QuickVina: Accelerating AutoDock Vina Using Gradient-Based Heuristics for Global Optimization

Stephanus Daniel Handoko, Xuchang Ouyang, Chinh Tran To Su, Chee Keong Kwoh, and Yew Soon Ong

Abstract—Predicting binding between macromolecule and small molecule is a crucial phase in the field of rational drug design. AutoDock Vina, one of the most widely used docking software released in 2009, uses an empirical scoring function to evaluate the binding affinity between the molecules and employs the iterated local search global optimizer for global optimization, achieving a significantly improved speed and better accuracy of the binding mode prediction compared its predecessor, AutoDock 4. In this paper, we propose further improvement in the local search algorithm of Vina by heuristically preventing some intermediate points from undergoing local search. Our improved version of Vina—dubbed QVina—achieved a maximum acceleration of about 25 times with the average speed-up of 8.34 times compared to the original Vina when tested on a set of 231 protein-ligand complexes while maintaining the optimal scores mostly identical. Using our heuristics, larger number of different ligands can be quickly screened against a given receptor within the same time frame.

Index Terms—Artificial intelligence, bioinformatics, global optimization, gradient methods.

1 Background

Molecular docking is a computational process trying to find the binding between a macromolecule (the receptor) and a small molecule (the ligand). Since it can be used in predicting binding conformations and affinities between drug molecules and their target proteins, leading to the understanding of the biological mechanism behind those bindings, molecular docking is with great value to drug design [1].

Generally, docking is an optimization problem that attempts to find the binding conformation with global lowest energy, the landscape of which is approximated by a scoring function. The introduction of flexibility in the ligand, or further in the receptor as well, will make the problem more sophisticated [1], [2]. The major issue of the difficulty comes from the large number of degrees of freedom in modeling the molecular system. Since 1980s, various programs and software have been developed in order to perform molecular binding, such as DOCK [1], AutoDock [3], GOLD [4], ICM [5], and FlexX [6] and different scoring functions have been proposed. However, after decades of development, docking is still a time-consuming task even with the most powerful computing resources to-date. In 2009, AutoDock Vina [7] (referred to as Vina afterward) was released by the same group who invented the earlier versions of AutoDock, which is one of the most popular docking software. Vina uses an empirical scoring function to evaluate the binding affinity between the molecules, and the iterated local search global optimizer for global optimization. This combination is reported to be successful to achieve approximately two orders of magnitude improvement in speed, and simultaneously, a significantly better accuracy of the binding mode prediction compared to AutoDock 4 [7].

In this paper, we proposed an improvement in the local search procedure of Vina. By heuristically preventing some of the intermediate points from performing local search, our improved version of Vina, named QuickVina (QVina), achieved a maximum speed-up of about 25 times with an average speed-up of 8.34 over a testing data set of 231 protein-ligand complexes from the PDBBind [8] and a tendency to have a higher speed-up with the larger number of degrees of freedom, without compromising the quality of docking result.

- *S.D. Handoko is with the Centre for Computational Intelligence, School of Computer Engineering, Nanyang Technological University, Blk N4, #B1a-02, Nanyang Avenue, Singapore 639798. E-mail: sdhandoko@ntu.edu.sg.*
- *X. Ouyang and C.T.T. Su are with the BioInformatics Research Centre, School of Computer Engineering, Nanyang Technological University, Blk NS4, #04-33, Nanyang Avenue, Singapore 639798. E-mail: {xouyang1, sutr0003}@e.ntu.edu.sg.*
- *C.K. Kwoh and Y.S. Ong are with the School of Computer Engineering, Nanyang Technological University, Blk N4, #02a-26, Nanyang Avenue, Singapore 639798. E-mail: {asckkwoh, asysong}@ntu.edu.sg.*

Manuscript received 8 Nov. 2011; revised 11 Mar. 2012; accepted 20 Apr. 2012; published online 23 May 2012.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBBSI-2011-11-0290.
Digital Object Identifier no. 10.1109/TCBB.2012.82.

2 Methods

2.1 Analyzing the Global Optimization Algorithm in Vina

At the time this paper is drafted, the source code of the AutoDock Vina is available free of charge at its website: http://vina.scripps.edu/. With the lack of detailed explanation on how exactly the search algorithm works in Vina, we performed a thorough analysis of the source code. In Fig. 1, we present the pseudocode of the global optimization approach employed by Vina. Fundamentally, it is a form of

1545-5963/12/$31.00 © 2012 IEEE Published by the IEEE CS, CI, and EMB Societies & the ACM

```
1:  Initialize conformation vector X, BEST
2:  For step := 1 TO MAX_STEP
3:    mutate(X);
4:    local_search(X);
5:    IF (metropolis_accept(X) AND score(X)<score(BEST))
6:      local_search(X);
7:      IF (score(X)<score(BEST))
8:        BEST:=X;
9:      ENDIF
10:   ENDIF
11: ENDFOR
```

Fig. 1. Pseudocode of the global optimization in Vina.

Memetic Algorithm (MA) [9] that interleaves global and local searches by means of stochastic and deterministic optimization algorithms, respectively. It is indeed mentioned in [7] that Vina employs the iterated local search global optimizer.

From our analysis, it is understood that Vina uses a Monte Carlo method to perform the global search. It is similar to a Markov chain Monte Carlo algorithm with restart, but differs in that after mutating the current solution to a neighbor randomly, a local search is initiated therefrom and the result of the local search is checked against some probability of acceptance according to the Metropolis criterion instead of checking the neighbor directly. Given sufficient number of steps, it is well understood that local search will converge to a local minimum. As the global minimum is one of the possibly many local minima, marrying local and global search increases the likelihood to identify the global minimum with a sufficiently high precision within a shorter time frame as global search is a powerful tool for identifying the region surrounding the global optimum.

To perform the local search, Vina employs the commonly used algorithm of Broyden-Fletcher-Goldfarb-Shanno (BFGS) for nonlinear unconstrained optimization [10]. In order to make progress toward better decision variables, BFGS—being essentially a Newton method—requires the first- and second-order derivatives of the objective function—which in this case is the scoring function—with respect to all design variables (i.e., the translation and rotation vectors between the receptor and the ligand as well as all the rotatable bonds). The second-order derivatives (also known as the Hessian), however, are approximated in BFGS as they are often very expensive to compute and not readily available. Hence, employing BFGS involves calculating the gradients of the target function, finding a direction along which the optimization progresses, and updating the approximation of the Hessian. After several iterative steps, the BFGS shall converge to a point with vanishing gradients in all directions that are either infinitesimally small or exactly zero. This point is the local minimum, at which the local search is terminated. The pseudocode of the BFGS used in Vina is shown in Fig. 2.

Specific to Vina is the parallelism that takes advantage of the multi-CPU or multicore architecture of many computer systems nowadays. When the main function of Vina is invoked, several parallel threads are initialized to perform the global optimization concurrently and cooperatively. This is believed to be partly responsible for the Vina's speed-up over its predecessor, AutoDock 4. It is reported in [7] that Vina has achieved a remarkable speed-up compared to the

```
1:  FUNCTION local_search(vector X)
    // X stores the conformation
2:    vector G:=calc_derivative(X);
3:    Initialize H;
      // H is an approximate to Hessian
4:    FOR step:= 1 TO MAX_STEP
5:      find a direction p by solving Hp=-G;
6:      do line search along p, and update X and G;
7:      IF (|G|->0)
8:        BREAK;
9:      ENDIF
10:     update H;
11:   ENDFOR
12: ENDFUNCTION
```

Fig. 2. Pseudocode of the BFGS for local search in Vina.

AutoDock 4, which adopts merely the Lamarckian Genetic Algorithm (GA) for the lowest energy conformation search. The introduction of local search in Vina has also contributed greatly to the speed-up attained as estimates of the global optimum produced by the global search method are allowed to converge rapidly to some local optima. From our in-depth analysis of the source code of Vina, however, we found out that the local search of Vina is the most time-consuming component of the optimization process. Should it be possible to either accelerate the local search or perform local search selectively only on some potential starting points, it will reduce the docking time even more. The latter approach has successfully been adopted in [11], [12], and [13].

2.2 Adapting the Local Search Frequency of Vina

In Vina, the most time-consuming component turns out to be the local search. This, as the matter of fact, is very much intuitive as the local search is responsible for refining an estimate of the global optimum such that it converges to some local optimum, hence requiring a number of intermediate steps that bridge the initial and the final locally lowest energy conformations. Meanwhile, the global search is responsible only for producing the estimates of the global optimum for further refinements by the local search. Thus, it should be easy to appreciate that the local search dominates the time spent to perform the global optimization process. Furthermore, Vina employs a local search algorithm in the class of quasi-Newton methods which requires computing the first-order derivatives of the scoring function for each newly generated set of design variables. The numerous intermediate steps produced by the local search adds to the time complexity of the entire global optimization procedure. Fortunately, the local search algorithm in this class is capable of approximating the second-order derivatives of the scoring function using the widely used and well-known BFGS update formula due to Broyden, Fletcher, Goldfarb, and Shanno [10].

With the first-order derivatives with respect to all design variables—the translation and rotation vectors between receptor and ligand as well as all the rotatable bonds—readily available in Vina, we propose in this paper a heuristics that makes use of this information to adaptively adjust the local search frequency of Vina. That means with minimum additional computational cost, we propose to assess the significance of each estimate of the global optimum produced by the global search to undergo a local search. In

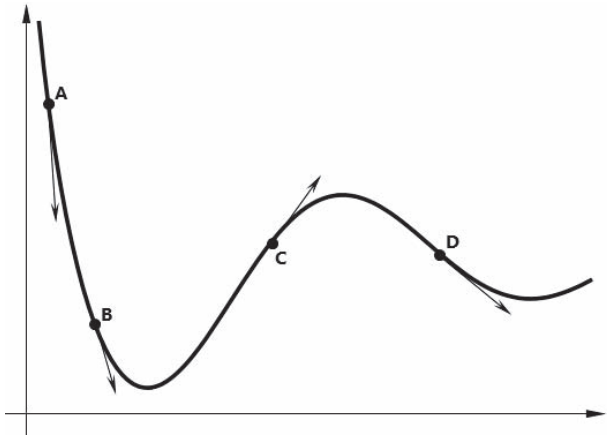

Fig. 3. Curve of a function of one design variable and its derivative at various points. Points A, B, and D are with negative derivative and point C is with positive derivative.

other words, only those estimates deemed significant by our heuristics shall undergo local refinements, hence reducing the time complexities of the entire global optimization procedure as there shall be less intermediate steps produced by the local search. This paper, therefore, not only presents software improvement but also, as shall be described next, a novel and sound mathematical-based strategy that efficiently makes use of the gradients of the scoring function of Vina. It shall be emphasized, therefore, that such strategy is the original idea of the authors, which in one way or another, is a continuation to their previous studies [11], [12], [13].

Considering the fact that local search is to find a local minimum, which must be a stationary point (in which the derivative is zero) in continuous functions according to Fermat's theorem, it is possible to prevent some of the point from performing local search. For illustrative purpose, an example of a continuous function in 1D space is shown in Fig. 3. Points A, B, and D are with negative derivatives while C is with a positive derivative. If the derivative of the function is also continuous, which stands true for the scoring function used in Vina, there must be a point with zero derivative between the points of A and C according to the Intermediate value theorem, but not guaranteed between the points of A and B. Based on this observation, we here propose an improvement in the global minimization in Vina by heuristically preventing some of the point from performing local search using the information of the points which has been seen in the previous searching steps.

The basic idea of the proposed heuristic is to test an intermediate point ready for local search against a database which stored the information of other points which have been previously reached by local search. If it is possible to find any neighbor of the point with totally opposite signs of derivatives in the database, this point passes the test, local search is performed and the result of the local search is sent to the database to be stored for possible future usage. Exception, however, can be observed from Fig. 3 when point D is the only neighbor of point A. Two points with derivatives of the same sign do not always suggest there are no stationary points in-between. Our experiments, however, suggest that this situation does not occur very often, or even if that is really happening, the randomness of the global search would have alleviated the problem. In Fig. 3, point C with a positive derivative would eventually be encountered by the global search algorithm, and since point A is already there, the local search would commence from there, reaching the same stationary point.

2.3 Implementing the Proposed Conceptual Framework as QuickVina

To gauge empirically the efficacy of our proposal detailed in the previous section, we have implemented our proposed conceptual framework as QuickVina. We, however, have kept the amount of modifications to the minimum possible so as to ensure the fairness when comparing the performance of QVina to that of the original Vina. We kept the program design and architecture of the original Vina and used the same programming language as what the group from the Scripps Research Institute had used: C++. We only added new components that are required for the utilization of our proposed heuristics, namely the collection of visited points, and did modification to the original code just before the local search procedure is invoked. A check against the criteria of being significant point for the execution of the local search is inserted thereat.

In order to store the additional information needed for the utilization of our proposed heuristics, we have implemented two new classes: 1) the database, and 2) the element of the database. For each element in the database, intuitively, two vectors of real numbers need to be stored to describe a visited point in local search: one for the design variables that defines where the point is, and the other for its corresponding derivatives. It is worth noting that for storing the derivatives, only their signs are of our interest, hence bit compression is introduced. Since Vina inherits the limitation of 32 rotatable bonds from the AutoDock 4, the 64-bit integer data type is sufficient to accomplish this purpose. Hence, the data stored in visited are finally a vector of real numbers and two integers (one integer is not representative enough because zero in derivative should be a distinct state rather than positive or negative). The database is constructed rudimentarily as an array of its elements, along with some additional information such as the number of dimensions.

Additionally, in the database class, necessary supporting functions are also created, among them is the most important function for checking whether to perform local search when a new start point presented. The invocation of this function is inserted into the original code just before the local search procedure is about to be performed. This function is implemented to check whether there are some existing points in the database—which are the neighbors of the point of interest upon which local search shall be initiated—with derivatives of entirely the opposite signs against those of the point of interest in every dimension of the design variables. In our implementation, the euclidean distance is adopted as the distance metric, and the neighbors of a particular point of interest are the set of the $2 * N$ nearest points to that particular point, where N is the number of dimensions. As the number of dimensions grows, it may become more and more difficult to find the so-called significant point of interest to undergo local search. An additional constraint is then introduced. We only consider dimensions on which both positive and negative derivatives are already encountered in

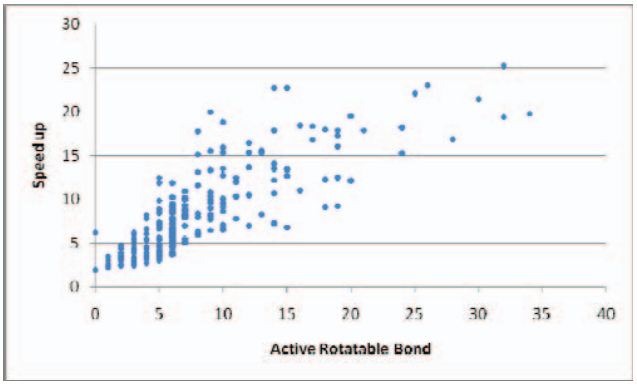

Fig. 4. Speed-up against the number of rotatable bonds. The speed-up is calculated by T_{Vina}/T_{QVina}, where T is the time used to perform docking. The larger the number of active rotatable bonds is, the more complicated the docking problem becomes. Our result shows that QVina has a trend to achieve higher speed-up with more active rotatable bonds.

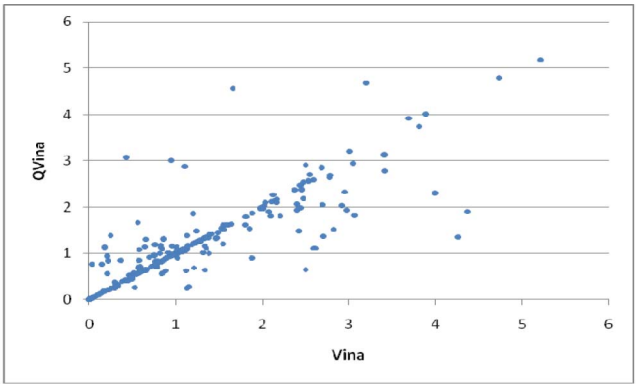

Fig. 6. The RMSE of the docking results with respect to the native conformations in PDB, showing the values for both QVina and Vina. The RMSE is calculated between the resulted structures and the known structures in PDB. The RMSE values of both QVina and Vina are shown. Most points lie around the diagonal. Some are scattered away due to the randomness of the Monte Carlo search algorithm.

the database. This proposition automatically deals with special situations such as the derivatives on some dimensions are always positive or negative, or always zero, since derivatives of this type would not be useful for our heuristics.

As an additional note, although the earlier AutoDock 4 as well as works in [11] and [12] had used Genetic Algorithm (GA) as the global search method, we have kept the global search procedure of our QVina to be the same as that of the original Vina, i.e., the Monte Carlo method with Metropolis criterion. This is to ensure that the improvements achieved are really due to the heuristics introduced herein. Further modifications, such as the use of GA, would be address in the near future.

3 Results

Following the implementation, we tested our QVina on the core set of PDBbind version 2010 [8]. The PDBbind data set is a well-established data set of protein-ligand docking. The data of PDBbind are selected from the Protein Data Bank (PDB), and several hierarchical subsets are defined for the convenience of different applications. First, 6,772 complexes including protein-ligand, nucleic-acid-ligand, protein-nucleic-acid, and protein-protein complexes with experimental measured binding affinity are identified and selected from the entire PDB to form the *general set* of PDBbind. From this *general set*, 2,061 entries of protein-ligand complexes only are selected based on several strict requirements on the data quality, among them are the minimum coordinate resolution of 2.5 Å and the absent of uncommon element types. These 2,061 entries are compiled as the *refined set*. Finally, a *core set* is compiled by clustering the *refined set* based on the protein sequence, forming 77 clusters and then selecting the highest, the lowest and the medium binding affinity from each cluster to ensure the diversity in both the structure and binding affinity [8]. The *core set* of PDBbind version 2010—comprising 231 protein-ligand complexes—was used for the test of our QVina.

In the test, both QVina and original Vina are compiled on the same cluster node with 8 2.6-GHz cores and 12-GB RAM running on the Red Hat Linux operating system. Compiling options, especially of the optimization level, were carefully kept the same. The test was to go through every individual receptor-ligand complex in the test data set of PDBbind core set with the same random seed and the same runtime options `"-cpu 8 -exhaustiveness 128."` Note that the default exhaustiveness value of the original Vina is 8 and the authors had claimed that it "should increase the time linearly and decrease the probability of not finding the minimum exponentially" [7]. The docking test is configured to set the boundary box centered at the centre of the native ligand conformation. The boundary box is essentially a cube aligned to the 3D Cartesian coordinate axes. The size of the boundary cube is either 22.5 Å or set to cover the native ligand structure with 5 Å margins on each axis, whichever is larger. This follows exactly the original Vina test setting.

The results of this performance test which consist of the running time, the docking result score and the docking result conformation of both QVina and Vina were collected. The speed-up against the number of rotatable bonds is plotted in Fig. 4. The docking result score of both QVina and Vina are shown in Fig. 5. The RMSE between the native structure in PDB and the docking result structure of both QVina and Vina is shown in Fig. 6.

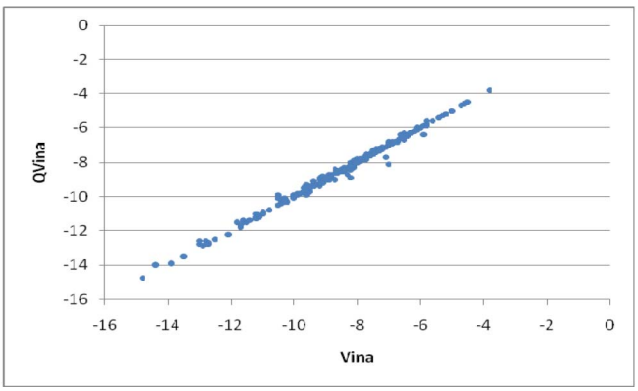

Fig. 5. The scores of the docking result of both QVina and Vina. The scores of the docking result on the test data set of QVina and Vina falls mostly around the diagonal line, showing that they are highly correlated. The correlation of the scores given by QVina and Vina is in fact 0.997.

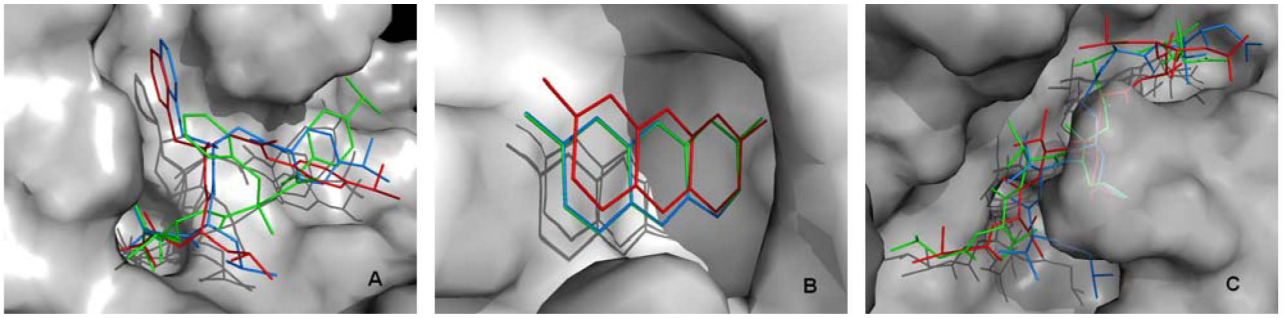

Fig. 7. Three examples of docking results from QVina (shown in green) and original Vina (shown in blue), both compared to the native ligand conformations found in PDB (shown in red). Both resultsfrom QVina and Vina have exactly the same score in each example. (A) The complex 1BMA with docking score of -7.9: the result from Vina is with RMSE of 1.272 Å; it is closer to the native conformation than the result from QVina with RMSE of 3.398 Å. (B) The complex 1BCU with docking score of -7.5: QVina and Vina give almost identical conformation, with RMSE of 0.497 Å and 0.478 Å, respectively. (C) The complex 4ER2 with docking score of -8.4: QVina gives a conformation with RMSE of 3.43 Å; it is closer to the native conformation than the result given by Vina with RMSE of 4.37 Å; it is also worth noting that at the lower left region, the blue conformation given by Vina goes to the opposite direction compared to the native conformation, contributed significantly to the larger RMSE.

4 Discussions

In order to maintain the database to a relatively small size so that nearest neighbors of a new point can be found very fast, the database in the visited class is implemented as a circular array whose tail would be warped back to be next to its head. When the size of the array grows to a predefined size of N2 where N is the number of dimensions, new data entries will be stored again from the beginning of the array, overriding the content in those array slot. This is based on the consideration that Monte Carlo method as a global optimizer will only mutate its current state to a neighboring state; and as the global search progresses, the search should be biased toward the global minimum conformation, hence only recently visited points would be of utmost relevance. In other words, the explored points would have a tendency to accumulate around the global minimum. This means the most recent intermediate result is more likely to give the information about the energy landscape around the best global minimum found thus far.

With the circular database and the assessment of significance to undergo local search, it can be clearly seen from Fig. 4 that our QVina is faster than the original Vina. Our QVina achieved an average speed-up of 8.35 times over the original Vina when tested on the test data set of PDBbind core set. Our QVina achieved the maximum speed-up of 25.24 times when tested on the protein-ligang complex 1ZEA, reducing the running time of docking from 6,640 seconds—roughly 1 hour and 50 minutes—to merely 263 seconds, which is 4 minutes 23 seconds, while maintaining the minimized score of docking the same. Such a tremendous speed-up will ultimately be useful for high-throughput screening of various ligands or small molecules, such as potential drug candidates, against a given receptor. This would eventually allow wet-lab experiments to be conducted only on the highly potential candidates, significantly reducing the cost and time needed for drug developments. Furthermore, it is also observed that the speed-up increases as the number of rotatable bonds increases. With greater flexibility—that is, more rotatable bonds—it is intuitive that the optimization process would be much more complicated. This implies larger search space to be explored by the global search method. Additionally, there would be more derivatives to evaluate as there are more design variables. There would also be more steps required by the local search method to converge to an optimum point. All these add to the time complexity of the global optimization procedure, increasing significantly the running time of the original Vina. Using the QVina, however, we manage to eliminate unnecessary local searches by means of our proposed heuristics. Consequently, computational time and resources which otherwise are supposed to be used for performing the "unnecessary" local searches in the original Vina could be eradicated in the QVina. This explains the tendency of having even more speed-up for higher number of rotatable bonds.

It should also be noted that the docking accuracy is not compromised while the running time is significantly reduced. Since the docking score serves as the objective function in the global optimization procedure of both QVina and the original Vina, we can use it as the estimation of the docking performance. From Fig. 5, it can clearly be seen that the quality of both docking software are identical. In fact, these two sets of scores have a correlation as high as 0.997—almost statistically identical—suggesting that the docking performance of QVina is never compromised. In other words, even with some local searches omitted, our QVina is able to find comparable global optimum conformations as those of the original Vina.

To further understand our results, the final conformations were also collected for structural analysis. In Fig. 6, the RMSE between the native structure in PDB and the docking result structure of both QVina and Vina is shown. As a stochastic method, it is quite normal for the docking programs to give different structures, as long as the score is stable. This is observed for example with the complexes 1BMA, 1BCU, and 1PB8, as shown in Fig. 7. For these three complexes, the docking results are scored identically by Vina and QVina, which are -7.9, -7.5, and -8.4, respectively. Since both Vina and QVina uses the same scoring function as the only objective function in optimization, the results from QVina and Vina are regarded equally optimal in the view of optimization. However, QVina and Vina gave very close docking result structure only on the complex 1BCU, which is also very close to the native structure shown in PDB. Meanwhile, QVina yielded a result closer to the native structure in the PDB on the complex 4ER2 than Vina did, while the opposite is true for the complex 1BMA. This suggests that there might be a need for a better scoring function that best describe the interaction between the receptor and the ligand. Furthermore, if the docking

accuracy is of the main concern, larger exhaustiveness could potentially be used as the docking time has been significantly reduced through the use of our method.

5 Conclusions

With the time significantly reduced compared to the original Vina, our proposed heuristics implemented as QVina shows a quite promising future for a high-throughput screening method of protein-small-molecule docking for rational drug design. A maximum acceleration of up to about 25 times was achieved when testing our proposed method on a test set of 231 protein-ligand complexes. Using our heuristics, more pairs of receptor-ligand complexes can be quickly screened within a given time frame. As time is of importance in virtual screening, our proposed method would undoubtedly be of significant value for the rational drug design.

As a proof-of-concept work, some simplifications and assumptions have been adopted in this project. In the future, different approaches to define the distance metric and the neighborhood, as well as special data structures for more efficient neighborhood searching will be explored.

Acknowledgments

This work was supported by Singapore MOE AcRF Grant No: MOE2008-T2-1-074.

References

[1] S.F. Sousa, P.A. Fernandes, and M.J. Ramos, "Protein-Ligand Docking: Current Status and Future Challenges," *Proteins: Structure, Function, and Bioinformatics,* vol. 65, no. 1, pp. 15-26, 2006.

[2] I. Halperin et al., "Principles of Docking: An Overview of Search Algorithms and a Guide to Scoring Functions," *Proteins: Structure, Function, and Bioinformatics,* vol. 47, no. 4, pp. 409-443, 2002.

[3] G.M. Morris et al., "Automated Docking Using a Lamarckian Genetic Algorithm and an Empirical Binding Free Energy Function," *J. Computational Chemistry,* vol. 19, no. 14, pp. 1639-1662, 1998.

[4] G. Jones et al., "Development and Validation of a Genetic Algorithm for Flexible Docking," *J. Molecular Biology,* vol. 267, no. 3, pp. 727-748, 1997.

[5] R. Abagyan, M. Totrov, and D. Kuznetsov, "ICM—A New Method for Protein Modeling and Design: Applications to Docking and Structure Prediction from the Distorted Native Conformation," *J. Computational Chemistry,* vol. 15, no. 5, pp. 488-506, 1994.

[6] M. Rarey et al., "A Fast Flexible Docking Method Using An Incremental Construction Algorithm," *J. Molecular Biology,* vol. 261, no. 3, pp. 470-489, 1996.

[7] 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. Computational Chemistry,* vol. 31, no. 2, pp. 455-461, 2010.

[8] R. Wang et al., "The PDBbind Database: Collection of Binding Affinities for Protein-Ligand Complexes with Known Three-dimensional Structures," *J. Medicinal Chemistry,* vol. 47, no. 12, pp. 2977-2980, 2004.

[9] Y.S. Ong and A.J. Keane, "Meta-Lamarckian Learning in Memetic Algorithms," *IEEE Trans. Evolutionary Computation,* vol. 8, no. 2, pp. 99-110, Apr. 2004.

[10] D. Shanno, "Conditioning of Quasi-Newton Methods for Function Minimization," *Math. of Computation,* vol. 24, no. 111, pp. 647-656, 1970.

[11] S.D. Handoko, C.K. Kwoh, and Y.S. Ong, "Feasibility Structure Modeling: An Effective Chaperone for Constrained Memetic Algorithms," *IEEE Trans. Evolutionary Computation,* vol. 14, no. 5, pp. 740-758, Oct. 2010.

[12] S.D. Handoko, C.K. Kwoh, and Y.S. Ong, "Classification-Assisted Memetic Algorithms for Equality-Constrained Optimization Problems," *Proc. 22nd Australasian Joint Conf. Advances in Artificial Intelligence (AI '09),* vol. 5866, pp. 391-400, 2009.

[13] S.D. Handoko, C.K. Kwoh, and Y.S. Ong, "Classification-Assisted Memetic Algorithms for Equality-constrained Optimization Problems with Restricted Constraint Function Mapping," *Proc. IEEE Congress on Evolutionary Computation,* June 2011.

Stephanus Daniel Handoko received the BEng degree (First Class) in computer engineering from NTU in 2005. He is now working toward the PhD degree. Currently, he is working as a researcher in the BioInformatics Research Centre (BIRC), School of Computer Engineering (SCE), Nanyang Technological University (NTU), Singapore. His primary research interests include real-valued optimization (operation research), soft computing, and data mining with applications in bioinformatics.

Xuchang Ouyang received the BEng degree in communication engineering from the University of Electronic Science and Technology of China in 2010 and he is now working toward the PhD degree in computer engineering in Nanyang Technological University, Singapore. His research interests include evolutionary algorithm, machine learning and high-performance computing and their applications in bioinformatics.

Chinh Tran To Su received the BSc degree in biotechnology-bioinformatics from the University of Sciences, Vietnam National University—Ho Chi Minh City and the MSc degree in bioinformatics from Nanyang Technological University, Singapore. Currently, she is working toward the PhD degree in bioinformatics in the School of Computer Engineering, Nanyang Technological University. Her interest focuses on implementing explicit solvent model in molecular dynamics simulation into protein docking to offer certain flexibility for this rigid-body docking.

Chee Keong Kwoh received the bachelor's degree in electrical engineering (First Class) and master's degree in industrial system engineering from the National University of Singapore, in 1987 and 1991 respectively, and the PhD degree from the Imperial College of Science, Technology and Medicine, University of London in 1995. Since 1993, he has been with the School of Computer Engineering. He is the programme director, MSc (bioinformatics) and deputy director, Biomedical Engrg Research Centre, NTU. His research interests include data mining and soft computing and graph-based inference; applications areas include bioinformatics and biomedical engineering. He has done significant research work his research areas and published many quality international conferences and journal papers. He is an editorial board member of *The International Journal of Data Mining and Bioinformatics*, *TheScientificWorldJOURNAL*, *Network Modeling and Analysis in Health Informatics and Bioinformatics (NetMAHIB)*, *Theoretical Biology Insights*, and *Bioinformation*. He has been a guest editor for many journals such as *JMMB*, *International Journal on Biomedical and Pharmaceutical Engineering*, and others. He has been often invited as a organizing member or referee and reviewer for a number of premier conferences and journals, including GIW, IEEE BIBM, RECOMB, PRIB, BIBM, ICDM, and iCBBE just to name a few.

Yew Soon Ong received the BS and MS degrees in electrical and electronics engineering from Nanyang Technological University (NTU), Singapore, in 1998 and 1999, respectively, and the PhD degree in artificial intelligence in complex design from the Computational Engineering and Design Center, University of Southampton, United Kingdom, in 2003. Currently, he is an associate professor and the director of the Center for Computational Intelligence, School of Computer Engineering, NTU. His current research interests include computational intelligence include memetic computing, evolutionary design, machine learning, agent-based systems, and cloud computing. He is the cofounding technical editor-in-chief of the *Memetic Computing Journal*, the chief editor of the Springer Book Series on *Studies in Adaptation, Learning, and Optimization*, an associate editor of the *IEEE Computational Intelligence Magazine*, the *IEEE Transactions on Systems, Man and Cybernetics B, Soft Computing*, *International Journal of Systems Science* and others. He is currently also the chair of the IEEE Computational Intelligence Society Emergent Technology Technical Committee and has served as a guest editor of several journals such as the *IEEE Transactions on Evolutionary Computation*.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.