1 D2.1 - WP2 (LIFL) Optimization algorithms designed

ALEXANDRU-ADRIAN TANTAR,
NOUREDINE MELAB, EL-GHAZALI TALBI

Laboratoire d'Informatique Fondamentale de Lille,
LIFL/CNRS UMR 8022, DOLPHIN Project - INRIA Futurs.

1.1 INTRODUCTION

With the evolution of distributed high-performance and high-throughput computing and with the support of nuclear magnetic resonance (NMR) data, we are at the dawns of a new era in molecular research and pharmaceutical drug design. Having been studied for more than a decade and of particular interest, protein-protein[1] docking is fundamental in understanding biomolecular processes, interactions between antibodies and antigens, intra-cellular signaling modulation mechanisms, inhibitor design, macromolecular interactions and assemblies, etc. In conceptual terms, the molecular docking describes the complexed macromolecule resulting from the binding of two separate folded molecules, exerting geometrical and chemical complementarity. From a computational standpoint, *in silico* docking simulates molecular recognition, although not relating to the molecular pathways of the process but to the final complexed result.

The docking problem combines three interrelated principal algorithmic components: a model for representing the molecular complexes, algorithmic mechanisms for performing conformational space search, and a scoring modulus for evaluating the potential solutions. The binding energy landscape has a funnel-like shape, thus

[1]proteins were discovered in 1838 by Jöns Jakob Berzelius, a Swedish chemist, the term being derived from the Greek *protas*, standing for "of primary importance".

the search for an optimal conformation aims for a global-minimum energy configuration.

The existing approaches may be differentiated in *bound* and *unbound* docking. Bound docking initiates having the receptor and the ligand molecules in a binded conformation, the computational approach being an attempt of reconstructing the originating complex. In this case the component parts of the process are obtained from an initial X-ray/NMR crystal structure enclosing both the ligand and the receptor, in a bound form. At a higher level of complexity, predictive computational schemes involve unbound structures for the ligand and the receptor which, in this case, may originate from crystallographic native-structure data, *ab-initio* calculations, etc.

The importance of the protein-protein docking problem is reinforced by the ubiquitousness of proteins in the living organisms, applications of computational molecular docking directing to computer assisted drug design and computer assisted molecular design. From a structural point of view, proteins are complex organic compounds composed of amino-acid residue chains joined by peptide bonds - refer to Fig. 1.1 for a schematic example. Proteins are involved in immune response mechanisms, enzymatic activity, signal transduction, etc.

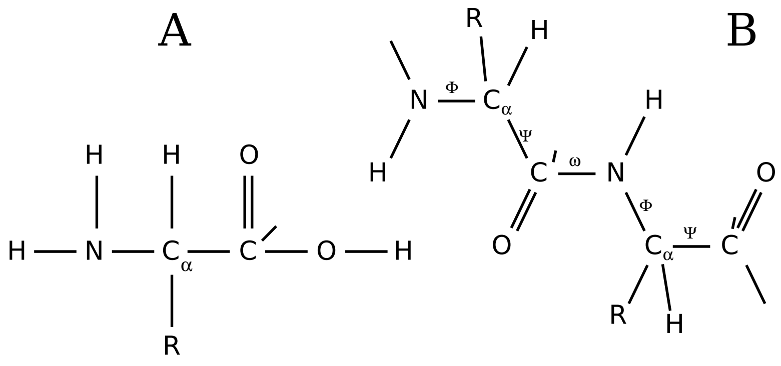


Fig. 1.1 Amino-acid: A - $NC_\alpha C$ back-bone structure; B - polymeric structure; ω, Φ and Ψ relate to dihedral angles; R designates the specific amino acid's *side chain* characteristic.

Due to the intrinsic relation between the structure of a molecule and its functionality, the problem implies important consequences in medicine and biology related fields. An extended referential resource for protein structural data may be accessed through the Brookhaven Protein Data Bank[2] [Bernstein et al., 1977]. For a comprehensive introductory article on the structure of proteins, consult [Neumaier, 1997]. Also, for a glossary of terms, see [H. Van de Waterbeemd, 1997].

[2]http://www.rcsb.org - Brookhaven Protein Data Bank; offers geometrical structural data for a large number of proteins.

1.1.1 Complexity Considerations

The accuracy of the computational docking is inherently determined by the flexibility level used in modeling the ligand and the receptor molecules. Simulating flexibility is computationally expensive, hence a bias is mandatory at the price of the accuracy of the results. A classification of the existing approaches by considering the flexibility as a criterion conducts to a classification consisting of three classes of docking:

- rigid docking - an extreme simplification of the docking process by considering both the ligand and the receptor as rigid entities, no flexibility being allowed at any point, nor the ligand, nor the binding site in the receptor;
- partially flexible docking - to some extent flexibility is modeled in the process by focusing on the smaller molecule - typically the ligand - or by defining comprehensive regions of significance;
- flexible docking - both the ligand and the receptor are modeled as flexible molecules, although, to some extent, limitations may be imposed in order to reduce the computational complexity. An example of molecular docking of a highly flexible ligand to a highly flexible macromolecule is described in [Teodoro et al., 2001].

The docking problem is computationally difficult, as the number of possible binding combinations exponentially increases in correlation with the magnitude of the involved molecular complexes. The corresponding degrees of freedom are determined by the three translational and the three rotational axes. As a consequence, ligand and binding-site flexibility is computationally expensive to model, increasing the number of possible conformations (resulting as a power of the number of rotatable bonds). As an example, [Lorber and Shoichet, 1998] note that for a molecular complex with ten rotatable bonds and with six minima allowed per bond, the number of possible conformations rises at 3.48×10^9. As a direct implication, no extensive conformational space exploration is possible, unless *a priori* information is provided. Different approaches may be considered for attaining complexity reduction, as by assuming and employing knowledge of the binding site as well as by performing rigid-body docking. The main drawbacks incurred in such approaches are the fact that there might exist multiple potential binding sites and that extensive simplification may lead to inaccurate models and predictions of the native molecular complexes.

1.2 INTRODUCTION TO LARGE-SCALE GRID COMPUTING

The proliferation of research and industrial projects on grid computing is leading to the proposition of several, sometimes confusing, definitions of the grid concept. As a consequence, a number of articles [Foster et al., 2001], [Krauter et al., 2002] especially address an attentive analysis of these definitions. A computational grid is a scalable pool of heterogeneous dynamic resources, geographically distributed across multiple administrative domains and owned by different organizations. Discussing the afferent defining characteristics, a summarizing description might be formulated as follows:

- *the grid includes multiple autonomous administrative domains* - the users and providers of resources are clearly identified. This allows to reduce the complexity of the security issue, however, the firewall traversal remains a critical problem to deal with. In global computing middlewares based on large scale cycle stealing, such as XtremWeb [Germain et al., 2000], the problem is solved in a natural way as communications are initiated within the boundaries of the domains, ”inside the domains”.

- *the grid is heterogeneous* - the heterogeneity in a grid is intensified by its large number of resources belonging to different administrative domains. The emergence of data exchange standards and platform independent technologies such as Java RMI allows to deal with the heterogeneity issue.

- *the grid has a large scale* - the grid has a large number of resources growing from hundreds of integrated resources to millions of PCs. The design of performant and scalable grid applications has to take into account the communication delays.

- *the grid id dynamic* - the dynamic temporal and spatial availability of resources is not an exception but a rule in a grid. Due to the large scale nature of the grid, the probability of a number of resources failing is high. Such characteristic highlights issues such as dynamic resource discovery, fault tolerance, and so on.

Furthermore, on an algorithmic scale, computational performance is achieved through hybrid parallel cooperative meta-heuristics - the gridification of such an approach requires taking into account at the same time the characteristics and underlined issues of the computational grids and the parallel cooperative models. Some of the issues related to grids may be solved by middlewares allowing to hide their inherent complexity to users. The number of issues that could be solved in a transparent way for the users depends on the middleware at hand. The choice of this later is crucial for performance and ease of use. Maintaining a logical communication topology in a volatile environment may be complex and inefficient due to the hight cost of the dynamic reconfiguration of the topology. For example, for an island-based model, one of the approaches allowing to deal with such issue is based on a shared space for storing the emigrant solutions between the islands. The island that initiates a migration operation sends the emigrants to the shared space, and these later are stored together with the identity of their source islands. Islands can also initiate immigration operations by sending requests to the shared space, and immigrants are randomly chosen from this later. In [Belding, 1995], it has been experimentally proven that random topologies (random selection of the target islands) could be as efficient as the common topologies (ring, mesh, etc.). Grid middlewares that support such approaches are Dispatch-Worker ones such as XtremWeb. In such systems, clients can submit their jobs to the dispatcher. A computational pool of volatile workers request the jobs from the dispatcher according to the large cycle stealing model. Then, they execute the jobs and return back the results to the dispatcher to be collected later by the clients. The islands could be deployed as workers and the dispatcher could serve to provide the global space.

One of the major limitations of such middlewares is that they are well-suited for embarrassingly parallel (*e.g.* multi-parameter) applications with independent tasks. In this case, no communication is required between the tasks, and thus workers. The deployment of parallel cooperative meta-heuristics that need cross-worker/task communication is not straightforward. The developer has the burden to manage and control the complex coordination between the workers. To deal with such problem existing middlewares must be extended with a software layer which implements a coordination model. Several examples of interesting coordination models may be found in the works of [Gelernter and Carriero, 1992] and [Papadopoulos and Arbab, 1998].

1.3 MOLECULAR DOCKING

1.3.1 Resolution Schemes for Molecular Docking

As no exploration method is applicable for performing an exhaustive conformational space search, due to computational complexity matters, approximation schemes are employed as resolution patterns for molecular docking. The related exploration algorithms, in basic form, can be classified in geometric complementarity methods, molecular dynamics based algorithms, Monte Carlo methods, evolutionary approaches, etc. In the followings, a few standard approaches are sketched, referential work being cited as starting point for further study - to no extent and by no means exhaustive as description, serving for no more than a basic outline.

1.3.1.1 Geometric Matching Molecular complementarity, on a geometrical and biochemical basis, may be computationally simulated by using geometric primitives for defining the molecular constituent entities, *i.e.* atoms, etc. Different geometrical attributes may be used at this point in order to define an optimal conformation as, for example, surface normals, molecular overlap, etc. Matching techniques utilize geometric hashing and indexing schemes, grid-based Fast Fourier Transformation (FFT) docking correlation, rigid three-dimensional geometric transformations (as affine and projective transformations), etc.

For a more in detail, geometry oriented, discussion of the aforementioned concepts, please refer to [Wolfson and Rigoutsos, 1997]. For an induced-fit, geometry based, docking exemplification, refer to [Sandak et al., 1998] - within their article, a computer vision inspired, hinge-bending docking approach is illustrated. As the authors remark, domain movements, at different structural levels, represent an essential element to consider when constructing a molecular docking resolution pattern. Conformational modifications were allowed for both the ligand and the receptor, thus providing an induced-fit docking procedure - although inside-domain conformational rigidity is maintained, molecular flexibility is simulated by allowing for domain movements.

The algorithm, as the authors note, has been tested on five complexes, using a flexible docking approach, allowing hinge-bending movements in the ligand molecules: (1) the HIV-1 protease with the U-75875 inhibitor, (2) the dihydrofolate reductase complexed with methotrexate and, (3) separately with NADPH, (4) lactate dehydrogenase complexed with NAD-lactate and, (5) a Fab fragment of an IgG antibody complexed with a peptide antigen. The obtained average RMS of the correct solutions is reported to be at 1.4Å with a one minute average execution time for each complex - the experiments were conducted on a SGI-Challenge R8000 machine.

1.3.1.2 Monte Carlo Techniques Monte Carlo methods, or, *in extenso*, importance sampling and Markov chain Monte Carlo, perform random conformational space exploration, discriminating the selected conformations by following a Boltzman probability distribution model. Monte Carlo methods are commonly employed as refinement components in multi-stage molecular docking, in correlation with terms for quantifying solvation and electrostatic effects, etc. A multi-stage molecular docking approach usually consists of (a) an initial, fast, conformational sampling phase, relying upon soft-docking techniques, geometric matching, etc, and (b) a final phase accounting for the refinement of the resulting conformations - achieved by performing accurate energy calculations, increased flexibility, etc.

In [Fernandez-Recio et al., 2002a] an approach including Monte Carlo algorithmic components is exposed, a more detailed study of the initial method being published in [Fernandez-Recio et al., 2002b]. The authors analyzed several protein-protein complexes, for all the reported cases, the obtained conformations falling within 3.0Å RMSD as compared to the crystal structure. As a first step, automated rigid-body Monte Carlo docking simulations are conducted on a non-redundant data set of protein-protein complexes, soft interaction energy potentials being pre-computed over a grid enclosing the potential active site. The sampling phase is carried by a pseudo-Brownian Monte Carlo algorithm, the Metropolis criterion being used as conformational filter. For the initial phase, a complete simulation cycle consists of 20000 energy evaluations, at the end of each random step, a local optimization being performed. The second phase, the refinement step, is focused on further optimizing the interface side-chains by using a Biased Probability Monte Carlo algorithm - the induced fit of the association is thus simulated, allowing for near native conformation solutions.

1.3.1.3 Evolutionary Algorithms Evolutionary algorithms are stochastic search iterative techniques, inspired from the Darwinian evolutionary theory, having a large area of appliance - epistatic, multi-modal, multicriterion and highly constrained problems [Cahon et al., 2005a]. Stochastic operators are applied for evolving an initial randomly generated population, in an iterative manner, *i.e.* in generations. Each of the individuals composing the population contains genotype information encoding its defining features - the phenotype. Each generation undergoes a selection process, the individuals being evaluated by employing a problem specific fitness function.

Algorithm 1.6.1. EA pseudo-code.

```
Generate($P(0)$);
$t$ := 0;
while not Termination_Criterion($P(t)$) do
    Evaluate($P(t)$) ;
    $P'(t)$      := Selection($P(t)$) ;
    $P'(t)$      := Apply_Reproduction_Ops($P'(t)$) ;
    $P(t+1)$   := Replace($P(t)$, $P'(t)$) ;
    $t$ := $t+1$;
endwhile
```

The pseudo-code above exposes the generic components of an EA. The main subclasses of EAs are the genetic algorithms, evolutionary programming, evolution strategies, etc.

Due to nontrivial addressed problems, requiring extensive processing time, different approaches were designed in order to reduce the computational costs. Complexity is also addressed by developing specialized operators or hybrid and parallel algorithms. We have to note that the parallel affinity of the EAs represents a feature determined by their intrinsic population-based nature. In [Cahon et al., 2004a] three main parallel models are identified - the island synchronous cooperative model, the parallel evaluation of the population and the distributed evaluation of a single solution.

Genetic algorithms (GAs) are population-based metaheuristics that allow a powerful exploration of the conformational space. However, they have limited search intensification capabilities, which are essential for neighborhood-based improvement (the neighborhood of a solution refers to part of the problem's landscape). Therefore, different approaches combine GAs with local search methods, in order to improve both the exploration and the intensification capabilities of the two techniques.

For a complete overview on parallel and grid specific metaheuristcs, please refer to [Cahon et al., 2005a], [Talbi, 2002], [Alba et al., 2005b], [Cahon et al., 2004a].

1.4 MOLECULAR DOCKING ON GRIDS

The main computational challenge of molecular docking consists in screening a large number of compounds against a protein target, in the search for a potential inhibitor to be further used in drug design. Considering a real-life approach, several problematic elements have to be overcame - screening chemical databases consisting of millions of compounds rises data transfer and storage difficulties as well as computational time problems. As specified in [Buyya et al., 2003], considering a few minutes to hours as the expense for screening a compound on a standard desktop computer, depending on structural complexity, results in years for screening the entire database. The authors extrapolate offering a more specific example for edifying on the computational complexity - for a screening involving 180000 compounds, and with a per-compound execution time expected to take about three hours on a desktop computer, the amount of required processing time raises at 540000 hours - more than 60 years. Moreover, as the authors indicate, a cluster-based supercomputer with 64 nodes might reduce the computation time to about one year, or, further, to be solved by a grid of hundreds of computers within a day.

As an outline for the molecular docking on grids, it may be stated that the process is largely biased towards the computational aspect, a reduced fraction of time being allocated for data transfer. Nevertheless, data transfer imposes as a problem as it is impractical to replicate an entire database of compounds over the nodes of a distributed environment. Alleviating the problem is possible by selective replication of the database over a reduced number of coordinating nodes, as well as by dynamic data distribution, inquiring remote databases as the computation advances.

Although extensively platform and grid-environment oriented and not offering relevant information regarding the details of the employed docking protocol, the work of [Buyya et al., 2003] represents a starting point in understanding the principles of a grid enabled approach. The docking code enclosed in the environment, DOCK, represents a research result of the University of California in San Francisco (UCSF) - a comprehensive insight of the algorithm is given in [Ewing and Kuntz, 1998].

The paper is concerned in offering the development details of a *Virtual Laboratory*, as the authors entitled the developed computing system, constructed on top of several existing Grid technologies.

Scheduling experimentations are reported to have been conducted on a grid resource in Australia (one Sun: Ultra-1 node) with the additional support of four resources in Japan (three sites regrouping twelve Sun: Ultra-4 nodes and two Sun: Ultra-2 nodes) and one in U.S.A. (eight Sun: Ultra-8 nodes). The experimentation consisted in a 200 molecules screening trial, on a endothelin converting enzyme (ECE) receptor.

Another impressive large scale docking on grids is reported in [N. Jacq, 2006], the article reporting on the effort of developing new drugs for fighting malaria - part of the WISDOM project (*World-wide In Silico Docking on Malaria*, a European initiative initially reuniting the Fraunhofer Institute for Algorithms and Scientific Computing, in Germany, and the Corpuscular Physics Laboratory, CNRS/IN2P3, in France). The goal of the project aimed in proposing new potential inhibitors for a family of proteins produced by *Plasmodium falciparium*, as drug resistance reduced the efficiency of classical pharmaceutics. The demonstration took place starting from the 11^{th} of July until the 19^{th} of August, 2005, enumerating over 46 million docked ligands and simultaneously gathering 1700 computers from 15 countries around the world. Another closely related report is offered in [Hurng-Chun Lee and Wu, 2006], an outcome of the first mentioned article, fighting against avian flu. Authors mention the work to be based on docking tools such as AutoDock. The performed experiments made use of more than 2000 CPUs, performing virtual screening over the extent of six weeks. Authors report to have screened eight protein targets against 308585 compounds, each target simulating a possible mutations of the avian flu virus. The estimated equivalent sequential computation time was estimated at more than 16 years, with an overall speed-up of 203, while having a distribution efficiency of 84%. The corroborated results for the two experiments are presented in Table 1.4, as exposed by the authors in the papers cited above:

Compl. dockings[a]	Duration	1CPU Exec.[b]	Speedup	Distr. Eff.[c]
$2 * 10^6$	6 weeks	88.3 years	767.37	38.4%
308585	30 days	16.7 years	203	84%

The first line of the table coresponds for the WISDOM experiment. [a]Total number of completed dockings. [b]Equivalent execution time on 1 CPU. [c]Distribution efficiency - defined as the approximation of the ratio between the overall speedup and the maximum number of concurently CPUs.

The experiment resulted in more than two millions of docking complexes, cumulating 123440 files with a total amount of 600 Gigabytes of data, stored for further study and in-depth biological analysis, to be performed in several research laboratories.

In the same line of ideas, in [Nakajima et al., 2004] a Grid-enabled conformational space approach is presented, CONFLEX-G, the exposed method being derived from CONFLEX, an sequential exhaustive conformational space search of low-energy regions. The implementation is based on a grid RPC system, called OmniRPC, a thread safe implementation of Ninf RPC. The original CONFLEX algorithm, as defined by the authors, consists of four distinct stages:

- an initial conformation is extracted from a previously discovered database of conformers;
- perturbations are applied on the initial selected structure, in order to generate trial structures;
- the previously generated trial structures are optimized on a geometrical basis;
- the optimized structures are compared with other conformers stored in a database, preserving the newly discovered structures.

Trial structures, as mentioned in the paper, are obtained by corner flapping and edge flipping for the ring atoms and stepwise rotation for side-chains or backbone-chains. Observing that the geometry optimization phase takes as much as 95% of the spent computational time, the method has been parallelized using a Master/Worker technique.

1.5 MOLECULAR DOCKING SOFTWARE

Considering the numerous techniques and applications developed in order to address the molecular docking problem, it is not possible to enumerate even an infinitesimal fraction of the existing software applications. As a consequence, in the followings, a resume is presented, enclosing a reduced number of the most representative developed applications. An ample review discussing the characteristics of different existing molecular docking software is presented in [R.D. Taylor and Essex, 2002], offering a categorized approach and discussing scoring function aspects and multiple method algorithms. The final part of the article offers bibliographic references regarding techniques comparison studies.

1.5.1 AutoDock

AutoDock[3], *Automated Docking of Flexible Ligands to Macromolecules*, is a collection of components with aims ranging from the interactive definition of the torsion angles to automatic docking. The software is distributed free of charge for academic and non-commercial use, a commercial version being also available. AutoDock represented the basis of several large-scale experiments on grids, as mentioned in the previous section, being also employed behind the *FightAIDS@Home*[4] project - reported to have cumulated over 2×10^{15} energy evaluations for HIV-1 protease candidate inhibitors. In addition, *AutoDockTools* or *ADT*, comes to complete the application by offering a visual interface. From an algorithmic point of view, AutoDock employs several evolutionary techniques, including also Monte Carlo simulated annealing and Lamarckian Genetic Algorithm. The AMBER-derived force field model is based on linear regression analysis, using empirical weighting factors determined from protein-ligand complexes for which the binding constants are known. A detailed description is available in [Morris et al., 1999].

1.5.2 DOCK

DOCK represents one of the pioneering programs in molecular docking, at its origins, approximating the process by considering both the ligand and the receptor as rigid molecules. The docking algorithm consists of a conformational search method and a scoring function, relying on graph theoretical techniques for superimposing ligand atoms onto predefined sets of points in the active site. The algorithm presumes on the chemical and geometrical complementarity of the molecules, hence inaccurately modeling docking processes involving conformational modifications. Furthermore, the algorithm is directed by considering *a priori* designated potential important regions, thus alleviating the computational complexity incurred by the conformational sampling process. Newer versions of DOCK offer the possibility of modeling the ligand in a flexible manner, as well as simulating protein flexibility by using multiple conformations. Details may be found in the work of [Ewing and Kuntz, 1998] which offers an insight over the graph techniques enclosed in the algorithm, presenting in the final part a number of experiment-extracted results.

[3]http://www.scripps.edu/mb/olson/doc/autodock
[4]http://fightaidsathome.scripps.edu

1.5.3 FlexX and FlexE

FlexX[5] is a fragment-based incremental construction algorithm, **FlexE** being derived from the aforementioned, essentially, it makes use of ensemble approach adapted concepts - scoring function, interaction scheme, incremental construction algorithm etc. The molecular docking process in FlexX considers multiple stages, initiating by selecting a base ligand fragment - determinant for the rest of the process; a geometry interaction database is used for the selection. Further, a geometric based alignment is performed, including hydrophobic interaction elements and geometric constraints - finally the ligand is built into the active site, in incremental fashion. A comparison test case of FlexX and FlexE is presented in [Holger Claußen and Lengauer, 2001], the experimentations being performed on ten protein structures ensembles - an overall of 105 crystal structures. FlexE is reported to find conformations with a 2.0Å RMSD for 67% of the cases, while, in the same context, FlexX amounts for 63%.

1.6 PARALLEL METAHEURISTICS FOR SOLVING THE PSP/DOCKING

1.6.1 Genetic Algorithm

Evolutionary algorithms are stochastic search iterative techniques, with a large area of appliance - epistatic, multi-modal, multicriterion and highly constrained problems [Alba et al., 2005a]. Stochastic operators are applied for evolving the initial randomly generated population, in an iterative manner. Each generation undergoes a selection process, the individuals being evaluated by employing a problem specific fitness function.

Algorithm 1.6.1. EA pseudo-code.

```
Generate($P(0)$);
$t$ := 0;
while not Termination_Criterion($P(t)$) do
    Evaluate($P(t)$);
    $P'(t)$      := Selection($P(t)$);
    $P'(t)$      := Apply_Reproduction_Ops($P'(t)$);
    $P(t+1)$   := Replace($P(t)$, $P'(t)$);
    $t$ := $t+1$;
endwhile
```

The pseudo-code in Alg. 1.6.1 exposes the generic components of an EA. The main subclasses of EAs are the genetic algorithms, evolutionary programming, evolution strategies, etc.

Genetic Algorithms (GAs) are Darwinian-evolution inspired, population-based metaheuristics that allow a powerful exploration of the conformational space. However, they have limited search intensification capabilities, which are essential for neighborhood-based improvement (the neighborhood of a solution refers to part of

[5]http://www.biosolveit.de/FlexX

the problem's landscape). A random population of individuals is evolved in generations through different strategies in order for convergence to be achieved. The *genotype* represents the raw encoding of individuals while the *phenotype* encloses the coded features. For each generation, individuals are selected on a fitness basis, genotype alteration being performed by means of crossover and mutation operators. Applying the genetic operators has as result the modification of the population's structure as to intensify exploration inside a delimited segment or for diversification purposes.

1.6.1.1 Encoding of the conformations The algorithmic resolution of the PSP, in heuristic context, is directed through the exploration of the molecular energy surface. The sampling process is performed by altering the backbone structure in order to obtain different structural variations.

Different encoding approaches were considered in literature, the trivial approach considering the direct coding of atomic Cartesian coordinates. The main disadvantage of direct coding is the fact that it requires filtering and correcting mechanisms, inducing non-negligible affected times. Moreover, by using amino-acid based codings [Krasnogor et al., 1999], hydrophobic/hydrophilic models were developed. In addition, several variations exist, making use of all-heavy-atom coordinates, C_α coordinates or backbone atom coordinates, where amino-acids are approximated by their centroids.

For the herein described method, an indirect, less error-prone, torsional angle based representation was preferred, knowing that, for a given molecule, there exists an associated sequence of atoms. More specifically, each individual is coded as a vector of torsion angle values - Fig. 1.2.

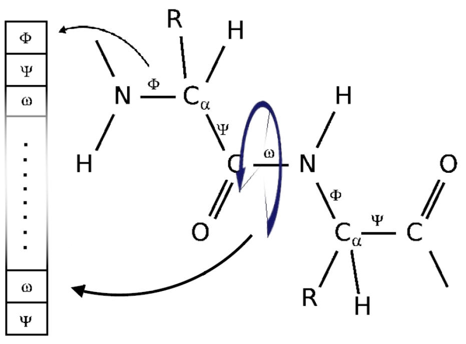


Fig. 1.2 Chromosome encoding based on specifying the backbone torsional angles.

The defined number of torsion angles represents the degree of flexibility. Apart from torsion angles which move less than a specified parameter, all torsions are rotatable. Rotations are performed in integer increments, energy quantification of covalent bonds and non-bonded atoms interactions being used as optimality evaluation criterion.

1.6.1.2 Scoring function The scoring function is computed by making use of bonded atoms energy and non-bonded atoms energy through an independently developed force field function. The quantification of energy is performed by using empirical molecular mechanics, as depicted in Table 1.6.1.2. An extensive discussion on force fields designed for protein simulations, with in-depth details, is offered in the article of [Ponder and Case, 2003]. The first part of the mentioned work covers the evolution of the force fields, starting from the 1980s and discussing various formulations which include the *Amber*, *CHARMM* and *OPLS* force fields.

Table 1.1 Scoring function quantifying the inter-atomic interactions.

$$
\begin{array}{lcc}
E = & \sum\limits_{bonds} & K_b(b-b_0)^2 \\
+ & \sum\limits_{bondangle} & K_\theta(\theta-\theta_0)^2 \\
+ & \sum\limits_{torsion} & K_\phi(1-\cos n(\phi-\phi_0)) \\
+ & \sum\limits_{Van\ der\ Waals} & \frac{K^a_{ij}}{d^{12}_{ij}} - \frac{K^b_{ij}}{d^{6}_{ij}} \\
+ & \sum\limits_{Coulomb} & \frac{q_i q_j}{4\pi\varepsilon d_{ij}} \\
+ & \sum\limits_{desolvation} & \frac{Kq_i^2 V_j + q_j^2 V_i}{d^4_{ij}}
\end{array}
$$

The involved factors model oscillating entities, the inter-atomic forces being conceptually simulated by considering interconnecting springs between atoms. A specific constant is associated with each type of interaction, notationally denoted by K_{inter}. An optimal value for the considered entity (bond, angle, torsion) is introduced in the equation as reference for the variance magnitude - $(A - A_0)$. A stands for the experimentation value, while A_0 specifies the natural, experimentally observed value. More specific, b represents the bond length, θ the bond angle, ϕ the torsion angle and q_a, d_{ij} and V_p the electrostatic charge associated with a given atom, the distance between the i and the j atoms and a volumetric measure for the p atom respectively. Although part of the designed algorithms, we considered out of scope for the current study to enter into further details concerning the employed force field. The use of empirical force fields has the drawback of offering results which are not directly comparable with results obtained through another differently-parameterized force field. This inconvenient is avoided by *ab initio* techniques although at the price of high computational demands for calculating the energy of the conformations.

An example of $\alpha - cyclodextrin$ energy surface is given in Fig. 1.3.

The set of corresponding molecular conformations was obtained by modifying a specified near-optimal initial conformation. Two torsional angles were chosen at

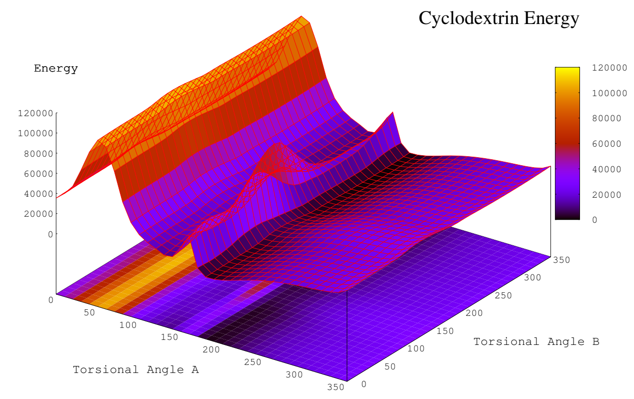


Fig. 1.3 Energy surface for $\alpha - cyclodextrin$. High energy points are depicted in light colors, the low energy points being identified by the dark areas.

random, for each of the two torsional angles, values between 0 and 360 being enumerated, in 10 degrees increments, all the other torsional angles being maintained rigid. The lighter areas on the obtained surface correspond to high-energy conformations. Furthermore, an energy-map representation is given, in the XY-plane - only the dark regions are meaningful. The hyper-surface, generated by varying the entire set of torsional angles has an extremely rough landscape, with a large number of local optima.

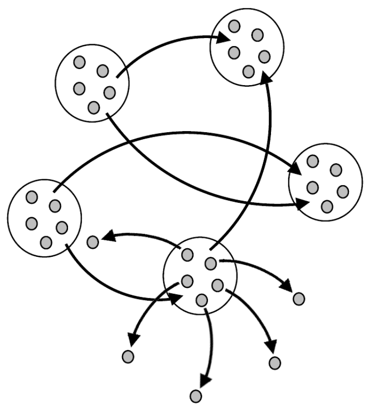
1.6.2 Parallelization of the Genetic Algorithm

The presented GA is parallelized in a hierarchical manner. First, several GAs cooperate by exchanging their genetic material (parallel island model [Alba et al., 2005a]). Second, as the fitness function of each GA is time-intensive the fitness evaluation phase of the GA is parallelized (parallel evaluation of the population model [Alba et al., 2005a]). These two models are provided in a transparent way through the ParadisEO-G4 framework [Cahon et al., 2005b][Melab et al., 2006], dedicated to the reusable design of parallel hybrid metaheuristics on computational grids.

The granularity of the problem, as counterpart term for the computationally expensive fitness evaluations, biased the resolution pattern towards a parallel, cooperative island-model approach. As a consequence, several populations evolve on a master machine, fitness function evaluations being distributed on remotely available computing units. We have to note that the evaluation of the fitness function consists of several stages, including the calculation of Cartesian atomic coordinates, interatomic distances determination, etc.

Complexity may also be addressed by developing specialized operators in conjunction with hybrid and parallel algorithms. The parallel affinity of the EAs represents a feature determined by their intrinsic population-based nature. The main parallel models are the island cooperative model, the parallel evaluation of the population and the distributed evaluation of a single solution.

For a complete overview on parallel and grid specific metaheuristcs refer to [Cahon et al., 2005b][Talbi, 2002][Alba et al., 2005a][Cahon et al., 2004b].



Island deployment model and parallel evaluation of the individuals: multiple algorithms are executed independently in a concurrent manner, migrations of the individuals occurring on a periodic basis. Parallel evaluation of the population is also employed by each algorithm - for simplicity the figure depicts only one algorithm performing the parallel evaluation step.

The designed genetic algorithm follows the above exposed pseudo-code, including in addition two levels of parallelism - island model and parallel evaluation of the population. An island model was adopted in the design, several independent algorithms being executed concurrently, the algorithms exchanging individuals at a predefined number of iterations - emigrants and immigrants. The exchange is performed in an asynchronous manner, *i.e.* no synchronization between the execution/generations of the algorithms is imposed - an important aspect to consider when there is an interest in having algorithms converging at different rates. The emigrant conformations are selected through a stochastic tournament technique, the integration of the immigrant individuals being performed by replacing the worst individuals in the population. At each migration phase, one third of the population is selected for the exchange. In addition, at each generation, the best obtained conformation till that point is preserved.

In order to exploit the local search capabilities of the conjugated gradient local search method, two different sets of operators were designed. For each operator type, crossover and mutation, a simple approach was followed - a two-point crossover and, respectively, a one-point torsion angle mutation. In addition, operators that apply the local search method on the outcome offsprings were designed. As an example, for the second case, the crossover generates two new conformations starting from two specified parents, the offsprings being optimized by applying the local search method. Similarly, for the mutation operator, after applying a random torsion angle variation on a random chosen angle, the local search method is applied. One potential drawback of this technique resides in the fact that it may lead to a premature convergence of the algorithm, thus a careful balancing of the two sets of operators being required, allowing in the same time for diversity and convergence.

The migrating individuals contribute to maintaining diversity while assuring for the coordinated convergence of all the islands.

1.6.3 Conjugated Gradient Local Search

The developed methods may benefit from relying on a hybrid architecture, combining, for example, a genetic algorithm with a conjugated gradient-based local search method - thus, a *Lamarckian* optimization technique is constructed.

The exploration and the intensification capabilities of the exploration algorithms, do not suffice as paradigm, when addressing rough molecular energy function landscapes. Small variations of the torsion angle values may generate extremely different individuals, with respect to the fitness function. As a consequence, a nearly optimal configuration, considering the torsion angle values, may have a very high energy value, and thus, it may not be taken into account for the next generations.

In order to correct the above exposed problem, a conjugated-gradient based method may be applied for local search, alleviating the drawbacks determined by the conformation of the landscape. Fig. 1.4 was obtained by applying the local search technique for each of the conformations that were previously used for generating the $\alpha - cyclodextrin$ energy surface in Fig. 1.3.

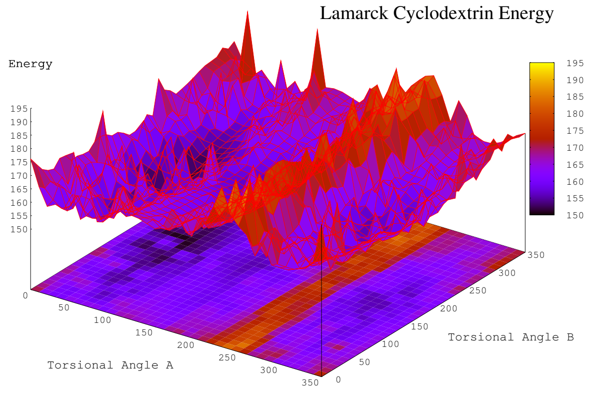


Fig. 1.4 Energy surface obtained after applying a Lamarck local search on the initial set of conformations.

1.7 ADAPTIVE SIMULATED ANNEALING

1.7.1 Simulated Annealing Algorithm

Simulated annealing algorithms are solution-based metaheuristics and were introduced as a generalization of Metropolis Monte-Carlo techniques, for simulating the evolution of a solid in the process of annealing - refer to Alg. 1.7.1 for a simple pseudo-code example. The system starts from an initial disordered state gradually following a cooling schedule, maintaining the thermodynamic equilibrium. Modifications of the current state are accepted on a Boltzmann probability distribution, *i.e.* the acceptance probability being computed according to a previous found state. Simulated annealing algorithms have a performance guarantee of finding the global optimum provided an idealistic long enough schedule is offered. The algorithm may act alternatively as a global search or as a local search method, depending on the schedule.

Algorithm 1.7.1. SA pseudo-code.

```
Generate($S_0$);
$k$ := 0;
while $T(k) > T_{threshold}$ do
    for $s$ := 1 to $nbSamples$ do
        $S_{rand} := randomMove(S_0)$;
        $\Delta E := eval(S_{rand}) - eval(S_0)$;

        if $\Delta E < 0$ then
                $S_0 := S_{rand}$;
        else
                $S_0 := S_{rand}$ with prob. $\frac{1.0}{1.0+e^{\Delta E/T(k)}}$
        endif
    endfor
    $k := k + 1$;
endwhile
```

The main drawback of the simulated annealing algorithm consists in the fact that it is difficult to parallelize without breaking the underlying philosophy, resulting in high computational-demanding methods. As a counterpart, there is no optimality guarantee proof for the genetic algorithm. For our study, a limited number of samples were generated at each step of the schedule, the samples being evaluated in parallel. Furthermore, a synchronous multi-start model is employed for launching several SA algorithms in parallel on a random generated set of initial solutions, at each step of the schedule, a specified number of sampled conformations being evaluated in parallel. The overall best found value is considered as final result. Although more complex simulated annealing variants may be constructed, for our purposes a minimalist version was preferred as to not induce an artificial bias between the compared algorithms.

Another problem to be considered for the simulated annealing algorithm would be the design of a cooling schedule to be followed. For our case a simple exponential decreasing schedule was considered, at each iteration of the algorithm, the temperature being reduced by multiplication by a fixed constant. In this case, the initial temperature must also be attentively selected. More sophisticated variants of simulated annealing algorithms render the method less sensitive to the different parameters involved, *i.e* adaptive versions, etc.

1.7.2 Adaptive Simulated Annealing Formalism Basis

The main directions to be followed in improving the basic simulated annealing technique consider modifications of the generating probability density function, the temperature schedule or parallel and hybridization schemes. The adaptive simulated annealing technique presented in the work of Ingber [Ingber, 1993a] relies on periodically reannealing the temperature while modifying different control parameters in concordance with the explored landscape - a simulated quenching technique may also be employed. As opposed to Boltzman simulated annealing, the Ingber's adaptive variant samples the ergodic space in a $D + 1$ dimensional space, where D represents the cardinality of a solution, i.e. the number of configuration parameters, the additional element standing for the associated fitness value.

Adaptive simulated annealing represents an outcome of a previous method, Very Fast Simulated Reannealing (VFSR) which exploited the characteristics of a specific-designed generating function in order to allow for an exponential faster annealing process, as compared to Boltzman annealing. Adding a reannealing step allows to adjust the control parameters of the algorithm as influenced by sensitivities measured in the multidimensional space, for each of the configuration parameters which model a solution.

In the followings a few basic notions are exposed in order to set the basis for formally sketching the phases of an adaptive simulated annealing algorithm - to no extent exhaustive as presentation. Please refer to the work of Ingber for detailed descriptions, including comparison and test case studies [Ingber, 2001][Ingber, 1996] [Ingber, 1993a][Ingber, 1993b][Ingber and Rosen, 1993] - the associated source code for the method is publicly available. For the herein presented case, a custom simplified implementation was preferred as starting point.

A standard simulated annealing consists of (1) a probability density function of the state space, $g(x)$, (2) an acceptance probability function, $h(\Delta E)$, and (3) an annealing schedule, $T(k)$. The annealing schedule is defined over a number of discrete steps, serving as control parameter for the acceptance probability function. The acceptance function offers a quantification for the probability of performing a transition from an E_k energy state to a new state with energy E_{k+1}, being defined as follows:

$$h(\Delta E) = \frac{\exp(-E_{k+1}/T)}{\exp(-E_{k+1}/T)+\exp(-E_k/T)} = \frac{1}{1+\exp(\Delta E/T)}$$

$$\frac{1}{1+\exp(\Delta E/T)} \approx \exp(-\Delta E/T), \Delta E = E_{k+1} - E_k$$

The acceptance function allows for high acceptance rates at high temperatures while leading to low acceptance rates with the decrease of temperature - thus a bias towards improved solutions is induced. High temperatures determine high acceptance rates with low discrimination between solutions the method acting like a global search procedure while local search is performed for low temperatures.

Considering a classical Gaussian-Markov system with a probability density state space function $g(x)$ defined as

$$g(\Delta x) = (2\pi T)^{-D/2} \exp[-\Delta x^2/(2T)],$$

for an appropiate initial temperature T_0, the global minimum can be found if the temperature is decreased no faster than $T(k) = T_0/\ln k$.

The main difficulty in designing a Boltzman SA consists in determining the starting temperature as well as an efficient schedule for the problem under study. In practice, a $T_0/\ln k$ schedule does not offer a fast enough annealing, an exponentially decreasing schedule being preferred, in different forms, like, for example, $T(k) = T_0 \exp((c-1)k), T(k) = c\,T_0$, with $0 < c < 1$, e.g. $c \approx 0.98$. To be noted that, although offering faster cooling schedules, the algorithm's proof of statistical asymptotic convergence does not stand for the aforementioned exponential annealing schemes.

As a general formalism, employing Ingber's notations, we consider a minimization problem under the following form:

$$\min_{\alpha\in\Lambda} F(\alpha),\ \alpha = (\alpha_1, ..., \alpha_n),\ \Lambda = \{\alpha : A_i \leq \alpha_i \leq B_i\}$$

According to the Ingber's proof, for a generating function g_T defined as

$$g_T(y) = \prod_{i=1}^{D} \frac{1}{2(|y^i| + T_i)\ln(1 + 1/T_i)} \equiv \prod_{i=1}^{D} g_T^i(y^i)$$

the associated cumulative probability distribution can be written as follows:

$$G_T(y) = \int_{-1}^{y^1} \cdots \int_{-1}^{y^D} dy'^1 ... dy'^D g_T(y') \equiv \prod_{i=1}^{D} G_T^i(y^i)$$

$$G_T^i(y^i) = \frac{1}{2} + \frac{sgn(y^i)}{2} \frac{\ln(1 + |y^i|/T_i)}{\ln(1 + 1/T_i)}$$

In this case, the global optimum is statistically attainable for a $T_i(k) = T_{0i} \exp(-c_i k^{1/D})$ annealing schedule:

$$\sum_{k0}^{\infty} g_k \approx \sum_{k0}^{\infty} [\prod_{i=1}^{D} \frac{1}{2|y^i| c_i}] \frac{1}{k} = \infty$$

The adaptive simulated annealing generation function is defined as follows, over a set of uniformly generated random variables, $u^i \in U[0, 1]$:

$$\alpha^i_{k+1} = \alpha^i_k + y^i(B_i - A_i), \text{ where } y^i \text{ is defined as}$$

$$y^i = sgn(u^i - 1/2)\, T_i[\,(1 + 1/T_i)^{|2u^i-1|} - 1\,], y^i \in [-1, 1]$$

In addition to the standard simulated annealing approach, a reannealing method is periodically applied making use of pre-computed sensitivities which offer information describing the local landscape structure. For each $\alpha_i \in \alpha$ the associated sensitivity s_i is computed. For the herein formalism, sensitivities are computed by including gradient information although no restriction is imposed on defining different sensitivities measures. At a specified number of accepted solutions, reannealing takes place, using the pre-computed sensitivities for updating temperatures and the k_i steps as follows:

$$T'_{ik} = \tfrac{s_{max}}{s_i} T_{ik}, k'_i = \left[\tfrac{\ln(T_{0i}/T'_{ik})}{c_i}\right]^D, s_i = \tfrac{\partial F}{\partial \alpha_i}$$

Considering also simulated quenching, the annealing schedule may be re-defined for including quenching factors:

$$T_{k_i} = T_{0i} \exp(-c_i k_i^{Q_i/D}), \text{ where } c_i = m_i \exp(-n_i Q_i/D)$$

m_i, n_i defining control parameters which may be adjusted for fine-tuning the algorithm for a specific problem.

REFERENCES

Alba et al., 2005a. Alba, E., G. Luque, E.-G. T., and Melab, N. (2005a). Metaheuristics and parallelism.

Alba et al., 2005b. Alba, E., G.Luque, E.-G. T., and Melab, N. (2005b). Metaheuristics and parallelism. In Alba, E., editor, *Parallel Metaheuristics*, Wiley Series on Parallel and Distributed Computing. Wiley.

Belding, 1995. Belding, T. C. (1995). The distributed genetic algorithm revisited. In Eshelman, L., editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 114–121, San Francisco, CA. Morgan Kaufmann.

Bernstein et al., 1977. Bernstein, F. C., Koetzle, T. F., Williams, G. J. B., Meyer, Jr., E. F., Brice, M. D., Rodgers, J. R., Kennard, O., Shimanouchi, T., and Tasumi, M. (1977). The protein data bank: a computer-based archival file for macromolecular structures. *Journal of Molecular Biology*, 112:535 – 542.

Buyya et al., 2003. Buyya, R., Branson, K., Giddy, J., and Abramson, D. (2003). The virtual laboratory: a toolset to enable distributed molecular modelling for drug design on the world-wide grid. *Concurrency and Computation: Practice and Experience*, 15(1):1–25.

Cahon et al., 2004a. Cahon, S., Melab, N., and Talbi, E.-G. (2004a). Paradiseo: A framework for the reusable design of parallel and distributed metaheuristics. *J. Heuristics*, 10(3):357 – 380.

Cahon et al., 2005a. Cahon, S., Melab, N., and Talbi, E.-G. (2005a). An enabling framework for parallel optimization on the computational grid. In *CCGRID*, pages 702 – 709.

Cahon et al., 2005b. Cahon, S., Melab, N., and Talbi, E.-G. (2005b). An enabling framework for parallel optimization on the computational grid. In *CCGRID '05: Proceedings of the Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05) - Volume 2*, pages 702–709, Washington, DC, USA. IEEE Computer Society.

Cahon et al., 2004b. Cahon, S., Melab, N., Talbi, E.-G., and Talbi, E. G. (2004b). Paradiseo: A framework for the reusable design of parallel and distributed metaheuristics. *Journal of Heuristics*, 10(3):357–380.

Ewing and Kuntz, 1998. Ewing, T. J. A. and Kuntz, I. D. (1998). Critical evaluation of search algorithms for automated molecular docking and database screening. *Journal of Computational Chemistry*, 18(9):1175 – 1189.

Fernandez-Recio et al., 2002a. Fernandez-Recio, J., Totrov, M., and Abagyan, R. (2002a). Screened charge electrostatic model in protein-protein docking simulations. *Pac Symp Biocomput*, pages 552–563.

Fernandez-Recio et al., 2002b. Fernandez-Recio, J., Totrov, M., and Abagyan, R. (2002b). Soft protein-protein docking in internal coordinates. *Protein Sci*, 11(2):280–291.

Foster et al., 2001. Foster, I., Kesselman, C., and Tuecke, S. (2001). The anatomy of the Grid: Enabling scalable virtual organizations. *Int. J. High. Perform. Comput. Appl.*, 15(3):200 – 222.

Gelernter and Carriero, 1992. Gelernter, D. and Carriero, N. (1992). Coordination languages and their significance. *Commun. ACM*, 35(2):96.

Germain et al., 2000. Germain, C., Neri, V., Fedak, G., and Cappello, F. (2000). Xtremweb: Building an experimental platform for global computing. In *GRID*, pages 91–101.

H. Van de Waterbeemd, 1997. H. Van de Waterbeemd, R.E. Carter, G. G.-H. K. Y. M. M. T. P. W. (1997). Glossary of terms used in computational drug design. *Pure and Applied Chemistry*, 69(5):1137 – 1152.

Holger Claußen and Lengauer, 2001. Holger Claußen, Christian Buning, M. R. and Lengauer, T. (2001). Flexe: Efficient molecular docking considering protein structure variations. *J. Mol. Biol.*, 308:377–395.

Hurng-Chun Lee and Wu, 2006. Hurng-Chun Lee, Jean Salzemann, N. J. L.-Y. H. H.-Y. C. V. B. I. M. L. M. S. C. L. and Wu, Y.-T. (2006). Grid-enabled high throughput in-silico screening against influenza a neuraminidase. NETTAB 2006, Santa Margherita, Italy.

Ingber, 1993a. Ingber, L. (1993a). Adaptive simulated annealing (asa). Technical report, Pasadena, CA.

Ingber, 1993b. Ingber, L. (1993b). Simulated annealing: Practice versus theory. *Mathematical Computer Modelling*, 18(11):29–57.

Ingber, 1996. Ingber, L. (1996).

Ingber, 2001. Ingber, L. (2001). Adaptive simulated annealing (asa) and path-integral (pathint) algorithms: Generic tools for complex systems. Technical report, Chicago, IL.

Ingber and Rosen, 1993. Ingber, L. and Rosen, B. (1993). Genetic algorithms and very fast simulated reannealing: A comparison. *Oper. Res. Management Sci.*, 33(5):523.

Krasnogor et al., 1999. Krasnogor, N., Hart, W. E., Smith, J., and Pelta, D. A. (1999). Protein structure prediction with evolutionary algorithms. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., and Smith, R. E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 1596–1601, Orlando, Florida, USA. Morgan Kaufmann.

Krauter et al., 2002. Krauter, K., Buyya, R., and Maheswaran, M. (2002). A taxonomy and survey of grid resource management systems for distributed computing. *Software Practice and Experience*, 32(2):135–164.

Lorber and Shoichet, 1998. Lorber, D. M. and Shoichet, B. K. (1998). Flexible ligand docking using conformational ensembles. *Protein Sci*, 7(4):938 – 950.

Melab et al., 2006. Melab, N., Cahon, S., and Talbi, E.-G. (2006). Grid computing for parallel bioinspired algorithms. *J. Parallel Distrib. Comput.*, 66(8):1052–1061.

Morris et al., 1999. Morris, G. M., Goodsell, D. S., Halliday, R. S., Huey, R., Hart, W. E., Belew, R. K., and Olson, A. J. (1999). Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function. *Journal of Computational Chemistry*, 19(14):1639–1662.

N. Jacq, 2006. N. Jacq, J. Salzemann1, Y. L. M. R.-F. J. M. Z. A. M. M. S. K. V.-K. H. S. M. H. V. B. (2006). Demonstration of in silico docking at a large scale on grid infrastructure. In Press, I., editor, *Challenges and Opportunities of HealthGrids, Proceedings of HealthGrid 2006*, Studies in Health Technology and Informatics.

Nakajima et al., 2004. Nakajima, Y., Sato, M., Goto, H., Boku, T., and Takahashi, D. (2004). Implementation and performance evaluation of conflex-g: grid-enabled molecular conformational space search program with omnirpc. In *ICS '04: Proceedings of the 18th annual international conference on Supercomputing*, pages 154–163, New York, NY, USA. ACM Press.

Neumaier, 1997. Neumaier, A. (1997). Molecular modeling of proteins and mathematical prediction of protein structure. *SIAM Review*, 39(3):407 – 460.

Papadopoulos and Arbab, 1998. Papadopoulos, G. A. and Arbab, F. (1998). Coordination models and languages. In *761*, page 55. Centrum voor Wiskunde en Informatica (CWI), ISSN 1386-369X.

Ponder and Case, 2003. Ponder, J. W. and Case, D. A. (2003). Force fields for protein simulations. *Adv Protein Chem*, 66:27–85.

R.D. Taylor and Essex, 2002. R.D. Taylor, P. J. and Essex, J. (2002). A review of protein-small molecule docking methods. *Journal of Computer-Aided Molecular Design*, 16:151–166.

Sandak et al., 1998. Sandak, B., Wolfson, H. J., and Nussinov, R. (1998). Flexible docking allowing induced fit in proteins: insights from an open to closed conformational isomers. *Proteins*, 32(2):159 – 174.

Talbi, 2002. Talbi, E.-G. (2002). A taxonomy of hybrid metaheuristics. *J. Heuristics*, 8(5):541 – 564.

Teodoro et al., 2001. Teodoro, M., Phillips, G., and Kavraki, L. (2001). Molecular docking: A problem with thousands of degrees of freedom.

Wolfson and Rigoutsos, 1997. Wolfson, H. J. and Rigoutsos, I. (1997). Geometric hashing: An overview. *IEEE Comput. Sci. Eng.*, 4(4):10 – 21.