A Constraint Programming Approach to Bioinformatics Structural Problems

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. In this paper we show how Constraint Programming (CP) techniques have been used to handle bioinformatics structural problems, namely in protein structure prediction and protein interaction (docking). Solving these problems requires innovative modelling of the problem variables and constraints, and the application of advanced CP features to handle the problems efficiently, namely the exploitation of global constraints and local search, in addition to more standard binary constraint propagation. Both applications, respectively PSICO (Processing Structural Information with Constraint programming and Optimisation), and BiGGER (Bimolecular complex Generation with Global Evaluation and Ranking) have been incorporated in a platform, Chemera, that aims at supporting (and has effectively supported, namely in protein docking), biochemists in their research.

1. Introduction

Bioinformatics is an increasingly important discipline where computer science methods are used to study biological entities and systems. This is especially important as a wealth of biological data have been gathered in recent years, in many cases freely accessible over the Web. By taking advantage of this availability, computational models may suggest what are the most promising outcomes, possibly decreasing the number (and cost) of experiments that are required to study certain biological properties.

Many interesting problems, requiring significant computational resources do arise in Bioinformatics. For example, sequencing problems are common, where one aims at reconstruct some large polymer (e.g. protein, DNA or RNA molecules) from a sequence of partially overlapping fragments of the polymer obtained in some experimental setting.

Other problems are related to the behaviour of biochemical systems, in particular metabolic pathways, that study the interaction of molecules along the time. Interesting pathways usually regard the processing of some biochemical molecules (for example, the citric acid cycle, glycolisis and other chemical reactions that are responsible for the metabolism of a cell), and are eventually related to gene expression, i.e. when genes become active and produce the corresponding enzymes and other proteins.

Finally, structural problems, such as the determination of the three-dimensional structure of proteins and other molecules, are also very important. Such determination is of particular interest in bioinformatics since in addition to other electro-chemical properties (e.g. hidrophobicity, hidrophily, polarity) a key factor that determines whether two proteins interact is that proteins have shapes that allow them to spatially fit in the contact surface areas (active regions) .

This wide variety of problems demand for several distinct computational techniques, ranging from data mining and machine learning to explore similarities between the problem under study and problems already solved (e.g. homology studies), or a variety of simulation techniques to study, with some adequate level of abstraction, the dynamic behaviour of biochemical reaction networks [1]. Among these techniques, we have been applying, with considerable success, constraint programming to structural problems, namely the determination of protein structure and the study of protein interaction ("docking").

Constraint Programming is usually applied to constraint satisfaction problems (CSPs) by providing a declarative and efficient way of modelling such problems, through the specification not only of a set of variables and respective domains (often finite), but also the constraints that restrict the values that the variables can take (some variants to classical CSPs also include the addition of an optimisation function, or a partial satisfaction of the constraints). Non trivial problems are NP-complete or harder, and so a significant amount of search is required to solve instances of even of moderate size.

Several techniques are exploited in constraint programming to make search more efficient. In addition to the inevitable use of heuristics, search is interleaved with some kind of "reasoning", in that whenever a choice is performed, the consequences of such choice are analysed to make the subsequent search more informed and effective.

Some problems (typically large optimisation problems) are adequately addressed with local search techniques, where search is driven by the optimisation of some objective function (even for constraint satisfaction problems where that function may simply be a measure of the constraints violation, that is to be minimised). In this case, reasoning is largely related with the study of the neighbourhoods of current hypotheses, to allow better heuristics in the search for the optima of the objective (e.g. COMET[2]).

In complete methods, relying on backtrack search, reasoning is rather performed through various types of "constraint propagation". Such propagation, aims at decreasing the number of possible values for the variables not yet assigned specific values, and hence decrease the search space. Some basic techniques exist to deal with binary constraints. In particular, maintaining arc-consistency (usually with an AC-3 type algorithm [3]) enforces that values in a variable that share a constraint with another variable have support in the values of the latter variable, otherwise they can be discarded.

Moreover, constraint programming languages are also quite flexible, in that they allow a declarative specification of even quite complex constraints, in contrast with other declarative approaches to constraint solving (e.g. SAT or integer programming) where all constraints have to be converted into a rigid and expressively poor language (e.g. clauses and linear constraints). Constraint programming languages allow an easy combination of

primitive constraints, into more complex ones, or even the specification of primitive global constraints, for which specific and efficient propagation methods exist that are seamlessly integrated into the general constraint propagation mechanisms [4]. The classical example is the global constraint all_different, that constrains a set of k variables to take distinct values and for which some graph-based algorithms propagate more efficiently than would the set with k(k-1)/2 difference constraints over all pairs of variables [5].

Despite the expressive power of constraint programming languages and systems, and the efficient search techniques provided, there is a non-negligible amount of ingenuity that is required to obtain the most adequate models to the problem in hand. This is particularly so in non-conventional problems, which might be usefully addressed by constraint programming only if an appropriate model is adopted, and if adequate and advanced constraint programming techniques are applied in the solving process.

In this paper we address the way in which, in our tool, CHEMERA (available in the Web [6]) two structural bioinformatics problems are modelled and solved by constraint programming techniques. Their models adopted unconventional domains, and their solving does apply some main features of CP, namely constraint propagation, improved propagation by means of global constraints and some complementarity between complete backtrack search methods and incomplete local search methods.

Section 2 introduces the main features of PSICO, the component that addresses the problem of structure determination of proteins, taking into account Nuclear Magnetic Resonance (NMR) induced constraints. Section 3 reports on BiGGER, and its use of constraint programming techniques to handle protein interaction, often referred to as protein docking. Section 4 discusses a number of ideas aimed at improving the current algorithms, specially for the problem of structure determination. The paper ends with a section on concluding remarks.

2. PSICO: modelling protein structure.

Constraint Programming has been used in protein structure prediction in two distinct approaches. In the first, the problem is addressed *ab initio*: all that is known is the primary structure of a protein (i.e. the sequence of its amino acid residues). In this case, the models assume that these residues are placed in a three dimensional lattice (e.g. cubic or more complex face-centred cubic lattices, as in [7]) and a solution should minimize some energy function. In practise, such function takes into account either the hydrophobic (H) or the hydrophilic, also known as polar (P), nature of the amino acids, and aims at maximizing the number of H-H contacts in the centre of the protein. Alternative models allow more sophisticated energy functions to minimise, and take into account secondary structures (alpha helices or beta-sheets) that might be known, or at least suspected from homology studies [8].

Although this H-P model is quite interesting from a computational viewpoint, it leads to significantly distorted solutions, since the dihedral angles that are possible (e.g. 90º in cubic lattices) are not those that are chemically admissible. Moreover, these models do not take into account the availability of biochemical data that should be used not only to test, but also to drive the search for a solution, in the spirit of constraint programming.

Hence, we adopted an alternative model for structure prediction that takes into account as much information as possible, including available experimental data. There are several sources of information that can help model the structure of a protein. First of all, the amino acid sequences of the protein chains determines most chemical bonds, restricting 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.

We can divide this information 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 adopt for protein structure determination is composed of two phases: firstly it adopts a backtrack search where enumeration of variables is interleaved with constraint propagation until an approximate solution is found. Secondly, this structure is improved by means of a local search optimisation.

2.1 Variable Domains and Propagation of Distance Constraints

The chemical information that is known from the protein sequence provides bond length and bond angle constraints. Bond length constraints are also distance constraints, and the bond angles can be modelled by sets of distance constraints. In fact, the structure and flexibility of an amino acid can be modelled by a conjunction of pair wise distance constraints between all the atoms. To model this information we consider two types of constraints: In constraints (eq. 1) and Out constraints (eq. 2).

$$\textit{In}\ \text{constraint} \quad \max(|x_1 - x_2|, |y_1 - y_2|, |z_1 - z_2|) \le k \tag{1}$$

$$\textit{Out}\ \text{constraint} \quad |x_1 - x_2| \ge \alpha k \vee |y_1 - y_2| \ge \alpha k \vee |z_1 - z_2| \ge \alpha k \qquad \alpha = 1/\sqrt{3} \tag{2}$$

These two constraint types are used to model all the chemical structural information, whether it is known beforehand or from the NMR spectroscopy experiments. Notice that rather then considering Euclidean distances and spherical regions, we use an approximation that considers cuboid regions which are much easier to propagate, as discussed below.

The variables we wish to determine are the positions of the geometric centres of the atoms, that is, the (x, y, z) coordinates in a single variable with a three dimensional domain, and this domain is represented as a set of cuboid regions. One cuboid defines the Good region, which is the volume that contains the possible positions for the atom. A set of non-overlapping cuboids contained in the Good region defines the NoGoods region, which contains the positions from which the atom must be excluded (see Figure 1).

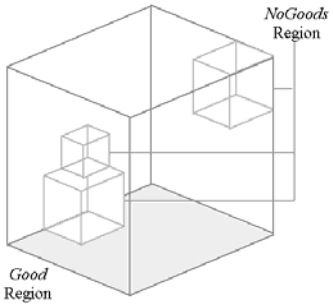

Fig. 1. The domain for the position of an atom is composed of two regions. The *Good* region is a cuboid that defines the positions for the atom that comply with the set of *In* constraints. The *NoGoods* region is a set of non-overlapping cuboids that define the volumes within the *Good* region from which the atom is excluded by the *Out* constraints.

We distinguished between the two types of distance constraints (In and Out) because of the way in which they are propagated (see Figure 2).

- The ***In*** *constraints* are propagated by simple intersection. The Good region of atom A will be the intersection of the current Good region of A with the neighbourhood of the Good region of atom B, defined as the Good region of B augmented by the distance value of the In constraint between A and B. The intersection of two cuboid blocks is very simple to calculate, requiring only Max and Min operations on the extremity coordinates, so propagation of In constraints is very efficient.
- For an ***Out*** *constraint* the propagation involves adding the exclusion region defined by the constraint to the NoGoods region of the affected atom. The most complex operation in this process is insuring that the NoGoods region consists of non-overlapping cuboids. This reduces propagation efficiency, but simplifies the task of determining the cases of failure when the NoGoods region becomes identical to the Good region.

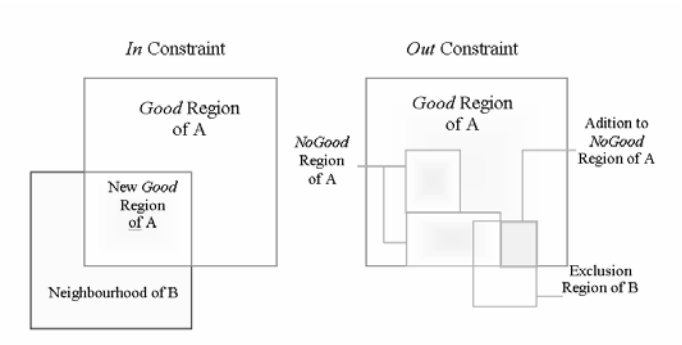

Fig. 2. This figure shows the propagation of both types of constraints. For *In* constraint propagation, the domain of atom A is reduced by intersecting the *Good* region of A with the neighbourhood of B. For *Out* constraint propagation a *NoGood* cuboid region is added, by intersecting the *Good* region of A with the exclusion region of B.

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

Enumeration interleaves the arc-consistency enforcement following a first fail approach on a round robin system. First, the atom with the smallest domain that was not selected in the current enumeration round is selected for enumeration. Exception is made if the coordinate domain is smaller than 2.0Å for all three coordinates, in which case the atom is considered sufficiently determined and no domain reduction is necessary. The domain of this atom is then split into two similarly sized domains by 'cutting' across the longest coordinate axis (x, y or z) of the domain. The domain of the atom will be one of these two 'halves'.

Enumeration heuristics now come into play. One simple heuristic that was shown to be successful [9,10] was to choose for the new domain the half cuboid that is less occupied by the domains of all other atoms, but additional considerations such as the chemical nature of the amino acid or the prediction of local structures can play a role at this stage to inform the choice of which regions of the domain to eliminate.

Since the domain for the enumerated atom is reduced, constraints are then propagated (as discussed above), and then another atom is selected for enumeration (the atom with the smallest domain not selected yet). 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 it is possible to backtrack and try different domain reductions, but backtracking is limited both for practical reasons and because it is often the case that the set of constraints is inconsistent due to experimental noise, and in these cases the user needs some structure, even if only partially correct, to help correct the inconsistencies by reassigning the constraints.

2.2 Propagation of Global Rigid Group Constraints

The last section outlined the basic framework for PSICO: the domain representations, arc-consistency interleaved with a round-robin enumeration, and limited backtracking. The propagation of rigid group constraints extends this framework to include the information on the configuration of groups of atoms. These can be prosthetic groups, secondary structures like alpha-helices, or more complex domains obtained by homology modelling, for which we can know the relative positions of all atoms but which fits within the structure of the protein in an unknown position and orientation.

Since rigid groups include many atoms that may only move together, reasoning globally with all these atoms, i.e. maintaining generalised arc-consistency, achieves better propagation than simply considering, one at a time, distance constraints between all the pairs of the atoms in the group (simple arc-consistency). This improvement is typical of reasoning with global constraints in constraint programming settings, and close to all-different reasoning [5], although with different domains. This section briefly outlines how generalised arc consistency is achieved with global rigid-group constraints.

Given a fixed orientation, it is trivial to reduce the domains of the atoms in a rigid group. This requires simply that we determine the limits for the translations of the group that do not place any atom outside its domain. Denoting by w_c one of the coordinates of the centre 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 centre c), such limits are related by the following equations (note that 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) :

$$w_{\max c} = Min_{j=1}^{n}(w_{\max j} + (w_c - w_j)) \qquad (3a)$$

$$w_{\min c} = Max_{j=1}^{n}(w_{\min j} + (w_c - w_j)) \qquad (3b)$$

Equations 3 assume 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, centred on the centre point of the group.

To determine the limits for the placement of the group as a function of the rotation around one axis, considering the rotation around the other axes fixed, we need but intersect the contributions of all atoms to these limits (see details in [11]).

Now we need to extend this to rotations around all three axes. Dividing the rotations into finite intervals, each orientation corresponds to an interval of angles, instead of just a single angle, and each coordinate to an interval of values. This way each rotation can be divided into a manageable number of orientations. Nevertheless, whereas rotating coordinates around an angular value gives a single values for the coordinates, rotating around an interval of angles results in intervals of coordinates. However, as long as certain conditions are met (more specifically, that the intervals for the angles partition the rotation with a step size that is a sub-multiple of 90º) then the intervals of the corresponding coordinates are trivial to calculate (more details in [11]).

2.3 More on Global Constraints – Propagation of Torsion Angle Constraints

In some cases, it is possible for a molecule to change 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. Some experimental techniques may provide constraints on torsion angles, and this is useful information when modelling a protein structure.

The propagation of these constraints is 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.

This procedure allows us to extend the rigid group constraint propagation to any number of rigid groups connected by torsion angles. There is a trade off between total group size and number of torsion angles to use, and the right trade off is also a function of the constraints on the torsion angles and the size of the atom domains at the time of propagation, so currently we are researching the best ways to optimise torsion angle constraint propagation taking into account all these factors.

2.4 Optimisation

Once enumeration terminates, each atom has a small cuboid domain, and a more exact position of the atom is obtained through an optimisation procedure, due to two main reasons. Firstly, enforcing smaller cuboids often leads to an exponential number of backtracks, unless a good heuristics is used. Secondly, if the geometric centre of the cuboids is considered the value for the atoms, then the resulting molecular structure does not respect the distance and angle values for the chemical bonds.

To address these problems, the Constraint Propagation method described so far, is complemented with a local search component that implements a simple torsion angle optimisation algorithm. Modelling the protein as a tree of rigid atom groups connected by rotatable bonds insures that the fine scale structure of the molecule is respected.

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 (details can be found in [12]).

3. BiGGER: the docking algorithm.

Another structural bioinformatics application where we have successfully applied constraint programming techniques is protein interaction (docking). 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 [13, 14, 15] or adapted [16] 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 [17].

Again, not only simple propagation of constraints is obtained by maintaining arc-consistency, but also 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.

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 using a regular cubic lattice of cells, similar to that commonly used in the Fast Fourier Transform (FFT) methods derived from [18]. 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. Figure 3 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.

This representation has several advantages over the FFT approach, requiring about a thousand times less memory (approximately 15Mb in BiGGER vs. 8Gb for FFT in large proteins) and being up to ten times faster than FFT [17]. BiGGER also models side-chain flexibility implicitly by adjusting the core grid representation [13] 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 to simultaneously improve the results and speed up the calculations.

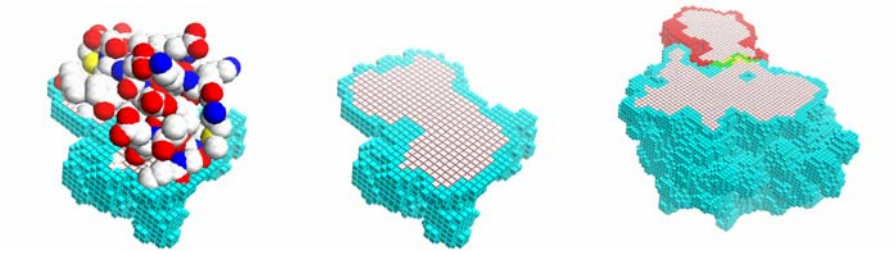

Fig. 3. The image on the left shows a protein structure overlaid on a cutaway of the respective grid, with spheres representing the atoms of the protein. The centre figure shows only the grid generated for this protein, cut to show the surface in light blue and the core region in grey. The rightmost image shows two grids (red and blue) in contact.

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. This is achieved by encoding the grids in a convenient way: instead of individual cells, grids are composed of lists of intervals specifying the segments of similar cells along the X coordinate. These lists are arranged in a two-dimensional array on the Y-Z plane.

This encoding not only reduces the memory requirements for storing the grids, but also leads naturally to searching along the X axis by comparing segments instead of by running through all the possible displacements along this coordinate. Given two surface segments, one from each structure and aligned in the same Y and Z coordinates, we can calculate the displacements where overlap will occur simply from the X coordinates of the extremities of the segments.

Representing by variable X, with domain Dx, the displacement of one structure relative to the other along the X direction, this approach of comparing segments efficiently enforces the constraint requiring surface overlaps, by reducing the domain of this variable to only those values where the constraint is verified. To begin with, each such variable is initialised to include all translations that may result in contacts by a bounds consistency check: if MaxA/MaxB and MinA/MinB are the maximum/minimum coordinate values along the X axis for the surface grid cells of the two structures, the domain of X is initialised to the interval [MinA-MaxB , MaxB-MinA]. This approach can be generalized for the translational search in the other 2 directions Y and Z.

3.2 Eliminating regions of core overlap

Another 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. By identifying the configurations where such overlaps occur, it is possible to eliminate from consideration those surface segments on each structure that cannot overlap surface segments on the other structure without violating the core overlap constraint. Some surface segments can thus be discarded from each search along the X axis. Figure 4 illustrates this procedure.

One structure, labelled A, is shown in the centre of the image. The other structure, labelled B, will be moved along the horizontal direction to scan all possible configurations but, from the overlap of core segments, a set of positions along the horizontal direction can be eliminated. Structure B is shown in position 1 to the right of A and in position 39 to the left of A. Clearly, in this case, B cannot occupy some positions in the centre.

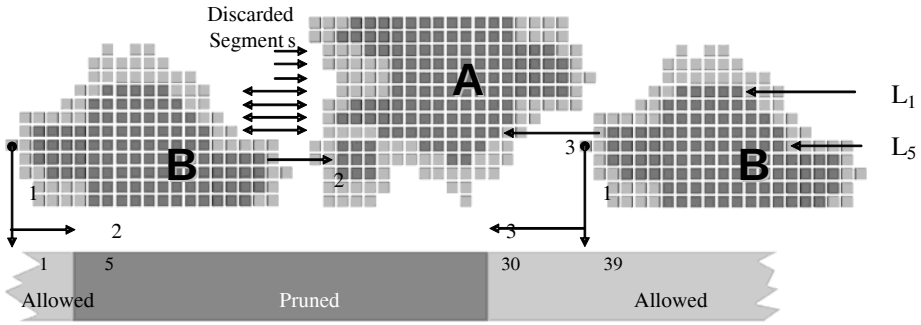

Fig. 4. Grid B is translated along the horizontal direction relative to grid A. The vertical arrows marked 1 indicate the position of B on the lower horizontal bar, which shows the allowed and forbidden values for the position of B. The arrows marked 2 and 3 show the allowed displacement of B. The group of horizontal arrows indicates segments to be discarded.

In particular, the domain of variable X, representing the displacement of one structure relative to the other along the X direction, can be pruned from the values 5 to 30. This is a contiguous interval in this example, but the domain of X can be an arbitrary set of intervals in the general case. This domain reduction due to the core overlap constraint propagates to the surface overlap, since some surface segments of A and B will not overlap in valid configurations. Some of these are shown in Figure 2 by the group of arrows to the left of structure A (Discarded Segments). For the last double arrow, for example, the surface cells of structures A and B would only overlap for X=7, a value pruned from the domain of X. In contrast, in the line below such overlap occurs for X = 3, a value kept in the domain. Thus the core overlap constraint allows us to reduce the number of surface segments to consider when counting surface overlaps.

The BiGGER algorithm imposes bounds consistency on these sets of core grids segments, which requires $O(k^2)$ operations, where k is the number of intervals defined by the core grid segments for each line and for each structure. This reduces the possible translation values, Dx, and affects the generation of the surface segments lists to take into

account Dx, including only those segments that could overlap given this domain (again, by imposing bounds consistency on the intervals). Finally, the overlap of surface cells is determined for each allowed translation value in Dx. This requires testing the bounds of the matching surface segments in a way similar to imposing bounds consistency, which is of $O(k^2)$ for each line, and then counting the contacts along X, which is of O(N).

The algorithm performs $O(N^2)$ steps by looping through the Dz and Dy, and in each of these steps it loops through the Z,Y plane twice to find the matching core and surface segments and compare the segment bounds. So each step in the z, y loop is $O(N^2k^2)$, where k is the number of segments per line. Except for fractal structures, k is a small constant. For convex shapes, for example, k is always two or less, and even for complex shapes like proteins k is seldom larger than two. Thus the time complexity of the search algorithm when imposing bounds constraints on the overlap of surface and core grid cells is $O(N^4)$, very close to the $O(N^3Log(N))$ of the FFT method. Furthermore, the comparisons done in the BiGGER algorithm are much faster and this constant factor makes BiGGER more efficient for values of N up to several hundred [17]. Finally, 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

Branch and Bound is a common technique that Constraint Programming often uses in optimisation problems, to restrict the domains of the variables to where it is still possible to obtain a better value for the function to optimise. In this case, we wish to optimise the overlap of surface cells, and restrict the search to those regions where this overlap can be higher than that of the lowest ranking model to be kept.

This constraint is applied to the Z and Y coordinate search loops, by counting the total surface cells for each grid as a function of the Z coordinate (that is, the sum over each X, Y plane) and as a function of each Y, Z pair (that is, the sum of each line in the X axis). The determination of the Z translation domain considers the list of total surface cells for each X,Y plane along the Z axis. For each Z translation value these two lists will align in a different way, as the one structure is displaced in the Z direction relative to the other. The minimum of each pair of aligned values gives the maximum possible surface overlap for that X,Y plane at this Z translation, and the sum of these minima gives the maximum possible surface overlap for this Z translation. Since there are O(N) possible Z translations to test and, for each, O(N) values to compare and add, this step requires $O(N^2)$ operations.

The same applies to restricting the Y translation domain, but taking into account the current value of variable Z. This is also an $O(N^2)$ operation identical to the pruning of the Z domain, but must be repeated for each value of the z translation variable, adding a total time complexity of $O(N^3)$ to the algorithm. Since the BiGGER algorithm has a time complexity of $O(N^4)$, these operations do not result in a significant efficiency loss.

By setting a minimum value for the surface contact count, or by setting a fixed number of best models to retain, this constraint allows the algorithm to prune the search space so

as to consider only regions where it is possible to find matches good enough to include in the set of models to retain. In general, this pruning results in a modest efficiency gain of up to 30% in medium-sized grids, but with decreasing returns as higher grid sizes lead to thinner surface regions and shift the balance between the total surface counts and the size of the grid [17]. However, this can benefit some applications like soft docking [13], 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 (4)

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 any combination 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 (4) by projecting it in each of the three directions:

At least K atoms of set A must be within R_ω of at least one atom of set B (**5**)

where R_ω replaces the Euclidean distance R and represents the modulus of coordinate differences on one axis Z, Y or X. R_ω 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 is to determine the neighbourhood of radius R of atoms in group B, projected on the coordinate axis being considered. The next step is to generate a list of segments representing the displacements for which at least K atoms of group A are inside the segments defining the neighbourhood R of the atoms in group B.

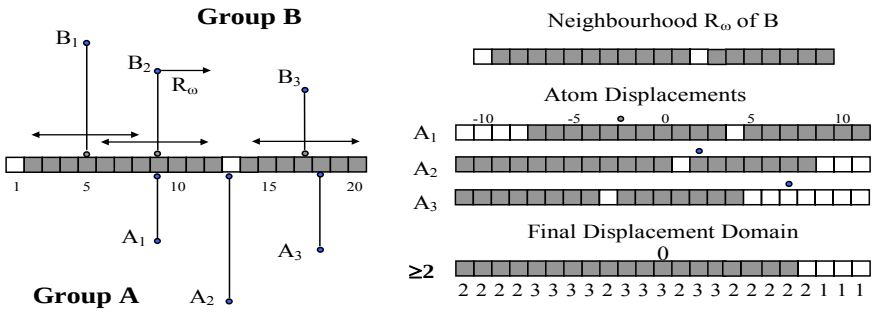

Fig. 5. Generating the displacement domain in one dimension. The left panel shows the generation of the neighbourhood of radius R of group B. The panel on the right shows the allowed displacements for each atom, and the final displacement domain for a K value of 2.

The calculation of the neighbourhood of B in some coordinate (either X, Y or Z) is illustrated in Figure 5. The positions of atoms B1, B2 and B3 in this coordinate are respectively 5, 9 and 17. Their neighbourhoods within a distance 3 are (2;8), (6;12) and (14;20). Merging the two first intervals, the neighbourhood 3 of the atom set B is thus (2;12) and (14;20).

To calculate the displacement values that place an 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, atom A1, with coordinate 9, lies inside the neighbourhood 3 of B if its displacement lies in the range (-7;3) or (5;11). Similarly, atoms A2 and A3, with coordinate values 13 and 18, respectively may be displaced by (-11;-1) or (1;7) and (-16;-6) or (-4;2).

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 (5) 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 (-11;3) and (5;7). In ranges (-7;-6) and (-4;-1) all 3 A atoms are in the neighbourhood 3 of B.

The propagation of constraints of type (5) 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 [17].

4. Results and Further Work

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 [13, 14, 19, 20, 21, 22, 23, 24, 25].

As to PSICO, it still is under development, namely to integrate the propagation of global constraints in the general algorithm. 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 10 minutes, 1 minute for the CP phase, and 9 minutes for various runs of the optimisation phase to produce 15 distinct solutions.

This is significantly faster than the reference system currently used in this area (DYANA [26]) that uses a simulated approach to the problem and took 10 hours to solve the problem. Nevertheless, the accuracy achieved with DYANA is significantly better, 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 to assist biochemists in the interpretation of NMR. In fact, in earlier phases, distances are not assigned to the correct atom pairs, and so a fast, if only approximate, interpretation is quite useful to alert biochemists that some of the distance constraints should be revised.

Although the integration of global rigid body constraints has not been done yet, we expect that PSICO 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 [11]. 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. The tuning of this propagation will be possible with Casper, a constraint propagation system that we started developing recently and that will be tested soon with PSICO problems [27].

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 [28].

Regardless of the rigid body global constraints, PSICO should perform better if better enumeration heuristics, namely value choice heuristics were used. The heuristics that is still used, choose the half domain less occupied by the domains of all other atoms, does

not take into consideration any biochemical properties of the amino acids. If these are taken into consideration, an initial 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% [29]. 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 [30]. 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 [31]). 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.

5. Conclusions

Constraint Programming is a computational paradigm quite adequate to address combinatorial problems given, 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, many problems are adequately addressed with Constraint Programming only if adequate models are used, which might require some degree of ingenuity from the users.

In this paper we have shown that structural bioinformatics problems can 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) which very likely will be necessary to handle the difficult problems arising in this domain.

Although in the heart of the algorithms being used, constraint programming is likely to be complemented with other advanced techniques, namely data mining on various databanks, increasingly available in the Web, for the development of complete practical applications. This has been shown in the applications described in this paper, 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. **N**. Chabrier and F. Fages. The biochemical abstract machine BIOCHAM. In C. Christophe, H.P. Lenhof, and M. F. Sagot, editors, Proceedings of the European Conference on Computational Biology, ECCB'03, pages 597-599, Paris, France. System available at http://contraintes.inria.fr/BIOCHAM, September 2003.
2. Laurent Michel, Pascal Van Hentenryck: Parallel Local Search in Comet, CP'2005 (Procs.), Peter van Beek (Ed.), Lecture Notes in Computer Science, vol. 3709, Springer, pp. 430-444, October, 2005.
3. A.K. Mackworth and E.C. Freuder, The complexity of some polynomial network consistency algorithms for constraint satisfaction problems, Artificial Intelligence, 25(1):65–73, 1985,
4. N. Beldiceanu, Global Constraint Catalog, http://www.emn.fr/x-info/sdemasse/gccat/.
5. J.-C. Régin, A Filtering Algorithm for Constraints of Difference in CSPs, Proceedings of AAAI-94, pp.362-367, 1994.
6. http://www.cqfb.fct.unl.pt/bioin/chemera/.
7. Backhofen 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
8. Dovier A., Burato M. and Fogolari F., Using Secondary Structure Information for Protein Folding in CLP(FD), In Procs. Workshop on Functional and Constraint Logic Programming, ENTCS, Vol 76, 2002
9. Krippahl, L., Barahona, P., PSICO: Solving Protein Structures with Constraint Programming and Optimisation, Constraints 2002, 7, 317-331
10. Krippahl, L., Barahona, P., Applying Constraint Programming to Protein Structure Determination, Principles and Practice of Constraint Programming, Springer, 1999 289-302
11. 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.
12. Krippahl L, Barahona P. PSICO: Solving Protein Structures with Constraint Programming and Optimisation, Constraints 2002, 7, 317-331
13. 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.
14. Krippahl L, Moura JJ, Palma PN. 2003. Modeling protein complexes with BiGGER. Proteins: Structure, Function, and Genetics. V. 52(1):19-23.
15. 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.

16. 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
17. 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.
18. 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.
19. 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.
20. 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.
21. 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
22. 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.
23. 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.
24. 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.
25. Impagliazzo A, Krippahl L and Ubbink M. Pseudoazurin : Nitrite Reductase Interactions (2005) ChemBioChem 6, 1648-1653
26. 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.
27. M. Correia, P. Barahona and F. Azevedo, 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.
28. 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.
29. J.C. Almeida Santos, Mining Protein Structure Data, M.Sc. Thesis, New University of Lisbon, 2006
30. Correia M. and 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
31. W. Harvey and M. Ginsberg, Limited Discrepancy search, in Proceedings of IJCAI, International Joint Conference on Artificial Intelligence, C. Mellish (ed.), Montreal, 1995.