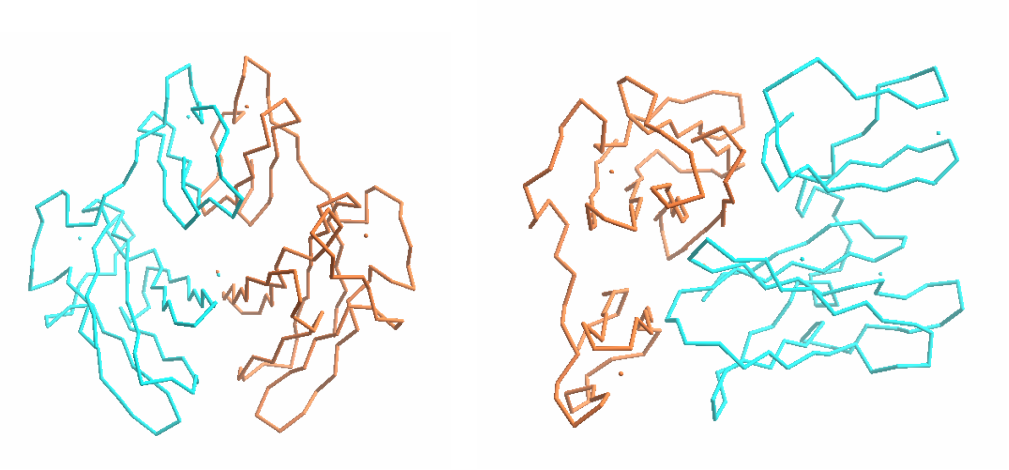


Figure 7-3 High and low symmetry complexes (left and right panels, respectively), from a docking simulation to reconstruct the desulfoferrodoxin dimer (PDB code 1DFX). Each monomer is shown in a different color.

7.4 Evaluating Docking Constraints

Chemera can score the models produced by BiGGER according to contacts or distances between atoms or residues. This is a useful and simpler alternative to constrained docking (section 6.5), allowing the user to try different scoring functions in the same set of models.

The scoring algorithm can score each model according to the minimum or average distance between selected groups of atoms or residues, or the number of atoms in contact, or the number of residues in contact. The user can specify the contact distances and the groups to consider, and so adapt this score to a wide range of experimental constraints, from NMR titration data to cross linking.

7.5 Electric Properties

Electromagnetic forces dominate all events at the chemical scale, where gravity is too weak and nuclear forces too short-ranged to be of significance. Chemera includes tools to estimate and visualise several aspects of protein electrostatics, using the atomic charges of the AMBER force field (Wang and others 2000; Cornell and others 1995), and to estimate the dynamics of electron transfer using a simple algorithm (Beratan and others 1987).

Chemera can display atomic or residue charges with user-defined colours, and estimate the dipole moment of a protein. The dipole moment estimate is the weighted sum of all the vectors from the geometric centre to each atom, with each vector weighted according to its charge. Figure 7-4

shows there representations for cytochrome 552 from Pseudomonas nautica, a small, polar protein (Brown and others 1999). Cythochrome 552 from P. nautica was crystalized as a dimer, but the figure only represents the monomer.

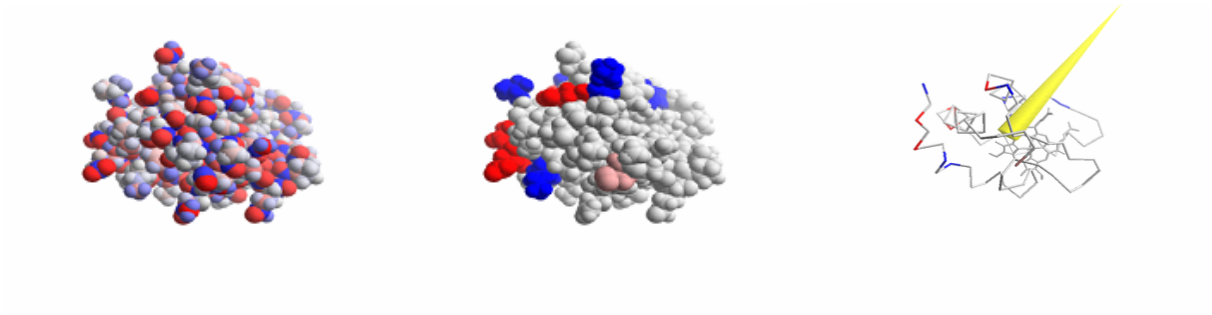


Figure 7-4 Three representations of the electrostatic properties of cytochrome C-522. The left panel shows the individual atomic charges (red for negative, blue for positive), the central panel shows the total charges for each amino acid residue, and the right panel shows the dipole moment vector of this protein.

Chemera can also calculate the electrostatic field surrounding a protein by solving the linear Poisson-Boltzmann equation with a multi-grid finite difference solver. The algorithm is based on the description by Klapper and others (1986), but with a correct implementation of the multi-grid solver.

The Poisson equation (7.1) is a differential equation that describes the special variation of the electric field (ϕ) as a function of the dielectric constant (ε) and the charge distribution (ρ):

$$\vec{\nabla} \cdot \left[e_{(x)} \vec{\nabla} f_{(x)} \right] - r_{(x)} = 0 \qquad (7.1)$$

The structure of a protein allows us to determine the charge distribution for the molecule, but not for surrounding ions in solution. To estimate the charge distribution due to these ions, we can use the Debye-Huckel estimate for the number of ions of one type (n_i) in a unit volume based on their average number per unit volume in the solution (n^0_i) and the electrostatic field in that region (ϕ). Equation (7.2) shows this relation, with k the Boltzmann constant and T the temperature (Debye and Huckel 1923).

$$n_i = n_i^o \exp^{-q_i f_{(x)}/kT} \qquad (7.2)$$

Since this is a Boltzmann distribution, the equation including the charge distribution due to the ions in solution is called the The Poisson-Boltzmann equation:

$$\vec{\nabla} \cdot \left[e_{(x)} \vec{\nabla} f_{(x)} \right] - r_{(x)} - \sum_i q_i n_i^o \exp^{-q_i f_{(x)}/kT} = 0$$

where $\rho_{(x)}$ is the charge density due to the known distribution of charges in the protein.

In physiological solutions, the ions are almost exclusively Na^+ and Cl^-. In this particular case, considering only two univalent ions of equal concentration and symmetric charges, we can rewrite the Poisson-Boltzmann equation as equation (7.3):

$$\vec{\nabla} \cdot \left[e_{(x)} \vec{\nabla} f_{(x)} \right] - q_i n_i \sinh(f_{(x)} / kT) - r_{(x)} = 0 \qquad (7.3)$$

For weak electric fields, we can linearize the Poisson-Boltzmann equation by ignoring the higher terms in the Taylor series expansion of the hyperbolic sine function:

$$\sinh a = a + \frac{a^3}{3!} + \frac{a^5}{5!} + \frac{a^7}{7!} + \ldots$$

and obtain equation (7.4).

$$\vec{\nabla} \cdot \left[e_{(x)} \vec{\nabla} f_{(x)} \right] - q_i n_i f_{(x)} / kT - r_{(x)} = 0 \qquad (7.4)$$

For the finite-difference approximation, we integrate all terms in small cubic volumes in a grid. The integral of the last term (ρ) is the sum of the charges from the protein that lie inside the cubic grid cell. We can estimate the volume integral of the second term by assigning to the grid cell a value for the Debye-Hückel parameter κ_0=qn/kT. This is zero on all cells inside the protein and a function of the ionic strength on all other cells, and the approximate value of the integral is:

$$\iiint \frac{qn}{kT} f_{(x)} d^3x \simeq \iiint k_0 f_0 d^3x = k_0 f_0 h^3$$

where h is the width of the cell.

Using Gauss's theorem, we can convert the volume integral in the first term into a surface integral and then calculate it by a finite difference formula. The integral becomes the sum over the six faces of the cube cell of the products of the dielectric constant associated with each face (ε_l), the difference between the field calculated for each neighbour and the cube cell (ϕ_i-ϕ_0) and the width of the cell (h):

$$\iiint \vec{\nabla} \cdot \left[e_{(x)} \vec{\nabla} f_{(x)} \right] d^3x = \iint e_{(x)} \vec{\nabla} f_{(x)} d^2A \simeq \sum e_i (f_i - f_0) h$$

Solving the complete equation for ϕ_0 — the field value in the grid cell — we obtain:

$$f_0 = \frac{\sum e_i f_i + q_0}{\sum e_i + k_0^2 h^2} \qquad (7.5)$$

The method described (Klapper and others 1986) uses a small initial grid to find an approximate solution for equation (7.5) at low resolution by the Jacobi relaxation method. This solution is interpolated to another grid with a higher resolution, this grid is relaxed to reduce the residual error, and then the process repeated until it reaches the highest resolution grid. According to Klapper and others 1986, the final solution is then obtained solely by Jacobi relaxation.

The algorithm implemented in Chemera is similar, but makes full use of the multi-grid approach. The Jacobi relaxation method applies the residual error — the difference between the field value at each cell and the result of equation (7.5) — directly to the field value in each cell. Since all cells change, all values for the neighbouring fields in equation (7.5) change, and there is a new, but smaller, residual error. This way, Jacobi relaxation slowly reduces the residual.

The multi-grid method not only uses the lower resolution grids to provide an initial guess, but also to estimate the best correction for each residual. The residuals are copied to the next lower resolution grid, which then determines the best correction to the higher resolution grid. The problem residual in the lower resolution grid is reduced in a similar way, using an even lower resolution grid to determine the best corrections to make. This way, the multi-grid algorithm improves convergence by using all the grids during all the calculation, and not merely to provide a first guess. See Press and others 1994 for more details on these numerical methods.

Chemera can display the calculated field in different ways. Figure 7-5 shows three representations, again for the monomeric form of cytochrome C552 from Pseudomonas nautica.

In addition to the electrostatic properties, Chemera can also estimate and display the dynamics of electron transfer within and between proteins. This is an important phenomenon, at the heart of photosynthesis, respiration, and many other parts of the metabolism of all living beings. Electron transfer in proteins is very hard to model, as the phenomenon can only be accurately described with quantum mechanics, which is very demanding for large systems like a protein. Furthermore, it is also dependent on redox potential differences, which depend on pH, mobility, local electric environment and other factors. A precise calculation of electron transfer pathways must take into account many factors, and is a complex and computationally demanding procedure (Balabin and Onuchic 1998, 2000).

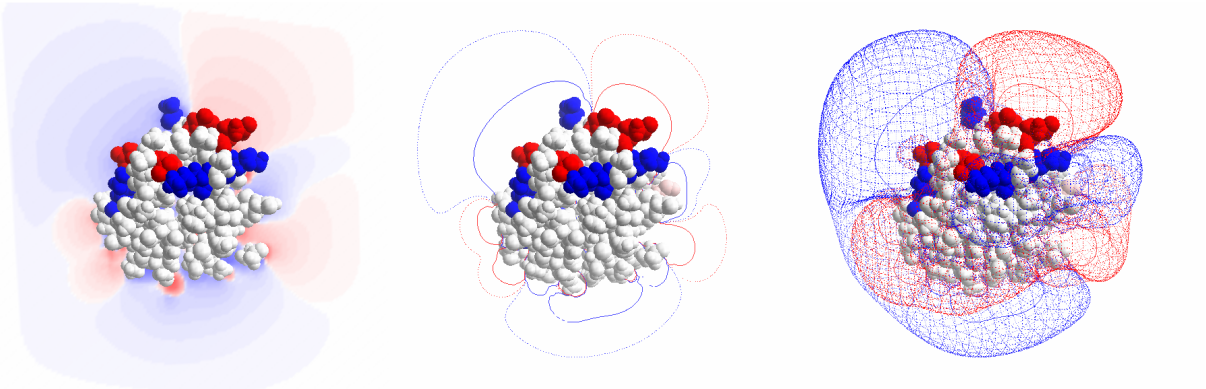


Figure 7-5 Shows three representations of the electrostatic field surrounding the cytochrome C552 monomer from Pseudomonas nautical. The left panel shows the field in a plane, with the colour indicating the intensity of the field (red for negative, blue for positive). The centre panel shows isopotential lines at ±0.1Kcal/mol (solid) and ±0.01Kcal/mol (dotted). The right panel shows a three-dimensional representation of the ±0.1 Kcal/mol isopotential surfaces.

Chemera uses a very simple approach, considering three classes of electron transfer events – through covalent bonds, through hydrogen bonds, and through space – and adding the contribution from all possible paths to estimate the transition probability between an origin atom and any other atom in the structure.

Figure 7-6 illustrates an application of this algorithm with a hypothetical model for the interaction between two desulfoferrodoxin dimers, built by docking simulations from the crystallographic structure of desulfoferrodoxin (Moura and others 1990; Coelho and others 1996, 1997). This model suggests how the protein exchanges electrons between the two different iron centres.

The iron atom of one desulforedoxin-type domain A was chosen as the origin of the electron transfer, and in this configuration, the iron centre A is closest to the neelaredoxin-type centre of the other dimer (B) than to the neelaredoxin-type centre of the same monomer (C). The intensity of the red colour indicates a greater probability of electron transfer to the neelaredoxin-type centre of the other dimer than to the one in the same monomer.

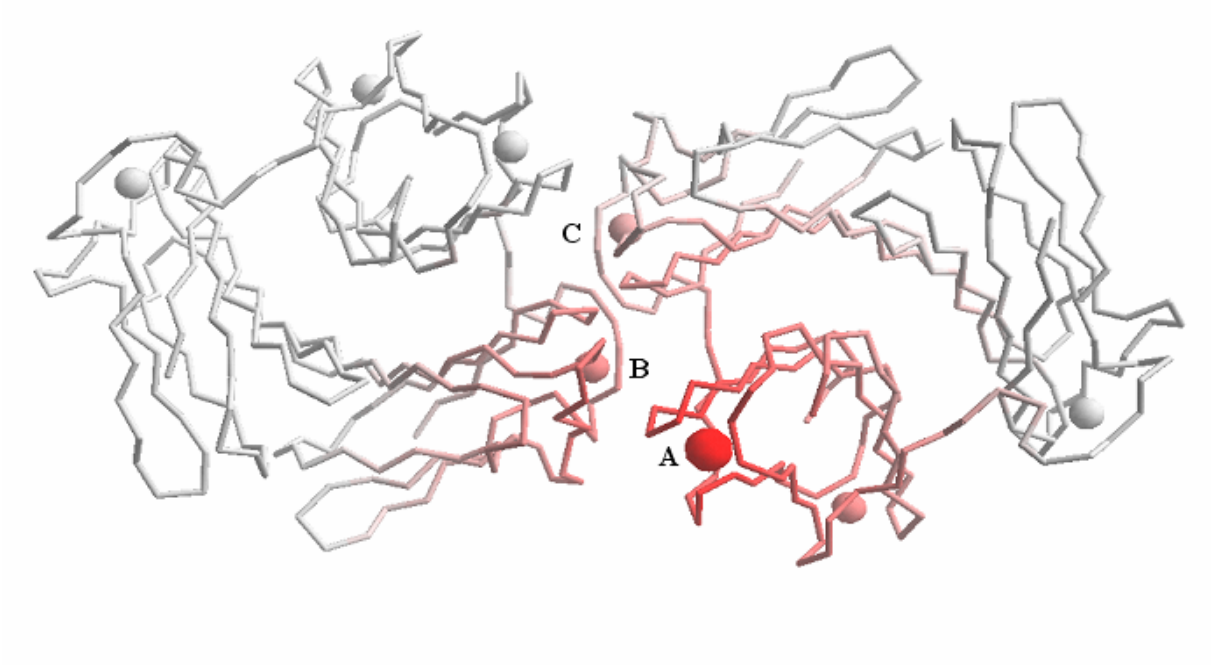


Figure 7-6 Electron tunnelling within and between two dimers of desulfoferrodoxin (Coelho and others 1996, 1997). The red colour indicates the probability of electron transfer from the iron atom at centre A, suggesting a possible pathway from atom A through B and to C.

PART

3

Tools, Applications, and Results

In This Part:

Chapter 8
PSICO: Protein Structure

Chapter 9
BiGGER: Protein Docking

Chapter 10
Applications

Chapter 11
Concluding Remarks

8 PSICO: Protein Structure

> For those who intend to discover and to understand, not to indulge in conjectures and soothsaying, and rather than contrive imitation and fabulous worlds plan to look deep into the nature of the real world and to dissect it— for them everything must be sought in things themselves.
>
> Sir Francis Bacon

This chapter presents the results of some of the tests evaluating the performance of different parts of PSICO. This algorithm is still at an early testing stage, and less developed than BiGGER. Not only because work on PSICO began later, but especially because this algorithm is considerably more complex. The PSICO DLL, for example, contains some seventy thousand lines of code (ten times the number of lines in this book). With such a complex system, much could go wrong; and much did, so most tests had the objective of checking and correcting the implementation.

Only recently was it possible to evaluate the performance of PSICO on more than a few test cases. PSICO was implemented with Delphi 5 (Borland Software Corporation ™).

8.1 Binary Propagation

This section evaluates the propagation of binary distance constraints using the algorithm described in section 5.2 (Krippahl and Barahona 1999, 2002; Krippahl, Trosset and Barahona 2001). The test set consisted of 182 protein chains of known structure selected from the SPIN-PP database (see section 9.2 for more details on this database). Table 2, on page 153, shows the PDB codes and chain identifiers for these strucures.

The SPIN-PP database is a protein interface database, but the structures it contains are non-redundant, having no more than 70% sequence homology with any other structure in the database, which made it a good source for structures with which to test PSICO.

Chains belonging to the same oligomer as another selected chain were rejected, as were those with gaps in the residue sequence or other problems. The 182 chains used were those that could be readily interpreted by the PSICO DLL routines with no problems.

The testing program generated a network of distance constraints for each structure. For each atom pair belonging to different residues and with a distance of 6Å or less, the testing program added

the distance between the atoms as a constraint with a probability of 25%. This is approximately the limit where random constraint generation can produce enough constraints to specify a globular structure, as previously determined by experiments in a case study (Krippahl 1999). The upper and lower limits for each distance constraint were, respectively, the measured distance plus or minus one Ångstrom.

Figure 8-1 shows the results of these tests. The left chart shows the RMSD from the target structure for each PSICO model, plotted against the chain size in residues. All RMSD values are for α-carbon deviations. The right chart shows the computation time scaled to a 1GHz PC.

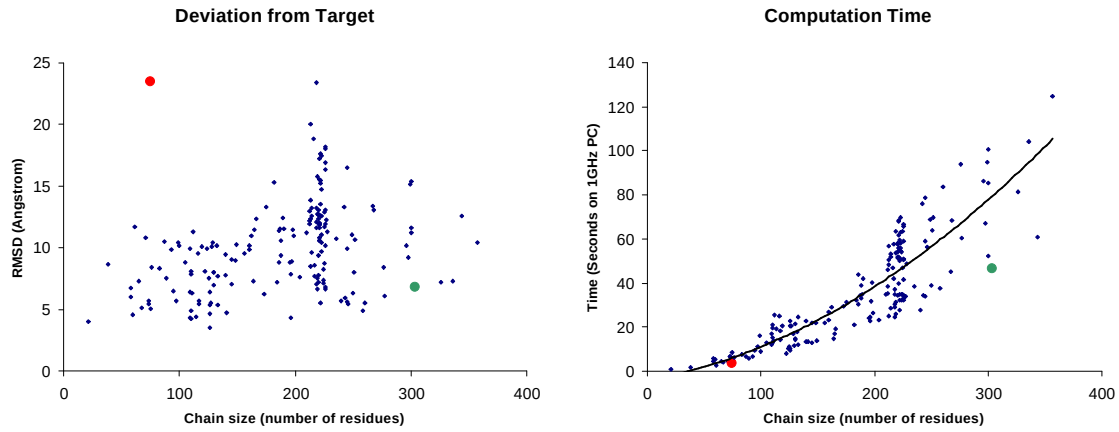


Figure 8-1 RMSD from the target structure and computation times for the 183 test structures. Computation times are scaled to a 1GHz PC. The red and green circles indicate the structures shown on Figure 8-2. See Table 2, page 153.

The computation time increases with the square of the size of the structure (illustrated by the quadratic trend line on the computation time chart), which is as expected. The RMSD of the PSICO model from the target structure also seems to increase as the size of the structure increases. This is also not surprising, but the large variation in RMSD values is due to more than simply the size of the structure.

Figure 8-2 illustrates the major factor in these results, showing a small protein whose PSICO model was very different from the target (23Å RMSD) and a large protein for which the model was fairly accurate (6Å RMSD). The small protein is a DNA binding protein (Wilson and others 1995; PDB code 1FJL) with two long, extended, segments. Short-range constraints like the ones simulated during this test are unable to restrict the mobility of these segments, which leads PSICO to place them in orientations that, though consistent with the constraints, are not similar to those in the

target structure. The RMSD for the region where the chain folds and short-distance constraints can specify the structure is less that 4Å. The flexible extremities account for most of the 23Å deviation.

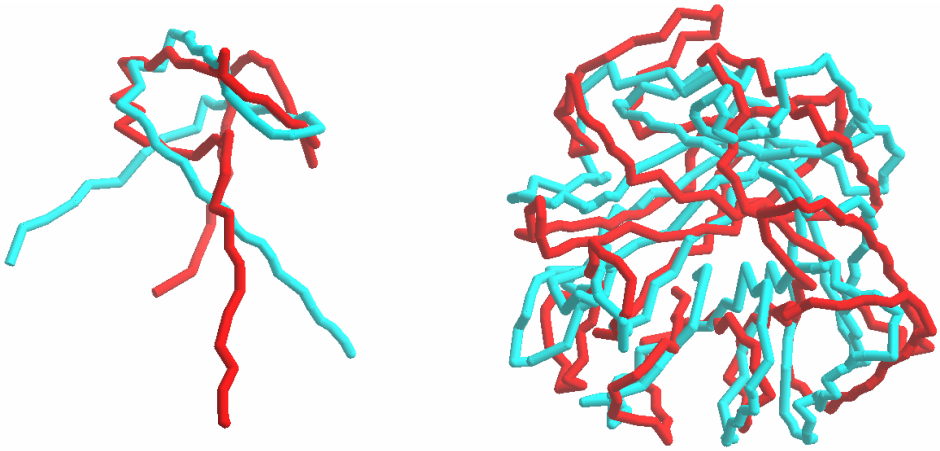


Figure 8-2 Two structures illustrating the performance of binary distance constraint propagation in PSICO. The left panel shows the target structure (in blue) for a DNA binding protein, PDB code 1FJL (Wilson and others 1995) and the PSICO model in red. The right panel shows the target structure of a 1,3-Beta-Glucanase (Varghese and others 1994), PDB code 1GHS, in blue and the PSICO model in red.

The second example is the opposite situation. This protein, 1,3-Beta-Glucanase (Varghese and others 1994; PDB code 1GHS) is a tightly packed globular protein. Since the distance constraints specify the structure to narrow limits, the PSICO model is close to the target structure, with a deviation of only 6Å for a structure with 304 amino acid residues.

These two examples are marked in Figure 8-1, a red circle indicating the DNA binding protein and a green circle the position of the Glucanase on the chart. It is interesting to note that the Glucanase model was both the most accurate and the fastest to compute for a structure of that size. This is not surprising, for the more constrained the structure the more efficient propagation will be and, other things being equal, the faster PSICO can calculate the model. The interesting aspect is that this is the inverse of what happens with the commonly used local search methods, like simulated annealing. In these, a greater number of constraints results in a longer computation and in an increased probability of being trapped in local optima.

8.2 Group Propagation

This section describes the performance tests run on the group propagation algorithm (section 5.3), and comparisons with algorithm to propagate binary distance constraints (section 5.2).

The first test was on set of random groups of block domains, which each algorithm had to reduce by propagation. Each group was generated creating a random group of points, uniformly distributed in a cube of 8 volume units per point. For example, for a group of ten points, the points were distributed in a cube of 80 volume units. The coordinates of the points provided one group constraint, and a set of Nx(N-1)/2 distance constraints specifying all distances between atom pairs.

The upper limit for each domain was the coordinate value of the respective atom plus a uniformly distributed random variable ranging from zero to two units, and the lower limit the coordinate value minus a similar random variable. Repeating the process for all three directions generated a set of random domains that included the position of the atom, thus ensuring a consistent set of domains.

The binary distance constraints were propagated to arc-consistency (AC), as described in section 5.6 (see also section 3.2). The group propagation algorithm used 10º steps for the rotation searches (see section 5.3). Figure 8-3 the propagation times and the final domain volumes for both algorithms.

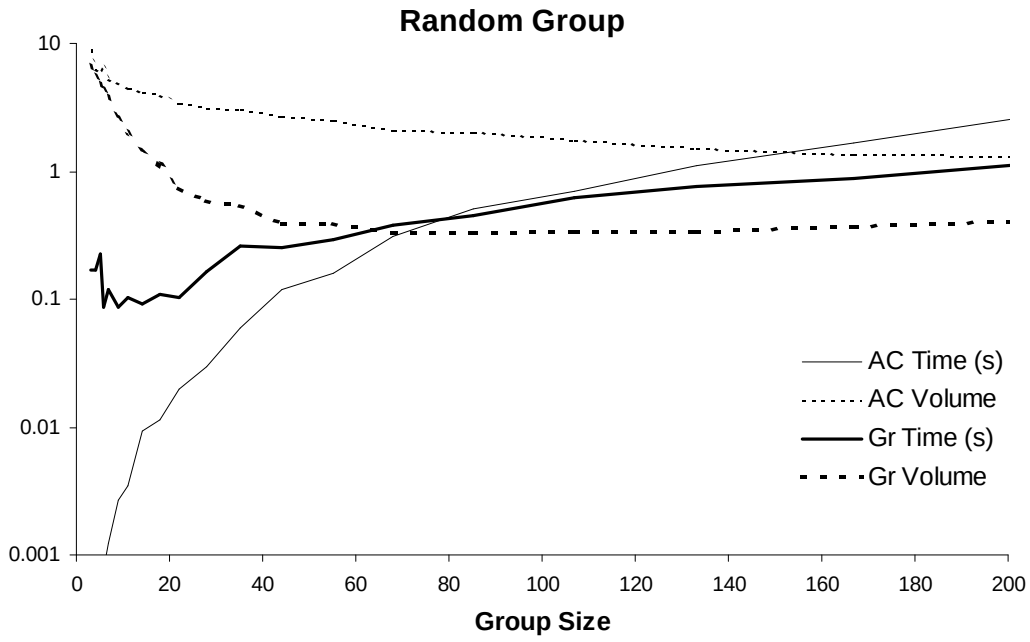


Figure 8-3 Propagation times and final domain volumes for arc-consistency in binary distance constraints and group propagation for a randomly generated group. Times are in seconds for a PII at 300Mz running Windows. Volumes are arbitrary units.

Each point on the chart corresponds to the average of 30 independent runs. In general, the group propagation algorithm is more effective than enforcing arc-consistency on a network of binary

constraints, even using all constraints, except for very small groups. For small groups, less than 10-20 atoms, group propagation is only slightly more effective than arc-consistency on binary constraints. For groups of over 20, the final domains with group propagation are an order of magnitude smaller than for arc-consistency of binary distance constraints.

The difference is even greater with more structured groups. Figure 8-4 shows the results of a similar experiment, but using a spiral structure with a radius of one unit, a step of two units per turn, and three points per turn. This approximates a α-Helix, a common structural motif in proteins, which has 3.6 residues per turn and approximately the same dimensions in Ångstrom.

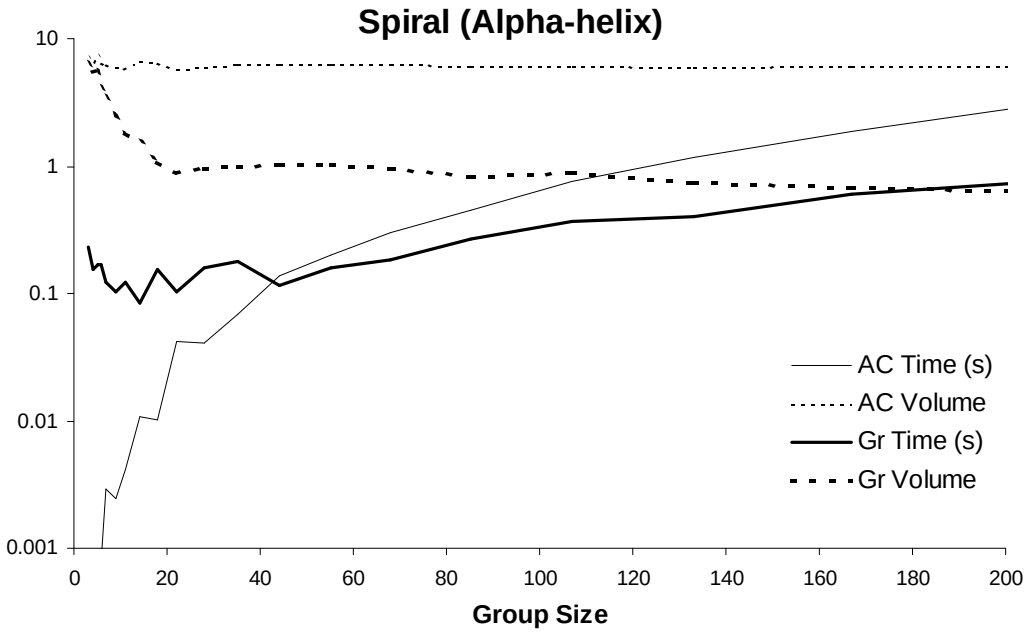


Figure 8-4 Propagation times and final domain volumes for arc-consistency in binary distance constraints and group propagation for a spiral generated group, simulating an a-Helix structure. Times are in seconds for a PII at 300Mz running Windows. Volumes are arbitrary units.

This domain reduction has a significant time cost for small groups, although, for large groups, group propagation is both more effective and faster. The reason for this cost in smaller groups is that, with little or no domain reduction, there is also no pruning of the rotational search. The interesting feature is that, the more the domains can be reduced, the faster the group propagation algorithm completes the search. This inverse correlation between domain pruning and computation time suggests that heuristics can make group propagation very efficient. Rules like a time limit for the propagation or checking the pruning on the φ rotation by the z limits are simple and effective

ways to use group propagation, because it is fastest when it is the most effective at reducing domains.

Furthermore, though AC on the binary constraints is about two orders of magnitude faster on average, the difference between the minimum times is of only one order of magnitude for the smallest groups, and with groups of 7 atoms the minimum time for group propagation is already lower than the average time for AC on the binary constraints. Finally, these tests were run without the optimisations described in sub-sections 5.9.4 and 5.9.5. These results suggest that, with structured groups, group propagation can both increase pruning and decrease computation time if properly applied with moderate or large groups. However, with small groups, such as those normally formed by the more rigid parts of amino acids, group propagation should be used more intelligently than as shown in Algorithm 5-9.

Another intended application of this algorithm is the integration of structural data from other sources, which can complement the experimental NMR constraints. These can be from secondary structure prediction (α-helices, β-sheets) or from homology modelling, for example, and can involve large rigid groups. As shown above, the group propagation algorithm is especially effective in these cases. But its use is not restricted to domain reduction; it is also useful for detecting inconsistencies, which is necessary to determine if the theoretical models assumed are compatible with the experimental data.

The test for this hypothesis was to change the coordinates of the points in each group after generating the constraints for the propagation algorithms, and then generating the domains from the altered coordinates. Each x, y, and z coordinate was replaced by a uniformly distributed random value within a given distance of the original value.

This simulated a situation where one set of constraints (e.g. from experimental data) reduced the domains to the current random configuration, and another constraint (e.g. from homology modelling) specifies a different configuration for the atoms. Thus the domains defined a structure that differed from the one that generated the constraints to test, and the difference could be controlled by the range of the random variable used to modify the coordinates of each point.

Figure 8-5 shows the percentage of failures detected as a function of this displacement parameter for groups of 20, 10, and 5 atoms. The results show that the group propagation algorithm can detect inconsistencies more reliably. Though the difference in this test is in the displacement

necessary to make the algorithms detect a failure, in practice this will also translate into detecting the failure sooner when testing different sets of constraints for consistency.

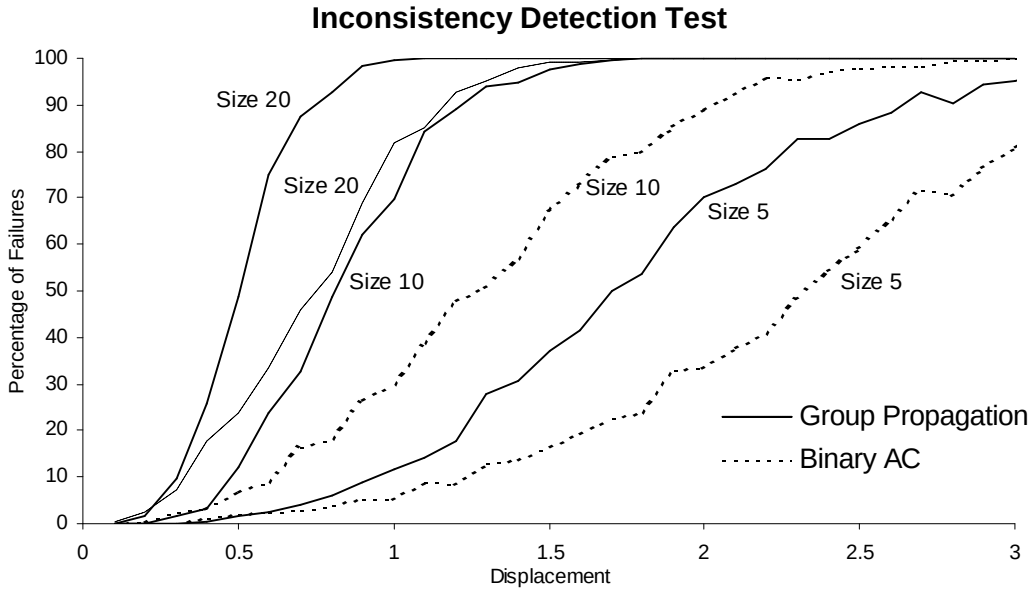


Figure 8-5.Percentage of detected failures as a function of the maximum displacements of the atom coordinates. The figure shows three different group sizes (5, 10, and 20), comparing the binary arc-consistency algorithm implemented in PSICO with the group propagation described here.

This is a potentially useful feature, because it can provide the ability to screen a database of structures from homology candidates, fragments, secondary structure prediction (theoretical or even from the NMR data itself) and other sources to find structures compatible with the data available. This can assist the research when the data is incomplete, such as during the assignment of cross peaks.

These preliminary results indicate that the group propagation algorithm is very efficient at detecting inconsistencies, because if the rigid group cannot fit the domains, the rotation searches are always significantly pruned (often completely) and the algorithm, in this case, runs very quickly.

8.3 Local Search Refinement

The final stage of PSICO consists in refining the approximate structure built by the constraint programming stage. Section 5.8 describes the local search algorithm, and this section gives an example of applying PSICO to a test case with real data. The target structure is desulforedoxin (PDB code 1DXG, Archer and others 1995) and the data is a set of NOESY constraints provided by Brian Goodfellow (Goodfellow and others 1998). Figure 8-6 shows the result.

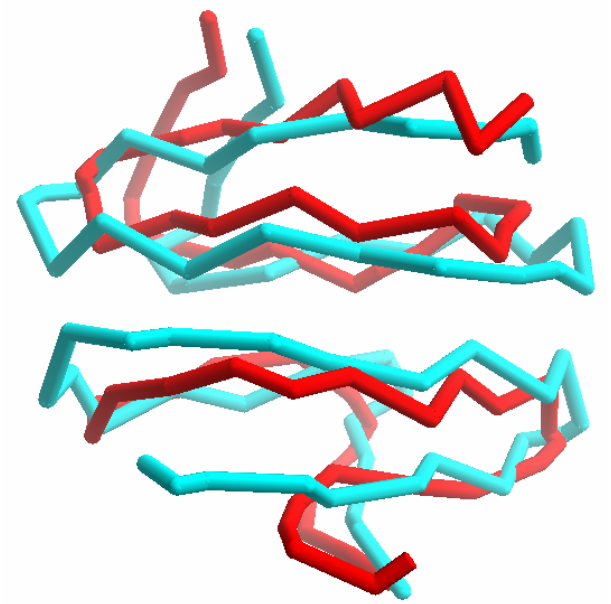


Figure 8-6 PSICO model for desulforedoxin using experimental NOESY data (Goodfellow and others 1998), shown in red, compared with the crystallographic structure of desulforedoxin, in blue (PDB code 1DXG, Archer and others 1995).

Desulforedoxin is a dimer, consisting of two identical monomers of 36 residues each. It contains two iron-sulphur centres, with one iron atom coordinated to four cysteines in each monomer. Ignoring the less constrained extremities of the two chains, the for the of the PSICO model RMSD is below 3Å. This is a significant improvement over the solution obtained in the constraint processing stage, with a RMSD of 6Å. The complete calculation took approximately a minute, with 15 seconds for the initial stage and 45 seconds for the local search. These computation times include a significant overhead with diagnostic and debugging routines.

8.4 Unresolved Issues

At the time of writing, there are still several problems to solve before applying PSICO to real problems. The value and variable enumeration heuristics (section 5.7) seem to be inefficient, as the performance is similarly poor for a wide range of alternatives and combinations (Marco Correia, MsC thesis, forthcoming).

The integration of the group and distance propagation algorithms is not straightforward, for the group propagation algorithm can be costly if applied indiscriminately. For large groups, such as in testing homology modelling candidates, there are no problems; the extensive pruning and the speed with which group propagation handles large groups make its use a simple choice in these cases. But it is still not clear what is the best method to take advantage of the many rigid groups of three or four atoms that make up a protein.

Finally, the local search optimisation algorithm is still mostly an unknown factor. The current implementation is a simple, naïve, approach, both inefficient and not adequate to producing multiple structures that can give information on the tightness of the constraints. Section 10.8 presents a more accurate minimisation algorithm, based on multidimensional scaling techniques, that could replace the current torsion angle minimisation using conjugated gradients. Another alternative would be to apply a more sophisticated combination of simulated annealing and conjugated gradient minimisation, as described by Guntert and others (1997).

9 BiGGER: Protein Docking

> Errors using inadequate data are much less than those using no data at all.
>
> Charles Babbage

Several factors make it difficult to properly optimise and test docking algorithms. The large number of possibilities in protein-protein interaction requires the examination of a wide range of cases. However, the more relevant cases involve weak interactions, and weakly biding complexes are the least well known of all structures. The computational demands of protein docking strongly limit the experimental possibilities

This chapter describes the parameterization of version 2.0 of Chemera and BiGGER (Palma and others 2002), and the most recent approach, still in progress, aimed at correcting weaknesses identified in the previous version and at improving the performance of the algorithm. Chemera and BiGGER were implemented with Delphi 5 (Borland Software Corporation ™).

9.1 Chemera and BiGGER Version 2.0

Though the docking algorithm (Chapter 6) remained constant through these years of development, some features and parameters have undergone almost constant change. This section briefly describes the parameters of the release version of Chemera (Krippahl and Palma 2001), and provides a baseline for the more recent results and methods, described in the following sections.

Most of this monograph provides the reader with only the more recent results and methods developed, despite being outnumbered by the many versions they superseded. But this case is an exception. The method in version 2.0 is publicly available, and so it is important to point out the problems identified during development, two years of application, and the work in progress to improve it.

This section will have a somewhat negative outlook, because its purpose show what can be improved. It will focus on the problems and mostly gloss over the good features of version 2.0 (Krippahl and Palma 2001; Palma and others 2002). So it is best to point out at the start that the parameters and filters described in this section have been successfully used to dock many different

complexes and that the docking applications shown on Chapter 10 all refer to this version. It is knowing that it works that we will look at what is wrong and how to fix it.

9.1.1 The geometric docking parameters

Until very recently, there were doubts about the choice of values for the geometric filter parameters. The decisions to use a 1Å resolution and radius modifier, a 12º or 15º rotation step, and to include hydrogen atoms were based on a very limited analysis (Krippahl 1996) of two test complexes: desulforedoxin (Archer and others 1995) and horse haemoglobin (Bolton and Perutz 1970). Despite these being commonly used values in docking algorithms, and despite many tests and applications of BiGGER never revealing problems with these parameters, it was still desireable to test these values with a more systematic analysis. Section 9.3 shows the results of such an analysis, which confirms the initial choice of all parameters except the use of hydrogen atoms, which does not seem to benefit the algorithm.

9.1.2 The need for additional filtering

If there is no correct model in the set with the highest surface contact score, we cannot use the geometric filter alone to select the likeliest models. An additional filter is necessary to exclude enough high ranking incorrect models to allow a lower ranking correct model into the final set. For each candidate model with a surface contact score high enough to enter the final set, the additional filter examines the side chain contacts to determine if the model is likely to be correct. The model enters the final set only if accepted by the side chain contacts filter.

Unlike the surface contact filter, this additional filter is not meant to retain the best N models, but to evaluate each model individually, either accepting it or rejecting it. Thus, the final set will contain the models with the highest surface contact score of all models accepted by the filter.

There are several alternatives proposed in the literature, using electrostatics, biochemical information, experimental data, and solvation effects (Walls and Sternberg 1992; Braci and others 2003; Guilquin and others 2002; Morelli and others 2000). The methods based on experimental data or biochemical knowledge are superior, but not applicable to all cases (see sections 6.5, 7.4, 10.3, 10.4 and 10.5), while the remaining methods do not result in dramatic improvements and are often constrained by the need for computation efficiency.

Therefore, we opted for machine learning techniques to identify features that distinguish correct models using the contact between the amino acid residues at the interface (Palma and others

2000). The choice of amino acid contacts as the determining factor was based on the assumption that local interactions have the greater role in complex formation. Electrostatic interactions can have a long-range effect, and it may be an advantage to include information on the electric field or polarity on the filter (Norel and others 2001). Another potentially important factor is the free energy – both entropy and enthalpy – of conformation change during complex formation (Searle and others 1992), but this is both difficult to estimate and possibly not applicable in a rigid docking algorithm. And while solvation effects are fundamental for protein folding and interaction, they have been successfully modelled by considering the solvent exposure of amino acid residues, which is a similar approach to the side chain contact criterion BiGGER uses to evaluate docking models. Although none of these simplified approaches can truly model the entropy of solvation, the side chain contact approach should be as successful as the more common approaches to modeling solvation effects.

Since BiGGER uses a search in real space, the additional filter needs only be applied to the minority of models that are candidates for the final set. All models with a surface contact score below the lowest of this set can be ignored. Compared to Fourier transform methods, this allows a greater flexibility in the type of filter to use, both because computation efficiency is not so critical and because the filter does not have to be encoded into a single correlation matrix (see Gabb and others 1997 or Walls and Sternberg 1992 for an example), although it is possible to improve the efficiency of some FFT approaches.

This section briefly describes the parameterization of the original contact filter, as implemented in the release version of Chemera, and focuses on the more recent work in a new filter, using a more methodical approach and better datasets to improve the results.

9.1.3 The data set for the contact filter and the global evaluation score

Table 3, on page 155, shows the PDB codes and chain identifiers for the structures used to generate the training set, forming 88 known complexes. This was the training set for the original side chain contact filter and for the global scoring network (sub-sections 9.1.4 and 9.1.5). Table 4, on page 155, lists the structures used to generate the test set. These test sets contained several redundant structures, where the same complex was formed by using the bound forms of the partners, the unbound forms, or one bound and one unbound partner.

The training sets were generated by docking simulations using a full rotation search at low angular resolution, with angular steps of 30º, 45º or 180º, depending on the computers available. The test

sets, with only 20 complexes, were generated with a full rotation search at 15º of angular resolution.

The docking simulations to generate training and test sets for the side chain contact filter used only the geometric filter with the soft docking modification (sections 6.2 and 6.3), while the simulations to generate the sets for the global score used both the geometric filter and the side chain filter, and the soft docking modifications (sections 6.2, 6.3, and 6.4). Thus, the global score as implemented on Chemera version 2.0 is best suited to soft docking with filtering, and the side chain contact filter for soft docking.

9.1.4 The original side chain contact filter

Our first side chain contact filter (Palma and others 2000) was motivated by the effects of the soft docking modification (sections 6.3 and 6.4), and was intended to balance the increase in incorrect models that threatened to exclude correct models from the final set. We opted for a Self-Organized Map (Kohonen 1982, in Haykin 1999), as a first approach to analysing potential features in a limited dataset. The training set consisted of 88 protein complexes, but was highly redundant, containing both bound and unbound versions of the same proteins and many homologous docking partners (see Table 3, page 155).

The Self-Organised Map (SOM) had a dual purpose: to be a classifier, and to find a pattern in the contacts of correct and incorrect models. The initial hypothesis was that the patterns of amino acid contacts at the interface could give us not only information on the accuracy of the model but also on the nature of the interaction. The SOM could thus be used both as a classifier to distinguish correct from incorrect models and to identify the driving force in the formation of the complex, such as solvation or electrostatics.

Unfortunately, the limitations on both the quantity and the quality of the data available presented some difficulties, and the results did not rise to the expectations.

A major problem was the dimension of the input vectors needed to encode amino acid contacts. With 20 amino acid types there can be 210 different contacts; an input vector of this dimension would be excessive with only 88 structures.

Our solution was to partition the amino acid types in five categories (Table 1, page 153), according to structural and chemical similarity. By considering only the category and not the amino acid type,

the contact vector was reduced to 15 values (the number of different pairs of five types). This vector is the V15 contact vector, to distinguish it from the V210 contact vector that uses all 20 amino acid types, and the V28 contact vector (see below) that groups amino acids into seven overlapping categories according to physical and chemical parameters (Table 7, page 156).

On hindsight, a Support Vector Machine (SVM) would have been a better classifier, since the main objective was to partition the models into two categories (correct and incorrect) using a limited dataset for learning. But inexperience and the possibility that the SOM could give us an additional insight into the type of interaction led to choosing this classifier.

Despite these limitations, the combination of the soft docking modification and this SOM to filter incorrect models resulted in an improvement in docking unbound structures. In a test of twelve complexes using the unbound structures of the partners, the original surface contact filter (section 6.2) retained correct models for four complexes. With the soft docking modification (section 6.3), it was successful in seven cases, including the first four. The combination of soft docking with the SOM filter for side chain contacts (section 6.4) was successful in an additional two cases, for a total of nine successes out of twelve (Palma and others, 2000).

More importantly, our experience implementing the original side chain contact filter showed us the problems and difficulties of designing such filter. We assumed a criterion for determining side chain contact (6.4) without comparing it to alternatives, and our choice for the V15 contact vector was dictated mainly by limitations in the data set and in the computing capabilities we had available. Lack of experience led to choosing a sub-optimal architecture for the classifier. The limited size of the data set used forced us to do without an independent validation set, and the redundancy in the data increased the correlation between the training and test sets. Finally, the data set was biased towards the more stable complexes. As a result, the performance of the algorithm decreased in some cases, like the weak interactions in electron transfer complexes, leading us to set the soft-docking and side chain filter modifications as an optional alternative to the initial hard docking algorithm, without additional filtering. But identifying the problem is the first step to solving it, and it is often the hardest. This was perhaps the most important benefit of our first attempt at implementing the side chain contact filter.

9.1.5 Global score

BiGGER uses a neural network to calculate the global score. As described in sub-section 6.7.4, this is a forward feeding network with two layers, 19 input neurons and 1 output neuron. The input

vector consists of the V15 contact vector plus the four interactions scores of surface contact, electrostatics, side chain affinity and solvation effects (see section 6.7).

The network was trained with backpropagation to respond with a value of 0.9 to a model that deviated less than 4Å from the correct structures, and with value of 0.1 to the other models. The training set contained 43,500 models, 500 from each of 87 different complexes (one complex, PDB code 4CPA, was rejected at this stage because of problems with the structure). These models were generated by docking simulations with angular steps of 180º, 45º, and 30º. The test set contained 100,000 models, 5,000 from each of 20 complexes, and was generated by docking simulations with an angular step of 15º.

The global score is an estimate of the probability that a model is correct, given the response of the network. BiGGER calculates the global score from the output of the network on the test set models: for a network response of x, the global score is the number of correct models in the test set with responses $\geq$x divided by the number of models with responses $\geq$x. For example, there are 11,158 incorrect models 1,448 correct models with a network response of at least 0.8 in the test set. The global score for a network response of 0.8 is 1148/(1148+11158)=9.3%.

The idea behind this conversion is not to give an accurate estimate of the probability but to give an intuitive score with a linear scale with a simple visual representation, because the error associated with this global interaction estimate makes it more useful as a means to examine large groups of models.

Figure 9-1 shows a histogram of the rank of the highest scoring correct model (below 4Å RMSD from the complex structure) for each complex in the test set.

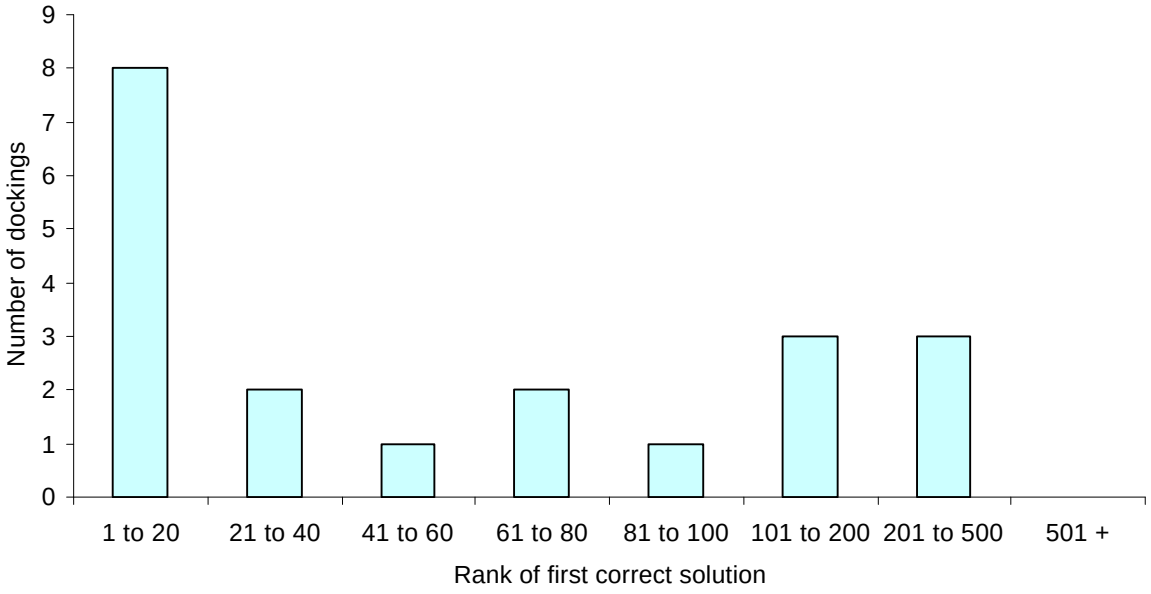


Figure 9-1 Ranking of the highest scoring model with less than 4Å deviation from the target complex for the 20 complexes in the test set.

In eight cases the highest scoring correct model was among the 20 highest scoring models, but in the remaining 12 complexes correct models were only found at lower ranks, down to 469 in the worst case.

9.2 The SPIN-PP Dataset

The SPIN-PP (Surface Properties of INterfaces - Protein Protein Interfaces) is a database of 6454 interfaces between two protein chains found in the PDB structure database. Each record includes the PDB code for the protein structure and the designations of the two chains in contact, plus information on several properties of each chain and the interface surface. The SPIN-PP database includes a non-redundant dataset containing no interfaces with more than 80% sequence similarity, and this set of 1029 interfaces from 760 protein structures was the chosen source for adjusting the added radius, angular step, and side chain contact filter parameters.

Though this set is representative of known protein chain interfaces, the set of known protein chain interactions is itself strongly biased towards stable complexes, because these are more likely to co-crystallise. Furthermore, a complex prediction algorithm is most useful in those cases where experimental determination of the complex is most difficult, which means that the set of cases

where BiGGER will tend to be used will have the opposite bias, towards weak interactions and unstable complexes.

This consideration determined the first choice in processing the dataset, which was to use as docking partners only the chains specified in the SPIN-PP database records, and not to dock one chain to the remainder of the protein structure. The latter would be more realistic for recreating the protein stucture, because the placement of each monomer in a multimeric structure is not determined by a single interaction but by the total interaction with all monomers it is in contact with. However, it would aggravate the bias for strong and stable complexes. For the geometric complementarity filter, taking one chain and docking it with the remaining structure would often be as simple cutting a slice off a cake and trying to put it back again, and not at all representative of docking two weakly binding proteins.

Unfortunately, using only the two chains specified in each interface record raises its own problems. For one, chain labels are somewhat arbitrary; often the same designation is used for all chains in a group (such as a protein that can be found isolated from the remaining chains) or all identical chains in a multimer structure, which raises the question of which chain or chains to use. For example, using all chains labelled "A" in a multimeric protein may result in an unrealistic structure formed by several separated chains, placed far apart; a most unlikely docking partner.

Another problem is that the protein chain interfaces are defined by the distance between the two chains involved, and do not necessarily reflect an affinity between chains. Their proximity may be due to an artefact of crystallization or caused by interactions with other chains in a multimer structure. Bovine F1-ATPase (Lutter and others 1993) provides an example of the latter case. As Figure 9-2 shows, the interface between monomers F and D, one of the 1029 interfaces in the non-redundant subset of the SPIN-PP database, is unlikely to represent an interaction between these protein chains.

Finally, within such a diverse set of 790 structure files containing multiple protein chains and prosthetic groups, there was likely to be a number of other problems derived from the particularities of each structure.

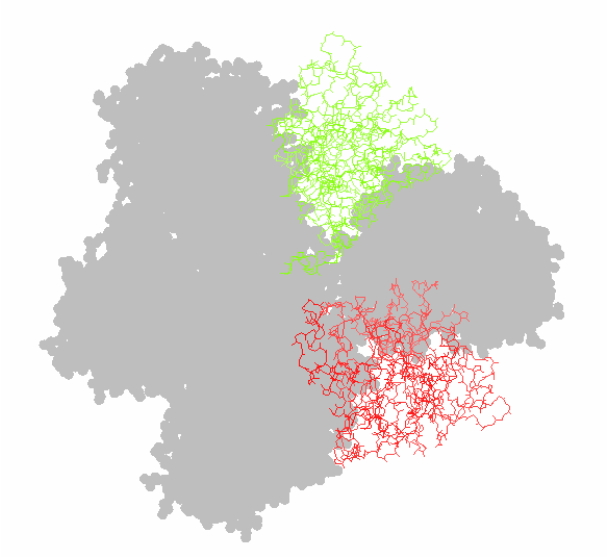


Figure 9-2 Shows monomers D (red) and F(green) of Bovine F1-ATPase (PDB code 1BMF). The remainder of the protein is shown in grey. It is more likely that the placement of these two monomers is determined by interactions with other chains in the protein than by a mutual affinity.

One possible way of solving these problems would be to examine each protein structure and manually select the correct chains. Though a laborious process with such a large number of structures, this was feasible, and was the solution we used in the first attempts to adjust the docking parameters (Palma and others, 2000). However, there is a serious drawback to this approach: human decisions on a case-by-case basis are difficult to reproduce, and would inevitably result in the criteria changing across the sample. Though this approach may be useful to get a first idea on the likelihood of an hypothesis (such as the simulation of experimental data in Krippahl and others 2003), the author felt a more precise and reproducible approach was necessary to fine-tune parameters from statistical information, and so wrote a program to process the non-redundant SPIN-PP dataset automatically.

On the first pass, the program selected only the first amino acid chains in each protein structure with each of the chain identifiers specified by the SPIN-PP interface records. For example, if the interface was between chains A and B, the program would use only the first amino acid chain labelled A and the first amino acid chain labelled B. This excluded all prosthetic groups and duplicate chains, and 68 interface records where the SPIN-PP chain identifiers referred only to prosthetic groups or did not correspond to identifiers on the PDB files.

On the second pass the program docked all chain pairs using a 1Å resolution grid, 1Å added radius value, and without rotation. The choice for the added radius parameter was determined by a preliminary analysis, similar to the one described in section 9.3. Only chain pairs with at least one model below 3Å RMSD from the correct configuration in the best 100 by the geometric score were

kept. This eliminated those interfaces that did not reflect a real interaction, but were due to other factors such as crystallisation, misidentification, or interaction with other monomers.

Some of the 136 interfaces eliminated at this stage could have represented real interactions, eliminated because geometric surface contact was not optimised. This could make the dataset biased towards those cases where the geometric filter works best. However, the criterion of finding a good model within the highest 100 without rotation is very permissive; extrapolating to a full rotation search, this is equivalent to keeping from half a million to more than a million models, depending on the angular resolution used. This is two to three orders of magnitude over the practical limit of one to five thousand models. Furthermore, the chains were the correct orientation and in the bound conformation, which greatly increases geometric complementarity. Finally, only some 15% of the dataset was rejected in this stage, and part of those for legitimate reasons, so the bias introduced in the set should not have been significant.

In any case, we cannot expect the docking algorithm to work where surface contact is not significant, nor can we generate a useful interface dataset without introducing some bias that distinguishes the relevant interfaces. Thus, the selected dataset consisted of 822 chain pairs found to interact in known structures, with significant surface contact, and in which no pair had more than 80% similar in sequence to any other pair in the set.

9.3 Rotation Search and Geometric Filtering

Three parameters characterize the geometric search and filter (sections 6.1 and 6.2): the grid resolution, the added atomic radius, and the angular search step. In addition, many docking algorithms don't use Hydrogen atoms in their representations of protein structures. This is justified by the small size of hydrogen atoms, and by their absence from protein structure files.

The main problem in determining the optimal value for the three numerical parameters is their interdependency. Grid resolution affects the overall shape of the geometric grid, which in turn will affect the ideal value of the atomic radii function and sensitivity to the orientation of the probe. Even sampling only ten values for each parameter, a total of one thousand runs for each complex used would be necessary to evaluate all combinations. This was beyond the available computational capabilities, so some compromises were necessary.

The first was the arbitrary choice of 1Å for the grid resolution. This may not be the best compromise between accuracy and efficiency, but it is difficult to weigh one factor relative to the other. There was no apparent overall ratio to optimise in order to decide on the best grid resolution value. Furthermore, grid resolution affects all other parameters; a comprehensive search for the best value for this parameter would be prohibitive, as it would require the examination of all combinations. Nevertheless, experience with a few other values, such as 1.2Å and 2Å, suggested that 1Å is the best choice, and since 1Å is the diameter of the smallest atom, there should be little gain in using a finer resolution.

The other concession to computation limitations was not to consider the interaction of the added radius parameter with angular sensitivity, by assuming them independent factors. This is a reasonable assumption, since angular sensitivity should be more dependent on the overall shape of the docking partners than on small adjustments of the order of fractions of an Ångstrom, but an untested assumption nonetheless.

The experiments explored the added radius parameter between 0Å and 2Å, with and without hydrogen atoms. The data suggested not using hydrogen atoms and an added radius value of 1Å; these were the parameters used to explore the angular sensitivity to rotations between 0º and 20º.

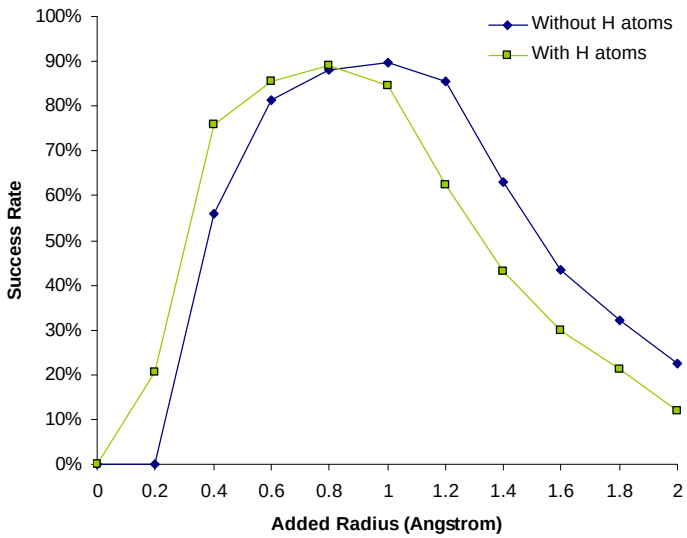
Each data point consisted of docking runs on all 822 chain pairs, using a fixed orientation, which was either the original relative orientation of the chains in the protein structure for the added radius and hydrogen atoms tests, or a rotation around the X coordinate axis between 0º and 20º. The axis of rotation was an arbitrary choice; without any reason to expect a correlation between the choice of coordinate reference frame and chain orientation, this is effectively identical to choosing a random rotation axis, and this simplified the implementation and the calculations.

It would have been best to optimise these parameters with a full search and retaining a set of one to five thousand models, the number of models kept on previous versions of the algorithm, which experience indicated was a good compromise between a practical sample a the risk of losing the correct models. However, a full rotational search can take from a few hours to a few days, depending on the angular resolution, and with 822 complexes to model for each data point, this would have been impractical.

Figure 9-3 shows the result for the optimisation of the added radius parameter. The results indicate optimal values of 1Å without hydrogen atoms and 0.8Å with hydrogen atoms, and that the

addition of hydrogen does not improve the success rate. These results differed from those that determined the use of a 1Å added radius with hydrogen atoms in previous versions of BiGGER (Krippahl 1996, Palma and others 2000), because the smaller datasets used were insufficient to detect the small difference in performance (5% lower success rate when using hydrogen atoms at 1Å added radius).

Figure 9-3 Shows the variation in the fraction of interfaces modelled with success as a function of the added radius parameter, both with and without hydrogen atoms. An interface was successfully modelled if there was at least one model at less than 3Å RMSD from the known structure within the 10 highest scoring models. Only the surface contact score was used in this test, and the chains were placed in the correct relative orientation.



Performance was measured by the number of interfaces where a close model (less than 3Å RMSD) was present within the highest ten ranking models using the surface contact score. With a 0.8Å added radius and hydrogen atoms, 732 of the 822 interfaces met this criterion. Using a 1Å added radius and no hydrogen atoms the number was 738. This improvement is statistically insignificant, being smaller than the standard deviation for this binomial distribution ($n=822$; $p=0.9$; $\sigma=8.6$). However, the difference at an added radius value of 1Å for both cases is significant (42 cases, five times the standard deviation).

Since the addition of hydrogen atoms seems not to improve performance, BiGGER will disregard hydrogen atoms in future versions. The 1Å value for the added radius parameter will remain. All following tests used a 1Å added radius value and no hydrogen atoms.

The last parameter for the geometric filter is the angular search resolution. Since the angular step affects computation time, we cannot simply choose the value that guarantees the greatest success rate. The time complexity is on the order of the cube of the angular resolution (halving the angular step increases computation times eightfold), which makes it necessary to find a compromise between effectiveness and computation cost.

The most rigorous way to determine the effects of angular resolution would be to run the complete simulation for each pair of protein chains and for each value of the angular resolution to test. However, this would require on the order of a year of CPU time, so the effects must be estimated from other data.

Aside from the practical issue of the computational cost of smaller angular steps, there is the matter of finding an accurate model in the final set of models retained. Using an elementary equality in probability, we can break down the probability of finding and retaining at least one accurate model into two components: the probability of finding at least one accurate model during the search, and the conditional probability that at least one accurate model found is retained in the final set, instead of all accurate models found being replaced by inaccurate models:

$$P(F \wedge R) = P(F) \times P(R \mid F) \qquad (9.1)$$

The probability of finding an accurate model in the search (event F) is a function of the angular step. This probability was estimated by sampling a single orientation at several rotation values. Each of the 822 complexes in the data set were reconstructed by searching the translational space after rotating one partner around the x coordinate axis. Since we expect no correlation between protein shape and the choice of coordinate references, this should be effectively a random axis. Figure 9-4 shows the percentage of accurate models found as a function of the rotation value, and the estimated probability that such models would be found as a function of the angular step value. For an angular step of θ, the minimum angular distance to the correct orientation with vary between zero and $\theta/2$, so the probability of finding an accurate model with a step of θ is the average of the probabilities of finding an accurate model at all rotation values from zero to $\theta/2$.

In three dimensions, this distribution is bell-shaped, not uniform, as the angular distance is the result of three rotation angles. Nevertheless, in the interval of interest, the probability decays approximately linearly with the increase in the angular step, so the average still provides a good estimate even if there is a greater probability of finding intermediate angular distances.

The right panel in Figure 9-4 show the estimated the probability of finding a correct model during the search, represented by P(F) in equation (9.1).

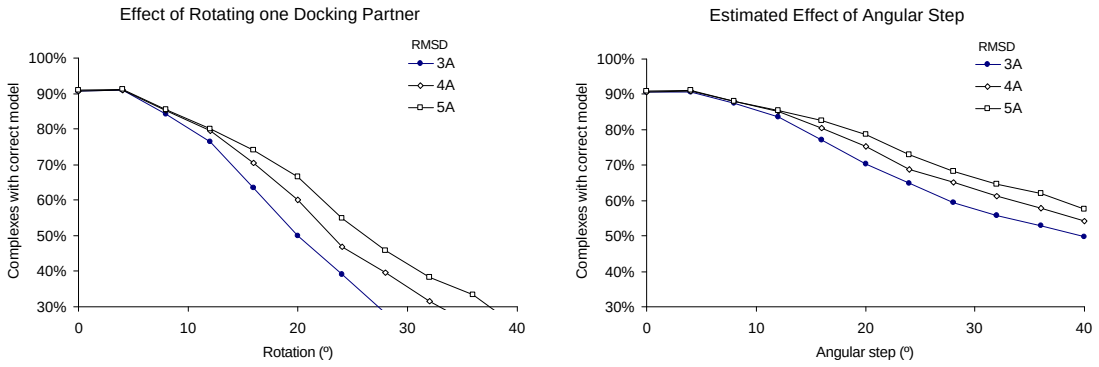


Figure 9-4 The left panel shows the percentage of complexes with at least one accurate model in the highest scoring 30 models for a single orientation, as a function of the rotation angle of one partner rotated. The right panel shows the estimated probability of finding an accurate model for each angular step value. Note that there are three different criteria for an accurate model: 3 Å, 4Å, and 5Å.

This experiment considered the 30 highest scoring models, using the surface contact score (see section 6.2). Assuming this number was large enough to ignore the effects of accurate models being displaced by inaccurate models, we now have an estimate for P(F), as a function of the angular step.

From the results obtained previously with docking unbound structures (Palma and others 2000), we can estimate $P(F \wedge R)$ in realistic situations. The soft-docking modification without the side chain contact filter was successful in retaining at least one correct model in seven out of twelve cases, or approximately 60% of the time. These simulations used a 15º angular step, so we can calculate:

$$P(R|F)_{15^o} = 0.6/0.85 = 0.7$$

Assuming the proportion of accurate models in the set of all models found is constant, and that the number of models retained in the final set is also constant, P(R|F) must be proportional to angular step, decreasing as we decrease the value of the angular step. This is because the total number of models found is proportional to the number of angles sampled, and the greater the total number of models, the more models will have a score higher than the best accurate model. The reason for keeping the size of the final set constant at five thousand models is that our experience indicates this to be the practical limit for scoring and analysing docking models.

Let us make the approximation that the average number of models found in the set of five thousand retained is linearly proportional to the inverse of number of angles sampled, and that this average can be equated with $P(R|F)_\theta$. Knowing that the total number of angles sampled is inversely proportional to the cube of the angular step, we have:

$$P(R \mid F)_q = a \times q^3 \qquad (9.2)$$

Where α is a proportionality constant that we can estimate from the value obtained for θ=15º, and with values greater than 1 for considered equal to 1.

Combining equations (9.1) and (9.2), and using $\alpha=2x10^{-4}$:

$$P(R \wedge F)_q = P(F)_q \times q^3 \times 2 \times 10^{-4} \qquad (9.3)$$

Figure 9-5 shows the estimated combined probability, as a function of the angular step. If the assumptions hold, the value of 15º is the best choice for the most stringent criterion of requiring a RMSD below 3Å for an accurate model. However, equation (9.2) is unlikely to be an accurate approximation, and the estimate for an angular step of 12º may be too pessimistic, especially if the algorithm uses a side chain contacts filter.

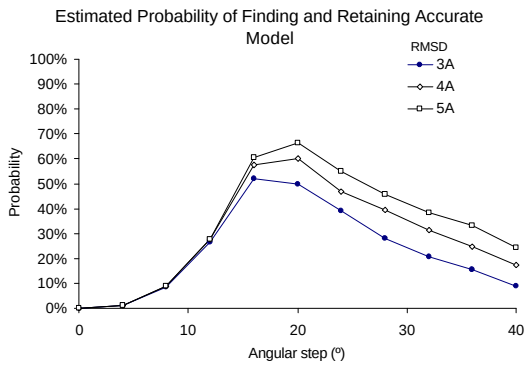


Figure 9-5 Effect of the angular step on the probability of finding and retaining at least one correct model. See equation (9.3).

Nevertheless, the main conclusion is reliable: even disregarding the practical issue of computation cost, too small an angular step is not necessarily an advantage. It seems the initial compromise due to computation cost led to, or close to, the optimal value for the angular step.

An interesting result of this experiment is the suggestion that larger values, such as 20º, may be equally good or even slightly better; an interesting possibility to test.

9.4 The new chain contact filter

With the SPIN-PP dataset and the experience acquired, it was possible to correct or mitigate these problems. Using 700 of the 822 complexes for the training set gave a sizeable sample of protein chain interactions, with 122 complexes to serve as a test set. Since all these examples are from bound structures, the set of 25 unbound structure pairs in the docking benchmark compiled at the

Wang laboratory at Boston university (Chen and others 2003) provided an independent validation set.

The problem with this larger data set was the increase in computation demands, now an order of magnitude greater than those of the initial filter. It was necessary to simulate the formation of each complex in the data set to produce a set of correct and incorrect models that reflected the conditions expected in a real application. Generating and processing the models for the first filter required a concerted effort using several computers, and took more than a month of calculations. Though computers have improved since then, it was still desirable to limit the computation requirements by a more intelligent selection of models to compare.

In a real application, the vast majority of the models produced by a full rotation search are incorrect. Even if only a small fraction happens to have a better contact score than a correct model, this small fraction may amount to a significant number of incorrect models preceding the correct models. Our initial approach to modelling this situation was to try to explore as much of the rotation search space as possible, and the set of models to train the original filter were generated by searching the full rotation space for each complex, albeit at lower resolutions of 30º-45º.

The current approach was to generate the models with sampling conditions that were unfavourable to the correct models. The docking simulation for each complex used a total of 16 orientations. Eight orientations were generated by rotating the sample $\pm$7º on each axis, to simulate the worse case scenario with a 15º angular resolution step, where the sampled orientations miss the correct orientation by half the width of the interval. The remaining eight orientations were rotations around each axis using a uniformly distributed random value between 30º and 60º.

This sampling scheme biased the model sets in two desirable ways: by generating a set of correct models that tended to have geometric contact scores inferior to the incorrect models, and by increasing the proportion of correct models relative to a real situation, where incorrect orientations can outnumber the correct ones by three orders of magnitude.

The first bias is desirable because the purpose of the filter is to exclude incorrect models with a better surface contact score than correct models. Incorrect models with lower surface contact scores cannot displace the correct models during the simulation, so these are not a problem, and this way even a small sample can provide a good number of such case. The second bias is useful because the correct models are important, even if severely outnumbered in real applications. In our

original approach, the proportion of incorrect models to correct models was 100 to 1. Using this sampling scheme and selecting only the 50 incorrect models with the highest surface contact score in each docking simulation, the proportion is now a more reasonable 2 to 1, reducing the problem of having a training set excessively biased against the cases where we wish to have the smallest classification error.

9.4.1 Contact parameters

There are two parameters to decide before evaluating side chains contacts: how to determine if there is a contact, and what types of contacts to consider.

Amino acid side chains can have great mobility, an important factor both in the stabilisation of the structure and in the formation of complexes (Doig and Sternberg 1995). Since the docking algorithm uses a rigid structure, it is the side chain contact model that must account for side chain mobility, and it does so by modelling each side chain as a sphere centred on the geometric centre of the side chain atoms. This way, the side chain can contact any other side chain in the neighbourhood even if the side chains are not oriented in the rigid structure so as to allow this contact.

The base radius for each side chain is that of the smallest sphere that contains the centres of all the side chain atoms plus the α-carbon. The parameter to adjust is the contact radius modifier, a value that is added to the sphere radius to determine the effective radius, and model the mobility of the side chains. Two side chains are in contact if the distance between the centres is lower that the sum of the modified radii of the two spheres.

The optimal combination is the one that best separates the correct cases from the incorrect cases. To estimate this without having to train classifiers in all possible combinations, we can measure the distance between the average contact vectors of correct models and incorrect models. Using the training set of 700 complexes, we calculate all contacts of all correct models for each V15, V28, and V210 and for each value of the contact radius modifier. We do the same for all incorrect models.

For each combination of vector dimension and contact modifier, we have a set of average contacts and the corresponding variances. The distance measure between correct and incorrect contacts was the average of the t statistics for those vectors.

Let C_a and I_a be the average contact vectors for correct and incorrect models respectively. Let C_σ and I_σ be the standard deviation vectors for the contacts between correct and incorrect models respectively. The distance between correct and incorrect models is:

$$d(C,I) = \frac{\sum_j \frac{C_{aj} - I_{aj}}{(C_{sj} - I_{sj})/2}}{N} \qquad (9.4)$$

where N is the cardinality of the vectors (15, 28 or 210).

This is not a standard measure of the statistical difference between two distributions, which is the χ^2 test. However, this statistical test is biased in favour of higher dimensions and higher numbers of events in each bin of the observed distribution. The average vectors of V210 with +4Å contact modifier would be the most statistically different simply for having the highest dimension and the largest number of contacts because of the modifier used.

The t average metric of equation (9.4) avoids these problems. Being an average, it cancels the bias for vectors with higher dimension, and the comparison of average and standard deviations is not affected by the overall increase in the number of contacts with the increase in the contact radius modifier. It is questionable to assume that the variances of these are similar, but the purpose of this metric is to provide an approximate ranking of the alternative configurations, not to give an exact absolute value of their statistical difference.

Figure 9-6 shows the results, comparing the three types of contact vector and contact radius modifiers ranging from -5Å to +4Å. The green circle corresponds to V28 at +1Å, the selected combination for the new contact filter. The yellow circle at V15 and +1Å marks the combination of parameters used in the original contact filter; a better choice than using the complete contact set of V210, but inferior to the new V28 with a similar contact criterion.

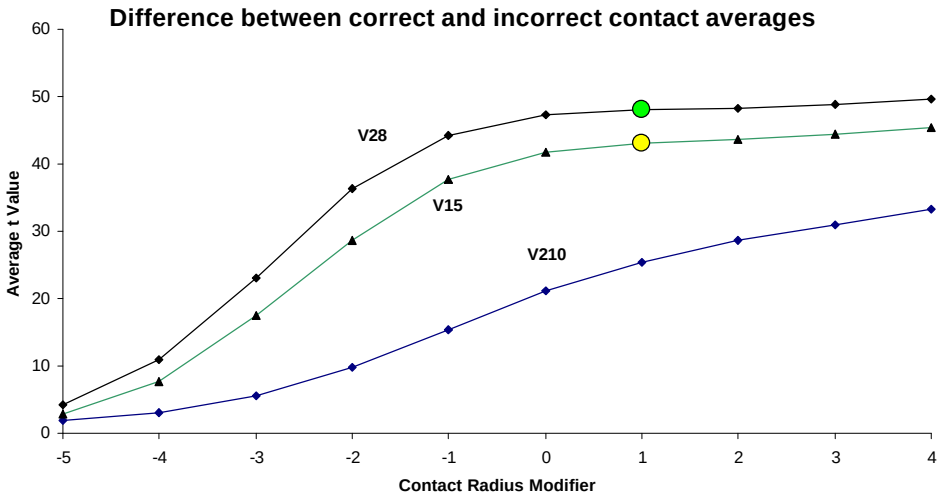


Figure 9-6 Shows the average t values for the differences between the average contacts of correct and incorrect complexes, for the three different contact vector dimensions and as a function of the contact radius modifier. The green circle shows the selected parameters (V28, with +1Å of contact radius modifier). The yellow circle shows the parameters for the previous version of the filter, using V15 and approximately +1Å of contact radius modifier. See Table 8 on page 156.

Based on these results, the side chain contact filters will use the V28 input vector and a +1Å contact radius modifier. There will be two side chain contact filters, for hard and soft docking. The global score classifiers will also be retrained with the V28 vector, with the possible addition of the four interaction scores (surface contact, side chain affinity, electrostatics, and solvation). There will also be different global scoring functions for hard and soft docking. It is necessary to train different classifiers for hard and soft docking because the soft docking option affects the models generated. Since soft docking improves the results in some cases but not in all cases, it is best to leave the soft docking modification as an option.

All classifiers will be support vector machines, which are better than SOMs or neural networks trained with backpropagation at coping with inadequacies in the datasets.

9.5 Soft-Docking

The soft docking modification (section 6.3) consists of removing some atoms from the representation of the protein core, to simulate conformational changes that could shift these atoms if they were blocking the formation of the complex.

Initially, the atoms removed were the side chain atoms from arginine, lysine, aspartic acid, glutamic acid and methionine, except for the β-carbon atoms, and only for the amino acids with more than 40% of surface exposure (Palma and others 2000). This list was later expanded to include some

atoms from cysteine, glutamine, isoleucine, leucine, phenylalanine, serine, and tyrosine (Krippahl and Palma 2001).

The list is currently being refined using a rotamer library (Lovell and others 2000) to determine the best choice of atoms to remove from the core representations. Rotamer libraries have been used with some success before to refine docking models (Jackson and others 1998) after the docking simulation. During the search for likely models, side chain flexibility is often accounted for by simplifying the structure representation of the docking partners (Li and others 2003, Zacharias 2003). The objective of the soft docking modification for BiGGER is to use the rotamer library to provide a more discriminate model of side chain flexibility to use during the search.

9.6 Unresolved Issues

The results shown in this chapter justify confidence in the search and geometric filtering procedures of BiGGER (sections 6.1 and 6.2), and in the values selected for the grid resolution, angular step, atomic radius modifier, and atoms used.

The soft docking modification and the side chain contact filter (sections 6.3 and 6.4) may require some fine-tuning, but these approaches seem to be the best compromise between efficiency and effectiveness. The soft docking modification implemented in BiGGER is more precise than the usual method of using a low resolution representation to account for side chain flexibility, without the computation cost of structure refinement by molecular dynamics or using rotamer libraries, something that is not practical to do during the search for likely models.

The side chain contact filter may be expanded to include other scores, such as solvation or electrostatic interactions, but this would only require changing the input vector and retraining the classifier. The basic concept seems to be the best option: with a real space search, additional filters evaluate only to the small fraction of models that pass the surface contact filter. Though still a sizeable sample, it is orders of magnitude smaller than the number of models other approaches need to examine. For example, to incorporate an additional filter in a FFT search, it is necessary to evaluate all possible configurations for both the surface contact and for the additional filter (Gabb and others 1997).

The implementation of the constrained docking procedure is still being tested (section 6.5), and it is too soon to tell how useful it may be. Nevertheless, a simplified version of constrained docking,

that consists in scoring the satisfaction of constraints by the models BiGGER produced (section 7.4) selected models, produced promising results so far (Morelli, Czjzek and others 2000; Morelli, Dolla and others 2000; Morelli and others 2001; Krippahl and others 2003). This justifies some confidence in the method, even at this early stage.

The greatest weakness in BiGGER is the global scoring function. Though it is relatively successful on average, placing correct models among the highest ranking 10% of the set of 5,000 models BiGGER retains (see Figure 9-1, in sub-section 9.1.5), this is not enough to be useful in practice, for the user is still left with a few hundred models to examine, any of which can be the correct model.

Furthermore, the scoring function is biased towards strong complexes (enzyme-inhibitor, dimers, and so forth), which were prevalent in the original training set. Its poor performance in modelling electron transfer complexes led to its replacement with other interaction scores such as electrostatics or older versions of the global score function in some cases (Pettigrew and others 1999; see also section 10.2).

The major problem with training the non-geometric filters and the global score is that the interface between proteins that form permanent complexes is likely different from that of proteins that form transient complexes, for the latter must be adapted to both solvated and unsolvated environments. This may result in different filter and score parameters depending on the nature of the complex to model, and different databases will have to be used to properly train these parameters. The SPIN-PP database mentioned previously provides good data on permanent complexes, while the database used by Gideon Schriber for the characterization of protein-protein binding sites will, once published, provide the information on transient complexes (Schriber 2003). Once the data is available, a comparison of the parameters for the two sets should determine whether or not using two different sets of filtering and scoring functions can improve the docking models.

Finally, there is still no clear way to interpret the results when the global function contradicts the experimental constraints. Such cases may indicate that no correct models were retained in the final set, that the global scoring function is not evaluating the models correctly, or that the experimental constraints were not correctly assigned. This is an important problem, but one that can probably only be solved with data from real cases, and not with simulated data.

10 Applications

"One cannot guess the real difficulties of a problem before having solved it."

Carl Ludwig Siegel

This chapter presents some applications of the algorithms that the previous chapters described. BiGGER accounts for most of the examples, because its development begun nearly two years before PSICO, and several researchers have used it to predict protein interactions. Their feedback has been invaluable in the development of the docking algorithms, for our simulations and tests alone could not reveal the richness of real life problems.

PSICO is at an earlier stage of development. Though the implementation is operational has produced results in test structures, there is still much to be done before other researchers can use it. There is need for graphical interfaces, compatibility with different types of data formats, interfaces where the user can specify special constraints, and so forth. This will likely be a long process, relying on the collaboration and suggestions of those working with real cases.

Section 10.1 presents the two software products of this work so far: the Chemera software package, which includes BiGGER and PSICO, and the PSICO Dynamic Link Library, which allows other researchers to integrate the PSICO algorithm in their own applications.

The remaining sections present some applications of these algorithms, chosen for their importance in the development and testing of PSICO and BiGGER. Section 10.3 describes the work of Graham Pettigrew and Sofia Pauleta, and section 10.4 describes the modelling and experimental work of Nuno Palma and Francoise Guerlesquin. The authors graciously supplied the docking models and figures presented in these sections.

10.1 Software Tools

Version 2.0 of Chemera is freely available since 2001 (Krippahl and Palma 2001). This version includes the graphics interface, BiGGER, and tools to process and evaluate docking models. With a few hundred registered downloads, it is probably the most visible product of this work. Through our email support service, we have helped approximately a dozen users who had questions or problems, and the feedback has been very positive, and the docking algorithm has been used in

several publications not involving the author's contribution (Crowley and others 2002; Morelli X, Czjzek and others 2000; Morelli and others 2001; Pettigrew and others 2003).

Though the release version of BiGGER still does not process experimental constraints as described in section 6.5, Chemera 2.0 can evaluate atom and residue contacts and so rank docking models using experimental constraints (see section 7.3). The release version also includes a comprehensive manual and some documentation on the algorithms.

The development version of Chemera is more advanced, currently implementing all the algorithms described in this book, though the more recent ones, such as constrained docking in BiGGER and some optimisations in PSICO algorithm, are still in an early test stage. Nevertheless, the development version is available to some researchers for field-testing, and this work owes much to their collaboration and tolerance for the problems of testing new software.

In the case of PSICO, the collaboration extended beyond the domain of biochemistry into computer science. The problem of protein structure determination is of interest to computer scientists too, both for its practical application and for its theoretical aspects. Being an important and difficult problem, there is interest in applying a various programming and artificial intelligence techniques in the search for solutions.

This motivated the production of a Dynamic Link Library (DLL), described in more detail in Appendix I: PSICO Dynamic Link Library. The PSICO DLL allows researchers to integrate the PSICO algorithms with their own code, and provides a set of tools to manage protein structural data on PDB files, amino acid templates and constraints.

10.2 Electron Transfer between Aldehyde Oxydoreductase and Flavodoxin

The xanthine oxidase family is composed of enzymes containing a molibdenum co-factor, and catalyse the hydroxylation of aromatic compounds and aldehydes (PROMISE database, Degtyarenko and others 1999). Except for aldehyde oxidoreductase (MOP) from Desulfovibrio gigas, these enzymes are organised into five domains, the first two coordinating two iron-sulphur clusters, followed by a flavin domain, and finally the two domains which bind the molybdenum co-factor. MOP is the exception for lacking the flavin domain.

Since the FAD co-factor in the flavin domain is the electron donor for the electron transfer reaction in these enzymes (Hille and Anderson 1991; Hille 1996), the Desulfovibrio gigas enzyme MOP would require a substitute donor protein, and a prime candidate would be the D. gigas flavodoxin.

The structures used to model the electron transfer complex between MOP and flavodoxin were the X-Ray structure for MOP (PDB code 1hlr, after correcting the transformation matrix to generate the dimer; Rebelo and others 2001) and an homology model for the D. gigas flavodoxin. This model was based on the Swiss-PROT sequence Q01095 (Helms and others 1992) and the structure from the D. vulgaris flavodoxin (PDB code 1fx1; Watenpaugh and others 1972).

Two other flavodoxin structures were used to test the suitability of the homology model for the D. gigas flavodoxin: the D. vulgaris flavodoxin (PDB code 1fx1; Watenpaugh and others 1972) homology model for the D. salexigens flavodoxin based on the D. vulgaris flavodoxin structure.

We used the BiGGER docking algorithm both with and without the soft docking modification and side chain filtering (sections 6.3 and 6.4). The soft docking algorithm (Palma and others 2000) was designed to predict a wide range of complexes, from tightly binding enzyme-inhibitor complexes to transient electron transfer interactions, and so must take into account the effects of side chain flexibility, which may be essential to allow the formation of tightly bound complexes. Furthermore, this general purpose algorithm must evaluate the candidate models taking into account different factors such as solvent exclusion, side chain contacts, and electrostatics.

Although this approach works moderately well in the general case, the downside is that it cannot take advantage of the particular nature of the complex to model. In this particular case, the weak binding of a transient electron transfer complex reduced the likelihood that incorrectly placed side chains would prevent the generation of the correct model. Therefore, we modelled all complexes with both hard and soft docking methods, and compared the results to see if we could use the more stringent hard docking algorithm without losing relevant models. Since the results are similar and the soft docking algorithm tends to generate a larger number of incorrect models, we present only the hard docking results.

Should the overall pattern of results of the hard docking simulation differ significantly from those of the soft docking simulation, we would have to use the soft docking results, because this could be an indication that the stringent criteria of the hard docking algorithm had eliminated the correct models.

The second stage of BiGGER is the ranking stage, in which the 5000 models are evaluated by a global scoring function (see sub-sections 6.7.4 and 9.1.5). Again, the objective of this general classification function is to make the algorithm applicable to a wide range of complexes, but in this case we decided to use only the electrostatic interaction score (see sub-section 6.7.3). MOP and all flavodoxins are highly charged, the flavodoxins have a dipolar moment in the order of 400 Debye, and, though both partners are acidic, MOP has a positively charged surface patch near the exposed Fe2S2 cluster, which suggest the complex is driven by electrostatic interactions. Furthermore, the global ranking scores did not indicate a clearly preferred site, probably because of considering factors like solvent exclusion effects or surface complementarity, which are important in most cases but apparently not as much as electrostatics in this particular complex. So we assumed that electrostatic attraction was the most important driving force for the formation of this complex.

Only the data on reaction kinetics for the D. gigas MOP-flavodoxin complex seemed at odds with this assumption, as the reaction rate increased with increasing ionic strength (Palma, unpublished data). However, the effect of ionic strength on the electrostatic field around MOP suggests an explanation. Figure 10-1 shows the isopotential lines near the Fe2S2 cluster at 0M and 0.5M ionic strength (see section 7.5 for details on the calculation of the electrostatic potential). Although at higher ionic strength the field is weaker, the overall negative charge of MOP has less influence near the positive patch surrounding the exposed Fe2S2 cluster, and no longer hinders the access of the negatively charged flavodoxin.

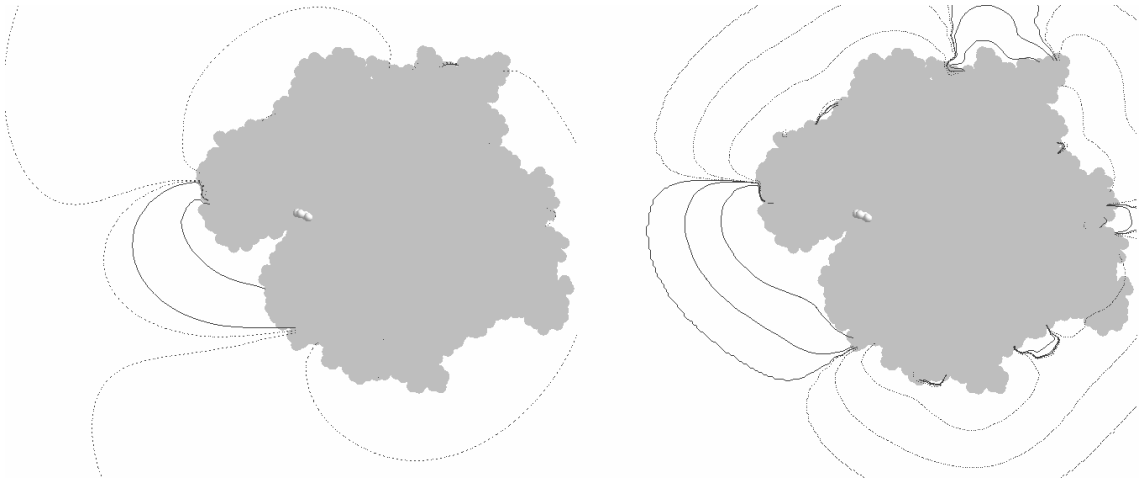


Figure 10-1 The electrostatic field profile of MOP at 0M and 0.5M ionic strength. Dotted lines indicate negative potentials, full lines positive potentials. The isopotential lines are drawn at 1 and 0.1 Kcal/mol for the 0M ionic strength, and at 0.1, 0.01, and 0.001 Kcal/mol for 0.5M ionic strength. The Fe2S2 cluster is shown at the centre of each figure.

Our main goal was to model the D. gigas complex, but flavodoxins from D. vulgaris and D. salexigens also form productive complexes in vitro. Since the structure for the D. gigas flavodoxin was an homology model, the other two — one a crystallographic structure and the other also an homology model — could be used to check the consistency of the results, and verify that the D. gigas flavodoxin model was appropriate for our purposes. Figure 10-2 shows the results, and in all three cases most complexes place the FMN residue of the flavodoxin close to the exposed Fe2S2 cluster (distances of 10-15Å).

We cannot give one unique model for the MOP-flavodoxin complex, and it is likely that the interaction is dynamic and no single structure exists that represents it. But these results allow us to predict that flavodoxin interacts by docking near the exposed Fe2S2 cluster. Though not unexpected, this is an interesting result when we compare the MOP-flavodoxin suggested by these docking simulations with Xanthine Oxidase (PDB code 1fiq; Enroth and others 2000) and CO-dehydrogenase (PDB code 1qj2; Dobbek and others 1999)

Figure 10-2A comparison of the three flavodoxin-MOP dockings. For each case the best ten models (according to the electrostatic score) are represented by the FMN residue relative to the MOP monomer.

As Figure 10-3 shows, the arrangement of the prostethic groups in the MOP-flavodoxin complex is similar to the arrangements found in CO dehydrogenase and xanthine oxidase, which indicates that, in D. gigas, two separate proteins cooperate to form the same electron transfer chain that exists in a single protein in eukaryotes.

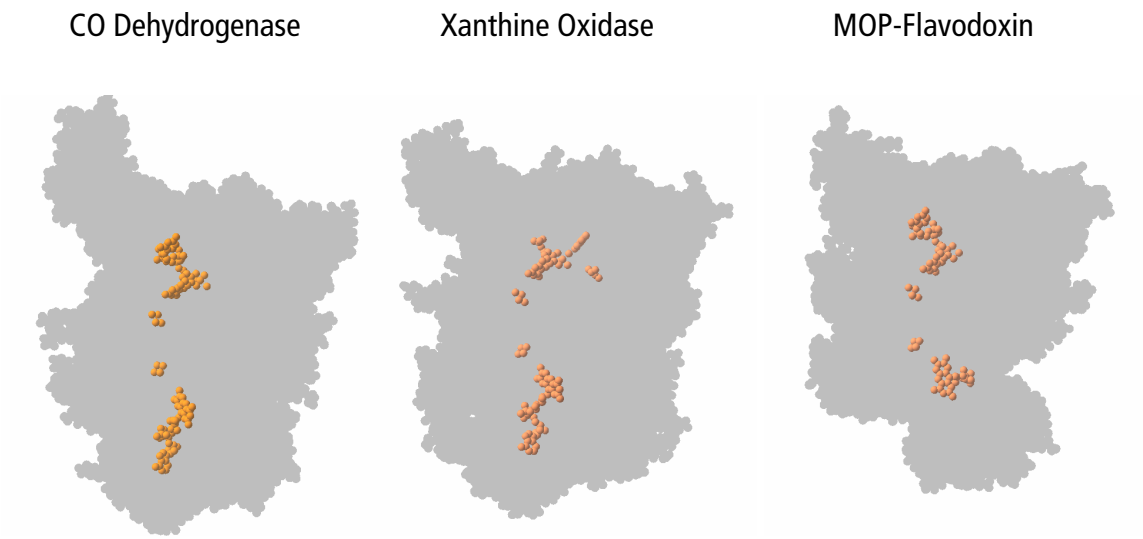


Figure 10-3 Comparison of the relative positions of the prosthetic groups for CO Dehydrogenase, Xanthine Oxidase and the best model for the MOP-Flavodoxin complex. The FMN group of flavodoxin is shown at the bottom, aligned with the FAD groups of CO Dehydrogenase and Xanthine Oxidase. The Fe2S2 clusters are shown at the center of each protein.

10.3 Electron Transfer Complexes between Cytochrome c550 and Cytochrome c Peroxidase

The objective of this work was to understand the mechanism of electron transfer between cytochrome c peroxidase from Paracoccus denitrificans with cytochrome c550 from the same organism, with horse cytochrome c (Pettigrew and others 1999; Pettigrew, Goodhew and others 2003; Pettigrew, Pauleta and others 2003), and with pseudoazurin (Pauleta 2002) the dimerization of cytochrome c550 (Pettigrew and others 1998). The importance of this work for the development of BiGGER came from the diversity of data to be considered and integrated with the simulations. This led to the development of the clustering algorithm (section 7.2), the electron transfer estimates (section 7.5), and revealed some of the shortcomings in the soft docking (section 6.3), side chain contact filter (section 6.4), and the more recent global scoring algorithms (sub-section 6.7.4). This section will focus on the generation of the complexes, which is only a small part of the effort to understanding these processes.

Electron transfer is a crucial and ubiquitous phenomenon in biological systems. Though there is still no consensus on the mechanisms that determine the specificity of these reactions, it seems that the simple lock-and-key model for protein recognition is not adequate in this case. Electron transfer complexes are likely to be fluid and dynamic, and no single model can account for this complexity.

Figure 10-4 shows families of models for the interaction of cytochrome c550 with the cytochrome c peroxidase. This is only one example, but it illustrates the approach. The interesting result from the perspective of the modeller was a clear family near the expected electron transfer heme (highlighted, near the heme marked E). But from a developer's perspective these patterns suggested several improvements, such as the usefulness of clustering models by their similarity and of additional filtering by criteria other than geometric contacts.

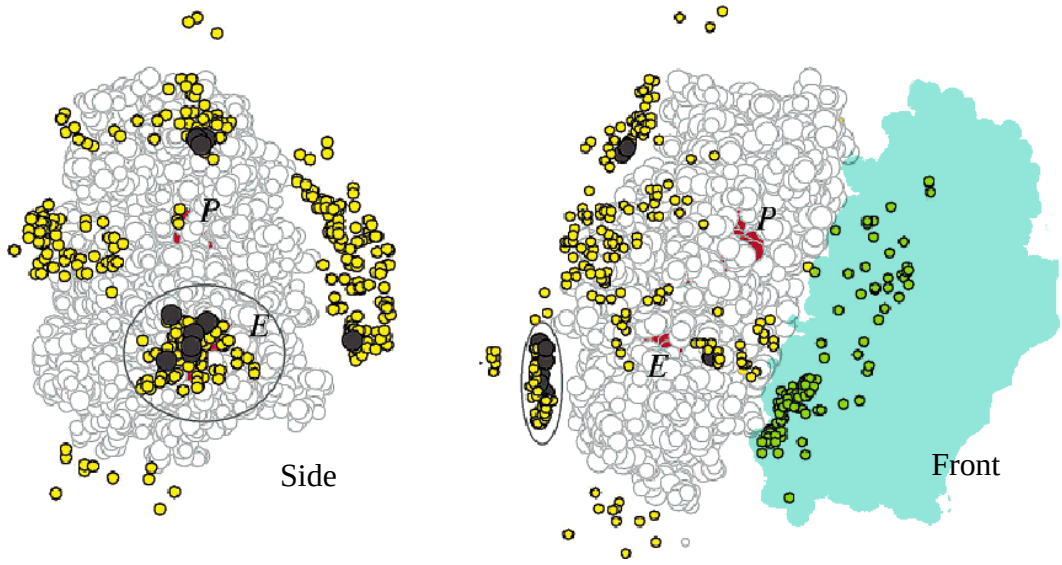


Figure 10-4 Families of docking models for cytochrome c550 and cytochrome c peroxidase. One peroxidase monomer is shown in grey, and the other monomer is shown in green in the front view. The yellow circles represent the placement of cytochrome c550 in different models. The two hemes on the peroxidase monomer are indicated by the letters E (electron transfer heme) and P (peroxidatic heme), and their atoms shown in purple. The highlighted family of solutions near the electron transfer heme contains several likely models for an electron transfer complex. Figure adapted from Pettigrew, Pauleta and others 2003.

Cytochrome c peroxidase is a dimer, and the front view (right panel of Figure 10-4) shows a large disperse family of models near the interface of the two monomers. This is due to the high surface contact scores on the groove formed at the interface, and unlikely to represent physiological complexes. The soft docking modification makes this artefact a serious problem (sections 6.2 and 6.3), and these results suggested the implementation of additional filters, such as the side chain filter (section 6.4).

Another important development motivated by the modelling work on these systems was the ability to score models according to electron transfer properties. This led to the implementation of simplified algorithm (7.5) that can give reasonable estimates of electron transfer probabilities even though the models have significant errors in geometry, due to the discrete search in 1Å and 15º steps (section 6.1).

Throughout most of the development of BiGGER and Chemera (and their predecessors) this electron transfer system has been an important test case. Some issues revealed by this application of the software are still not completely resolved: the applicability of the new scoring functions to electron transfer complexes (section 6.7) and the when to use the soft docking modifications.

10.4 Electron Transfer Complex between Cytochrome c553 and Ferredoxin

The modelling of the electron transfer complexes of cytochrome c553 and ferredoxin (Morelli, Czjzek, and others 2000; Morelli, Dolla, and others 2000) and other systems (Morelli, Palma, and others 2000) provided the framework for developing the contact score (section 7.4) and, later, for the contact filter in constrained docking (section 6.5).

The essence of the method is to find the agreement between the interaction prediction and NMR titration data. Observed perturbations on the chemical shifts and broadening NMR peaks during titration provide the experimental data. If the resonance frequencies are assigned, it is possible to identify regions in one or both partners that are affected during the formation of the complex.

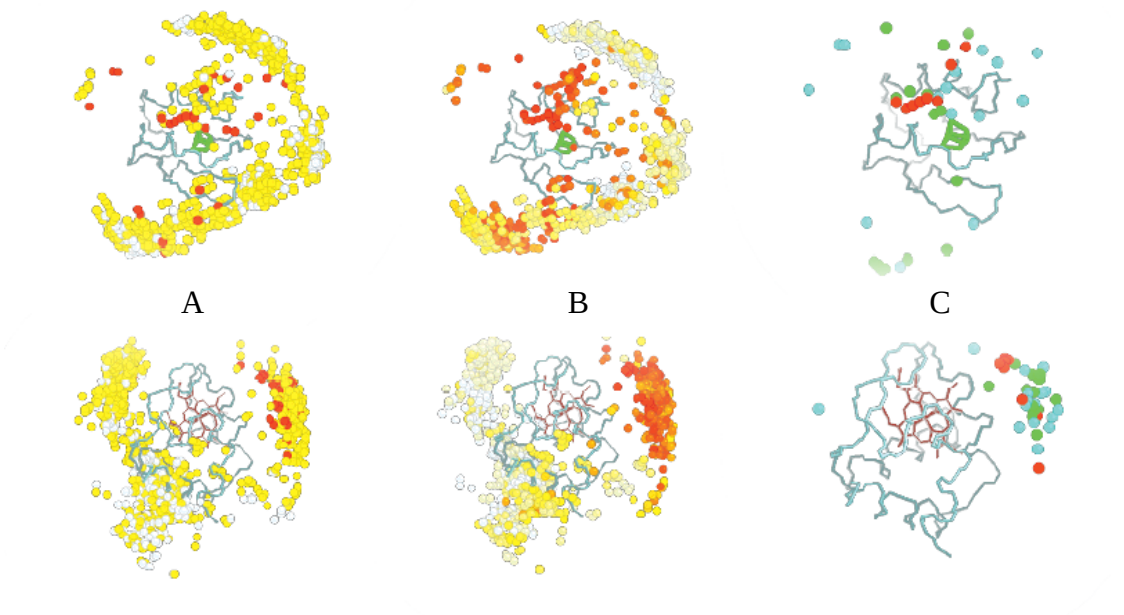


Figure 10-5 The top row shows the positions of cytochrome c553 in the best models (circles) relative to ferredoxin (shown in a backbone representation). The bottom row shows the same models but displaying the backbone of cytochrome c553 and the position of the ferredoxin as coloured circles. Red circles indicate a higher score, and columns A and B represent the best 1000 models according to the global score (column A) and the contact score from NMR data (column B). Column C shows the best models according to each score, and shows in red those models that score highest in both.

Figure 10-5 shows the results for this complex. The selection criterion uses both the global score and the contact score (A and B in the figure, respectively). These theoretical and experimental data are combined by intersecting the sets of the best models according to each score (column C).

The models were scored after the search and filtering stage, and not by using constrained docking as described in section 6.5. Constrained docking is a recent development, derived from the success of the contact score method, and still not fully tested. However, these results and those for the complex of ferredoxin with ferredoxin NADP+ reductase indicate that constrained docking is a promising approach.

10.5 Electron Transfer Complex between Ferredoxin and Ferredoxin NADP+ Reductase

Ferredoxin NADP+ reductase (FNR) catalyzes the production of NADPH from NADP+ and two electrons transferred by a ferredoxin from photosystem I. This section describes the modelling of the FNR-Fd electron transfer complex in the cyanobacterium Synechocystis sp. PCC 6803.

The structures used were both homology models, based on the homologous proteins from the Anabaena cyanobacterium (Kurisu and others 2001; Morales and others 1999).

2D ^{1}H-^{15}N-HSQC spectra provided the experimental data used to score the models (section 7.4) by counting the contacts for all residues with a ^{15}N chemical shift of over 15ppm or whose HSQC cross peak vanished during titration. The selected contact residues for the ferredoxin were: L25, L35, R40, G49, L64, H90 and K91 for line broadening during titration, and D26, G32, Y80, D84, and Y96 for the ^{15}N chemical shift. There was no NMR data on the FNR.

The data on the effect of a charge-reversal mutation of Glu94 in the Anabaena ferredoxin (Hurley and others 1997), which disrupts electron transfer between the Fd and FNR in Anabaena, provided an additional constraint to evaluate. This residue seems to stabilize the complex by forming a salt bridge with Lys75 of FNR (Navarro and others 1995), and an homologous residue in spinach ferredoxin (Glu92) can be cross-linked with FNR (Zanetti and others 1988). This data was used to score the Synechocystis FNR-Fd complex models by assuming that the homologous residue in this ferredoxin (Glu92) is an interface residue. Finally, the need for the two redox centres to be in proximity provided the third criterion to score the models produced by BiGGER.

Figure 10-6 shows the best models according to these three different scores. To represent a large number of models, the top panels show the complete structure only for FNR, representing the position of ferredoxin in each model by a single sphere located at the geometric centre of this protein. The bottom panels use the inverse representation, showing FNR as spheres and the main chain of ferredoxin.

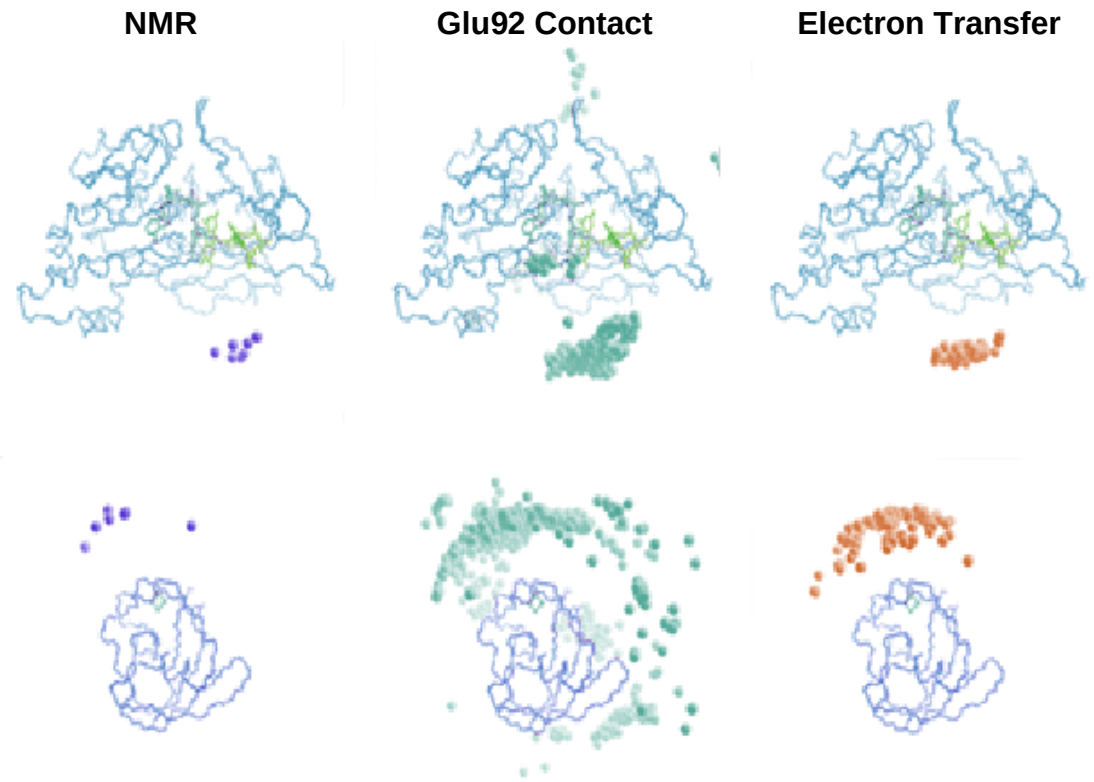


Figure 10-6 The highest scoring solutions for the Synechocystis FNR-Fd complex, using three different scores. From left to right: NMR titration data (blue), site-directed mutagenesis data (green), and electron transfer (red).

There were only two different models with high scores on all three criteria. Because both models had the same global score of 2.6 (see sub-section 6.7.4), it was not possible to select only one, so there are two alternative models for this electron transfer complex (Figure 10-7).

BiGGER produced similar results for the Anabaena and maize complexes. Since the crystallographic structure of these complexes was available, these docking simulations served to validate the method, as the models produced had a RMSD from the known structure of 4.9Å for Anabaena and 2.5Å for maize.

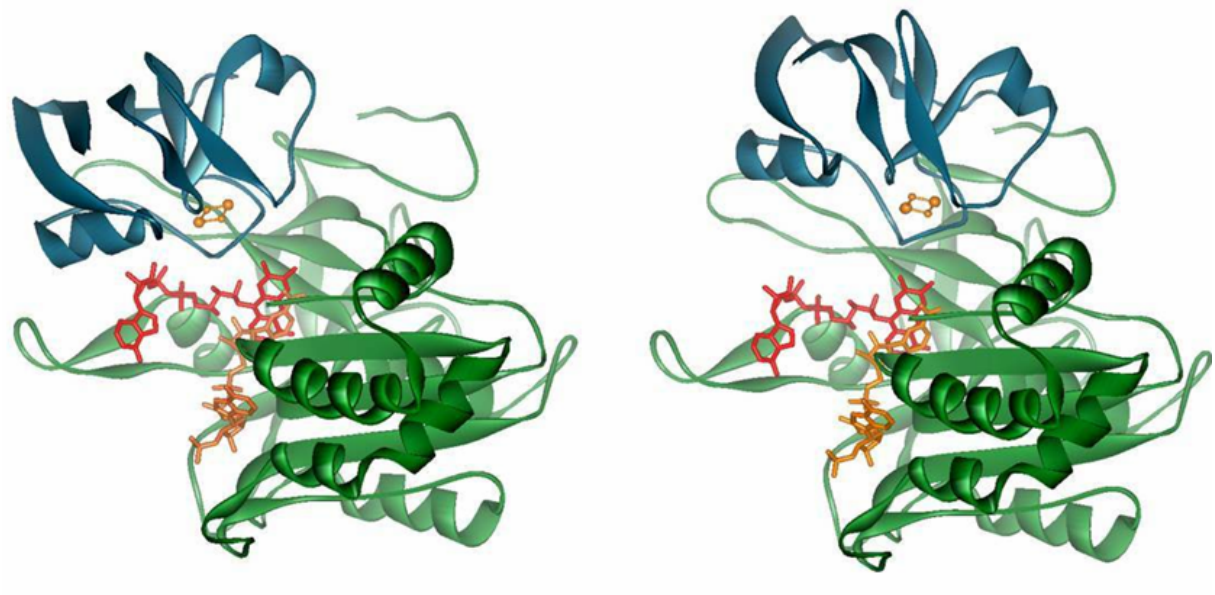


Figure 10-7 The two models for the Synechocystis FNR-Fd complex. The ferredoxin is shown in blue, on top, and the FNR in green.

10.6 Electron Transfer Complex between Cytochrome b5 and Cytochrome C

Another test case for the contact score approach, using NMR titration data on both docking partners (Banci and others 2003). The modelling approach and results were similar to those described in sections 10.4 and 10.5.

10.7 The CAPRI Experiments

The Critical Assessment of PRediction of Interactions (CAPRI) experiments (Janin 2002, Janin and others 2003) ran in three rounds from 2001 to 2003. The organizers collected a total of 9 protein complex structures provided by crystallographers, and distributed the docking partners to the contestants in random orientations. The goal was to predict the configuration of the complex prior to the publication of its structure by the crystallography teams who supplied the targets[1]. This section outlines our participation in CAPRI (Krippahl and others 2003), the results of the experiment, and the implications for the integration of experimental data in these simulations.

[1] CAPRI documentation uses "target" to refer to the complexes to model, instead of to the molecule that remains fixed during docking (see section 6.1). In this section, when "target" is followed by a number (e.g. target 7) it refers to the CAPRI complexes, otherwise it means the largest of the docking partners.

Of the nine CAPRI targets, we attempted to predict seven. The docking algorithm was only operational close to the deadline for the first round of CAPRI, so there was no time to model targets 1 and 3. Targets 4, 5, 6, 8, and 9 were modelled using the soft docking geometrical search (see 6.3). Target 7 was modelled with hard docking, which is done by disabling the core grid alterations and the side chain contact filter described in sections 6.3 and 6.4. These simulations used version 2.0 of BiGGER. Target 2 was modelled during the development of BiGGER, and the different parameters and implementation details result in a slight difference between the models submitted and the models that would be obtained using the current version of the algorithm.

In addition to using hard docking, the T cell receptor protein in target 7 was truncated: all residues up to D118 were used; those from L119 onward were discarded during docking. This was done because the V shape of this structure was favouring contact at the inner portion of the hinge, in detriment to the alternatives. The models submitted to CAPRI were obtained by fitting the complete structure with the fragment used in docking. This was the only case where additional information was used to generate the models, since we assumed that the region from L119 onward was not important for docking.

For CAPRI rounds one and two, we were to submit five models for each target. We clustered all models using a 3Å cutoff value (see section 7.2), and considered only the highest scoring model in each cluster to represent the whole cluster. We submitted the five highest scoring representatives, except when visual inspection indicated that two models were too similar. In this case, we replaced the lowest scoring with the highest scoring representative outside the selected group.

For round 3, targets 8 and 9, we submitted 10 models, ranked from most likely to least likely. We increased the clustering threshold to 5Å and 7Å for target 8 and target 9 respectively, to try to give a more representative sample of the models retained by BiGGER in the set of 10 candidates submitted. However, the overall performance was similar to that of the previous rounds.

If we evaluate the models we submitted using the RMSD from the target complex, only one prediction was accurate (target 6). In all other cases, the RMSD value was too high. Finding the right model in such a small set is difficult for any docking algorithm, as the overall results of the CAPRI experiment show. A less stringent criterion is to evaluate the interface residues correctly predicted in each model, one of the criteria used by the CAPRI evaluation team. A model can deviate significantly from the target structure, but still accurately predict the interface regions on both partners; for example, if the partners are misaligned by a rotation around an axis

perpendicular to the interface plane. This still provides valuable information on the interaction, and it may even be a better model of the real complex than a single crystal structure, since protein complexes may be more dynamic than the static image crystallography often provides.

Figure 10-8 shows a comparison of the results obtained by all the groups participating in CAPRI. The score attributed to each model is the fraction of correct interface residues in the model. As an example, one of our models for target 2 predicted 22 out of 27 interface residues on one protein, and 2 out of 27 on the other partner. The score for this model is (22+2)/(27+27), or 0.44.

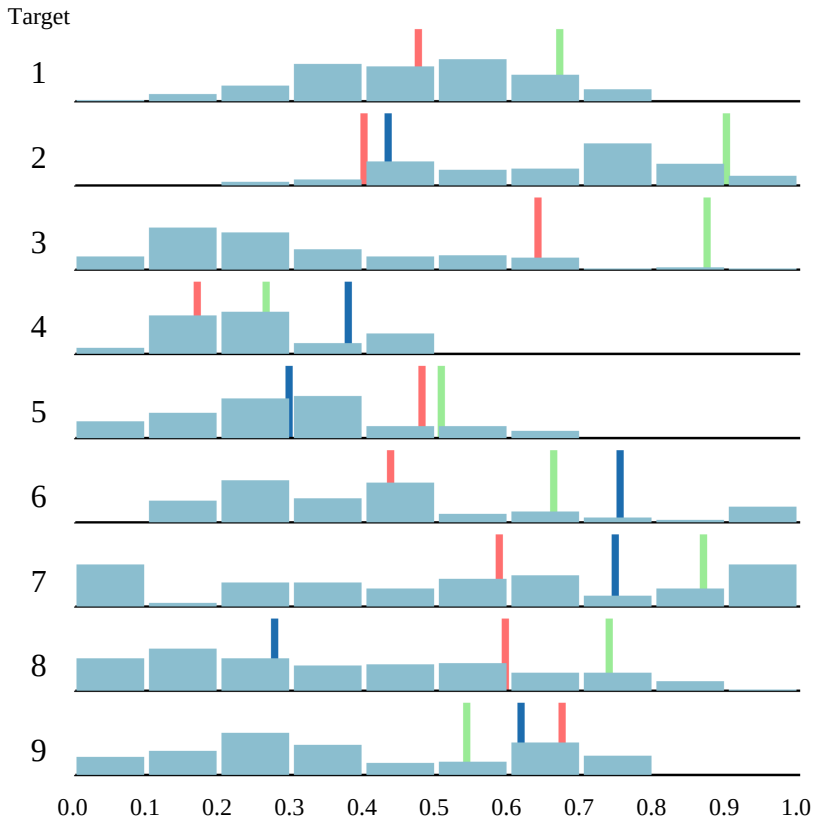


Figure 10-8 Summary of the results for all CAPRI participants. The histograms represent all models submitted, evaluated by the fraction of correct residues at the interface. The red bars indicate the best models of the group with the lowest average, the green bars the best models of the group with the highest average, and the blue bars the best BiGGER models.

The nine histograms, one for each target, show the number of models scoring from 0 to 0.1, from 0.1 to 0.2, and so forth, for all models submitted by all groups. The thin bars indicate the best models generated by BiGGER (thin blue bars), and the best models generated by the highest and lowest scoring of the groups that participated in all three CAPRI rounds (thin green and red bars, respectively). The group scores are the average across the targets of the best score for each target.

These results show no clearly superior method; the highest scoring method produces better results than BiGGER only in four out of seven cases, and is not significantly better than the lowest scoring method in three out of nine targets. The most important factor determining the accuracy of the best models seems to be the nature of the target to model, and not the method.

The implication is that purely theoretical prediction of protein complexes is not reliable, at least with current methods. The alternative is the integration of experimental information, such as data on interface residues. To test the hypothesis that this integration improves results, the author simulated these data on the CAPRI targets by randomly selecting four interface residues; three residues from the larger partner (the target in the docking simulation) and one residue from the other (the probe in the docking simulation. Table 9 (p. 157) lists the proteins on CAPRI and shows the selected residues. The models were then scored by counting the number of atoms, in the 3 amino acids chosen on the target, that were within 5Å of any atom on the probe plus the number of atoms, in the amino acid chosen on the probe, that were within 5Å of any atom on the target.

These scores meant to simulate the effects of experimental data on the quality of the models. This is the sort of scoring that could be used if some amino acids were known to be important for complex formation (e.g. by site directed mutagenesis), or present at the interface (e.g. by NMR data).

Table 9 (p. 157) indicates the structures in each target and the residues chosen to simmulate contact data. No residues were picked for target 2 because there were no models in the final set close enough to the target structure. This experiment could not include targets 8 and 9 because the structures of the complexes were not available at the time.

Table 10 (p. 157) summarizes the results, showing the rank and RMSD value for the highest ranking acceptable solution, or the best if no acceptable solution was kept. The last four columns show the same results using simulated information on the interface region. In some cases, there were several models with the same number of contacts. The numbers in parentheses indicate the number of models tied for first place in these cases.

An acceptable model was any model with less thatn 2.5Å of RMSD from the target structure, or the model with the lowest RMSD if there were none below 2.5Å and the model showed the two partners in an approximately correct configuration.

The closest model to the X-Ray structure for target 2 has an RMSD of 10Å. This is not an acceptable model at all, and, in this case, additional information could not improve these results because there was no good model in the set of 5000 structures kept by BiGGER.

For target 4, of three clusters of models with less than 2.5Å RMSD from the X-Ray structure (ten individual structures in total), the one with the highest global score was in position 1150. The highest scoring models all had RMSD values over 10Å, so in this case it would not be possible to identify an accurate model without additional information. Scoring for contacts with the three residues chosen for the target molecule, we find an acceptable model among the 22 models tied for the first place with 25 contacts (RMSD of 1.2Å). Scoring for all four amino acids we find a model with a 3.3Å RMSD in the 15 top scoring models (all with 49 contacts). Figure 10-9 compares these three models with the crystallographic structure.

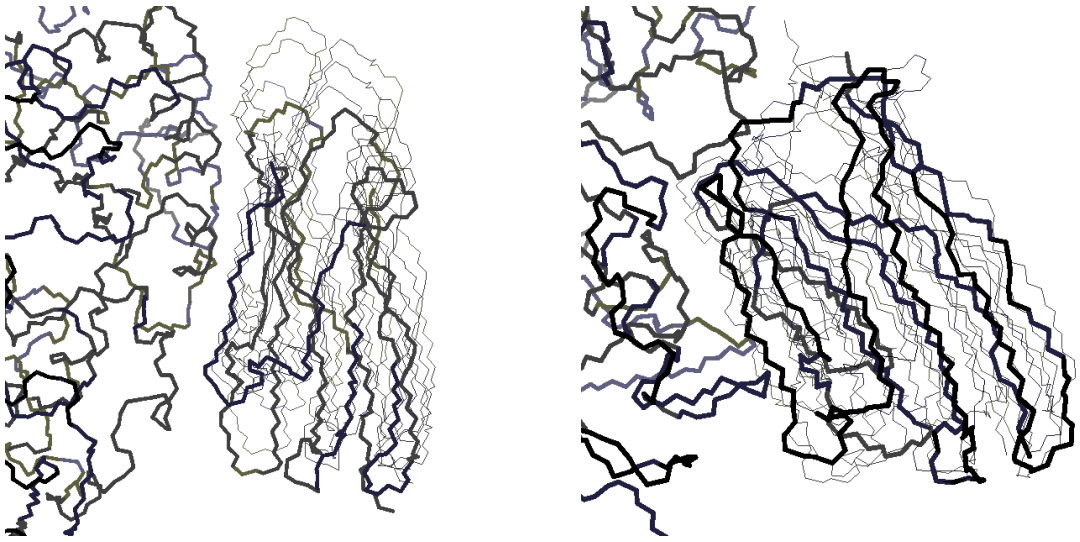


Figure 10-9 Models for target 4 (left panel) and target 5 (right panel). The backbone of the crystallographic structure is shown in thick lines and the backbone of the docking models in thin lines. The largest partner, on the left on each panel, is in the same position for all models.

The closest model to target 5 had a RMSD of 3.4Å, and ranked in position 1374 by the global score. The ranking of this model improved to 49 with the contact scores for the 3 amino acids chosen on the target molecule, and then to 43 when considering all 4 amino acids. Even when sorting according to the contact scores, the high ranking models had large RMSD values: 6.6Å for the first position (18 models tied with 25 contacts) using three amino acids for the contact score, 5.4Å in the second place (the best of the first five, with 40 contacts) when using all four amino acids. Figure 10-9 shows these three models and the crystallographic structure of CAPRI target 5.

A 2.1Å RMSD model for target 6 ranked in eighth place on the global score. Because 3 higher ranking models due were similar to others in the set to submit, they were discarded and the eighth ranking model was included, and this turned out to be the most successful prediction we submitted. Adding the contact information for the three amino acids on the target (see Table 1) promoted a worse model to the top rank (3.3Å RMSD), one of 63 models with the 41 contacts, but the difference is not significant. Adding the contact score for R51, on the probe, reduced this set to 6 models tied for the first rank, with the 3.3Å RMSD model still the best among them, but with a good model (1.0Å RMSD) at position 15. Figure 10-10 shows these three models.

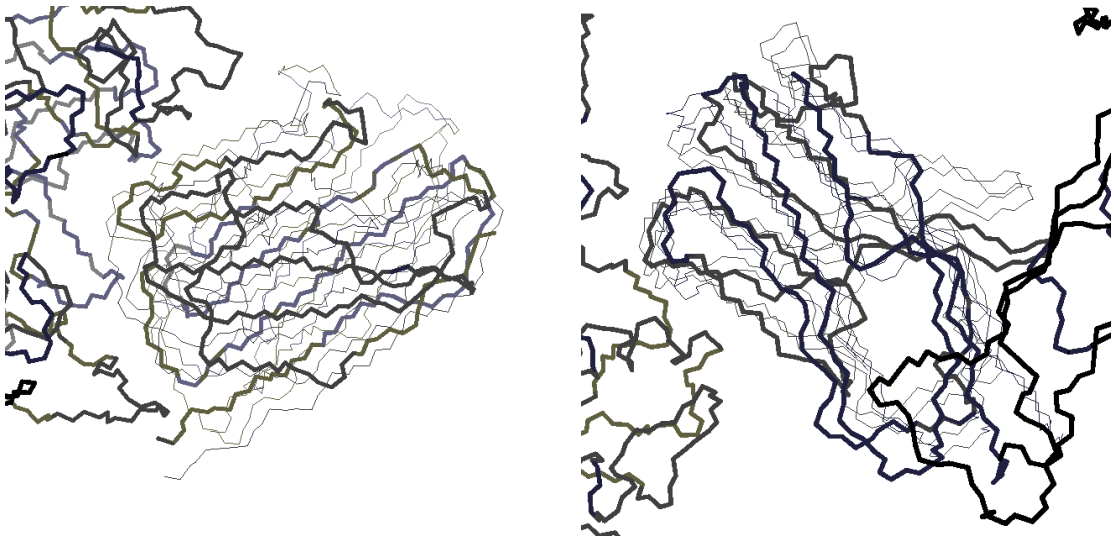


Figure 10-10 Models for target 6 (left panel) and target 7 (right panel). The backbone of the crystallographic structure is shown in thick lines and the backbone of the docking models in thin lines. The largest partner, on the left on each panel, is in the same position for all models. Note that the probe molecule for CAPRI target 7 was truncated before docking, and the docking models show only the fragment used for docking.

For target 7, a 1.9Å RMSD at 21 on the global score, but there was no similarly accurate model in the 5 candidates submitted. Using the contact scores for the 3 amino acids on the target promoted another good model (1.8Å RMSD) to third place, and adding the contact score for the probe amino acid brought it to second place on the contact ranking. These models are shown in Figure 10-10. Note that the chain of the probe (T Cell Receptor protein) is shown only up to D118, because only this fragment was used in this docking.

These results indicate that docking simulations can benefit significantly from even a little additional information on the residues or regions critical for docking. There are between 38 and 66 interface residues in these complexes, (targets 7 and 6, respectively), and knowing only 5-10% of these was enough to produce accurate models.

By themselves, these results are not conclusive. Picking 4 amino acids from the interface region is not a good substitute for experimental data. But these results agree with those obtained using NMR data (Morelli, Dolla, and others 2000; Morelli, Czjzek, and others 2000; Morelli and others 2001).

In cases like target 2, where the initial filtering stage eliminates all acceptable models, there is no possibility of promoting correct models because they are no longer available. It would be interesting to re-run this experiment using the constrained docking algorithm as described in section 6.5

10.8 Combining PSICO with Multidimensional Scaling

This section describes the combination of PSICO with a Multidimensional Scaling (MDS) algorithm, a general-purpose solver for distance geometry problems. The work was done in collaboration with Michael Trosset (Krippahl, Trosset and Barahona 2001; Trosset 2000).

The objective was to show how the combination of these two approaches could help solve problems where each method alone performs poorly. Though all test problems were protein structure determinations, there is nothing in the combined method that restricts it to proteins or even molecules. The method can be applied to the general problem of finding a configuration of points in an N-dimensional space from a set of constraints on distances between points.

Section 5.2 describes how PSICO processes such distance constraints. Though the algorithm operates in three-dimensions, it is trivial to extend it to the general case of an N-dimensional problem. Both the domain representation (section 5.1) and the propagation of constraints (section 5.2) can be applied in N dimensions, as long as the distance metric is the same.

The two basic concepts behind the MDS method are the distance matrix and the dissimilarity matrix. A dissimilarity matrix is a symmetric $n \times n$ matrix in which all diagonal elements are zero, and all other elements are positive or zero. A symmetric $n \times n$ matrix $\Delta = (d_{ij})$ is a p-dimensional Euclidean distance matrix if there exists a configuration of n points in p dimensions such that d_{ij} is the distance between point i and point j.

The data stored in a dissimilarity matrix represent information about the dissimilarity of each pair of objects. In classical applications of MDS, these objects can be psychometric data points (such as

colour preferences, for example) or statistical measures. The objective of metric MDS is to determine the configuration of objects that corresponds to the dissimilarity matrix.

It is possible to determine the configuration of points from the distance – in other words, if all the distances are known exactly (Torgerson 1952). For the specific problem of determining the structure of a protein, this amounts to knowing the exact distances between all atom pairs.

But this information is not readily available. What the experimental data provide is a set of upper and lower bounds for all the distances. The method proposed by Trosset (2000) aims at finding a dissimilarity matrix, bound by the matrixes of the upper and lower distance bounds, which is the most similar to a distance matrix.

The experiments reported in this section used a limited memory algorithm (Byrd and others 1995) to find this matrix and thus the configuration of atoms in the protein from the set of distance constraints. Implementation used was the L-BFGS-B, version 2.1 (Zhu and others 1996).

The strength of the constraint programming approach is its ability to find an approximate solution (in which some constraints are violated) very quickly. However, as one demands solutions that violate fewer and fewer constraints, the computational expense grows exponentially. In contrast, the most difficult aspect of the MDS approach is finding good initial dissimilarities from which to begin the optimisation. Although this is not a substantial difficulty in simple problems with a small number of atoms, experiments with larger molecules and more realistic problems suggest that it will become so as the number of atoms increases. Furthermore, in real life problems only a relatively small set of interatomic distances are known in advance to be constrained, which makes it necessary to propagate these constraints to obtain enough information to generate a good initial dissimilarity matrix.

We tested this combination of PSICO with the MDS algorithm on known protein structures. For PSICO, we used only the constraint propagation stage and binary distance constraints; there was no optimisation or rigid group constraints. Two structures (Mutant P53 Anti-Oncogene and Phosphotransferase, PDB codes 1AU1 and 1PHO respectively) were obtained by Nuclear Magnetic Resonance (NMR) spectroscopy. The remaining three (Desulforedoxin, Bovine Trypsin inhibitor and Barstar, PDB codes 1DXG, 1BPI and 1BRS respectively) are X-Ray structures.

All atom pairs closer than 7Å were used to generate a constraint pair, where the distance between two atoms is constrained by both a lower and upper boundary. The values for the boundaries were taken to be the distance value in the known structure plus or minus 0.1Å.

This method provides a large number of very tight constraints. The number of constraints expected from an actual NMR experiment would be an order of magnitude smaller, and the distance boundaries at least an order of magnitude larger than the ones used for these tests. Although this way we are not accurately simulating a real NMR structural determination, these 'tighter' problems allow for a more stringent test on the performance of the algorithm.

PSICO was run on a PII processor at 300MHz, and the calculations took between 20 seconds (Desulforedoxin) and 2 minutes (Barstar). The MDS stage was run on a PIII processor at 600MHz, with the calculations taking between 15 minutes (Desulforedoxin) and 6 hours (Barstar), although computation time can be reduced by allowing for less accurate solutions.

The figures below (10.11 to 10.15) show the name of the protein, the number of atoms and of the distance constraint pairs (upper an lower bound) taken from the known structure. These were all interatomic distances below 7Å and the boundaries were considered to be equal to the interatomic distance $\pm$0.1Å. For all other distance constraints, for a total of $n\times(n-1)$ pairs, the boundaries were considered to be 7.0Å and infinity (100.0Å).

Each case also shows the quality of the PSICO and MDS structures by measuring the root mean square deviation (Rmsd) from the target structure and the average constraint violations for the constraints taken from the structure. The violations that can still be observed are caused by the premature termination of the optimisation. Since the dissimilarity matrix when the process is terminated is still significantly different from a configuration matrix, when the nearest configuration matrix is found constraints are still violated. Given enough time the program would converge to a solution with no constraint violations, but the trade-off between accuracy and calculation economy is still being under study, so these results show a tentative exploration of the possibilities.

As these results show, the combination of these two complementary approaches makes it possible to solve problems of a size and with a precision beyond the practical capabilities of each method alone, for neither can PSICO reach an accurate solution in an acceptable time, nor can the MDS algorithm do so reliably without an approximate initial guess.

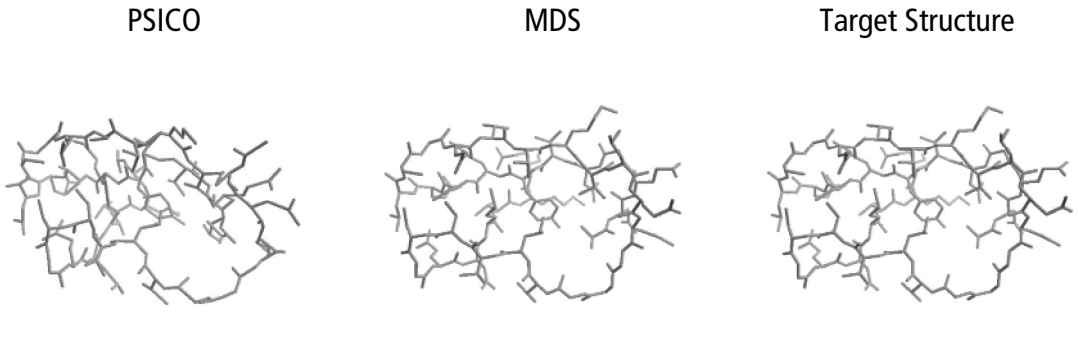


Figure 10-11 Desulforedoxin Monomer (Archer and others 1995). 260 Atoms, 5951 Constraint pairs. The RMSD from the crystallographic structure was 2.1Å for PSICO and 0.02 Å for MDS. The average constraint violations were, respectively, 0.82 Å and 0.04 Å

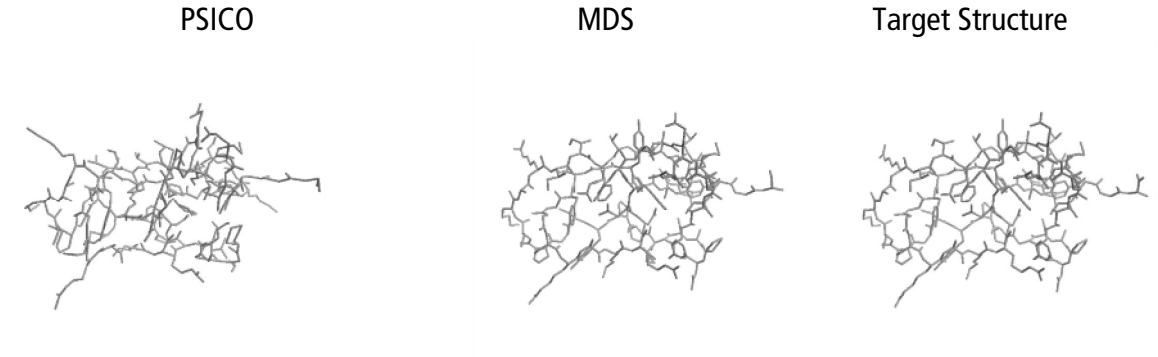


Figure 10-12 Trypsin Inhibitor 448 Atoms, 11613 Constraint pairs. The RMSD from the crystallographic structure was 2.8Å for PSICO and 0.02 Å for MDS. The average constraint violations were, respectively, 0.73 Å and 0.05 Å

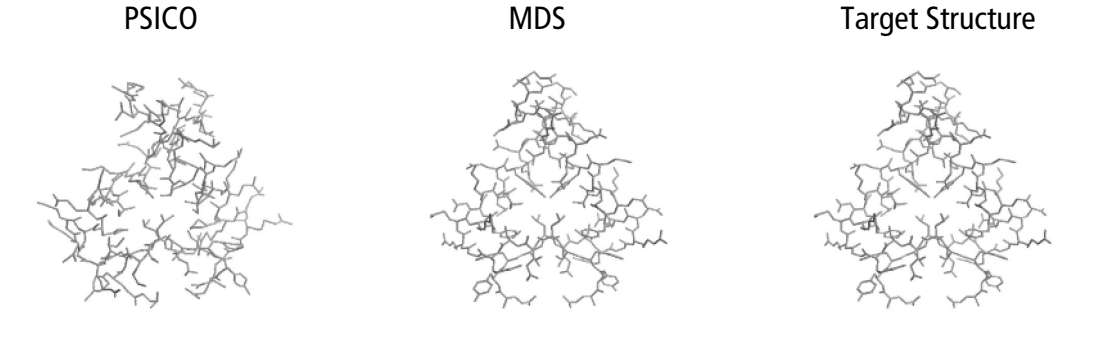


Figure 10-13 Mutant P53 Anti-Oncogene (McCoy and others 1997): 514 Atoms, 12938 Constraint pairs. The RMSD from the crystallographic structure was 2.5Å for PSICO and 0.02 Å for MDS. The average constraint violations were, respectively, 0.68 Å and 0.05 Å

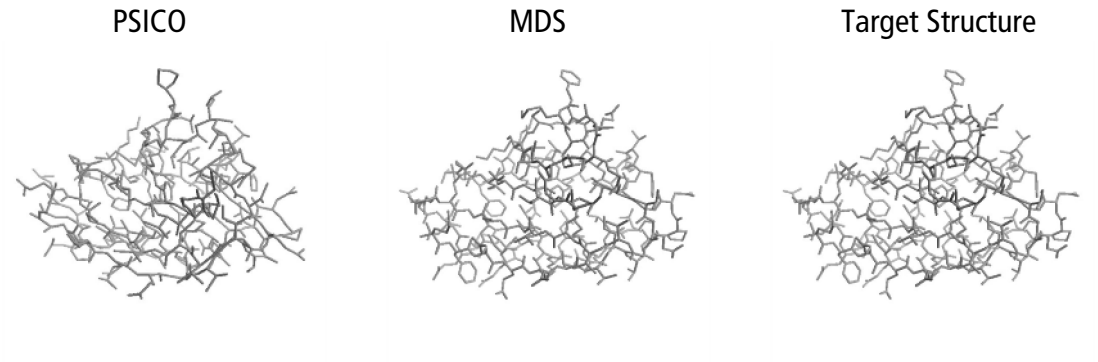


Figure 10-14 Phosphotransferase (Jia and others 1993): 639 Atoms, 17206 Constraint pairs. The RMSD from the crystallographic structure was 2.8Å for PSICO and 0.02 Å for MDS. The average constraint violations were, respectively, 0.67 Å and 0.01 Å

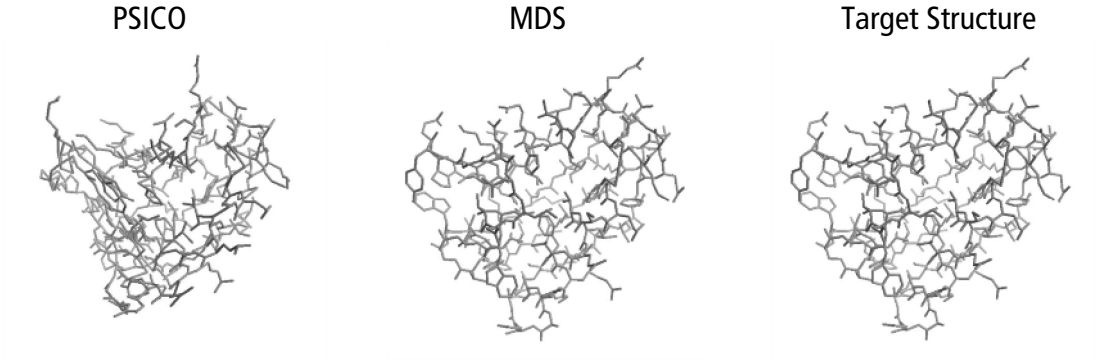


Figure 10-15 Barstar Mutant (Buckle and others 1994) 693 Atoms, 18996 Constraint pairs. The RMSD from the crystallographic structure was 4.1Å for PSICO and 0.02 Å for MDS. The average constraint violations were, respectively, 0.76 Å and 0.01 Å.

These results show the potential of combining constraint propagation with local search in order to solve problems that neither technique alone can solve efficiently. However, the examples shown in the previous section are not a realistic representation of the experimental data available in real life structural NMR problems. Although we know that the PSICO algorithm can perform satisfactorily in real life situations (Krippahl and Barahona 2000), the MDS algorithm was not yet tested with experimental data. Nevertheless, the MDS optimisation algorithm gave extremely accurate solutions to the problems tested so far.

Regarding the efficiency of the method in the particular problem of protein structure determination from NMR data, at this moment we do not know what is the best trade-off between the time to run the MDS stage (the PSICO stage is very fast) and the quality of the solution produced. The calculation times for the examples in the previous section were between 20 minutes and 3 hours, but the accuracy obtained in most cases is still significantly higher than is necessary in real problems. It is possible that the MDS method is more efficient than the torsion angle minimization described in section 5.8 and by Guntert and others (1997). Furthermore, the MDS method has the

added advantage of being a generic distance geometry method, and thus independent of structural features particular to proteins and other macromolecules, and not restricted to three-dimensional problems.

The MDS method cannot solve these problems reliably by itself, because the results are dependent on the starting point for the local search. Without using PSICO to provide the initial guess, MDS would often stop at a completely incorrect solution.

11 Concluding Remarks

> One never notices what has been done; one can only see what remains to be done.
>
> Maria Sklodowska-Curie

The work presented here is a starting point for an approach to protein structure modelling. It is not likely that it will ever be finished, for the integration of experimental data in theoretical prediction methods is always subject to improvements in both areas, and in new ways of using old techniques.

BiGGER has been in use by several research groups for some years now, and the experience in developing the algorithm alongside those using it in different ways has shown how important it is for those working with the theoretical models to do so in close collaboration with those solving the real problems.

Chemera has also benefited from the input of many kind and patient colleagues who helped test the software as it developed. Version 2.0 has been distributed on the Internet to hundreds of users, and feedback has been encouraging so far.

BiGGER and Chemera were a major achievement of this work; two useful tools with demonstrated applicability to modelling protein interactions.

PSICO is still at an earlier stage. While BiGGER can be used in parallel, independently of other research efforts to obtain information about protein interactions, PSICO must be made part of the process for the determination of protein structures. It must interface with other applications for data acquisition and pre-processing, and must do so without inconveniencing the user. This raises programming and design problems that are not very relevant for this work, and thus the author decided to postpone this task and focus instead on the perfecting the algorithm as much as could be done without working on real cases.

Once PSICO starts being used on protein structure research it is bound to suffer many changes and improvements, for there are several open problems to solve. The most efficient ways to combine the propagation binary distance constraints and rigid group constraints is likely to depend on the constraints involved, and so can only be properly determined with real problems. The same is true

for search heuristics, and of the possibility of using local search during the constraint processing stage instead of only afterwards.

Much can also be done to improve BiGGER. The scoring functions that evaluate the models retained would benefit from a better parameterization using a larger number and different types of complexes, and the integration of experimental data in the search and filtering stage must still be used in real problems to identify any possible weaknesses or ways to improve the algorithm and its application.

The possibility for improvement is, perhaps, the greatest strength of the approach. The tools produced so far and described in this document are efficient at solving protein structure and interaction problems, but the ultimate goal is not to solve a specific problem; rather, it is to solve a general and open-ended problem. In a sense, the objective is to solve the problem of solving protein structural problems. This makes development an integral part of the method, constantly responding to the needs of researchers and to innovative uses of techniques and data.

Hopefully, this will keep the author employed and happily busy for years to come.

References

Cited References

Abagyan R., Batalov S, Cardozo T, Totrov M, Webber J, Zhou Y. 1997. Homology Modeling With Internal Coordinate Mechanics: Deformation Zone Mapping and Improvements of Models via Conformational Search. Prot. Struc. Func. Gen, Suppl. 1:29–37 (1997)

Abe H, Braun W, Noguti T, Go N. 1984. Rapid calculation of first and second derivatives of conformational energy with respect to dihedral angles for proteins. General recurrent equations. Computers & Chemistry, II-4, 239-247, 1984.

Archer M, Huber R, Tavares P, Moura I, Moura JJ, Carrondo MA, Sieker LC, LeGall J, Romao MJ. 1995.Crystal structure of desulforedoxin from Desulfovibrio gigas determined at 1.8 A resolution: a novel non-heme iron protein structure. J Mol Biol. 1995 Sep 1;251(5):690-702.

Arun K, Huang T, Blostein S.1987. Least-squares fitting of to 3-D point sets. IEEE Transactions on Pattern Analysis and Machine Intelligence, 9(5):698-700.

Asturby WT and Street A. 1932. X-Ray Studies of the Structure of Hair, Wool and Related Fibres. Philosophical Transactions of the Royal Society of London A 230:75-101.

Backofen R, Narayanaswamy NS, Swidan D. 2002. Protein similarity search under mRNA structural constraints: application to selenocysteine incorporation. In Silico Biology 2: 26

Backofen R, Will S, Bornberg-Bauer E. 1999. Application of constraint programming techniques for structure prediction of lattice proteins with extended alphabets. Bioinformatics. 1999 Mar;15(3):234-42.

Backofen, R, 1998. Constraint Techniques for Solving the Protein Structure Prediction Problem, CP98, Lecture Notes in Computer Science, Springer-Verlag, V. 1520: 72, 1998

Backofen, R. 2001. The Protein Structure Prediction Problem: A Constraint Optimisation Approach using a New Lower Bound, Constraints, V.6: 2&3: 223

Bailey-Kellogg C, Widge A, Kelley JJ, Berardi MJ, Bushweller JH, Donald BR. 2000. The NOESY jigsaw: automated protein secondary structure and main-chain assignment from sparse, unassigned NMR data. J Comput Biol. 2000;7(3-4):537-58.

Balabin IA, Onuchic JN. 2000. Dynamically Controlled Protein Tunneling Paths in Photosynthetic Reaction Centers. Science; 2000; 290 (5489); 114

Balabin, IA, Onuchic JN. 1998. A New Framework for Electron Transfer Calculations - Beyond the Pathways-like Models. J. Phys. Chem. B; 1998; 102 (38);7497

Baldi P, Brunak S.1998. Bioinformatics: the machine learning approach. Dietterich T, editor.MIT Press, 351p.

Banci L, Bertini I, Felli IC, Krippahl L, Kubicek K, Moura JJ, Rosato A. 2003. A further investigation of the cytochrome b5-cytochrome c complex. J Biol Inorg Chem. 2003 Sep;8(7):777-86. Epub 2003 Jul 19.

Bartak R. 1999. Constraint Programming -- What is behind?. Proceedings Workshop CP in Decision and Control 1999 (Figwer J editor)

Beratan DN, Onuchic JN, Hopfield JJ. 1987. Electron tunneling through covalent and non-covalent pathways in proteins," J. Chem. Phys. 86 (8), 4488

Bessiere C, Freuder EC, Regin JR. 1999. Using constraint metaknowledge to reduce arc consistency computation. Artificial Intelligence 107, pp 125-48

Bessiere C. 1994. Arc-consistency and arc-consistency again. Artificial Intelligence 65, pp179-90

Betts MJ, Sternberg MJE. 1999. An analysis of protein conformational changes on protein-protein association: implications for predictive docking. Prot Eng 1999;12:271–283.

Bhansali S, Kramer GA, Hoar TJ. 1996. A Principled Approach Towards Symbolic Geometric Constraint Satisfaction Journal of Artificial Intelligence Research 4 (1996) 419-443

Bloembergen N, Purcell EM, Pound EV. 1948. Relaxation effects in nuclear magnetic resonance absorption. Phys. Rev. 73, 679-712.

Bolton, W., Perutz, M. F. 1970. Three dimensional fourier synthesis of horse deoxyhaemoglobin at 2.8 Angstrom units resolution. Nature 228 pp. 551 (1970)

Bracci L, Pini A, Bernini A, Lelli B, Ricci C, Scarselli M, Niccolai N, Neri P. Biochemical filtering of a protein-protein docking simulation identifies the structure of a complex between a recombinant antibody fragment and alpha-bungarotoxin. Biochem J. 2003 Apr 15;371(Pt 2):423-7.

Bragg WL and Perutz MF. 1954. The structure of haemoglobin: Fourier projections on the 010 plane. Proc. R. Soc. Lond. A, 225, 315–329.

Branden C, Tooze J, 1999, Introduction to Protein Structure, 2nd Ed. Garland

Brown K, Nurizzo D, Besson S, Sheppard W, Moura J, Moura I, Tegoni M, Cambillau C. 1999. MAD Structure of Pseudomonas nautica dimeric cytochrome C552 mimics the C4 dihemic cytochrome domain association, J.Mol.Biol, V.289, 1017

Brumlik MJ, Leroy G, Bruschi M, Voordouw G. 1990. The nucleotide sequence of the Desulfovibrio gigas desulforedoxin gene indicates that the Desulfovibrio vulgaris rbo gene originated from a gene fusion event. J. Bacteriol. 172, 7289-7292.

Brünger AT, Adams PD, Clorec GM, DeLanod WL, Grose P, Grosse-Kunstlevea RW, Jiang JS, Kuszewskic J, Nilgesg M, Pannuh NS, Readi RJ, Rice LM, Simonsonj T, Warren GL. 1998. Crystallography & NMR System: A New Software Suite for Macromolecular Structure Determination Acta Cryst. (1998). D54, 905-921

Buckle AM, Schreiber G, Fersht AR. 1994. Protein-Protein Recognition: Crystal Structural Analysis of a Barnase-Barstar Complex at 2.0-A Resolution Biochemistry V. 33 8878

Byrd RH, Lu P, Nocedal J, Zhu C. 1995. A limited memory algorithm for bound constrained optimization. Journal on Scientific Computing, 16:1190—1208.

Chen R, Mintseris J, Janin J, Weng Z (2003) A Protein-Protein Docking Benchmark, Proteins 52:88-91

Chen R. & Weng Z. 2002 Docking Unbound Proteins Using Shape Complementarity, Desolvation, and Electrostatics, Proteins 47:281-294

Chothia C, Lesk AM. 1986. The relation between the divergence of sequence and structure in proteins. EMBO J. 1986 Apr;5(4):823-6.

Coelho AV, Matias P, Fulop V, Thompson A, Gonzales A, Carrondo MA. 1997. Desulfoferrodoxin structure determined by MAD phasing and refinement to 1.9 Angstrom resolution reveals a unique combination of a tetrahedral FeS4 centre with a square pyramidal FeSN4 centre. J. Biol. Inorg. Chem, V2 507

Coelho AV, Matias PM, Carrondo MA, Tavares P, Moura JJ, Moura I, Fulop V, Hajdu J, Le Gall J. 1996. Preliminary crystallographic analysis of the oxidized form of a two mono-nuclear iron centres protein from Desulfovibrio desulfuricans ATCC 27774. Protein Sci. 1996 Jun;5(6):1189-91.

Cooley JW, Tukey OW. 1965. An Algorithm for the Machine Calculation of Complex Fourier Series. Math. Comput. 19, 297-301, 1965.

Cornell WD, Cieplak P, Bayly CI, Gould IR, Merz Jr KM, Ferguson DM, Spellmeyer DC, Fox T, Caldwell JW, Kollman PA. 1995. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. J. Am. Chem. Soc. 117, 5179-5197

Crowley PB, Rabe KS, Worrall JAR, Canters, GW, Ubbink M. 2002. Complex of Cytochrome f and Cytochrome c: Identification of a Second Binding Site and Competition for Plastocyanin Binding. ChemBioChem 3:526-533

Darby NJ, Creighton TE, 1993, Protein Structure: In Focus, Oxford University Press

Debye P, Hückel E. 1923. Zur Theorie der Elektrolyte. I. Gefrierpunktserniedrigung und verwandte Erscheinungen. Physik. Z. 24, 185.

Degtyarenko KN, North ACT, Findlay JBC .1999. PROMISE: a database of bioinorganic motifs. Nucleic Acids Res. 27, 233-236.

Deisenhofer J, Epp O, Miki K, Huber R, Michel H. 1984. Electron density map at 3 A resolution and a model of the chromophores of the photosynthetic reaction center from Rhodopseudomonas viridis. J Mol Biol. 1984 Dec 5;180(2):385-98.

Desmyter A, Spinelli S, Payan F, Lauwereys M, Wyns L, Muyldermans S, Cambillau C. 2002. Three camelid VHH domains in complex with porcine pancreatic a-amylase. J. Biol. Chem. 277:23645-50

Dobbek H, Gremer L, Meyer O, Huber R, 1999, Crystal Structure and Mechanism of CO Dehydrogenase, a Molybdo Iron-Sulfur flavoprotein containing S-Selanylcysteine. Proc.Nat.Acad.Sci.USA V. 96 8884.

Doig AJ, Sternberg MJ. Side-chain conformational entropy in protein folding. Protein Sci. 1995 Nov;4(11):2247-51.

Dominguez C, Boelens R, Bonvin AM. 2003.HADDOCK: a protein-protein docking approach based on biochemical or biophysical information. J Am Chem Soc. 2003 Feb 19;125(7):1731-7.

Duan Y, Kollman PA. 1998. Pathways to a Protein Folding Intermediate Observed in a 1-Microsecond Simulation in Aqueous Solution Science 282, 740-44 (1998).

Eggert D, Lorusso A, Fisher RB, 1997. Estimating 3-D rigid body transformations: A comparison of four major algorithms. Machine Vision and Applications vol 9, pp. 272-290.

Enroth C, Eger BT, Okamoto K, Nishino T, Nishino T, Pai EF, 2000, Crystal Structures of Bovine Milk Xanthine Dehydrogenase and Xanthine Oxidase, Structure-based Mechanism of Conversion, Proc.Nat.Acad.Sci.USA, V.97 10723

Fernandez-Recio J, Totrov M, Abagyan R 2002. Soft protein-protein docking in internal coordinates. Protein Sci. 2002 Feb;11(2):280-91.

Fogel GB, Corne DW. (editors). 2003. Evolutionary Computation in Bioinformatics. Morgan Kaufmann, 393 p.

Foley JD, van Dam A, Feiner SK, Hughes JF. 1996. Computer Graphics Principles and Practice, 2nd Edition in C. Addison Wesley

Frishman D, Argos P. 1997. Seventy-five percent accuracy in protein secondary structure prediction. Proteins: Structure, Function, and Genetics Volume 27, Issue 3, pp 329-35

Gabb HA, Jackson RM, Sternberg MJ. 1997 Modelling protein docking using shape complementarity, electrostatics and biochemical information. J Mol Biol. 1997 Sep 12;272(1):106-20.

Gallaire H. 1985. Logic Programming:Further developments, in: IEEE Symposium on Logic Programming, Boston, 1985

Gardiner EJ, Willett P, Artymiuk PJ. 2001 Protein docking using a genetic algorithm. Proteins. 2001 Jul 1;44(1):44-56.

Gent IP, MacIntyre E, Prosser P, Smith BM, Walsh T, 1996. An empirical study of dynamic variable ordering heuristics for the constraint satisfaction problem. Principles and Practice of Constraint Programming-CP'96, 179–193, (1996).

Gille C, Goede A, Preissner R, Rother K, Frommel C. 2000. Conservation of substructures in proteins: interfaces of secondary structural elements in proteasomal subunits. J Mol Biol. 2000 Jun 16;299(4):1147-54.

Gilquin B, Racape J, Wrisch A, Visan V, Lecoq A, Grissmer S, Menez A, Gasparini S. 2002. Structure of the BgK-Kv1.1 complex based on distance restraints identified by double mutant cycles. Molecular basis for convergent evolution of Kv1 channel blockers. J Biol Chem. 2002 Oct 4;277(40):37406-13. Epub 2002 Jul 19.

Goodfellow BJ, Rusnak F, Moura I, Domke T, Moura JJ. 1998 NMR determination of the global structure of the 113Cd derivative of desulforedoxin: investigation of the hydrogen bonding pattern at the metal center. Protein Sci. 1998 Apr;7(4):928-37.

Goodsell DS, Olson AJ. 1990. Automated Docking of Substrates to Proteins by Simulated Annealing Proteins: Str. Func. And Genet., 8: 195-202.

Goodsell DS, Olson AJ. 2000. Structural symmetry and protein function. Annu Rev Biophys Biomol Struct. 2000;29:105-53.

Guntert P, Mumenthaler C, Wuthrich K. 1997. Torsion angle dynamics for NMR structure calculation with the new program DYANA. J Mol Biol. 1997 Oct 17;273(1):283-98.

Halgren TA. 1996. Merck Molecular Force Field. II. MMFF94 van der Waals and Electrostatic Parameters for Intermolecular Interactions. J Comp Chem 1996; 17:520-552.

Halperin I, Ma B, Wolfson H, Nussinov R. 2002. Principles of docking: An overview of search algorithms and a guide to scoring functions. Proteins. 2002 Jun 1;47(4):409-43

Hansson T, Oostenbrink C, van Gunsteren W. 2002 Molecular dynamics simulations. Curr Opin Struct Biol. 2002 Apr;12(2):190-6.

Haralick RM, Elliott GL. 1980. Increasing tree search efficiency for constraint satisfaction problems. Artificial Intelligence, Vol. 14 (3) 263-313

Haykin S. 1999. Neural networks: A comprehensive foundation. 2nd ed. Prentice Hall, 842p.

Helms LR, Swenson RP, 1992, The primary structures of the flavodoxins from two strains of Desulfovibrio gigas. Cloning and nucleotide sequence of the structural genes. Biochim. Biophys. Acta 1131:325-328.

Hentenryk PV, Deville Y, Teng CM. 1992. A generic arc-consistency algorithm and its specializations. Artificial Intelligence 57, pp291-321

Herrmann T, Güntert P, Wüthrich K. 2002. Protein NMR structure determination with automated NOE assignment using the new software CANDID and the torsion angle dynamics algorithm DYANA. J. Mol. Biol. 319, 209-227.

Hille R, Anderson R.F. 1991. Electron transfer in milk xanthine oxidase as studied by pulse radiolysis. J. Biol. Chem. 266, 5608-5615.

Hille R. 1996. The mononuclear molybdenum enzymes. Chem. Rev. 96, 2757-2816.

Horn B, Hilden H, Negahdaripour S. 1988. Closed-form solution of absolute orientation using orthonormal matrices Journal of the Optic Society A, Vol. 5, No. 7

Horn B. 1987. Closed-form solution of absolute orientation using unit quaternions. Journal of the Optic Society A, Vol. 4, No. 4.

Hurley JK, Weber-Main AM, Stankovich MT, Benning MM, Thoden JB, Vanhooke JL, Holden HM, Chae YK, Xia B, Cheng H, Markley JL, Martinez-Julvez M, Gomez-Moreno C, Schmeits JL, Tollin G. 1997. Structure-function relationships in Anabaena ferredoxin: correlations between X-ray crystal structures, reduction potentials, and rate constants of electron transfer to ferredoxin:NADP+ reductase for site-specific ferredoxin mutants. Biochemistry. 1997 Sep 16;36(37):11100-17.

Jackson RM, Gabb HA, Sternberg MJ. 1998. Rapid refinement of protein interfaces incorporating solvation: application to the docking problem. J Mol Biol. 1998 Feb 13;276(1):265-85.

Jaffer J, Lassez JL. 1987. Constraint Logic Programming, in proc. The ACM Symposium on Principles of Programming Languages, ACM, 1987

Janin J, Henrick K, Moult J, Eyck LT, Sternberg MJE, Vajda S, Vakser I, Wodak SJ. 2003. CAPRI: A Critical Assessment of PRedicted Interactions. Proteins: Structure, Function, and Genetics. V. 52-1, pp2-9

Janin J. 2002. Welcome to CAPRI: A Critical Assessment of PRedicted Interactions. Proteins: Structure, Function, and Genetics Volume 47, Issue 3, 2002. Pages: 257

Jia Z, Quail JW, Waygood EB, Delbaere LT. 1993. The 2.0-A resolution structure of Escherichia coli histidine-containing phosphocarrier protein HPr. A redetermination. J Biol Chem. 1993 Oct 25;268(30):22490-501.

Katchalski-Katzir E, Shariv I, Eisenstein M, Friesem AA, Aflalo C, Vakser IA. 1992 Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques. Proc Natl Acad Sci U S A. 1992 Mar 15;89(6):2195-9.

Kendrew JC, Dickerson RE, Strandberg BE, Hart RG, Davies RD, Phillips DC. and Shore VC (1960) Structure of myoglobin: a three-dimensional Fourier synthesis at 2.0Å resolution. Nature, 185, 422–427

Klapper I, Hagstrom R, Fine R, Sharp K, Honig B. 1986. Focusing of electric fields in the active site of Cu-Zn superoxide dismutase: effects of ionic strength and amino-acid modification. Proteins. 1986 Sep;1(1):47-59.

Kohonen T. 1982. Self organized formation of topologically correct feature maps. Biological Cybernetics 43. 59-69.

Kramer, G. A. 1992. Solving Geometric Constraint Systems: A Case Study in Kinematics. Cambridge, MA: MIT Press.

Krippahl L, Barahona P. 2000. PSICO: Combining Constraint Programming and Optimisation to Solve Macromolecular Structures, presented at ERCIM-2000, Padova, Italy

Krippahl L, Barahona P. 2002. PSICO: Solving Protein Structures with Constraint Programming and Optimisation, Constraints 2002, 7, 317-331

Krippahl L, Moura JJ, Palma PN. 2003. Modeling protein complexes with BiGGER. Proteins: Structure, Function, and Genetics. V. 52(1):19-23.

Krippahl L, Palma NP, 2001 Chemera and BiGGER, v2.0. Available from http://www.cqfb.fct.unl.pt/bioin/chemera/ System requirements: IBM PC compatible Windows 98, 2000 or XP, with DirectX 6.0 or above.

Krippahl L, Palma NP. 1996. Cyberzyme 1.0. IBM PC DOS, (obsolete)

Krippahl L-. 1996. BoGIE - Boolean Geometric Interaction Evaluation. Final project report, available from the author.

Krippahl L. 1999. Determinação de estrutura de proteínas através de programação por restrições. Master's thesis, FCT, Universidade Nova de Lisboa. Available from the author.

Krippahl L., Barahona, P., 1999, Applying Constraint Programming to Protein Structure Determination, Principles and Practice of Constraint Programming, Springer Verlag, 1999 289-302

Krippahl L., Barahona, P., 2003, Propagating N-ary Rigid-Body Constraints, Principles and Practice of Constraint Programming – CP 2003, pp 452-65.

Krippahl, L., Trosset, M., Barahona, P. 2001. Combining Constraint Programming and Multidimensional Scaling to solve Distance Geometry Problems. Proceedings of CP-AI-OR'2001, Imperial College, Kent UK, 67-80, 2001.

Kurisu G, Kusunoki M, Katoh E, Yamazaki T, Teshima K, Onda Y, Kimata-Ariga Y, Hase T. 2001. Structure of the electron transfer complex between ferredoxin and ferredoxin-NADP(+) reductase. Nat Struct Biol. 2001 Feb;8(2):117-21.

Lamdan Y, Schwartz JT, Wolfson HJ. 1988. On Recognition of 3-D Objects from 2-D Images. Proceedings of IEEE International Conference on Robotics and Automation, pp. 1407-1413, 1988.

Li CH, Ma XH, Chen WZ, Wang CX. 2003. A soft docking algorithm for predicting the structure of antibody-antigen complexes. Proteins. 2003 Jul 1;52(1):47-50.

Lovell SC, Word JM, Richardson JS, Richardson DC. 2000. The penultimate rotamer library. Proteins: Structure Function and Genetics 40 389-408.

Lunin VY, Urzhumtsev A, Bockmayr A. 2002. Direct phasing by binary integer programming. Acta Crystallogr A. 2002 May;58(Pt 3):283-91.

Lutter R, Abrahams JP, Raaij MJ, Todd RJ, Lundqvist T, Buchanan SK, Leslie AG, Walker JE. 1993. Crystallization of F1-ATPase from bovine heart mitochondria. J. Mol. Biol. 229, 787-90

Mackworth AK. 1977. Consistency in networks of relations. Artificial Intelligence 8, pp88-118

Mathieu M, Petitpas I, Navaza J, Lepault J, Kohli E, Pothier P, Prasad BVV, Cohen J, Rey FA. 2001. Atomic structure of the major capsid protein of rotavirus: implications for the architecture of the virion. EMBO J. 2001 Apr 2;20(7):1485-97.

McCoy M., E.S.Stavridi, J.L.Waterman, A.M.Wieczorek, S.J.Opella, T.D.Halazonetis 1997. Hydrophobic Side-Chain Size is a Determinant of the Three-Dimensional Structure of the P53 Oligomerization Domain Embo J. V. 16 6230 1997

MolSoft 2003. ICM-Pro, version 3.0-23a. Available from www.molsoft.com. System requirements: Windows, Linux or SGI systems.

Morales R, Charon MH, Hudry-Clergeon G, Petillot Y, Norager S, Medina M, Frey M. 1999. Refined X-ray structures of the oxidized, at 1.3 A, and reduced, at 1.17 A, [2Fe-2S] ferredoxin from the cyanobacterium Anabaena PCC7119 show redox-linked conformational changes. Biochemistry. 1999 Nov 30;38(48):15764-73. Erratum in: Biochemistry 2000 Mar 7;39(9):2428.

Morelli X, Czjzek M, Hatchikian CE, Bornet O, Fontecilla-Camps JC, Palma NP, Moura, JJ, and Guerlesquin F. 2000 Structural model of the Fe-hydrogenase/cytochrome c553 complex combining transverse relaxation-optimized spectroscopy experiments and soft docking calculations. Journal of Biological Chemistry 275, 23204-23210.

Morelli X, Dolla A., Czjzek M, Palma PN, Blasco, F, Krippahl L, Moura JJ, Guerlesquin F. 2000. Heteronuclear NMR and soft docking: an experimental approach for a structural model of the cytochrome c553-ferredoxin complex. Biochemistry 39:2530-2537.

Morelli X, Palma PN, Guerlesquin F, Rigby AC. 2001. A novel approach for assessing macromolecular complexes combining soft-docking calculations with NMR data. Protein Sci. 10:2131-2137.

Morris GM, Goodsell DS, Halliday RS, Huey R, Hart WE, Belew RK, Olson AJ. 1998. Automated Docking Using a Lamarckian Genetic Algorithm and and Empirical Binding Free Energy Function J. Computational Chemistry, 19: 1639-1662.

Morris GM, Goodsell DS, Huey R, Olson AJ . 1996. J. Distributed automated docking of flexible ligands to proteins: Parallel applications of AutoDock 2.4 Computer-Aided Molecular Design, 10: 293-304.

Morris GM, Goodsell DS, Halliday RS, Huey R, Hart WE, Belew RK, Olson AJ. 1998. Automated Docking Using a Lamarckian Genetic Algorithm and Empirical Binding Free Energy Function. , J. Computational Chemistry, 19: 1639-1662.

Moura I, Tavares P, Moura JJG, Ravi N, Huynh BH, Liu MY, LeGall J. 1990 Purification and characterization of desulfoferrodoxin. A novel protein from Desulfovibrio desulfuricans (ATCC 27774) and from Desulfovibrio vulgaris (strain Hildenborough) that contains a distorted rubredoxin center and a mononuclear ferrous center. J. Biol. Chem. 265, 21596-21602.

Mulder GJ. 1839. On the composition of some animal substances. Journal für praktische Chemie 16, 129

Navarro JA, Hervas M, Genzor CG, Cheddar G, Fillat MF, de la Rosa MA, Gomez-Moreno C, Cheng H, Xia B, Chae YK, and others.1995. Site-specific mutagenesis demonstrates that the structural requirements for efficient electron transfer in Anabaena ferredoxin and flavodoxin are highly dependent on the reaction partner: kinetic studies with photosystem I, ferredoxin:NADP+ reductase, and cytochrome c. Arch Biochem Biophys. 1995 Aug 1;321(1):229-38.

Needleman SB, Wunsch CD. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins.J Mol Biol. 1970 Mar;48(3):443-53.

Norel R, Sheinerman F, Petrey D, Honig B. Electrostatic contributions to protein-protein interactions: fast energetic filters for docking and their physical basis.Protein Sci. 2001 Nov;10(11):2147-61.

Nussinov R, Wolfson HJ. 1999. Efficient Computational Algorithms for Docking, and for Generating and Matching a Library of Functional Epitopes I. Rigid and Flexible Hinge-Bending Docking Algorithms Combinatorial Chemistry & High Throughput Screening, 2, 277-287.

Palma PN, Krippahl L, Wampler JE, Moura, JJG. 2000. BiGGER: A new (soft) docking algorithm for predicting protein interactions. Proteins: Structure, Function, and Genetics 39:372-84.

Palma PN, Moura I, LeGall J, Van Beeumen J, Wampler JE, Moura JJ. Evidence for a ternary complex formed between flavodoxin and cytochrome c3: 1H-NMR and molecular modeling studies. Biochemistry. 1994 May 31;33(21):6394-407.

Pauleta SR, 2002. Pseudoazurin as an alternative electron donor to cytochrome c peroxidase. In EUROBIC 2002, 2002 Jul 29 to 2002 Aug 3

Pauling R and Corey RB. 1951a. Atomic Coordinates and Structure Factors for Two Helical Configurations of Polypeptide Chains, Proceedings of the National Academy of Sciences Vol. 37, Nos. 5, pp. 235-240.

Pauling R and Corey RB. 1951b. The Pleated Sheet, A New Layer Configuration of Polypeptide Chains. Proceedings of the National Academy of Sciences Vol. 37, Nos. 5, pp. 251-256.

Pettigrew GW, Gilmour R, Goodhew CF, Hunter DJ, Devreese B, Van Beeumen J, Costa C, Prazeres S, Krippahl L, Palma PN, Moura I, Moura JJ. 1998. The surface-charge asymmetry and dimerisation of cytochrome c550 from Paracoccus denitrificans--implications for the interaction with cytochrome c peroxidase. Eur J Biochem. 1998 Dec 1;258(2):559-66.

Pettigrew GW, Goodhew CF, Cooper A, Nutley M, Jumel K, Harding SE. 2003, The electron transfer complexes of cytochrome c peroxidase from Paracoccus denitrificans. Biochemistry. 2003 Feb 25;42(7):2046-55.

Pettigrew GW, Prazeres S, Costa C, Palma N, Krippahl L, Moura I, Moura JJ. 1999. The structure of an electron transfer complex containing a cytochrome c and a peroxidase. J Biol Chem. 1999 Apr 16;274(16):11383-9.

Pettigrew GW, Pauleta SR, Goodhew CF, Cooper A, Nutley M, Jumel K, Harding SE, Costa C, Krippahl L, Moura I, Moura J. 2003. Electron Transfer Complexes of Cytochrome c Peroxidase from Paracoccus denitrificans Containing More than One Cytochrome. Biochemistry 2003, 42, 11968-81

Press WH, Teukolsky SA, Vetterling WT, Flannery BP. 1994. Numerical recipes in C. Cambridge, 994p.

Rebelo JM, Dias JM, Huber R, Moura JJG, RomaoMJ, 2001,Structure Refinements of the Aldehyde Oxidoreductase from Desulfovibrio gigas, J.Biol.Inorg.Chem, V.6, 791

Ritchie DW, Kemp GJ. 2000 Protein docking using spherical polar Fourier correlations. Proteins. 2000 May 1;39(2):178-94.

Rodošek R. 2001. A Constraint-Based Approach for Deriving 3-D Structures of Cyclic Polypeptides, Constraints, V.6: 2&3: 257, 2001

Searle MS, Williams DH, Gerhard U. 1992. Partitioning of free energy contributions in the estimation of binding constants: Residual motions and consequences for amide-amide Hydrogen bond strengths. J. Am. Chem. Soc. 114. 10697-704

Schneidman-Duhovny D, Inbar Y, Polak V, Shatsky M, Halperin I, Benyamini H, Barzilai A, Dror O, Haspel N, Nussinov R, Wolfson HJ. 2003. Taking geometry to its edge: fast unbound rigid (and hinge-bent) docking. Proteins. 2003 Jul 1;52(1):107-12.

Schreiber G. 2003. The structure of protein-protein interfaces: from characterization to prediction to docking. In Workshop on Weak Protein-Protein Interactions, Sevillha, 2003 July 9-13.

Sellers PH. 1974. On the theory and computation of evolutionary distances. SIAM J. Appl. Math. 26, 787-793

Simons KT, Kooperberg C, Huang E, Baker D. 1997. Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. J Mol Biol 268, 209-25

Singh J, Thornton JM. 1992. Atlas of protein side chain interactions. Oxford IRL Press.

Stanford University. 2000. FOLDING@HOME Available from http://www.stanford.edu/group/pandegroup/folding/. System requirements: IBM PC compatible Windows 98, NT, 2000 or XP OS, Linux Intel or Mac OS.

Stetefeld J, Mayer U, Timpl R, Huber R. 1996 Crystal structure of three consecutive laminin-type epidermal growth factor-like (LE) modules of laminin gamma1 chain harboring the nidogen binding site. J Mol Biol. 1996 Apr 5;257(3):644-57.

Stryer, L, 1998, Biochemistry, Freeman

Sundberg EJ, Li H, Llera AS, McCormick JK, Tormo J, Schlievert PM, Karjalainen K, Mariuzza RA. 2002. Streptococcal superantigens bound to TCR_ chains reveal diversity in the architecture of T cell signaling complexes. Structure 2002, 10:687-699.

Sutherland I. 1953. Sketchpad: a man-machine graphical communication system, in: Proc. IFIP Spring Joint Computer Conference 1963

Swain MT, Kemp GJ. Modelling protein side-chain conformations using constraint logic programming. Comput Chem. 2001 Dec;26(1):85-95.

Szyperski T, Yeh DC, Sukumaran DK, Moseley HN, Montelione GT. 2002. Reduced-dimensionality NMR spectroscopy for high-throughput protein resonance assignment. Proc Natl Acad Sci U S A. 2002 Jun 11;99(12):8009-14.

Tilbeurgh H, Le Coq D, Declerck N. 2001.Crystal structure of an activated form of the PTS regulation domain from the LicT transcriptional antiterminator. EMBO J. 2001 Jul 16;20(14):3789-99.

Todd BA, Rammohan J, Eppell SJ. 2003. Connecting Nanoscale Images of Proteins with Their Genetic Sequences. Biophysical Journal 84:3982-3991 (2003)

Torgerson WS. 1952. Multidimensional scaling: I. Theory and method. Psychometrika, 17:401--419, 1952.

Trosset M.W. 2000. Distance matrix completion by numerical optimisation. Computational Optimization and Applications, 17:11--22, 2000.

Varghese JN, Garrett TP, Colman PM, Chen L, Hoj PB, Fincher GB.1994 Three-dimensional structures of two plant beta-glucan endohydrolases with distinct substrate specificities.Proc Natl Acad Sci U S A. 1994 Mar 29;91(7):2785-9.

Walker M, Shao L, Volz R. 1991. Estimating 3-D location parameters using dual number quaternions. CVGIP: Image Understanding, Vol. 54, No. 3.

Walls PH, Sternberg MJ.1992 New algorithm to model protein-protein recognition based on surface complementarity. Applications to antibody-antigen docking.J Mol Biol. 1992 Nov 5;228(1):277-97.

Waltz DL. 1975. Understanding line drawings of scenes with shadows, in Psychology of Computer Vision, McGraw Hill, New York

Wang J, Cieplak P, Kollman PA. 2000. How well does a restrained electrostatic potential (RESP) model perform in calcluating conformational energies of organic and biological molecules?. J. Comput. Chem. 21, 1049-1074.

Wang JTL, Shapiro BA, Shasha D.1999. Pattern discovery in biomolecular data. Oxford, 251p.

Wang Y, Zhang H, Li W, Scott RA. 1995. Discriminating compact nonnative structures from the native structure of globular proteins. Proc Natl Acad Sci U S A 92(3):709-13.

Wang Y, Zhang H, Scott RA. 1995. A new computational model for protein folding based on atomic solvation. Protein Sci 4(7):1402-11.

Watenpaugh KD, Sieker LC, Jensen LH, Legall J, Dubourdieu M, 1972, Structure of the oxidized form of a flavodoxin at 2.5-Angstrom resolution: resolution of the phase ambiguity by anomalous scattering. Proc Natl Acad Sci U S A. 1972 Nov;69(11):3185-8

Watson JD, Crick FHC. 1953. A structure for Deoxyribose Nucleic Acid. Nature, V171, p737

Wilson DS, Guenther B, Desplan C, Kuriyan J. 1995. High resolution crystal structure of a paired (Pax) class cooperative homeodomain dimer on DNA.Cell. 1995 Sep 8;82(5):709-19.

Wlodawer A, Miller M, Jaskolski M, Sathyanarayana BK, Baldwin E, Weber IT, Selk LM, Clawson L, Schneider J, Kent SB. 1989. Conserved folding in retroviral proteases: crystal structure of a synthetic HIV-1 protease. Science. Aug 11;245(4918):616-21.

Wolfson HJ, Rigoutsos I. 1997. Geometric Hashing: An overview.. IEEE Computational Science and Engineering, 1997, pp. 10-21

Zacharias M. 2003. Protein-protein docking with a reduced protein model accounting for side-chain flexibility. Protein Sci. 2003 Jun;12(6):1271-82.

Zagrovic B, Sorin EJ, Pande V. 2001Beta-hairpin folding simulations in atomistic detail using an implicit solvent model. J Mol Biol. 2001 Oct 12;313(1):151-69.

Zanetti G, Morelli D, Ronchi S, Negri A, Aliverti A, Curti B. 1988. Structural studies on the interaction between ferredoxin and ferredoxin-NADP+ reductase . Biochemistry 27, 3753-3759

Zheng W, Doniach S. 2002. Protein structure prediction constrained by solution X-ray scattering data and structural homology identification. J Mol Biol. 2002 Feb 8;316(1):173-87.

Zhu C, Byrd RH, Lu P, Nocedal J. 1996. L-BFGS-B: Fortran Subroutines for Large-scale Bound Constrained Optimization. Technical Report NAM-11, Department of Electrical Engineering & Computer Science, Northwestern University, Evanston, IL 60208, December 1994. Revised October 8 1996

Zimmerman D, Kulikowski C, Wang L, Lyons B, Montelione GTJ. 1994. Automated sequencing of amino acid spin systems in proteins using multidimensional HCC(CO)NH-TOCSY spectroscopy and constraint propagation methods from artificial intelligence. Biomol. NMR 1994, 4: 241 – 256

Other References

Harris J, Stocker H. 1998. Handbook of mathematics and computational science. Springer.

[CBE] Council of Biology Editors.1994. Scientific style and dormat. 6th ed. Cambridge.

Fowler H. 1926. A Dictionary of modern English usage. 1st ed. Oxford.

Trask RL. 1997. The Penguin guide to punctuation. Penguin.

University of Chicago Press. 1993. The Chicago manual of style. Chicago. UCP.

PART

4

Appendixes

In This Part:

Appendix I

PSICO Dynamic Link Library

Appendix II

Tables

Appendix I: PSICO Dynamic Link Library

AI. 1. Initialisation, Finalisation, and Problem Selection functions

BuildPivotSet
Selects the rigid group with the largest number of constraints with other residues in the protein as a pivot. This group can be fixed in space to start propagation.

CalculateSolution
Sets the coordinates of the atoms in the internal representation of the protein structure to the centres of the good regions in their domains. Can be used to visualise the structure during the calculations.

ClearCurrentProblem
Clears all variables and constraints from the current problem.

ExportSolution(FileName:PChar)
Saves the current solution to a PDB file specified by the FileName parameter. The coordinates for each atom are the central point of the Good region of the domain.

FinalizePsicoDll
Frees memory used by PsicoDll. Should be called at the end of the calculations.

GetProblemCount:Integer
Returns the number of problems active at the time.

InitializePsicoDll(DataPath:PChar)
Must be called once before using the Dll. Reads the residue information file Resinfo.Cyb in the DataPath folder, and initialises the Dll.

LoadProblem(SequenceFile,ConstraintsFile:PChar)
Loads a problem file. If ConstraintsFile is null, SequenceFile is the name and path of a PSICO problem file. Otherwise, SequenceFile is the name and path of a sequence file, and ConstraintsFile of a DYANA constraints file.

PivotSetCount:Integer
Returns the number of the atoms in the Pivot set generated by BuildPivotSet.

SetCurrentProblem(ProblemIndex:Integer)
The problems are indexed from zero to Count-1 (see SetProblemCount). Typically, this procedure will be called with ProblemIndex=0. Must be called after SetProblemCount and before other function calls, but can be called at any time if there are several problems loaded to switch processing to another problem.

SetMaxChoicePoints(MaxCP:Integer)
Sets the maximum number of choice points to be stored by the DLL. If additional choice points are necessary, the DLL disposes of the oldest choice points and keeps only the only the last MaxCP choice points.

SetOmegaLock(Lock:Integer)
If lock is not zero all omega torsion angles are considered locked at 180º.

SetProblemCount(Count:Integer)
Sets the number of problems to create. For typical applications, it will be run with a Count value of 1. Must be called after InitializePsicoDll, but before any other function call.

SetSplit(Split:Integer)
Sets the fraction of the domain that is kept when splitting during enumeration. Units are 0.001.

AI. 2. Atom Handling Functions

AtomConstraint(AtomIndex,ConstraintIndex:Integer;
var HighB,LowB,ToAtomIndex:Integer)
Returns the data for constraint ConstraintIndex of the atom AtomIndex. HighB and LowB are the upper and lower distance bounds, and ToAtomIndex the index of the other atom constrained. AtomIndex and ToAtomIndex range from 0 to AtomCount-1. ConstraintIndex ranges from 0 to AtomConstraintCount.

AtomConstraintCount(AtomIndex:Integer):Integer
Returns the number of binary distance constraints involving atom AtomIndex. AtomIndex ranges from 0 to AtomCount-1.

AtomCount:Integer
Returns the number of atoms in the current problem.

AtomicNumber(AtomIndex:Integer):Integer
Returns the atomic number of the atom specified by AtomIndex. AtomIndex ranges from 0 to AtomCount-1.

AtomNode(AtomIndex:Integer):Integer
Returns the index of the node in the torsion angle model where the atom specified in AtomIndex belongs. AtomIndex ranges from 0 to AtomCount-1.

AtomResidueId(AtomIndex:Integer):Integer
Returns the Id number for the residue of the atom AtomIndex. The residue Id number is the number on the PSICO problem file. AtomIndex ranges from 0 to AtomCount-1.

GetConnection(ConIx:Integer):Integer
Gets the index of the atom at conection ConIx from the connection buffer. See MakeConnectionBuffer.

IsBackBoneAtom(AtomIndex:Integer):Integer
Returns 1 if the atom AtomIndex belongs to the backbone, zero otherwise.

MakeConnectionBuffer(AtomIx:Integer):Integer
Creates an stack of atoms that are connected to the atom indicated, and returns the length of the stack (indexed at 0).

NodeAtom(NodeIndex,AtomNodeIndex:Integer):Integer
Returns the global index of the atom number AtomIndex in the torsion angle specified by NodeIndex. NodeIndex ranges from 0 to NodeCount-1, and AtomIndex from 0 to NodeAtomCount-1.

NodeAtomCount(NodeIndex:Integer):Integer
Returns the number of atoms in the torsion angle node specified by NodeIndex. NodeIndex ranges from 0 to NodeCount-1.

NodeCount:Integer
Returns the total number of nodes in the torsion angle model.

NodeOffspring(NodeIndex,OffspringIndex:Integer):Integer
Returnd the index of the offspring number OffspringIndex of node NodeIndex. NodeIndex ranges from 0 to NodeCount-1, and OffspringIndex ranges fom 0 to NodeOffspringCount-1.

NodeOffspringCount(NodeIndex:Integer):Integer
Returns the number of offspring of the torsion angle node specified by NodeIndex. NodeIndex ranges from 0 to NodeCount-1.

PivotAtom(Index:Integer):Integer
Returns the global index of the Pivot atom specified by the Index parameter. The Index parameter ranges from 0 to PivotSetCount-1. The Pivot set is generated by BuildPivotSet

SetAtomicCollisionRadius(Rad:Integer)
Sets the atomic radius value used to determine collisions.

SetOracleCoord(AtomIndex,x,y,z:Integer)
Sets the oracle coordinate for atom AtomIndex. Units are 0.001Å.

TemplatePosition(AtomIndex:Integer; var cx,cy,cz:Integer)
Sets variables cx, cy, and cz to the coordinates of amino acid template for the atom specified in AtomIndex. AtomIndex ranges from 0 to AtomCount-1

AI. 3. Enumeration and Backtracking Functions

BackTrack:Integer
Causes the enumeration process to return to the previous choicepoint, and choose the next search branch in the next enumeration.

ChoicePointCount:Integer
Number of choice points currently stored.

GetEnumerationFlag(AtomIndex:Integer):Integer
Returns the enumeration state of atom AtomIndex in the current enumeration round. The value is -1 if AtomIndex is not the variable currently chosen for enumeration, or the number of previous attempts at enumerating AtomIndex in this enumeration node. With current value enumeration heuristics, this value can only be 0 – the atom is being enumerated at this node for the first time – or 1 – the atom was enumerated at this node once, but execution returned to this node due to backtracking.

PopChoicePoint
Returns enumeration to the last stored choice point and sets the enumeration flags to force value enumeration into the alternative branch of the search tree.

PushChoicePoint(EnumeratedAtom:Integer)
Stores a choice point in the current enumeration

QueryDefaultEnumeration(var AtomIndex,Hx,Hy,Hz,Lx,Ly,Lz:Integer)
Queries the heuristic for the next atom to enumerate (returned in AtomIndex) and the domain to restrict it to (returned in the high and low coordinates Hx, Hy, Hz, Lx, Ly, and Lz).

QueryDefaultValueEnumeration(AtomIndex:Integer; var Hx,Hy,Hz,Lx,Ly,Lz:Integer)
Queries the value enumeration heuristic to returns the block to restrict atom numbered AtomIndex in the next enumeration (returned in high coordinates Hx, Hy, Hz, and low coordinates Lx, Ly, Lz)

RestrictAtom(AtomIndex,Hx,Hy,Hz,Lx,Ly,Lz:Integer)
Intersects the domain of the atom numbered AtomIndex with the block defined by the high coordinates Hx, Hy, Hz, and the low coordinates Lx, Ly, and Lz, and marks the atom as active for propagation.

SetCurrentHeuristic(Heuristic:Integer)
Sets the value enumeration heuristic to use.

SetCurrentVarHeuristic(Heuristic:Integer)
Sets the heuristic for variable selection.

SetMaxWidToEnumerate(Wid:Integer)
Sets a maximum value for the size of the domain of an atom to be enumerated. If no atom below this threshold is left to enumerate in the current enumeration round, a new round starts. The default value is zero.

AI. 4. Propagation and Domain Handling Functions

AtomGood(Index:Integer;var Hx,Hy,Hz,Lx,Ly,Lz:Integer)
Sets the Hx, Hy, Hz, Lx, Ly, Lz to the upper and lower boundaries for the Good block of the atom Index.

AtomNoGood(AtomIndex,NoGoodIndex:Integer;var Hx,Hy,Hz,Lx,Ly,Lz:Integer)
Sets the Hx, Hy, Hz, Lx, Ly, Lz to the upper and lower boundaries for the No-Good block NoGoodIndex of the atom AtomIndex.

AtomNoGoodCount(Index:Integer):Integer
Returns the number of No-Good blocks for atom Index.

ExcludeAtoms(Atom,Radius:Integer)
Propagates an Out constraint of radius Radius from atom Atom to all atoms that do not have constraints specified with this atom.

Propagate:Integer
Calls the Arc-Consistency propagation procedure. Returns a value of one if propagation reduced all atom domains to less than the target size for enumeration, a value of two if one propagation failed), or zero otherwise.

ReadFailureData(var AtomIx,Cause:Integer)
Returns the atom index of the failure detected in Propagate. The Cause variable returns zero if the reason for failure was an empty atom domain, a value of one if the failure was detected by the atomic collision detection algorithm.

SetMaxDomainEdge(Size:Integer)
Sets the maximum length for the Good region for an atom domain to be considered sufficiently reduced.

TotalVolumes(var Good,NoGood:Double)
Sets variables Good and NoGood to the total volumes of Good and No-Good blocks for all atoms.

AI. 5. Group Handling Functions

GroupCount:Integer
Returns the number of rigid groups defined.

GroupSize(Ix:Integer):Integer
Returns the total number of atoms in group number Ix. Atoms are numbered within the group from zero to GroupSize-1.

GroupAtomIndex(GIx,AIx:Integer):Integer
Returns the global index of atom Aix of group Gix

ImportRigidGroups(FileName:PChar)
Imports a PDB file containing the coordinates for rigid groups. Each rigid group is a set of atom records using the PDB format ending in the TER keyword.

GroupAtomCoords(Gix,Aix:Integer;var x,y,z:Integer)
Returns the coordinates for the atom number Aix in group number Gix.

SetGroupPropagationFlag(Ix,F:Integer)
Marks the propagation flag of group number Ix with False if F is 0, True otherwise. A True flag forces a call to the group propagation algorithm for this group at the next call of Propagate.

SetRigidGroups(MinSize:Integer)
Builds rigid groups of size equal to or greater than MinSize from the torsion angle model.

AI. 6. Optimisation Functions

Minimize(FitIters:Integer;FitVar:Single; MinIters:Integer;MinVar:Single)
Calls the local search minimisation to reduce constraint violations. FitIters and FitVar are the maximum number of iterations and maximum variation for fitting the torsion angle model to the central points of the atom domains. MinIters and MinVar are the maximum number of iterations and maximum variation for minimizing constraint violations. These parameters set the stopping conditions for the local searches.

SetCPCoords
Sets the domains of all atoms to the coordinates in the torsion angle model.

AI. 7. Debugging Functions

LoadOracle(OracleFile:PChar)
Loads a PDB file with the structure of the correct solution to the current problem.

OraclePosition(AtomIndex:Integer; var cx,cy,cz:Integer)
Returns the Oracle coordinates for atom AtomIndex.

SetCallbackFlag(Id:Integer;Status:Boolean);stdcall
The Status flag determines whether the Callback function supplied in SetDebugCallback will be called, and the Id value determines which step in the Propagate procedure calls the function.

SetDebugCallback(Callback:TPDDCallback)
Supplies a pointer to a callback function of the form:
CallBack(ID:Integer;IntCount:Integer;Ints:Pointer;SingleCount:Integer;Sings:Pointer)
This function is called by the Dll on different steps depending on the value of the Id sent in the SetCallbackFlag function. The parameters of the CallBack function will vary depending on where it is called from.

AI. 8. Other Functions

DestroyGroups
Removes all rigid group constraints

EditConstraint(AtomIndex,ConstraintIndex,HighB,LowB:Integer)
Sets the upper and lower distance bounds (HighB and LowB) of the constraint specified by ConstraintIndex in atom AtomIndex. AtomIndex ranges from 0 to AtomCount-1, and ConstraintIndex from 0 to AtomConstraintCount-1.

ExportConstraints(FileName:PChar)
Exports all constraints to a text file specified in FileName.

OracleRMSD(Mirror:Integer):Integer
Calculates the Root Mean Square Deviation between the current structure and an oracle. The coordinates used for the atoms are the centres of the domains. If the Mirror flag is not 0, the coordinates are mirrored before the calculation.

OracleRMSDFit(Mirror:Integer):Integer
Returns the RMSD of the central points of the current domains to the oracle structure. If Mirror is not zero the mirror image of the structure is used. The oracle is placed at the orientation and position that minimises the RMSD value.

Refresh(AtomIndex:Integer)
Refreshes the statistics on the specified atom. A value of -1 on AtomIndex refreshes all atoms.

RotateCoords(AngX,AngY,AngZ:Integer;var x,y,z:Integer)
Rotates the coordinates x, y and z by the rotation angles supplied in AngX, AngY, AngZ.

Version:Integer
Returns the Dll version number

Appendix II: Tables

Table 1 Classes of similar amino acids used in the side chain contact filter (Section 6.4)

Class	Amino Acids
2	Gly, Met, Ala, Val, Ile, Leu, Phe, Pro
4	Asp, Glu
5	Ser, Thr, Cys
6	Asn, Gln, His, Trp, Tyr
7	Arg, Lys

Table 2 Set of structures (PDB code and chain ID) used to test the binary distance constraint propagation (continued on next page).

Name	Residues	RMSD	Time	Name	Residues	RMSD	Time	Name	Residues	RMSD	Time
137lA	173	6.3	29.6	1cgsL	226	16.9	34.6	1hyxL	223	11.1	54.8
1aa7A	162	11.0	29.3	1choE	93	9.9	6.8	1iaiL	223	10.5	70.0
1aalA	60	4.6	5.5	1chpD	109	6.4	12.5	1ibcA	175	13.3	31.5
1aapA	58	6.8	6.0	1ciqA	38	8.7	1.9	1ibgL	223	12.6	56.0
1ab9B	130	8.0	21.4	1cksA	87	10.5	6.7	1iesA	186	8.8	30.4
1ac1A	197	8.9	31.9	1cloL	218	12.1	34.8	1igcL	220	12.8	51.7
1ac6A	114	4.4	19.2	1clyL	224	7.4	32.3	1igfL	226	6.8	66.5
1acyL	221	12.0	67.8	1clzL	227	12.3	34.1	1igiL	224	8.2	54.0
1ad0A	217	7.7	57.7	1cnpA	95	6.5	9.6	1igmL	117	5.7	14.2
1ad9L	226	10.2	59.0	1cnsA	245	5.6	33.8	1igtA	219	12.8	63.6
1afvA	156	9.5	22.0	1cpjA	258	4.9	37.5	1ihvA	58	6.0	4.3
1ag1O	251	8.0	64.1	1cweA	105	8.9	13.0	1ikfL	220	12.4	51.3
1agbA	300	15.4	100.7	1cwpA	149	10.3	13.7	1indL	213	12.3	48.3
1ahsA	126	3.5	20.4	1dbaL	224	7.1	34.8	1ivlA	109	4.3	12.1
1ahwA	223	9.7	62.9	1dclA	214	12.6	28.8	1jckA	250	6.4	38.8
1ai0A	21	4.0	0.7	1dfbL	218	11.7	30.5	1jhlL	111	6.4	13.8
1ai4A	221	15.5	54.7	1dfjE	127	5.6	20.6	1kbaA	67	5.1	4.0
1aifL	221	10.8	51.5	1dhmA	89	7.5	5.9	1kelL	225	11.7	63.6
1aihA	182	15.3	21.0	1dkkA	133	7.7	12.1	1kipA	110	4.2	21.0
1aisA	189	12.4	34.7	1dkyA	213	20.0	24.9	1klaA	119	10.1	20.7
1aj7L	220	7.2	34.5	1dvfA	110	7.9	15.4	1knoA	221	11.9	68.4
1akjA	300	11.6	52.5	1dynA	126	4.7	9.5	1kxiA	65	7.3	4.6
1aksA	124	10.0	10.3	1eapA	221	11.7	40.0	1lbeA	268	13.1	68.5
1anwA	336	7.3	104.3	1efnA	61	11.7	2.5	1lemA	188	10.6	39.1
1ao7A	298	9.2	67.0	1enxA	196	4.3	22.8	1lenA	188	9.4	33.3
1aokA	126	5.4	10.2	1extA	166	12.3	19.2	1lgbA	187	11.5	34.6
1apyA	160	10.1	23.4	1faiL	222	17.7	32.1	1lilA	213	13.9	50.7

Name	Residues	RMSD	Time	Name	Residues	RMSD	Time	Name	Residues	RMSD	Time
1aqdA	196	7.9	24.6	1fbiL	223	17.5	58.1	1lmkA	245	16.5	78.8
1aqkL	212	12.2	35.5	1fccA	221	10.6	55.8	1lynA	140	9.5	22.8
1as4A	357	10.4	124.6	1fdlL	222	15.4	59.3	1mamL	222	13.5	57.9
1askA	133	5.4	24.4	1fgvL	109	10.0	17.0	1mcoL	214	13.3	51.9
1atzA	184	7.2	32.2	1ficA	277	6.1	60.5	1mcpL	225	12.9	35.1
1autC	252	10.7	69.9	1figL	218	23.4	29.1	1mctA	219	6.7	26.0
1avdA	128	10.1	16.9	1fjlA	75	23.4	3.5	1mdaH	344	12.6	60.8
1awbA	276	8.5	94.1	1fleE	243	6.0	34.5	1meeA	260	5.5	83.8
1axsL	220	13.1	54.0	1flrL	226	16.3	32.3	1melA	130	8.0	14.8
1azsA	198	11.4	40.0	1fonA	235	10.7	38.5	1mf2L	221	17.3	55.8
1azzA	220	9.4	58.3	1forL	216	18.9	46.9	1mfbL	214	9.6	53.3
1bafL	220	12.7	57.8	1fptL	226	6.6	33.1	1mhcA	299	15.1	94.8
1bbdL	228	11.3	48.8	1frgL	225	8.7	47.0	1mlbA	221	10.6	61.6
1bbrH	165	11.5	16.8	1ftpA	140	7.7	13.3	1mlcA	221	11.9	51.8
1bcpB	204	7.6	23.4	1fvcA	112	11.3	25.6	1mpaL	226	11.9	51.1
1bcrA	267	13.4	45.3	1fvdA	219	7.5	53.6	1mrcL	223	14.8	51.0
1bebA	164	7.3	14.9	1gamA	97	5.7	11.1	1mspA	132	10.2	17.7
1berA	210	11.2	35.0	1gdtA	190	11.5	42.2	1ndlA	160	9.9	26.8
1bfmA	71	10.8	4.6	1ggbL	222	12.6	34.9	1ngpL	213	8.5	56.8
1bgsA	117	5.4	18.5	1ghfL	219	15.8	31.4	1nldL	226	13.1	65.9
1bjmA	212	11.9	46.2	1ghsA	304	6.8	46.7	1npoA	76	8.4	6.1
1blbA	199	11.0	26.6	1gigL	212	13.0	38.5	1nqbA	242	13.3	76.1
1bmsA	129	10.4	11.4	1hiiA	100	10.1	9.0	1nsnL	226	18.0	50.6
1bplA	195	8.8	24.0	1hiwA	122	8.1	8.2	1opaA	145	7.1	21.8
1bqlL	218	13.3	37.3	1hngA	186	11.4	43.7	1opgL	220	7.8	57.9
1brcE	222	15.2	27.9	1hocA	296	10.2	86.5	1ospL	222	5.5	50.3
1breA	110	8.1	12.8	1hrjA	73	5.4	6.5	1pauA	148	9.0	22.0
1bunA	125	7.5	11.3	1hsaA	300	11.2	85.5	1pciA	326	7.2	81.6
1ca0B	130	6.9	14.3	1hslA	246	5.5	56.4	1pfxC	245	9.9	59.2
1cbiA	145	9.0	13.1	1hstA	75	5.0	8.3	1pmkA	83	8.3	7.4
1cbvL	226	18.2	41.6	1htiA	249	11.1	68.9	1pp2R	126	6.4	22.9
1cfpA	99	10.5	16.2	1htlD	110	4.9	19.5	1ppeE	217	8.6	42.1
1cfyA	141	4.7	13.3	1hulA	116	9.6	25.2	1ppfE	218	7.1	24.5
1cgiE	240	5.7	27.6	1humA	73	5.7	6.5				

Table 3 PDB codes and chain identifiers for the complexes used to train the side chain contact filter on version 2.0 of BiGGER

Dimers	Protease Inhibitor	Other Enzyme Complexes		Antibody Antigen	Electron Transfer
1a2kAB	1avw	1a2kBC	1gua	1bzqAL	1bvy
1ask	1bthHP	1agrAE	1hwgAB	1fbiLH	1ewym
1cc0AC	1cgj	1aipAC	1qfk	1fc2	1MDAm
1cdt	1cho	1ak4AD	1seb	1fdl	2MTAm
1cta	1cse	1ao7AB	1smp	1jhl	2pcb
1dg1GH	1fle	1aro	1stf	1kb5	
1dlfLH	1hia	1atnAD	1stf	1melAL	
1dvfAB	1mct	1dfj	1tx4	1mlc	
1f2xKL	1ppf	1dhkAB	1ycs	1nca	
1fc1	1tab	1efnAB	2btf	1nmb	
1il8	1tbqHR	1efuAB	2dlf	1nsn	
1msb	1tgs	1egp	2sdp	2jel	
1oun	2sni	1ernAB	2trcBP	1aifLH	
1pp2	2tec	1fin	3hla	1cfq	
1ypi	3sgb	1fss	4cpa		
2ccy	3tgi	1gla			
2rus	3tpi				
2sod					
2tsc					
4mdh					
6ebx					

Table 4 PDB codes for the structures used to test the side chain contact filter and the soft docking modification on version 2.0 of BiGGER

Bound	Pseudo-Unbound	Unbound		
1acb	1dxg	1a19	1fss	2ovo
1sbn	1lza	1a2p	1hpt	2pcb
1tec	2mip	1bpi	1hrc	2pcc
2pcc	3hfl	1brs	1lza	2pka
2sic	3hfm	1ccp	1mlb	2ptc
3sdh	3sdh	1ccp	1mlc	2ptn
	6ebx	1cgi	1sup	2sic
		1chg	1vfa	2sni
		1cho	1ycc	2st1
		1cta	2ace	3ssi
		1fdl	2ci2	4pti
		1fsc	2kai	5cha

Table 5 Percentage of complexes in the SPIN-PP dataset with at least one correct model, as a function of the angular step (º)

RMSD Cutoff	0º	4º	8º	12º	16º	20º	24º	28º	32º	36º	40º
3Å	90%	86%	72%	61%	54%	51%	48%	48%	47%	46%	45%
4Å	90%	88%	78%	67%	60%	54%	51%	50%	48%	47%	47%
5Å	90%	89%	81%	71%	64%	58%	54%	52%	50%	48%	47%

Table 6 Percentage of complexes in the SPIN-PP dataset with at least one correct model, as a function of the atom radius modifier.

Case	0	0.2	0.4	0.6	0.8	1	1.2	1.4	1.6	1.8	2
Without H atoms	0%	0%	56%	82%	88%	90%	86%	63%	43%	32%	23%
With H atoms	0%	21%	76%	85%	89%	85%	63%	43%	30%	21%	12%

Table 7 Categories for the 20 basic amino acid residues used in the V28 contact vector .

Residue	Hydrophobic	Polar	H Donor	H Acceptor	Aromatic	Neg	Pos
Alanine	y						
Arginine			y				y
Asparagine		y	y	y			
Aspartic Acid				y		y	
Cysteine		y	y				
Glutamic Acid				y		y	
Glutamine		y	y	y			
Glycine	y						
Histidine			y				y
Isoleucine	y						
Leucine	y						
Lysine			y				y
Methionine		y					
Phenylalanine					y		
Proline	y						
Serine		y	y	y			
Threonine		y	y	y			
Tryptophan					y		
Tyrosine			y	y	y		
Valine	y						

Table 8 Average t values for the differences between correct and incorrect contact averages (Figure 9-6)

C. Vector	-5Å	-4Å	-3Å	-2Å	-1Å	0Å	1Å	2Å	3Å	4Å
V210	1.97	3.15	5.60	9.75	15.30	21.12	25.45	28.58	31.03	33.22
V28	4.21	10.94	23.08	36.29	44.28	47.34	48.09	48.24	48.87	49.66
V15	2.84	7.71	17.44	28.69	37.60	41.75	43.08	43.64	44.43	45.37

Table 9 Residues selected to simulate contact information for selected CAPRI targets. References are: [1] Mathieu and others 2001, [2] Desmyter and others 2002, [3] Sundberg and others 2002, [4]Stetefeld and others 1996, [5] Tibeurgh and others 2001.

ID	Target	Residues selected			Probe	Residue
2	Viral Capsid VP6 [1]	—	—	—	FAB (K + IG)	—
4	Pig Amylase [2]	S243	S245	G249	Camel Amyd10 VHH [2]	F47
5	Pig Amylase [2]	S270	G271	G285	Camel Amy07 VHH [2]	F52
6	Pig Amylase [2]	N53	S145	V349	Camel Amy09 VHH [2]	R51
7	Strp. Pyrog. Exotoxin A1 [3]	N20	N54	Y84	14.3.D T Cell A. R. [3]	G53
8	Laminin[4]	-	-	-	Nidogen-G3	-
9	Wild type LicT	-	-	-	LicT (homodimer)	-

Table 10 Simulating experimental data on amino acid contacts with the selected CAPRI targets

			Best in 5 highest			
CAPRI	Highest acceptable		With 3 AAs		With 3+1 AAs	
Target	Rank	rmsd	Rank	rmsd	Rank	rmsd
2	136	10.8	-	-	-	-
4	1150	2.3	1 (22)	1.2	1 (14)	3.3
5	1374	3.4	1 (18)	6.6	2	5.4
6	8	2.1	1 (63)	3.3	1 (6)	3.3
7	21	1.9	3	1.8	2	1.8

Glossary

AND: Boolean operation that is 1 if both operands are 1, 0 otherwise

BiGGER: Bimolecular complex Generation with Global Evaluation and Ranking. See chapters 6, 9, and 10)

Bound Structure: The structure of a protein found in association with another protein in a complex. May differ significantly from the Unbound Structure.

CAPRI: Critical Assessment of Prediction of Interactions. An experiment for the evaluation of docking algorithms and the prediction of protein complexes (Janin 2002; Janin and others 2003)

Chemera: A Windows application for molecular modeling and the display of protein structures. Includes the BiGGER and PSICO modules, several modelling tools, and can display structures and calculation results.

Condensation Reaction: Chemical reaction in which two reactants combine into a single product with the formation of water.

Core: The inner region of a protein structure, as opposed to the Surface.

CSG tree: A Constructive Solid Geometry tree a procedural representation of solid shapes. The tree stores a hierarchy of elementary shapes, such as cuboid blocks, spheres, cones, and so forth, and Boolean operations that connect them (intersection, union, and difference). The shape can be reconstructed by following the procedure of combining the shapes using the operations in the order specified by the tree.

Distance Constraint: A constraint that specifies an allowed range of distances between two points, generally two atoms.

Docking: Predicting the structure of a complex of two molecules (for example, protein docking)

DNA: Deoxyribonucleic Acid

Exclusion Region: Region defined by an Out Constraint from the Good region of the atom domain, and from which the other atom must be excluded.

FFT: Fast Fourier Transform

Geometric Hashing: A technique of dividing a set of coordinates into regions of space to speed up searche and matching.

Good Region: A cuboid block defining a volume where the atom can be located

In Constraint: A constraint specifying the maximum distance between two atoms.

Neighbourhood: The Good region of an atom domain expanded by the value of the In Constraint under consideration.

NOESY: Nuclear Overhauser Enhancement Spectroscopy

No-Good Regions: A set of zero or more cuboid blocks within the Good Region that specify volumes from where the atom is excluded.

OR: Inclusive Or, Boolean operation that is 1 if one of the operands is 1, 0 otherwise

Out Constraint: A constraint specifying the minimum distance between two atoms.

PDB: Protein Data Bank. A repository for protein structures.

Primary Structure: Amino acid sequence of a protein chain.

Probe: One of the two docking partners that is rotated and translated to search for the best complex configurations.

Protein complex: A protein structure formed by two or more protein chains. See Quaternary Structure.

PSICO: Processing Structural Information with Constraint Programming and Optimisation

Quaternary Structure: The structure formed by several protein chains. Se Protein Complex.

Rigid Group Constraint: An n-ary constraint specifying the relative position of a set of atoms.

RNA: Ribonucleic acid.

Secondary Structure: In a protein chain, a structure stabilised by local interactions.

Surface Contact Score: The number of surface grid cubes of one docking partner that overlap the surface grid cubes of the other.

Target Complex: A known complex to be modelled.

Target Structure: The known structure to be modelled, either a complex in the case of BiGGER or a protein structure in the case of PSICO

Target: Refers to the largest of the docking partners in BiGGER, which is fixed while the Probe is rotated and moved to explore possible docking configurations. See also Target Structure and Target Complex.

Ternary Structure: The folding of the protein chain, that confers it its three dimensional structure.

Unbound Structure: The structure of a protein as found outside a complex (see Bound Structure).

V15: Contact vector for the side chain contact filter and for the global scoring function. See sections 6.4, 6.7, 9.1. See Table 1 for the five classes of amino acids that give the contact vector of dimension 15.

V28: Contact vector for the side chain contact filter. See sections 6.4 and 9.1. See Table 7 for the seven amino acid categories that result in a contact vector of dimension 28.

XOR: Exclusive Or, Boolean operation that is 1 if the operands differ, 0 otherwise

Index

A

Aldehyde Oxydoreductase, 114

B

backjumping, 17

backtracking, 17

BiGGER, 57
  applications, 113
  constrained docking, 64
  Core grid, 60
  geometric filter, 59
  probe, 59
  scoring, 69
  search, 58
  soft docking, 62
  surface contact score, 60
  Surface grid, 60
  target, 59

C

CAPRI, 123

Chemera, 71
  applications, 113
  contact score, 75
  symmetry score, 74

Clustering, 72

constraint
  In Constraint, 32
  Out Constraint, 32
  Rigid Group, 34

Constraint Programming, 14

Constraints, 15

Contact parameters, 108

Contact score. See Chemera contact score

crystallography. See X-ray crystallography

cuboid, 30

Cytochrome b5, 123

Cytochrome C, 123

Cytochrome c Peroxidase, 118

Cytochrome c550, 118

Cytochrome c553, 120

D

Dll. See PSICO:Dll

Docking, 12

Domains, 28

Dynamic Link Library. See PSICO:Dll

E

Enumeration, 48

F

fail first, 17

Ferredoxin, 120, 121

Ferredoxin NADP+ Reductase, 121

Flavodoxin, 114

Folding. See Protein folding

G

generation and test, 16

geometric hashing, 12

Good region, 28

H

Heuristics, 17

L

local search, 17

M

Multidimensional Scaling, 129

N

NMR spectroscopy, 9

NoGood region, 28

O

Optimisation. See PSICO optimisation

P

Poisson-Boltzmann, 76

Propagation, 15

Protein
  docking. See Docking
  folding, 5
  modelling, 8
  primary structure, 3
  secondary structure, 5
  structure, 3
  ternary structure, 5

PSICO
  Distance constraint, 31
  Dll, 147
  Exclusion region, 33
  Good Region, 30
  No-Good regions, 30
  Optimisation, 50
  PSICO Algorithm, 27

R

RMSD, 71

S

Search algorithms, 16

Symmetry. See Chemera symmetry score

X

X-Ray crystallography, 9