Novel Algorithm for Efficient Distribution of Molecular Docking Calculations

Luigi Di Biasi[1,2], Roberto Fino[1], Rosaura Parisi[1], Lucia Sessa[1], Giuseppe Cattaneo[2], Alfredo De Santis[2], Pio Iannelli[1], and Stefano Piotto[1(✉)]

[1] Department of Pharmacy, University of Salerno, Via Giovanni Paolo II, 132-84084 Fisciano, SA, Italy
piotto@unisa.it
[2] Department of Informatics, University of Salerno, Via Giovanni Paolo II, 132-84084 Fisciano, SA, Italy

Abstract. Molecular docking is a computational method to study the formation of intermolecular complexes between two molecules. In drug discovery, it is employed to estimate the binding between a small ligand (the drug candidate), and a protein of known three-dimensional structure. Docking is becoming a standard part of workflow in drug discovery. Recently, we have used the software VINA, a de facto standard in molecular docking, to perform extensive docking analysis. Unfortunately, performing a successful blind docking procedure requires large computational resources that can be obtained by the use of clusters or dedicated grid. Here we present a new tool to distribute efficiently a molecular docking calculation onto a grid changing the distribution paradigm: we define portions on the protein surface, named hotspots, and the grid will perform a local docking for each region. Performance studies have been conducted via the software GRIMD.

1 Introduction

Drug discovery is a time-consuming, risky, and expensive process. To shorten the research cycle and to lower the failure rate, Computer-Aided Drug Design is applied in the early drug discovery phases. Molecular docking is one of the most popular strategies to evaluate the drugability of a molecule. An efficient search of the best binding interaction of a ligand is extremely computationally demanding, and existing software suffer several limitations. As result, the predictive ability of docking software is severely limited, especially for blind docking. Different approaches have been used by different research groups to identify the receptor binding site and to estimate all the contributions to the total binding energy of the ligand – receptor complex. No commercially available or free-to-use software for molecular docking consider the importance of conserved sequence in proteins. Often the active site of the receptor is unknown, so the only option left to identify all the possible binding sites on the receptor is to extend the search box to all the receptor, a type of molecular docking also known as blind docking, reducing the accuracy of the procedure and leading to an increase of computing times. The existing software are, from a computational point of view, extremely inefficient. In fact, programs like Vina [1] can generate thousands of

F. Rossi et al. (Eds.): WIVACE 2015, CCIS 587, pp. 65–74, 2016.
DOI: 10.1007/978-3-319-32695-5_6

poses with a small coverage of the conformational space. What is worst is that the best pose is often rejected even after an exhaustive pose generation. This means that the program found a pose close to the experimental one but the scoring function poorly evaluated it. The pose ranking is based on a calculated binding energy that shows a poor correlation with experimental binding energies. For this reason, one cannot simply rely on massive calculation of an astronomic number of poses, because the ranking functions are not properly calibrated. During extensive docking analysis, we observed that conserved residues often lie on binding sites [2, 3]. Our idea was to drive ligands toward conserved regions on the surface adding an extra term to the force field. We decided to use the software Vina because it is efficient and open source. We could observe that, in most cases, binding sites lie on conserved portion on the protein surface [2, 3]. The opposite in not always true, so we can assume that the presence of conserved residues is a necessary but not sufficient condition to predict a binding site. Conserved residues are rarely isolated. Normally, a binding site can be made of several spatially closed but non-adjacent residues.

2 Experimental Part

We define as hotspot (HS) the barycenter of spatially related conserved residues. The conserved regions can be easily obtained by multiple sequence analysis, but an easier way consists in downloading essential information from the server PDBFinder [4]. The distance of a pose from the HS is then used to modify the Vina function.

The poses that satisfied the Vina criteria are checked out in terms of distance from the HS. The binding energy is than modified according to Eq. (1), adding a term that depends on the minimal distance (d) between the ligand barycenter and the nearest HS. The new energy takes into account also the conservation value of the residue (conservation weight, C_W):

$$E_{QN} = E^0_{QN} + \frac{E^0_{QN} C_w}{d^2} \tag{1}$$

These values of E_{QN} (quasi-Newton energy) are saved in the Prop channel and used to train the genetic algorithm.

```
void quasi_newton::operator()(model& m, const
precalculate& p, const igrid& ig, output_type& out,
change& g, const vec& v) const { // g must have correct
size
        quasi_newton_aux aux(&m, &p, &ig, v);
        fl res = bfgs(aux, out.c, g, max_steps, aver-
age_required_improvement, 10);
        fl hsval = m.eval_conservation(out.coords,out);
        out.yadaRankProp = res*hsval;
        out.dist = hsval;
        out.e = res;
        }
```

In red is highlighted where the calculation is called in the original quasi_newton.cpp Vina file. The definition of the function eval_conservation is in the file model.cpp.

```
fl model::eval_conservation(vecv poseCoords,output_type&
out)
{
        //find nearest hot spot
        if (poseCoords.size()==0)
                return 0;
        // for each hot spot check which of them is close
to current ligand
        int j;
        int hsmin = 0;
        fl min_distance =
std::numeric_limits<float>::max();;
        for (j=0;j<yada_hscoords.size();j++)
        {
                fl r;
                int k;
                r = 0;
                int c;
                for (k=0;k<=poseCoords.size();k++)
                        for (c=0;c<=3;c++)
                                r+=  pow(poseCoords[k][c]
-yada_hscoords[j][c],2);
                r = sqrt(r);
                if (r<min_distance)
                {
                        min_distance = r;
                        hsmin=j;
              }
        }
        // nearest hotspot found. Looking for reference
atom.
        int k;
        min_distance = std::numeric_limits<float>::max();
        for (k=0;k<=poseCoords.size();k++)
        {
                // check distance between atom and
'reference atom' in nearest
                // hotspot
                 int c;
                 float r;
                 r=0;
                 for (c=0;c<=3;c++)
                         r+=  pow(poseCoords[k][c]-
 yada_hscoords[hsmin][c],2);
                 r = sqrt(r);
                 if (r<min_distance)
                 {
                         min_distance = r;
                 }
         }
         // save nearest atom
         vec
 hsrefatompos(yada_hscoords[hsmin][0],yada_hscoords[hsmin]
 [1], yada_hscoords[hsmin][2]);
         out.hsrefatomcoords = hsrefatompos;
         // return prop
         return
 pow(yada_hscoords[hsmin].hsval,2)*(1/min_distance);
 }
```

When we want to explicitly consider the presence of water molecules, the program checks if there is space to introduce an oxygen atom at 2.9 Å from electronegative atoms in the ligand. If so, the QN function is called with and without oxygen and only the pose with the maximum QN is chosen.

The choice of poses is made with a traditional Metropolis approach. Metropolis algorithm is a Markov chain Monte Carlo method for obtaining a sequence of random poses from a probability distribution for which direct sampling is difficult. When the energy results are to be higher, the new conformation will be accepted or rejected if an acceptance probability law

$$P = e^{\left[-\frac{E_2 - E_1}{k_b T}\right]} \tag{2}$$

is randomly satisfied, where T is temperature and kB the Boltzmann's constant. The acceptance condition is verified if generating a pseudo-random number u, uniformly distributed between 0–1, will result $u < P$.

The following changes have been introduced in metropolis.cpp.

```
bool metropolis_accept(fl old_f, fl new_f, fl tempera-
ture, rng& generator,fl dist,output_type& m ) {
        if (dist>5) return false;
        if(new_f < old_f) return true;
        const fl acceptance_probability = std::exp((old_f
- new_f) / temperature);
        // flip coin here,
        return random_fl(0, 1, generator) < ac-
ceptance_probability;
}
```

The initial idea was to increase the acceptance probability nearby conserved regions in order to drive the ligands toward those sites. The acceptance rate was therefore modified as:

$$AccProb = AccProb \cdot \frac{1}{\left(distance_{pose-hotspot}\right)^2} \tag{3}$$

The distance is obtained from the routine Quasi_Newton to decrease the computing time. The instruction m.metEnable, reads the flag to tell the program when use the modified Metropolis.

```
bool metropolis_accept(fl old_f, fl new_f, fl tempera-
ture, rng& generator,fl dist,output_type& m ) {
        if (dist>5) return false;
        if(new_f < old_f) return true;
        fl yada_increase = 0;
        if (m.metenable)
            yada_increase = 1/pow(m.hsdistance,2)

        const fl acceptance_probability = std::exp((old_f
- new_f) / temperature) + yada_increase;
return random_fl(0, 1, generator) < ac-
ceptance_probability;
}
```

The flowchart representing the implementation of HS in the original Vina code is shown in Fig. 1.

2.1 Improving the Scoring Function

One of the typical problems with docking software, and Vina makes no exception, is that the pose ranking is made in terms of energy. Vina uses a semi-empiric calculation of the pose energy. Unfortunately, the calculation of free energy is far from being optimal and, consequently, the ranking process is poor. This means that the best pose, i.e. the one with the minimum RMSD from experimental data, is not the first in rank. We have performed an extensive genetic algorithm study to improve the ranking. As result, in 95 % of the cases, the best pose is the one with the highest score.

2.2 Porting on Grimd

Grimd is a software that can easily create a computer grid [5]. Grimd can chunk a complex calculation in a number of smaller jobs. The jobs are sent to available PCs (slaves) and, after completion, the most relevant results are collected and made available via web interface. The Master, which is a dedicated machine, runs a program that takes care of the input data partitioning, the scheduling, the tasks execution across a set of machines (dynamically updated), the handling of machine failures, and the managing of the required inter-machine communication. The Master implements a basic authentication mechanism when a new slave subscribes to the "Grimd Network", managing communication privacy through channel encryption (a sort of VPN) and client-side strong authentication through session key negotiation. The distributed grid was already successfully applied for a wide range of applications [6–10]. Grimd was used to perform a flexible ligand-flexible receptor docking encoding the conformational spaces of molecules through a protocol of molecular dynamics, followed by the generation of an ensemble of rotamers. These conformational subspaces can be built to span a range of conformations important for the biological activity of a protein.

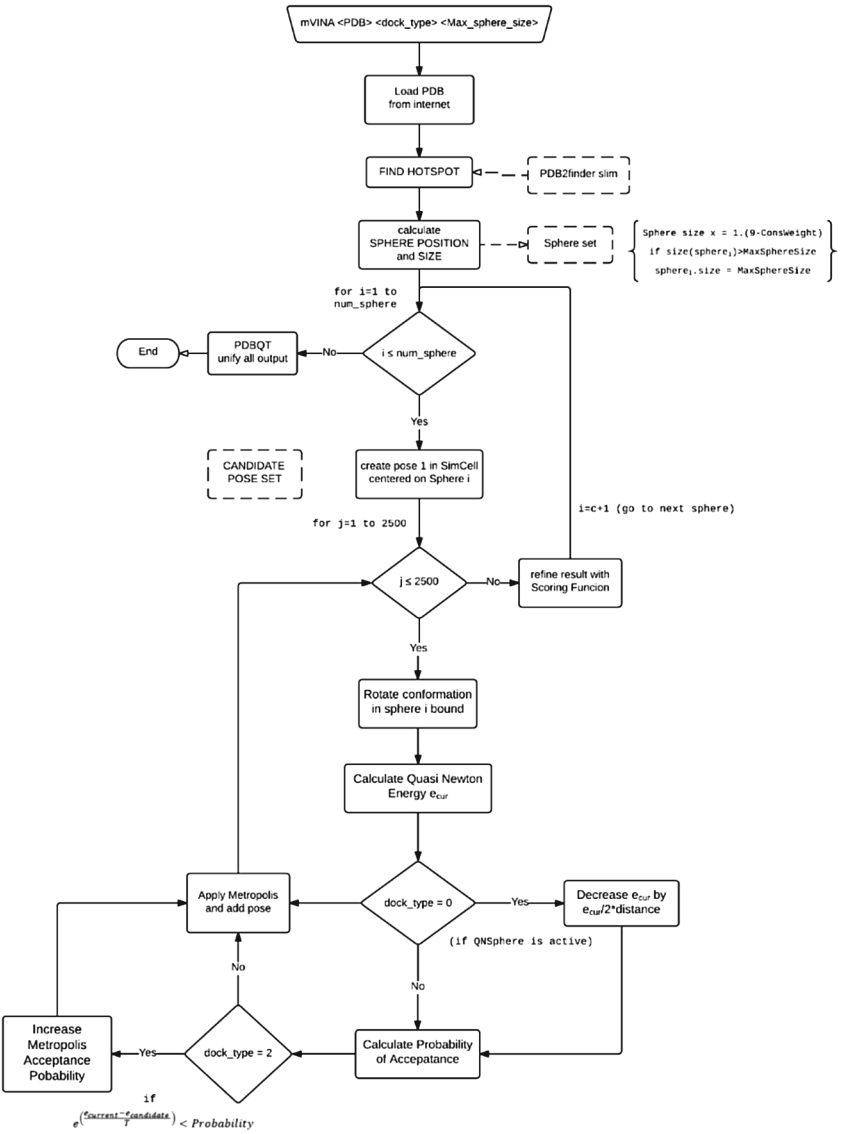

Fig. 1. Flowchart of the implementation of HS on Vina code

A variety of motions can be combined, ranging from domains moving as rigid bodies or backbone atoms undergoing normal mode-based deformations, to side chains assuming rotameric conformations. In addition, Grimd can be easily used for the screening of several receptors against a large library of ligands. Because of the underlying architecture, Grimd is not limited to docking or molecular dynamics, but it has been also applied to extend coarse grain dynamics [11, 12], to distribute quantum mechanical calculations [13, 14], and to improve of orders of magnitude the speed of

Monte Carlo simulations [15]. The concept of hotspot permitted a straightforward distribution on a grid. In fact, each hotspot can be computed on a different node of the grid. To run a full blind docking calculation, it is necessary to send three files to the Master: the receptor and the ligand in pdbqt format, and a text file containing the conservation string. An example of the text file is:

```
##EXPLOSION(1) = WRITE_RANGE[30,60,1] @ENDEXP
1au2_ligand.pdbqt 1au2_receptor.pdbqt
893579884593865885983998997994486897531346847549949698992
138399399473389497227297658439891625419274322257232661475
254925832886136877767874125955933989263273333669687589992
326245977899983299349962757634717888328789 26
#min_range = '@EXPLOSION(1)@'
#hotspot_pos = @HS@
#exit
```

Once the Master received the job request, each HS is assigned to a different Slave. Once the local calculation is completed, the Slaves create two files: a file with results and a log file. These files are sent back to the Master that reduces the information into a global result file, sorts the poses using the binding energy and print out one or more poses. No changes are necessary on the Master to distribute the calculation. The flowchart representing the porting of Vina on Grimd is shown in Fig. 2.

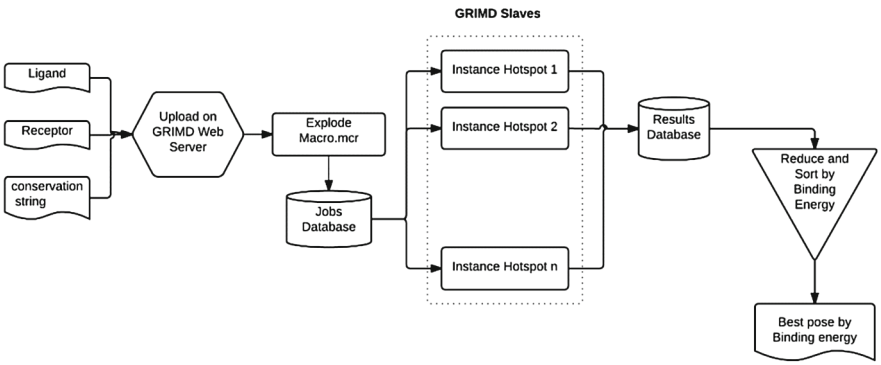

Fig. 2. Flowchart of VINA porting on GRIMD

3 Results

3.1 Docking Validation

We tested the accuracy of the new approach on a customized version of the Aspex list of PDBs available in literature. The validation of a docking software is always a critical task. Several works already discussed this point [3]. Among others, two aspects appear

to be critical: the choice of the validation data set, and the position (and dimension) of the docking box. In fact, it is evident that an ad hoc choice of the proteins to be docked can give the illusion of fantastic performance. Few publications offer a systematic comparison of software performances. The validation did not use any prior information on the docking box, or ligand orientation, nor had we used a particular (and benevolent) pdb validation set. The evaluation of was performed on a set of 180 proteins and the results compared to Vina, a de facto standard in molecular docking. The docking procedure was total blind docking, 250 runs, Amber03 ff, no water molecules. We have considered two aspects in blind docking: the goodness of the first pose in terms of RMSD between the docked pose and the experimental data, the free energy of binding and the execution time.

3.2 Grid Tests

Standard procedures were followed to set up the virtual docking. AutoDock requires that the ligands and receptor be formatted in pdbqt files. This format is similar to a PDB file and also has charge and AutoDock atom-type information. These files can be created with the AutoDockTools (ADT) [16] interface or with scripts provided with the software. ADT is a graphical interface provided with the AutoDock software and can be used to carry out serial docking jobs, prepare files, and analyze results. The provided scripts were used to add partial charges to each atom, merge nonpolar hydrogens with the heavy atoms to which they are covalently bound and determine the AutoDock atom types of each atom for all the ligands and decoys. Using HSs, a docking calculation with Vina can be easily distributed on several machines.

In Fig. 3, it is shown the computing time for the system 1 fkg distributed on a number of nodes between 1 and 10. The behavior of the net is almost linear demonstrating a perfect distribution of the docking.

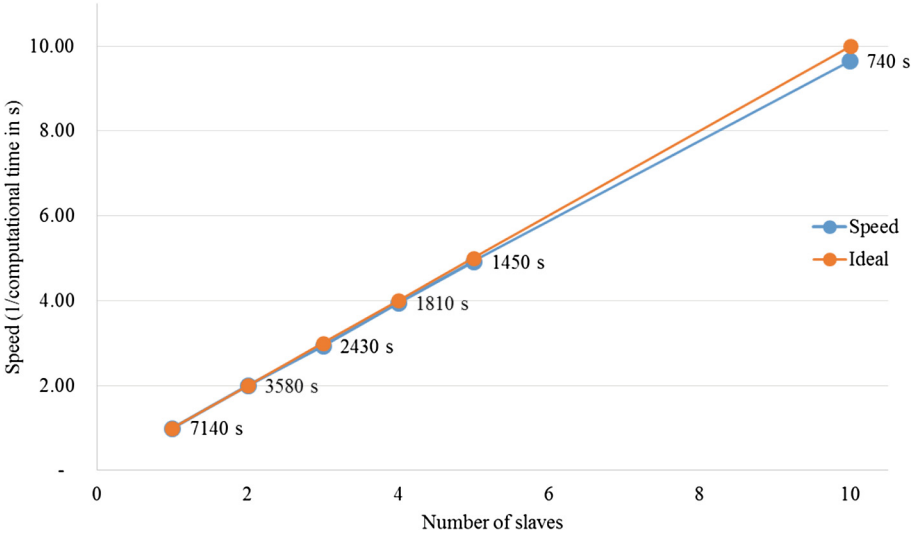

Fig. 3. Running time as function of slave number

4 Conclusions

We presented here a modification of Vina that permitted to increase the accuracy of docking as well as a load distribution on a dedicated grid that permitted a drastic reduction of the computational time. The pose generation algorithms and the scoring function for pose ranking have been modified in order to consider the conservation of residues in the protein sequence. To assign the conservation weights to each residue we used a customized version of the HSSP database [4]. In correspondence to each conserved region, the program places new local search boxes. The size of these boxes is inversely depending on the conservation of the residue: the highest the conservation, the smallest is the size. The choice of the best pose follows a completely novel approach. We have used Genetic Algorithms (GA) to develop a scoring function that takes into account force field related energy as well as the distance of the ligand from conservation regions. According to this, our new ranking function allows the final user to pick the best pose after the molecular docking with a better accuracy and reliability. Finally, the introduction of hotspots, i.e. highly conserved residues in a protein, permitted a straightforward and efficient distribution onto a dedicated grid.

Acknowledgments. This work was partially supported by the "Data-Driven Genomic Computing (GenData 2020)" PRIN project (2013–2015), funded by the Italian Ministry of the University and Research (MIUR).

References

1. Trott, O., Olson, A.J.: AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. J. Comput. Chem. **31**(2), 455–461 (2010)
2. de Vries, S.J., van Dijk, A.D., Bonvin, A.M.: WHISCY: What information does surface conservation yield? Application to data-driven docking. Proteins Struct. Funct. Bioinf. **63**(3), 479–489 (2006)
3. Ouzounis, C., Pérez-Irratxeta, C., Sander, C., Valencia, A.: Are binding residues conserved? In: Pacific Symposium on Biocomputing, pp. 401–412 (1997)
4. Hooft, R.W., Sander, C., Scharf, M., Vriend, G.: The PDBFINDER database: a summary of PDB, DSSP and HSSP information with added value. Comput. Appl. Biosci. CABIOS **12**(6), 525–529 (1996)
5. Piotto, S., Di Biasi, L., Concilio, S., Castiglione, A., Cattaneo, G.: GRIMD: distributed computing for chemists and biologists. Bioinformation **10**(1), 43 (2014)
6. Concilio, S., Bugatti, V., Neitzert, H.C., Landi, G., De Sio, A., Parisi, J., Piotto, S., Iannelli, P.: Zn-complex based on oxadiazole/carbazole structure: synthesis, optical and electric properties. Thin Solid Films **556**, 419–424 (2014)
7. Lopez, D.H., Fiol-deRoque, M.A., Noguera-Salvà, M.A., Terés, S., Campana, F., Piotto, S., Castro, J.A., Mohaibes, R.J., Escribá, P.V., Busquets, X.: 2-Hydroxy arachidonic acid: a new non-steroidal anti-inflammatory drug. PLoS ONE **8**(8), e72052 (2013)
8. Piotto, S., Concilio, S., Bianchino, E., Iannelli, P., López, D.J., Terés, S., Ibarguren, M., Barceló-Coblijn, G., Martin, M.L., Guardiola-Serrano, F.: Differential effect of 2-hydroxyoleic acid enantiomers on protein (sphingomyelin synthase) and lipid (membrane) targets. Biochim. Biophys. Acta (BBA)-Biomembr. **1838**(6), 1628–1637 (2014)

9. Piotto, S., Trapani, A., Bianchino, E., Ibarguren, M., López, D.J., Busquets, X., Concilio, S.: The effect of hydroxylated fatty acid-containing phospholipids in the remodeling of lipid membranes. Biochim. Biophys. Acta (BBA)-Biomembr. **1838**(6), 1509–1517 (2014)
10. Scrima, M., Di Marino, S., Grimaldi, M., Campana, F., Vitiello, G., Piotto, S.P., D'Errico, G., D'Ursi, A.M.: Structural features of the C8 antiviral peptide in a membrane-mimicking environment. Biochim. Biophys. Acta (BBA)-Biomembr. **1838**(3), 1010–1018 (2014)
11. Caracciolo, G., Piotto, S., Bombelli, C., Caminiti, R., Mancini, G.: Segregation and phase transition in mixed lipid films. Langmuir **21**(20), 9137–9142 (2005)
12. Piotto, S., Concilio, S., Mavelli, F., Iannelli, P.: Computer simulations of natural and synthetic polymers in confined systems. Macromol. Symp. **286**(1), 25–33 (2009)
13. Piotto, S., Nesper, R.: CURVIS: a program to study and analyse crystallographic structures and phase transitions. J. Appl. Crystallogr. **38**(1), 223–227 (2005)
14. Acierno, D., Amendola, E., Bugatti, V., Concilio, S., Giorgini, L., Iannelli, P., Piotto, S.: Synthesis and characterization of segmented liquid crystalline polymers with the azo group in the main chain. Macromolecules **37**(17), 6418–6423 (2004)
15. Piotto, S., Mavelli, F.: Monte Carlo simulations of vesicles and fluid membranes transformations. Orig. Life Evol. Biosph. **34**(1–2), 225–235 (2004)
16. Morris, G.M., Huey, R., Lindstrom, W., Sanner, M.F., Belew, R.K., Goodsell, D.S., Olson, A.J.: AutoDock4 and AutoDockTools4: automated docking with selective receptor flexibility. J. Comput. Chem. **30**(16), 2785–2791 (2009)