Multithreaded version of AutoDock 4.2 suitable for massive virtual screening of potential biologically active compounds (enzyme inhibitors)

Vsevolod Tanchuk, Volodymyr Tanin, Andriy Vovk

Institute of Bioorganic Chemistry and Petrochemistry, National Academy of Sciences of Ukraine, Murmanska, 1, 02660, Kyiv-94, Ukraine

v_tanchuk@yahoo.com

Abstract. *A modified version of a well-known docking package AutoDock 4.2 was created. It is specially designed for virtual screening of a large number of potential inhibitors. AutoDock uses a set of pre-calculated grids and other parameters describing intermolecular interaction which are the same for the same target molecule. Multithreading allows more efficient use of memory. Alternatively it can help to increase precision. The process becomes much more efficient and more suitable for massive screening. Besides that a new search algorithm is proposed to handle many ligands which are too flexible to be thoroughly investigated by AutoDock's stochastic search engine in any reasonably time.*

Keywords

Parallel processing, multithreading, molecular docking, global optimization.

1 Introduction

Search for new drugs is one of the most important problems in medicine, chemistry and biology. In the field of molecular modeling, docking is a method which predicts the preferred orientation of one molecule to a second when bound to each other to form a stable complex. Docking is frequently used to predict the binding orientation of small molecule drug to their protein targets in order to in turn predict the affinity and activity of the small molecule (ligand).

2 Related works

AutoDock 4.2[1] is one of the most widely used docking packages. Besides that it is distributed under GPL license which allows its modifying. Being quiet easy to use AutoDock 4.2 has some limitations:

1) Usage of precomputed special grids. A separate grid is build for each type of ligand atoms over the region of space where molecular interaction is expected. The energy of interaction in each node is computed and stored. Such approach requires considerable amounts of computer memory. Some energy functions are also stored in tabular form which further increases the amount of required memory. If AutoDock were able to use several threads this memory could be common for all of them and would not limit the number of simultaneously processed ligands. On the other hand grid cells could be made smaller to improve the precision of calculations.
2) AutoDock's search engine is based on a special version of genetic algorithm (GA), the so called Lamarckian genetic algorithm [2]. It is not very fast. Sometimes docking of a molecule requires many hours, which makes screening of a large number of molecules impractical. Besides that the algorithm may fail in case of a large binding cavity of an enzyme (e.g. thrombin, protein-tyrosine phosphatase 1B, etc.) and a flexible ligand.

The authors tried to address these problems.

3 Modified version of AutoDock

The modifications of AutoDock include:

1) The ability to apply one or several spatial constraints on docking poses. In this case one or several atoms are positioned in some predefined areas. It might be very helpful if it is known which groups play important role in binding of inhibitors.
2) The ability to use standard multidimensional optimization techniques instead of stochastic search. In combination with special constraints make the search process much faster and more reliable. If the search is constrained binding optimization requires approximately 100 times less time than the original version. Besides that modified AutoDock usually finds lower energy minima.
3) The ability to make a sequential search where the best conformation is passed to the next run. This leads to global optimization.
4) The ability to use charges calculated by MOPAC 2009 PM6 method known to improve binding poses [3].
5) An auxiliary program PrepRDock to prepare multiple files for docking by AutoDock 4.2. The utility prepares multiple task files from a template with or without special constraints. Spatial constraints can be applied to different atom types of the ligand. Multiple task files with different binding modes can be generated for the same substance. Despite that, docking takes much less time than in case of original Autodock 4.2.
6) Multithreading. Several ligands can be docked at the same time using the same precomputed grids and lookup tables. The number of threads is specified.

3.1 Details of a new search algorithm

Adding ligand constraints.

It seemed to be a good idea to add some constraints, so that some atoms of inhibitor may be located only in certain regions of space. This has been realized by adding a new feature such as atom position constraint (a keyword ATPOSCONSTR in a dpf file). ATPOSCONSTR keyword is followed by atom number of an atom in a ligand's pdbqt file, maximal distance from the center of the allowed region, x, y, and z coordinates of the center and weight parameter (relative importance of this constraint). If the distance from the atom to the center of the region exceeds maximal value a penalty term is added to the energy calculated by AutoDock:

*Penalty = (dist – maxdist) * 1000 * weight,*

Where *dist* is a distance and *maxdist* – maximal allowed distance between the atom and the center of its allowed region, weight is a number defining relative importance of this constraint.

New optimization mode.

Adding such penalty term helped to find some suitable docking poses for our ligands, but due to stochastic nature or the search procedure it happened only from time to time. Most computer time and efforts have been wasted. This led to a second modification. A new search mode based on standard multidimensional optimization has been added. If there is a constraint the procedure is following:

Molecule is moved so that a constrained atom is placed exactly into the center of allowed region. Multiple random orientations of the ligand are generated (default number 100). Constrained atom remains in the centre of coordinates while the ligand is rotated over x, y, and z axis by random angles, generated by Sobol sequence [4]. Random coordinates can be generated optionally as well.

For each random orientation a minimization routine based on Direction Set (Powell's) method [4] is applied. This method is suitable because it does not require calculation of derivatives. Optimization is done over all torsional, translational and rotational degrees of freedom. A constrained penalty term is added to the energy calculated by standard AutoDock's procedure.

In case of several constraints the procedure is different:

The most important constraint is chosen (by weight) as constraint A. Constraint A is used as in item 1 above. Molecule is moved so that a constrained atom is placed into the center of allowed region. The second constraint is chosen as constraint B. This is either the second constraint by weight or the most distant constraint from the constraint A. Molecule is rotated so that direction between the constrained atoms coincides with the direction between the centers of the respective allowed regions. The direction will be called A-B axis. The ligand is rotated around the A-B axis with a step of 0.5°. For each of 720 positions the same minimization procedure as above is applied. Penalty terms are added for all constraints. After initial minimization the standard search procedure of AutoDock is applied.

Stepwise (global) optimization.

Besides that there is a new option called stepwise optimization. If it is used (keyword 'stepwise' added to a dpf file) the best global result is not lost. After finishing a run the best individual survives and is passed to a next run. The rest individuals of the initial population are generated randomly and compete with the best result from the previous runs (global best). The idea is similar to elitist selection in genetic algorithm, but in this case the best individual is passed not to the next generation, but to the next optimization which has been completely independent originally. In such way parallel runs of GA are maid sequential.

The result of docking is shown on **Fig 1.** Proper positioning of a ligand is hard to achieve relying on pure GA.

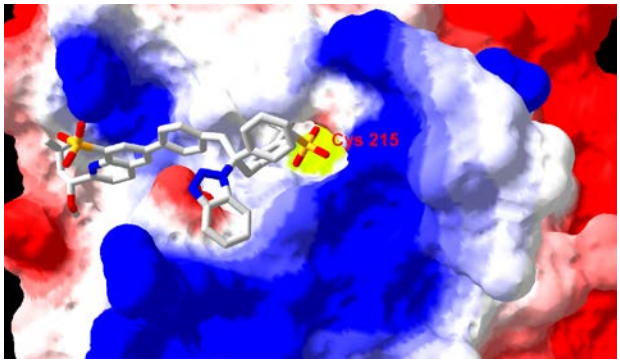

Fig. 1. Proper binding of inhibitor is ensured by special constraint on P atom of the phosphonate group. P atom must not be located more than 2 Å from the position of P which is near catalytic Cys215 of PTP1B in original PDB file (2CM8).

3.2 Multithreading

The new search procedure is 10 to 30 times faster than the original GA. Virtual screening of a large number of molecules becomes feasible. The remaining problem is the use of memory. The original AutoDock requires about 300-700 M of memory most of which is occupied by precomputed grids and lookup tables. This limits the number of processes that can be launched simultaneously. Multithreading leads to minimization of memory requirements (all precomputed data are the same for all simultaneously processed ligands) allowing to use the full power of the available processors. Some time ago there was an attempt to speedup AutoDock by porting it to CUDA [5]. The authors claim that their version of AutoDock is up to 50 times faster that it's basic version AutoDock 4.0.1 of 2007. Though porting AutoDock to CUDA made the package much faster, it required substantial efforts to rewrite the source code. It seems that because of that newer versions have not been ported to CUDA.

In our work, the acceleration is not achieved only by parallelization and using more computing power, but mostly by a new optimization algorithm. Parallelization is also a substantial improvement which allows to save memory and run multiple threads simultaneously facilitating the high-throughput screening of potential drugs. AutoDock Vina is a new generation of docking tools[6]. It is much faster and gives better average accuracy of the binding mode predictions. Though its advent does not make the AutoDock obsolescent. The authors of both packages say on their Web site (http://autodock.scripps.edu/): "Because the scoring functions used by AutoDock 4 and AutoDock Vina are different and inexact, on any given problem, either program may provide a better result". This means that it is a completely different approach and detailed comparison with our version is not needed. AutoDock Vina is multithreaded, but this multithreading might lead to some problems. AutoDock Vina determines the number of processors in the system and starts the same number of threads trying to gain full control over the computer. It is not always the best strategy. In out vesrion of AutoDock the user can specify how many threads to run.

4 Conclusion

As a result a new version of AutoDock that is much more suitable for the investigation of ligand binding into large binding cavities and for faster and more reliable virtual screening is created. At the present time there is a version for MS Windows that can be compiled by Visual C++. A version for Linux compilable by g++ is being created. Both versions will be made available under GPL license shortly after the conference (at first by e-mail request from the authors).

References

[1] R. Huey, G.M. Morris, A.J. Olson, D.S. Goodsell: A semiempirical free energy force field with charge-based desolvation. Journal of Computational Chemistry, 28(6): 1145-1152, 2007.

[2] G. M. Morris, D. S. Goodsell, R. S. Halliday, R. Huey, W. E. Hart, R. K. Belew, A. J. Olson: Automated docking using a Lamarckian genetic algorithm and an empirical binding free energy function. Journal of Computational Chemistry, 19(14): 1639-1662, 1998.

[3] Z. Bikadi, E. Hazai: Application of the PM6 semi-empirical method to modeling proteins enhances docking accuracy of AutoDock. *J Cheminform.* 1, 15, 2009.

[4] W. H. Press, S. A.Teukolsky, W. T.Vetterling, B. P.Flannery: *Simulated Annealing Methods.* Second Edition ed.; Cambridge University Press: 1996.

[5] S. Kannan, R. Ganji: Porting Autodock to CUDA. WCCI 2010 IEEE World Congress on Computational Intelligence. July, 18-23, 2010 - CCIB, Barcelona, Spain.

[6] O. Trott and A. J. Olson: AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization and multithreading. J Comput Chem. January 30; 31(2): 455–461. 2010.