Constraint Programming in Structural Bioinformatics

Pedro Barahona and Ludwig Krippahl

Dep. de Informática, Universidade Nova de Lisboa, 2825 Monte de Caparica, Portugal
ludi@di.fct.unl.pt, pb@di.fct.unl.pt

Abstract. Bioinformatics aims at applying computer science methods to the wealth of data collected in a variety of experiments in life sciences (e.g. cell and molecular biology, biochemistry, medicine, etc.) in order to help analysing such data and eliciting new knowledge from it. In addition to string processing bioinformatics is often identified with machine learning used for mining the large banks of bio-data available in electronic format, namely in a number of web servers. Nevertheless, there are opportunities of applying other computational techniques in some bioinformatics applications. In this paper, we report the application of constraint programming to address two structural bioinformatics problems, namely protein structure prediction and protein interaction (docking). The efficient application of constraint programming requires innovative modelling of these problems, as well as the development of advanced propagation techniques (e.g. global reasoning and propagation), which were adopted in Chemera, a system that is currently used to support biochemists in their research.

1. Introduction

For the last decades, a huge set of data is being collected in a variety of experiments in life sciences, ranging from several branches of biology (e.g. cell and molecular or evolutionary), to biochemistry or medicine. Such data is usually kept in some electronic format and made widely available through various web servers. Moreover, many such servers and services are maintained for scientific purposes and, with some notable exceptions (e.g. pharmaceutical databases), their access is public and free.

Such wealth of data cannot be conveniently analysed “by hand”, which justifies the increasing importance of Bioinformatics, roughly defined as the application of computational methods and tools to biological data to allow the analysis of such data as well as the extraction of new knowledge from it.

Although bioinformatics is often identified with string processing and data mining, we show that there are opportunities for applying other computational techniques, namely constraint programming, to some problems in this area. In this paper, we aim at showing the relevance of constraint programming techniques in solving two such fundamental bioinformatics problems in the domain of structural biology, namely protein structure determination and protein interaction (docking).

The structure of the paper is as follows. The next section addresses these structural biology problems of in the wider context of bioinformatics, and discusses alternative approaches to solve them. Section 3 presents our work in the protein structure determination, regarding the modelling that we used and how constraint programming techniques take advantage of it. Section 4 focuses on the constraint programming components of our handling of protein docking and their relevance in improving the overall efficiency and expressiveness. Finally, section 5 presents a summary of the results achieved and some concluding remarks.

2. Bioinformatics and Constraint Programming

During the past decades, a huge set of data has been collected in a variety of experiments in life sciences. Given that most data concerns DNA, RNA and proteins, which are all long polymer molecules that can be coded by means of appropriate alphabets (4 letters to identify DNA and RNA bases and 20 letters to identify the amino acid residues of proteins), many bioinformatics tools involve string processing, namely algorithms such as BLAST or FASTA, that find sequences, similar to those of interest, in many biological databanks. Another important class of computational methods regards machine learning, often used for mining these databanks (and more general bioinformatics servers) in order to discover interesting patterns hidden in the information being stored (e.g. the identification of active sites or secondary patterns in proteins, protein interactions reported in scientific journals, etc).

Notwithstanding their importance, string processing and data mining are not the only computational methods used in structural bioinformatics (the branch of bioinformatics that addresses problems concerning the structure of biologic molecules). In particular, we have been adopting the constraint programming paradigm in two important structural bioinformatics problems, namely a) the prediction of protein structure (the 3D shape of proteins) and b) the interaction (docking) of proteins.

These are two related problems, since in addition to other physico-chemical properties (e.g. charge, hydrophobicity, hydrophily, and polarity) a key factor that determines whether two proteins (or a protein and some other ligand of pharmaceutical interest) interact is that they have shapes that allow them to spatially fit in the contact surface areas (active regions).

These problems are central in molecular biology. On the one hand, biological processes are usually complex networks of interacting molecules (metabolic pathways), where proteins often act as catalyzing enzymes. Although a global understanding of the complete metabolic pathways needs support from techniques (e.g. quantitative or more qualitative simulations [1]), they ultimately rely on “individualised” protein interactions.

Such interaction strongly depends on the structure of proteins. Since this primary structure is often known, as there is a simple mapping from genes to proteins (each DNA codon, sequence of 3 DNA bases, maps into a specific amino acid), it is tempting to predict the tertiary (3D) structure of a protein from its primary structure (the sequence of amino acid residues that compose them). Though it is often assumed that the structure of a protein is mostly determined by its amino-acid sequence, as proteins are naturally folded by nature during their assembly by sequential addition of amino-acids, the folding is determined by the kinetics of the process and thus modelling based on minimising some energy function does not scale up to more than the smallest proteins, for both theoretical reasons and due to the computational costs of dealing with large proteins.

Hence, alternative approaches either adopt some simplified models or seek additional information. Among the first, it is worth referring to lattice models, where proteins are modelled at the amino acid level (not atom level) where, for simplicity, it is imposed that the amino acid residues be placed in the vertices of some lattice structure (e.g. cubes or face-centred cubes). The energy function is also drastically simplified. Rather than considering a number of molecular forces (electrostatic, van der Waals, and entropy), some of which act at a distance, the model scores only the contacts between amino acids which are not contiguous in the protein sequence. By so doing, structures obtained tend to place hydrophobic amino acids in the centre of the protein, whereas the polar amino acids, which interact positively with the water that typically surrounds the proteins, tend to be placed on the protein surface. Although these simplified models are quite interesting from a computational point of view, and indeed they have been addressed with constraint programming [3, 4], their interest to biochemists is not clear. Moreover, it does not seem possible to include additional experimental information without deeply affecting the optimisation algorithms and heuristics that are used.

In alternative, information may be added to the primary structure of a protein to support the prediction of its tertiary structure. For example, homologies of parts of the protein primary sequence with known similar structures, enable the composition of the protein tertiary structure from these known structures, namely secondary motifs such as alpha-helices or beta-sheets, or more general domains [29].

The approach we have been using takes into account another type of information, namely distance constraints between pairs of atoms, indirectly obtained by Nuclear Magnetic Resonance experiments. As such the structure prediction problem is transformed into one of constraint satisfaction, where the constraints are bounds to the allowed distances between some atom pairs. As explained in the following section, handling this problem in constraint programming makes use of many advanced techniques exploited in this paradigm, namely propagation of global constraints, heuristic search, and complementary used of backtrack search and local search.

Two proteins bind together with relatively weak interactions, such as charge complementarity, hydrogen bonds, or hydrophobicity effects. Such weak interactions are only effective if the structures have a large contact area, and to calculate these interactions and the surface contact between the proteins it is necessary to know the structure of each protein. Once the structures are known, surface matching can be modelled by a number of techniques, e.g. FFT [14] and Geometric Hashing [30]. The “naïve” modelling in real space is not as efficient as these approaches, but it may be improved by advanced techniques. We explain in section 4, the improvements induced by our use of constraint propagation techniques in such real-space models, namely when the search for such surfaces is directed to active sites of the proteins (i.e. the areas of the protein surfaces where there is some evidence, obtained for example from homology studies or from philogenetic information on conserved regions of the proteins, that suggests that interactions are more likely to occur). These two applications of constraint programming to structural bioinformatics are explained in more detail in the next two sections.

2. Modelling protein structure.

Since determining the structure of proteins with *ab initio* or first principle methods faces major computational difficulties, we adopted an alternative model for structure prediction, that takes into account as much information as possible, including available experimental data, in the form of distance constraints between pairs or more global sets of atoms.

Several sources of information can help modeling the structure of a protein. Firstly, the primary structure of the proteins, i.e. the amino acid sequences of the protein chains, constrains inter atomic distances in many atom pairs, angles formed by atom triplets, of even larger groups of atoms that are effectively rigidly bound together by the chemical bonds. NMR data provides distance constraints by showing that two atoms must be close enough for the Nuclear Overhauser Effect to be felt, limits the angles of rotation around some chemical bonds, or can even suggest limits for relative special orientations of groups of atoms with Residual Dipolar Coupling data. Furthermore, homology with known structures or modelling secondary structure can provide detailed information of the structure of parts of the protein being modelled.

This information can be divided into three types of constraints: distance constraints between two atoms, group constraints that fix the relative positions of a group of atoms in a rigid configuration, and torsion angle constraints that restrict the relative orientation of two groups joined together by a chemical bond. The constraint programming approach that we adopted considers these constraints in two phases: firstly, a backtrack search is performed where enumeration of variables is interleaved with propagation of distance constraints between pairs and sets of atoms, until an approximate solution is found. Such approximation does not guarantee chemically sound structures, as the dihedral angles of chemical bonds are distorted. Hence, a second phase starts by producing the closest solution to the previous, but with correct dihedral angles, which is then subsequently subject to a local search optimisation, where the dihedral angels are the variables considered.

2.1 The Basic First Phase Algorithm: Dealing with Pairwise Distance Constraints

Since we are modelling distance constraints between atoms, the first issue to address is the representation of domains and constraints. To simplify constraint propagation rather than adopting variable domains (integers, or real numbers) usual in constraint programming, we considered specialised domains for the centre of the atoms, which should represent the positions in space where these centres may lie, and that will be subsequently reduced by constraint propagation and enumeration. Although distance constraints to a point are ideally represented by means of (the inside or outside of) spherical regions, the intersection of such spherical region is rather complex to maintain. Hence, we adopted a cuboid (the Good region) representation for the spatial domain where an atom centre may lie, with the exception of a (possibly empty) set of included cuboids (the NoGoods). Such domains are decreased by the propagation of distance constraints as well as the interleaved enumeration, as explained below.

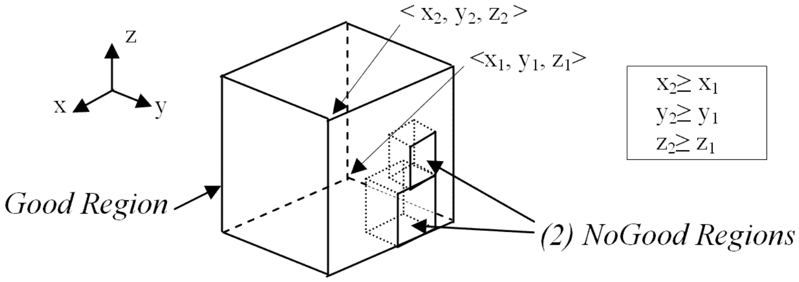
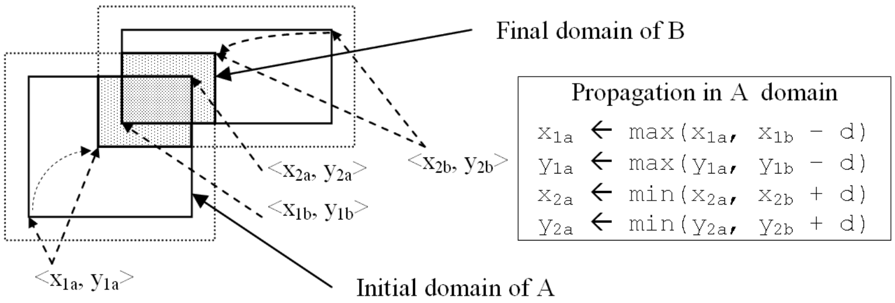


Fig. 1. Domain representation of the centre of an atom

Because the cuboids are aligned with the x,y and z coordinates, they can be simply stored as the coordinates of two opposed vertices $<x_1,y_1,z_1>$ and $<x_2,y_2,z_2>$, respectively, the minimum and maximum coordinates in each dimension. As already mentioned, distance constraints should ideally be considered as Euclidean, but exact propagation of such constraints would be too expensive, and rather than using the Euclidean norm for distances we used the Manhattan Norm, as explained below.

Distance constraints arise not only from NMR spectroscopy experiments, but also from bond length constraints and bond angle constraints between atoms in the same amino acid, which are known beforehand. All these constraints can be divided into two types of distance constraints: **In** constraints impose that two atom centres are within a certain distance d, whereas **Out** constraints do the opposite. With our representation of domains and models they are propagated quite easily, as shown in Figures 2 and 3 (shown in 2D).

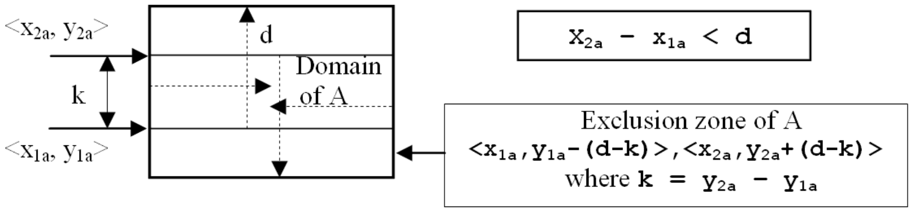
The ***In*** constraints, **(A-B) = d**, where A and B denote the centre of atoms A and B, and d, the maximum distance between them, are propagated by simple intersection. The Good region of an atom becomes the intersection of its current Good region with the Good region of the other atom, augmented by d.



Final domain of B
Propagation in A domain
$x_{1a} \leftarrow \max(x_{1a}, x_{1b} - d)$
$y_{1a} \leftarrow \max(y_{1a}, y_{1b} - d)$
$x_{2a} \leftarrow \min(x_{2a}, x_{2b} + d)$
$y_{2a} \leftarrow \min(y_{2a}, y_{2b} + d)$
<x2a, y2a>
<x2b, y2b>
<x1b, y1b>
<x1a, y1a>
Initial domain of A

Fig. 2. Propagation of an **In** constraint, **(A-B) = d**

For an ***Out*** constraint, **(A-B) = d**, the propagation adds a NoGood region to an atom, corresponding to the exclusion zone of the other, obtained from the current domain of the latter, augmented by max(0, d-k), where k = w_2-w_1, is the length of the Good region of this atom in any of the 3 dimensions x, y or z.



<x2a, y2a>
k
<x1a, y1a>
d
Domain of A
$x_{2a} - x_{1a} < d$
Exclusion zone of A
$<x_{1a}, y_{1a}-(d-k)>, <x_{2a}, y_{2a}+(d-k)>$
where $k = y_{2a} - y_{1a}$

Fig. 3. Propagation of an **Out** constraint, **(A-B) = d**

Arc-consistency is guaranteed by propagating all the constraints on each atom that suffered a domain restriction until no domain changes (as in AC-3). After complete propagation, one atom is selected for enumeration, and the propagation step is repeated.

Enumeration interleaves with arc-consistency maintenance. Each enumeration corresponds to selecting an atom and halving its domain. The variable selection heuristic is a typical first-fail heuristic: select the atom with smallest domain, in this case the volume of the Good Region. However, there is an adaptation suitable to the bisection of the domain: atoms are selected in round robin, i.e. before an atom is selected twice, all other atoms must be selected.

The value selection heuristic being used is also a typical maximum likelihood heuristic. The cuboid representing the Good region is bisected across its largest dimension, and the selected half cuboid is the one least constrained by other atoms, i.e. with least intersection with other atoms good regions. Additional considerations, such as the chemical nature of the amino acid or the prediction of local structures should also inform the choice of which regions of the domain to eliminate, as will be discussed later (section 2.4).

This process of selection and domain reduction is repeated until all atoms were selected once, after which a new round of enumeration starts (in case of failure backtrack occurs). Due to uncertainty on exact distances, as well as the approximations made, it is not worth to allocate very precise locations to the atoms centres, and so the whole enumeration process stops when the atoms have an acceptable size of their Good region (i.e. the length in any dimension does not exceed a certain threshold, typically 1 to 2 Å, since the typical atom size is about 1 Å^3). This policy is also appropriate to cases when the set of constraints is inconsistent due to experimental noise, as even if only partially correct the structure can help the user identify and correct the inconsistencies by reassigning the constraints, rather than just being informed, after a long time backtracking search, that the constraints are unsatisfiable.

2.2 Global Reasoning for Propagation of Rigid Groups of Atoms

There is often information about the structure of the proteins, additional to pairwise individual constraints. In particular, groups of atoms are known (or suspected) to form rigid groups, namely prosthetic groups, secondary structures like alpha-helices, or more complex domains obtained by homology modelling. For these rigid groups, we can know the relative positions of all atoms within the structure of the group, although in an unknown position and orientation relative to the whole protein structure. Hence, the independent propagation of the constraints between all pairs of atoms of the group is not as informative as the propagation of the whole set of constraints, as a single global constraint.

More formally, reasoning globally with all the atoms of rigid groups, i.e. maintaining generalised arc-consistency, achieves better propagation than simple arc-consistency on the pairwise distance constraints, a common situation in constraint programming in finite domains. Nevertheless, specialised and appropriate algorithms are required for global reasoning, so that the balance between better propagation and the longer reasoning which is required tips towards the former. This section briefly outlines how generalised arc consistency is achieved with global rigid-group constraints.

As common in global reasoning, a feasible solution is obtained first, and from that solution values for variables that do not belong to any solution are eliminated from the domains of the variables. We will illustrate this procedure both for translations and rotations of the rigid groups.

Given a fixed orientation, pruning the domains of the atoms in a rigid group, through translations, can be achieved quite simply, in three steps. Let us assume a rigid group is composed of a set on n atoms (i in 1..n), and the center of these atoms have domains (good regions) defined by their lower and upper bounds, respectively triples <x_{1i}, y_{1i},z_{1i}> and <x_{2i}, y_{2i},z_{2i}> (where w_{2i} > w_{1i} for all coordinates, and w denotes any of the {x,y,z} coordinates). Firstly, a solution <x_i, y_i, z_i> is obtained, where all atoms are inside their domains (w_{1i}=w_i =w_{2i}). Secondly, given such solution, the translation in any single direction by any atom is limited not by the distance to the limit of its own good region, but by the smallest distance to the limit of its Good Region of any of the atoms of the rigid group. Without loss of generality, the maximum increase/decrease in the w coordinate that any atom can suffer is given by

$????????_{w\,=}\ ?_w^+ + ?_w^-$ where $?_w^+ = \min_i(w_{2i} - w_i)$;$???????_w^- = \min_i(w_i - w_{1i})$,

Thirdly, these ? s can be subsequently added to the current positions, to obtain the new upper and lower bounds of the good regions of all atoms. This is shown (for 3 atoms, in 2D in Figure X, below).

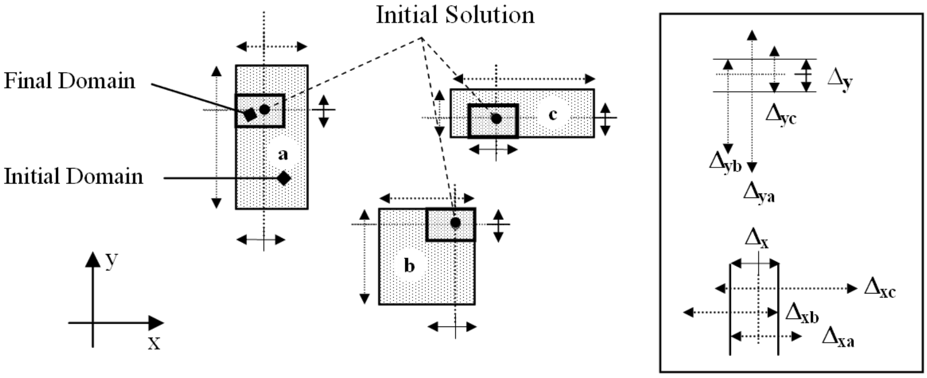


Fig. 4. Possible translations (in 2D) of a group of 3 atoms (a, b, c). On the right, the computation of the maximum ? used to obtain the new Good regions for all the atoms

The procedure to handle constraints on rotation angles between rigid groups is a simple extension of this approach to additional degrees of freedom. The details cannot be described in this paper, but are discussed in [7] and [31] for full details. The whole procedure is not too expensive, as it only involves the computation of angles for which two sine functions intersect.

2.3 Second Phase: Optimisation on Dihedral Angles

As discussed above, when enumeration terminates, each atom has a small cuboid domain, and a more exact position of the atom is obtained through an optimisation procedure. This procedure should take into account the need to produce solutions which are chemically acceptable. In fact, if the geometric centre of the cuboids is considered to place the centres of the atoms, then the resulting molecular structure does not respect the distance and angle values for the chemical bonds.

In fact, a molecule changes configuration by groups of atoms rotating around a chemical bond. It is this process that allows proteins to fold into their shapes, and the angle of such a rotation is called the torsion angle.

To address these problems, the constraint propagation method described above is complemented with a local search component that implements a simple torsion angle optimisation algorithm. This type of algorithm is a particularly good choice for this problem because of the significant reduction in the number of variables used; there is approximately one torsion angle for every 5 atoms, thus a 15 fold reduction from the x, y and z coordinates for each atom to the one torsion angle coordinates. Hence, not only is calculation speed improved but also chemically sensible solutions are produced.

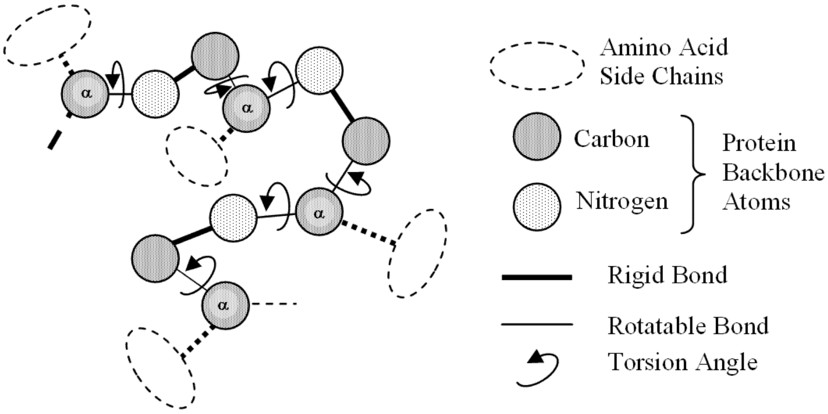


Fig. 5 Dihedral angle model for protein folding.

The minimisation proceeds in two steps. In the first step the torsion angle values for the torsion angle model are adjusted to minimize the distance between the atomic positions in the structure provided by the CP stage and the respective positions in the torsion angle model. This fits the torsion angle model to the CP solution, thus providing a chemically sound structure close to respecting the distance constraints. The second step is to minimize constraint violations. The variables are again the values for the torsion angles that define the structure, but now the function to minimize includes the violation of constraints and inter-atomic repulsion.

Both minimization steps use the conjugated gradient method, which is essentially a steepest descent method, modified to ensure that the search proceeds along conjugated directions, which improves efficiency (details can be found in [8]).

Finally, an additional simulated annealing search can be included after the fitting of the torsion angle model and before the final minimization. Thus several slightly different solutions can be generated, and since less constrained regions will result in a wider dispersion of structures, this allows the user to estimate how well the constraint set defines the structure

2.4 Results and Future Improvements

PSICO [5] is the first implementation of the algorithms above (without rigid groups) that is integrated in the Chemera system (although not yet in the public distribution version of Chemera), a tool developed to assist biochemists and other researchers in their protein structure prediction studies [2]. Initial tests performed with PSICO with real data (the Desulforedoxin dimer, with 520 atoms and about 8000 constraints where over 800 are provided from NMR data and the rest from amino acid knowledge) shown acceptable results achieved in approximately twenty seconds for the CP phase, plus a few minutes for the optimisation phase to generate several models.

This is significantly faster than the reference system currently used in this area (DYANA [22]) that uses a simulated annealing approach to the problem and can take hours to solve the problem. Even state of the art algorithms [23] take about fifty times longer than the CP phase of PSICO. Nevertheless, the accuracy achieved with DYANA is significantly better, due to an optimized alternating scheduling of minimization and molecular dynamics, achieving RMSD distances of about 1?Å between the predicted and the actual structures, compared with 2.3 Å, achieved by PSICO. Although significant, this error does not prevent PSICO from assisting biochemists in the interpretation of NMR data. In fact, in earlier stages of peak assignment and structure determination, distances are not assigned to the correct atom pairs, and so a fast, if only approximate, calculation is quite useful to alert biochemists that some of the distance constraints should be revised. It is also worth noting that the 2.3Å RMSD is close to the 2Å or 2.5Å threshold values typically used for the final domain size in PSICO.

The integration of global rigid body constraints has not been done yet, but we expect that PSICO performance should improve considerably with such integration. Preliminary results have shown that the propagation of alpha-helices with 20 atoms or over (i.e. with 5 residues or more) typically decreases the union of the domains of the atoms by a factor of 10, with no sensible increase in run time [7]. However, run times depend significantly on the size of the rigid bodies that are considered and the actual propagation policy, i.e. the interplay between propagation of fast binary constraints, and heavier global constraints.

Of course, the choice of the rigid bodies to consider is also a key factor for the integration of rigid body constraints. Currently, secondary structures such as alpha-helices and beta-sheets can be predicted quite accurately by homology reasoning, taking into account the vast amount of proteins whose structure is already known, and maintained in the PDB data bank, publicly accessible via the Web. In fact, this is a study we are currently undertaking in the Rewerse European Network of Excellence, that aims at developing Semantic Web tools and apply them to Bioinformatics, among other domains [25].

Some experimental techniques may provide constraints on torsion angles, which is a useful information when modelling a protein structure, namely when the propagation of these constraints is seen as an extension to the rigid group constraint propagation discussed in the previous section. We can consider that two rigid groups connected by a bond allowing rotation is a single rigid group if the torsion angle is fixed. If the torsion angle is an interval, we can account for the relative coordinates of all atoms in the two groups by using the corresponding intervals, in a way similar to that discussed in the previous section.

The tuning of propagation of all these types of constraints over atoms and/or rigid groups, propagation will be possible with CaSPER, a constraint propagation system that we started developing recently [24]. CaSPER allows an easy tuning of the strategies for propagation of fast local constraints and heavier global constraints (e.g. over rigid groups). At the moment we are using CaSPER with a simplified model that only considers the backbone of the proteins, but it will be tested soon with PSICO problems with a variety of rigid groups.

Regardless of the propagation, the performance of PSICO is quite dependent on the enumeration heuristics used. The heuristics that is still being used chooses the half domain less occupied by the domains of all other atoms, and does not take into consideration any biochemical properties of the amino acids. A data mining study was performed at amino acid level to predict whether the amino acids are buried in the protein complex or at its surface, with a success rate of around 80% [26].

This is quite close to another study we have performed that indicates a sensible decrease in the overall RMSD error of different proteins if this rate of success was achieved (but at an atom level). For example, before the optimisation phase, we achieved RMSDs below 4Å if the rate of success in the heuristics is 80%, rather than around 7Å when choices are correct only 50% of the time [27]. As with global constraints, more data mining and homology studies should be performed in the PDB data to improve the quality of the heuristics being used.

Finally, no heuristic is perfect, and a pure backtrack search will very likely be insufficient, given the size of the problems. A possible trade-off between completeness of search and efficiency is the use of limited discrepancy search, where regions of the search space are visited only if they do not involve overriding the heuristic choice more than a limited amount of times (the discrepancy level accepted [28]). Nevertheless this discrepancy search might have to be complemented with some form of local search in the first choices, which are critical for the performing of backtrack search, and which are very badly informed in the early stages of the search, where the likely positions of the atoms are still very much undefined. This is also a feature of the CaSPER system that is planned for the near future.

3. A docking algorithm

The other structural bioinformatics application where we have successfully applied constraint programming techniques is protein interaction (docking). At the core of our protein docking algorithm is the representation of the protein shapes and the measure of surface contact. The former is a straightforward representation in real space using a regular cubic lattice of cells, similar to that commonly used in the Fast Fourier Transform (FFT) methods derived from [14]. In BiGGER the cells do not correspond to numerical values, but each cell can be either an empty cell, a surface cell, or a core cell. The surface cells define the surface of the structure, and the overlap of surface cells measures the surface of contact.

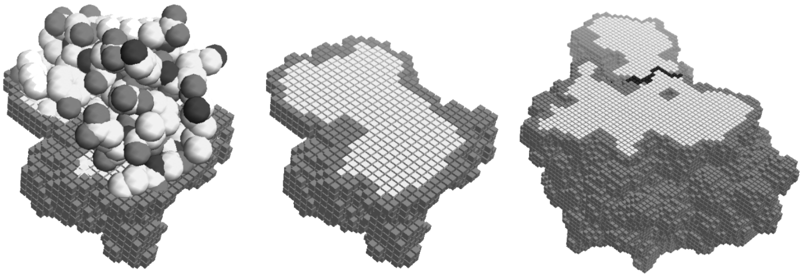


Fig. 6. The image on the left shows a protein structure overlaid on a cutaway of the respective grid, with spheres representing the atoms of the protein. The centre figure shows only the grid generated for this protein, cut to show the surface in darker grey and the core region in lighter grey. The rightmost image shows two grids in contact, with the black line indicating the overlap of surface grid cells.

Figure 6 illustrates these concepts, showing on the first two panels a cutaway diagram of the grid representing a protein structure, and on the third panel a cutaway diagram of two grids in contact, showing the contact region corresponding to a set of overlapping surface cells. With such representation, and for a fixed orientation of the two structures, our goal is to obtain the translation that maximises the number of surface cells of the two structures, constrained to the fact that no core cells of the two structures should overlap.

A naïve algorithm to obtain optimal solutions would require the comparison of N^3 cells of one structure with N^3 cells of the other structure, with time complexity of $O(N^6)$, which would be unacceptably inefficient (typical values of N range from 100 to 200). We show in this section how simple constraint programming techniques (maintenance of bounds consistency) helped improving the algorithm so as to make it competitive with alternative approaches. Moreover, we show that generalised arc consistency is achieved to deal with a special global constraint that can be used to enforce specific activity regions in the docking proteins.

To achieve these results, we encode the grids in a convenient way: instead of individual cells, grids are composed of lists of intervals specifying the segments of similar cells along one coordinate. These lists are arranged in a two-dimensional array on the plane formed by the 2 other coordinates.

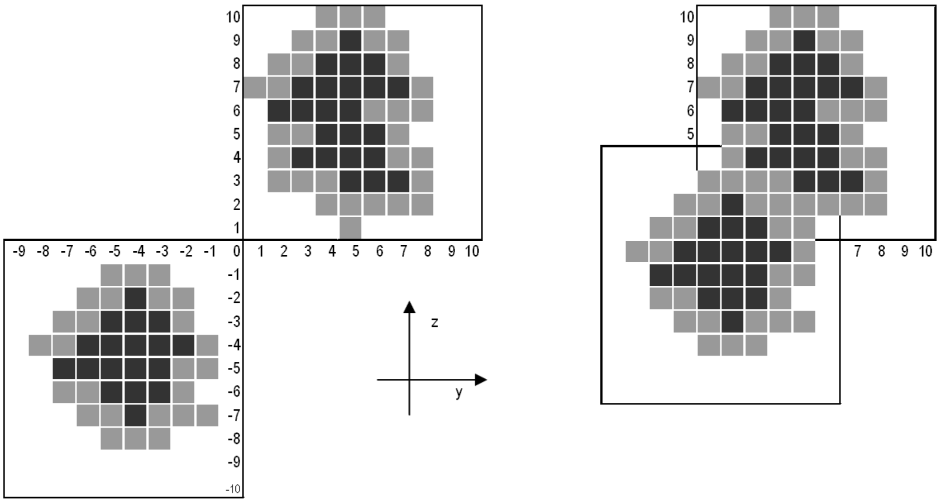


Fig. 7 View of a cut along the YZ plane of structures A and B, in the initial position and when B is shifted 6 cells along the Y axis and 4 cells along the Z axis.

Figure 7 shows a cut through the YZ plane of two structures A and B. These structures are modelled by $2N^2$ lists for structure A, A^c_{ij} and A^s_{ij}, respectively for core and surface cells, and similarly $2N^2$ lists for structure B, B^c_{kl} and B^s_{kl}. Indices i,j take values in interval [-N-1 .. 0], whereas indices k,l take values in the interval [1 .. N]. In the initial position, shown in the left of the figure, no lists for A and B are aligned, so that no surface or core cells overlap. If A is moved p cells along Y and q cells along Z, (as shown the right of the figure, where p= 6 and q = 4) then all lists A_{ij} and B_{kl} for which i+p = k and j+q = l are aligned. Overlap of surface and core cells can be checked in aligned lists A^s_{ij} and B^s_{kl}, and A^c_{ij} and B^c_{kl}, respectively. The study for all displacements p and q may be performed in the nested loop

```
for p in 1 to N                % Y displacement
   for q in 1 to N             % Z displacement
      count_surface_cells(p, q)
```

We show now how this study can be improved by constraint propagation.

3.1 Restricting the search to surface overlapping regions.

A significant proportion of all possible configurations for the two grids results in no surface overlap. Much can be gained by restricting the search to those configurations where surface cells of one grid overlap surface cells of the other.

Figure 8, shows (in 2D), aligned lists of structures A and B, where A^s = [4 ..5] ?? [11..11], B^s = [15..15] ? [23..23] ?for the superficial cells, and A^c = [6,10] and B^c = [16, 22] for the core cells (for simplicity, the indices of the lists were dropped).

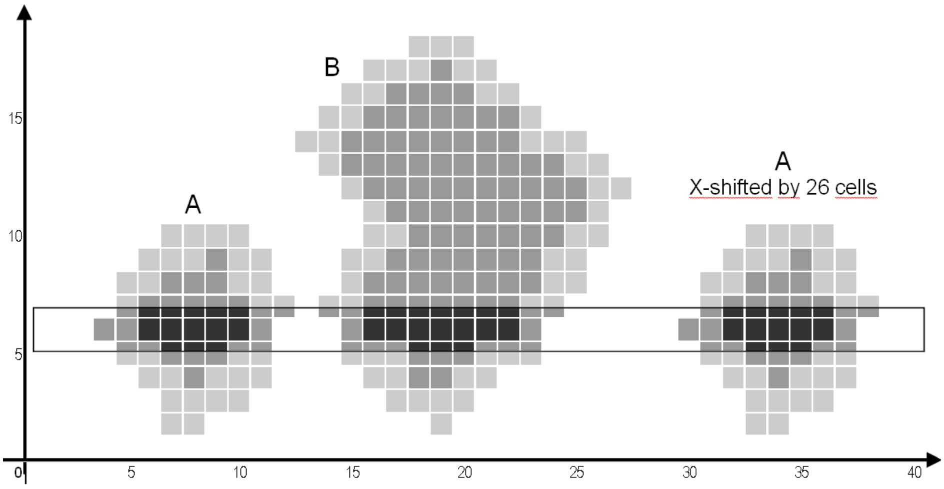


Fig. 8. View of a cut along the XY plane of structures A and B. In the left, A is in its initial position, whereas on the right A was shifted 26 cells along the X axis.

As can be noticed, the encoding of cells in lists, not only reduces the memory requirements for storing the grids, but also simplifies searching along the X axis by comparing segments rather than running through all the possible displacements along this coordinate. Given two aligned surface lists, the X translations that may lead to the overlap of surface cells must be in the interval

$$X \text{ in } [\min B^s\text{-}\max A^s , \max B^S\text{-}\min A^S].$$

In this case, it is easy to check that X in [15-11 .. 23-4] = [4 .. 19]. Now, the relevant displacement of structure A in the X direction is obtained by the union of all these intervals, obtained in the Y, Z nested loops shown above i.e.

$$X \text{ in } [\min_{p,q} X_{p,q} \text{ .. } \max_{p,q} X_{p,q}],.$$

3.2 Eliminating regions of core overlap

The important constraint in this problem is that core regions of the grids cannot overlap, for that indicates the structures are occupying the same space instead of being in contact. This further restricts the X intervals previously obtained from the analysis of surface cells. A similar reasoning on the bounds of the intervals of core cells, allows the identification of the forbidden displacements to occur in the intervals

$$X' \text{ in } [\min A^C\text{-}\max B^C , \max B^C\text{-}\min A^C].$$

In the example, displacement is forbidden in the interval X’ = [16-10 .. 22-6] = [6 .. 16]. Taking into account the previous results for these lists, the possible displacements lie in the intervals X \ X’ = [4 .. 19] \ [6 .. 16] = [4..6[?]16..19]. In general, the relevant displacement of structure A in the X direction, is obtained by the intersection of all these intervals obtained in the Y, Z nested loops, i.e.

$$X' \text{ in } [\max_{p,q} X'_{p,q} \;..\; \min_{p,q} X'_{p,q}],$$

Notice that the number of allowed displacements along X for all p,q pairs is usually a small number, d, usually corresponding to 2 intervals, one where structure A is to the left of structure B and the other where A is to the right of B, as is the case of the example.

Therefore, in both the surface and core cells lists, the BiGGER algorithm imposes bounds consistency on the lists, which requires $O(m^2 N^2)$ operations, where m is the number of intervals defined for each line. Except for fractal structures, m is a small constant. For convex shapes (as shown in the figure) m is always 2 or less, and even for complex shapes like proteins, m is seldom larger than two.

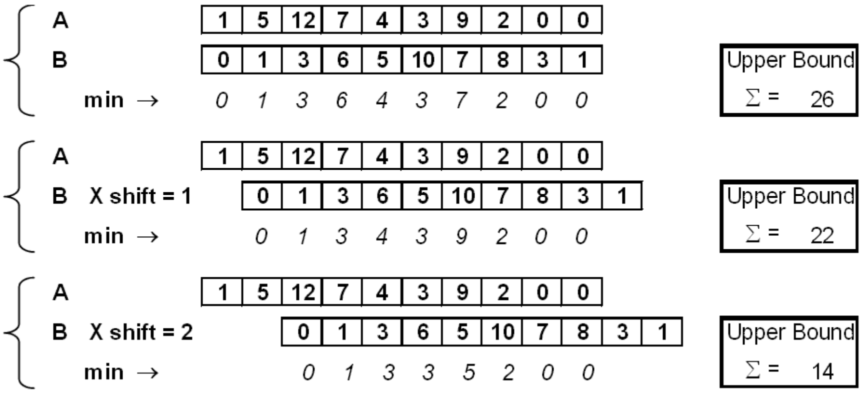
Finally, the overlap of surface cells is determined for each allowed translation value in each coordinate. This requires testing the bounds of the matching surface segments in a way similar to imposing bounds consistency, which is of $O(N^2)$ for all aligned lists of the structures, and then counting the contacts along X, which is of O(d). This procedure has to be repeated for all displacements along Y and Z (i.e. for all p,q of the nested loops).

Taking all these factors into account, the time complexity of the search algorithm when imposing bounds constraints on the overlap of surface and core grid cells is $O(d\ m^2 N^4)$, or simply $O(N^4)$, since m^2 and d are bound by small constant values. Though greater than the $O(N^3 Log(N))$ complexity of the FFT method, the operations done in the BiGGER algorithm are much faster and this constant factor makes BiGGER more efficient for values of N up to several hundreds [13], which include the vast majority of real cases. Moreover, the space complexity of BiGGER is $O(N^2)$, significantly better and with a lower constant factor than the FFT space complexity of $O(N^3)$.

3.3 Restricting the lower bounds on surface contact

Since the docking problem involves an optimisation of the number of contact cells, search can be pruned by branch and bound technique, whenever the search path cannot lead to a solution where these contact cells are less than those in the lowest ranking model being kept. This section explains how this optimisation is implemented in BiGGER, namely in the counting procedure in the Y, Z loops, presented above.

For every value of the X coordinate, each structure has a number of surface cells (that can be determined in the N^2 lists of surface cells, in time $O(N^2)$. The number of overlapping cells for that value of the X coordinate is upper bounded by the minimum of surface cells of the two structures (corresponding to a situation where all cells of the structure overlap with cells of the other). An upper bound for the total of overlapping surface cells is obtained by a sum of these minima over all values of X.



A 1 5 12 7 4 3 9 2 0 0
B 0 1 3 6 5 10 7 8 3 1
min → 0 1 3 6 4 3 7 2 0 0
Upper Bound Σ = 26

A 1 5 12 7 4 3 9 2 0 0
B X shift = 1 0 1 3 6 5 10 7 8 3 1
min → 0 1 3 4 3 9 2 0 0
Upper Bound Σ = 22

A 1 5 12 7 4 3 9 2 0 0
B X shift = 2 0 1 3 6 5 10 7 8 3 1
min → 0 1 3 3 5 2 0 0
Upper Bound Σ = 14

Fig 9. Change in upper bounds of surface cells when structure B is displaced along the X axis. If a docking was already found with 20 overlapping surface cells counting for shift = 2 is avoided.

For every pair p, q of the Y and Z coordinates, and prior to determination of the actual number of overlapping surface cells (which requires a counting in lists of surface cells after their alignment) this upper bound can be compared with some threshold value (e.g. the best model so far), avoiding the counting if the upper bound is less than the threshold. If a fixed number of best models to retain is set, this constraint also allows the algorithm to prune the search space so as to only consider regions where it is possible to find matches good enough to include in the set of models to retain.

These bounds can be computed only once and used for all values of the Y and Z displacements of the structures, and require the sum of the minima obtained in time $O(N^2)$. Hence the whole complexity of this procedure is $O(N^3)$, which does not impose any significant loss in efficiency, given the $O(N^4)$ complexity of the BiGGER algorithm.

This pruning results in a modest efficiency gain of up to 30% in medium-sized grids. However, with larger grid sizes the relatively thinner surface regions shift the balance between the total surface counts and the size of the grid [13], reducing the gain in performance. Still, this can benefit some applications like soft docking [9], where the surface and core grids are manipulated to model flexibility in the structures to dock, or if the minimum acceptable surface contact is high.

3.4 Constraining the Search Space to Active Regions

In some cases there is information about distances between points in the structures, information that can be used to restrict the search region. If this information is a conjunction of distance limits, then it is trivial to restrict the search to the volumes allowed by all the distances. However, real applications may be more complex.

For modelling protein interactions, it is often the case that one can obtain data on important residues or atoms from such techniques as site directed mutagenesis or NMR titrations, or even from theoretical considerations, but it is rare to be absolutely certain of these data. The most common situation is to have a set of likely distance constraints of which not all necessarily hold. Typically, we would like to impose a constraint of the form:

At least K atoms of set A must be within R of at least one atom of set B (1)

where set A is on one protein and set B on the other, and R a distance value. This constraint results in combinatorial problem with a large number of disjunctions, since the distances need only hold for at least one of many combinations of K elements of A.

Since the real-space (geometrical) search of BiGGER can be seen as three nested cycles spanning the Z, Y, and X coordinates, from the outer to the inner cycle, we can decompose the enforcement of constraint (1) by projecting it in each of the three directions:

At least K atoms of set A must be within $R_?$ of at least one atom of set B (2)

where $R_?$ replaces the Euclidean distance R and represents the modulus of coordinate differences on one axis Z, Y or X. $R_?$ has the same value of R; the different notation is to remind us that this is not a Euclidean distance value, but its projection on one coordinate axis. This makes the constraint slightly less stringent, by considering the distance to be a cube of side 2R instead of a sphere of diameter 2R, but this can be easily corrected by testing each candidate configuration to see if it also respects Euclidean distance.

The propagation algorithm is the same for each axis and consists of two steps. The first step determines, for all displacements along one coordinate, X, what atoms of set A are possibly in the neighbourhood of radius R of atoms in group B, i.e. for which atoms A_i and B_j is $|A_{i,Y} – B_{j,Y}| = R$ and $|A_{i,Z} – B_{j,Z}| = R$. The neighbourhoods for each of the atoms B along coordinate X are considered. Then for each displacement in coordinate w1, we count the number of atoms of B that lie on the neighbourhoods of corresponding atoms of the set A. This is shown in Figure 10, for a displacement along the X axis.

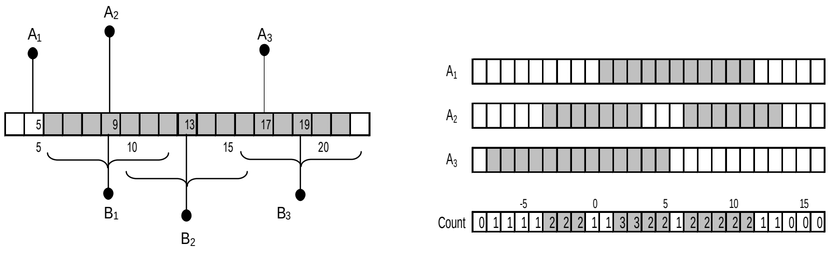


Fig. 10. Generating the displacement domain in one dimension. The left panel shows the generation of the neighbourhood of radius R of group B. The panel on the right shows the allowed displacements for each atom, and the final displacement domain for a K value of 2.

Let us assume that A_1 is in the neighbourhood R = 3 of atoms B_1 and B_2, with respect to coordinates Y and Z, (i.e. $|B_{1y} - A_{1y}| = 3$ and $|B_{1z} - A_{1z}| = 3$, and similarly for A_2) but not in the neighbourhood of atom B_3 (i.e. $|B_{1y} - A_{1y}| > 3$ or $|B_{1z} - A_{1z}| > 3$. In the figure, A_2 is in the neighbourhoods of B_1 and B_3, and A_3 is in the neighbourhoods of B_2 and B_3.

The X coordinates of B_1, B_2 and B_3 are respectively 9, 13 and 18. In the initial relative position of structures A and B, the X coordinates of A_1, A_2 and A_3 are respectively 5, 9 and 17. Let us assume that structure A moves along the X axis.

A_1 is within distance R= 3 of atom B_1 if the X displacement is in the interval [9-3-5 , 9+3-5] = [1 , 7], since the displacements of A_1 in the interval [1 , 7] leads to positions X_1 in 5 + [1 , 7] = [6 , 12], which is the allowed interval [-3, +3] around A_1, 9 + [-3 , +3] = [6 , 12]). Hence A_1 is within distance R of some atom in set B if the X displacement lies in the interval [1, 7] (atom B_1) or in interval [13-3-5 , 13+3-5] = [5, +11] (atom B_2). Similarly, atom A_2 is within distance R = 3 of some atom of set B if the X displacement is [9-3-9,9+3-9] = [-3 , +3] (atom B_1) or [19-3-9,19+3-9] = [7 , 13] (atom B_3) and atom A_3 is within distance R = 3 of some atom of set B if the X displacement is [13-3-17,13+3-17] = [-7 , -1] (atom B_2) or [19-3-17,19+3-17] = [-1 , 5] (atom B_3).

Once we have the displacement segments for all atoms, we must generate the segments describing the region at least K atoms are in the neighbourhood of B, which is a simple counting procedure (hence, constraint (2) need not be limited to specifying a lower bound for the distances to respect. The value of K can also be an upper bound, or a specific value, or even any number of values). In this case, there are at least two atoms of set A within neighbourhood 3 of atom set B if the displacement lies in ranges [-3,-1] or [2, 5] or [7,11]. In range [2,3] all 3 A atoms are in the neighbourhood 3 of B.

The propagation of constraints of type (2) thus restrict the translation domains that are used in the translation search (see last section). The time complexity of enforcing constraint (2) in one axis is O(a+b+N), where a is the number of atoms in group A and b the number of atoms in group B, and N is the grid size. Since this must be done for the translation dimensions the overall complexity contribution is $O(N^3)$, which does not change the $O(N^4)$ complexity of the geometric search algorithm, and pruning the search space speeds up the search considerably [13].

4. Results and Further Work

As discussed, our representation of the structures is quite economic in space, $O(N^2)$, namely when compared with alternative approaches, such as the FFT approach. Moreover, only integers are stored, contrary to FFT, which requires maintaining $O(N^3)$ floating points. Hence, for grids of around N = 100, BiGGER requires about one thousand times less memory (approximately 15Mb in BiGGER vs. 8Gb for FFT in large proteins) and being up to ten times faster than FFT [13]. BiGGER also models side-chain flexibility implicitly by adjusting the core grid representation [9] and allows for hard or soft docking simulations depending on the nature of the interaction to model. Furthermore, this representation and the search algorithm can take advantage of information about the interaction (namely, the active site) to simultaneously improve the results and speed up the calculations.

A common trend is to model interactions using only knowledge derived from the structure and physicochemical properties of the proteins involved. Some algorithms have been developed [9, 10, 11] or adapted [12] to use data on the interaction mechanisms, but this approach is still the exception rather than the norm. BiGGER is one of these exceptions, as it has been developed from inception to help the researcher bring into the modelling process as much data as available, and Constraint Programming techniques have much improved the efficiency and expressiveness of earlier versions [13].

Previous results show that BiGGER can be a powerful modelling tool when used in this manner, even when the experimental data are only applied after the search stage to score the models produced [9, 10, 15, 16, 17, 18, 19, 20, 21]. However, there are two advantages to using the experimental data to constrain the search space. One is in efficiency, since a reduced search space results in faster computations (approximately one order of magnitude, depending on the constraints). The most important advantage is in improving the quality of the results. Due to computational costs, only a limited number of models can be retained for evaluations beyond the geometric search (typically five thousand). If the constraints are only applied to evaluate this set it may be that no acceptable models were retained. By applying the constraints during the search stage it is guaranteed that the models retained will be agree with the experimental data

5. Conclusions

Constraint Programming is a computational paradigm quite adequate to address combinatorial problems given, since on the one hand, its declarative nature that allows problems to be easily modelled and adapted and, on the other hand, the efficiency of the underlying constraint solvers. Of course, to address many problems, namely some arising in Bioinformatics, it is necessary to adopt sophisticated modeling techniques to represent the problem, and thus render it adequate for the application of Constraint Programming techniques.

In this paper we have shown that structural bioinformatics problems can indeed be handled quite successfully with a constraint programming approach, making it possible to incorporate many sources of information, including experimental data (e.g. NMR data). Such inclusion is highly desirable, or even mandatory, since modeling these applications from first principles is strongly limitied due to the sheer size of the search space.

Notwithstanding its key role in the algorithms being used, constraint programming is therefore likely to be complemented, in the development of complete practical applications, with other advanced techniques, namely data mining on the many web available bioinformatics data banks. This has been suggested in the applications described in this paper (e.g. to improve the heuristics in structure determination), for which we expect to obtain soon better results with the integration of such complementary techniques.

Acknowledgements: We thank Nuno Palma and José Moura for their role in the development of BiGGER and Chemera. Developments of Chemera are currently being funded by the European Commission and by the Swiss Federal Office for Education and Science within the 6th Framework Program project REWERSE number 506779 (cf. http://rewerse.net).

References

1. Fages, F., Soliman, S. and Chabrier-Rivier, N. (2004) Modelling and querying interaction networks in the biochemical abstract machine BIOCHAM. Journal of Biological Physics and Chemistry 4(2), pp.64-73. October 2004.
2. http://www.cqfb.fct.unl.pt/bioin/chemera/.
3. Backofen R. , Will S. A Constraint-Based Approach to Fast and Exact Structure Prediction in Three-Dimensional Protein Models, Constraints, Vol.11, N. 1, Springer, January 2006
4. Dal Palú A., Dovier A., Fogolari F., (2004) Constraint Logic Programming approach to protein structure prediction, BMC Bioinformatics 2004, 5:186 (30 November 2004)
5. Krippahl, L., Barahona, P., PSICO: Solving Protein Structures with Constraint Programming and Optimisation, Constraints 2002, 7, 317-331
6. Krippahl, L., Barahona, P., Applying Constraint Programming to Protein Structure Determination, Principles and Practice of Constraint Programming, Springer, 1999 289-302

7. Krippahl L. and Barahona P., Propagating N-Ary Rigid-Body Constraints, Principles and Practice of Constraint Programming, CP'2003 (Procs.), Francesca Rossi (Ed.), Lecture Notes in Computer Science, vol. 2833, Springer, pp. 452-465, October, 2003.
8. Krippahl L, Barahona P. PSICO: Solving Protein Structures with Constraint Programming and Optimisation, Constraints 2002, 7, 317-331
9. 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.
10. Krippahl L, Moura JJ, Palma PN. 2003. Modeling protein complexes with BiGGER. Proteins: Structure, Function, and Genetics. V. 52(1):19-23.
11. Dominguez C, Boelens R, Bonvin AM. HADDOCK: a protein-protein docking approach based on biochemical or biophysical information. J Am Chem Soc. 2003 Feb 19;125(7):1731-7.
12. Moont G., Gabb H.A., Sternberg M. J. E., Use of Pair Potentials Across Protein Interfaces in Screening Predicted Docked Complexes Proteins: Structure, Function, and Genetics, V35-3, 364-373, 1999
13. Krippahl L. and Barahona P., Applying Constraint Programming to Rigid Body Protein Docking, Principles and Practice of Constraint Programming, CP'2005 (Procs.), Peter van Beek (Ed.), Lecture Notes in Computer Science, vol. 3709, Springer, pp. 373-387, October, 2005.
14. 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.
15. 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.
16. 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.
17. 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
18. 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.
19. 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.
20. Palma PN, Lagoutte B, Krippahl L, Moura JJ, Guerlesquin F. Synechocystis ferredoxin / ferredoxin - NADP(+)-reductase/NADP+ complex: Structural model obtained by NMR-restrained docking. (2005) FEBS Lett. 2005 Aug 29;579(21):4585-90.
21. Impagliazzo A, Krippahl L, Ubbink M. Pseudoazurin : Nitrite Reductase Interactions (2005) ChemBioChem 6, 1648-1653
22. Güntert, P., Mumenthaler, C. & Wüthrich, K. (1997). Torsion angle dynamics for NMR structure calculation with the new program DYANA. J. Mol. Biol. 273, 283-298.

23. Wang L., Mettu, R, Donald, B. (2006) A Polynomial-Time Algorithm for De Novo Protein Backbone Structure Determination from Nuclear Magnetic Resonance Data, J. Comp. Biol. Vol 13, N 7, 2006, 1267–1288
24. Correia M, Barahona P, Azevedo F, CaSPER: A Programming Environment for Development and Integration of Constraint Solvers, in Proceedings of the First International Workshop on Constraint Programming Beyond Finite Integer Domains (BeyondFD'05), Azevedo et al. (Editors), pages 59-73, 2005.
25. Krippahl, L. Integrating Web Resources to Model Protein Structure and Function. RW-SISS-'2006 (Procs.), Pedro Barahona (Ed.), Lecture Notes in Computer Science, vol. 4126, Springer, pp. 184-196, September 2006.
26. J.C. Almeida Santos, Mining Protein Structure Data, M.Sc. Thesis, New University of Lisbon, 2006
27. Correia M., Barahona P., Machine Learned Heuristics to Improve Constraint Satisfaction, 17th Brazilian Symposium on Artificial Intelligence, SBIA’04 (Procs.), Ana.L.C. Balzan and Sofiane Labidi (eds.), LNCS, vol. 3171, Springer, pp.103-113, Maranhão, Brazil, 2004
28. Harvey W, Ginsberg M, Limited Discrepancy search, in Proceedings of IJCAI, International Joint Conference on Artificial Intelligence, C. Mellish (ed.), Montreal, 1995.
29. Simons KT, Kooperberg C, Huang E, Baker D: Assembly of protein tertiary structures from fragments with similar local sequences using simulated annealing and Bayesian scoring functions. J Mol Biol 1997, 268:209-25.
30. Wolfson HJ, Rigoutsos I, Geometric Hashing: An Overview, IEEE Computational Science & Engineering archive Volume 4 , Issue 4 (October 1997) 10 – 21, 1997
31. Krippahl, L. Integrating Protein Structural Information, PhD Dissertation, FCT/UNL, 2003