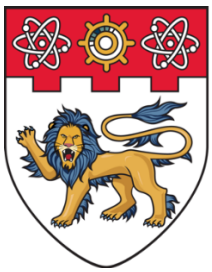


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Accurately Accelerating Drug Design Workflow

Amr Ali Mokhtar ALHOSSARY

School of Computer Science and Engineering

Thesis Supervisors

Dr. KWOH Chee Keong, Dr. MU Yuguang

A thesis submitted to the Nanyang Technological University

in partial fulfilment of the requirement for the degree of

Doctor of Philosophy

March 2018

Acknowledgment

All praise to Almighty God, who emphasized the importance of knowledge when he started his holy revelation in the Quran with the command "read"[1] and directed us in myriads of positions in the Quran and Hadith to *observe*, *contemplate,* and *ask for more knowledge*[2].

This thesis would not have been possible without my supervisors Dr. Kwoh Chee Keong and Dr. Mu Yuguang. Their efforts both *seen,* and *behind-the-scene* were essential, to stand the difficulties I met during my PhD, and their guidance was essential to build the reading, writing, thinking and presentation skills I need as a researcher. As a human being, I learned from them about being kind, supportive, and genuinely caring.

I am extremely indebted to Professor Thambipillai Srikanthan, the former chair, and professor Ong Yew Soon, the current chair of SCSE who helped me stand from my fall, gave me a scholarship, and extended it, to be able to continue my research with peace of mind. Special thanks go to my friend Supriya Sathyanarayana, who introduced me to prof. Thambipillai.

I am deeply thankful for the knowledge and advice provided by Dr. Lu Lanyuan which are not restricted to the thesis work, but also is extended to what is beyond my PhD.

I'd like to thank my wife, Nafisa, who actively participated in parts of my journey as a lab mate and collaborator and bore me in the rest as a caring wife.

I'd like also to acknowledge all my lab mates in SBS and SCSE, especially Dr. Wang YaoFeng, Mr. Zheng Liangzhen, Dr. Yaw Awuni, Dr. Chua KhiPin, Mr. NG Tze Yang Justin, Mr. Liu Yang, Dr. Ning Lulu for the fruitful discussions we had over the last 5 years during our group meetings and daily interactions.

[1] First verse of the Quran to be revealed was "**Read** in the name of your lord who created" (Quran 96:1).
[2] "Say, [O Muhammad]: (Travel through the land and **observe** how He began creation)." (Quran 29:20),
"Do they not **contemplate** within themselves?" (Quran 30:8),
"say, (My Lord, **increase me in knowledge**.)" (Quran 20:114).

Finally, I'd like to express my gratitude to my family (my mother, father, and sister) for they raised me since my early childhood, and their support didn't stop on personal, moral humanitarian, emotional, and even monetary level, until I finished my PhD work.

Dedication

This work is dedicated to my country Egypt, the first country in written history, and the gem of the old world, the link between Africa and Asia and their gateway to Europe. Peace upon the land of peace, from the islands of Tiran and Sanafir to the passageway of Essalloum. Peace upon the souls of everyone who strive to ensure the peace and unity of the land and people.

May whatever knowledge included in this thesis or can be built upon from this thesis benefit my country and the whole humanity and make the world better and free of pathogens. May my son and his generation, who are the future, live a better life.

Table of Contents

List of Figures

List of Tables

Abbreviations and Notations

3D	Three dimensions (X, Y, and Z)
Å	Angstrom
a	*Acceleration*
AMBER	Assisted Model Building with Energy Refinement
$ASoF_t$	Average Sums of Proximity relative Frequencies at time t
a_{To}	Overall-time acceleration
a_{Ts}	Search-time acceleration
B	QuickVina base (X1, X2, X4 …)
BE	Binding Energy
BFGS	Broyden-Fletcher-Goldfarb-Shanno method
C	Set of all available clusters
C’	Set of selected clusters out of all available ones
CG	Coarse Graining
CHARMM	Chemistry at HARvard Macromolecular Mechanics
D	Proximity cutoff radius
DENV	Dengue virus
DOFs	Degrees of freedom
E	Exhaustiveness level
ED	Essential Dynamics
ES	Electrostatic Force
F	Trajectory containing all frames
FBDD	Fragment Based Drug Discovery
Fr	Relative Frequency of occurrence of an event
Fr(i,j,t)	Relative frequency of the head of thread i to pass by any point of the history of thread j at time t
G	Global step/Global search
GROMACS	GROningen MAchine for Chemical Simulations
GROMOS	Groningen Molecular Simulation
H	Head (H hits) of ranked ligand list
H1N1	Influenza A virus subtype H1N1(Swine Flu)
I	Individual step/Individual search
MA	Memetic Algorithm
MC	Monte Carlo
MD	Molecular Dynamics
MRC	Multiple Receptor Conformation
MS-CG	Multiscale Coarse-Graining
N	Degrees of freedom (6 + number of rotatable bonds)
n	Number of running threads
NMR	Nuclear Magnetic Resonance
NP	Nucleoprotein
NS5	Non-structural protein 5
OPLS	Optimized Potentials for Liquid Simulations
P	Maximum number of checks to do
P	Number of processors running a parallel code
P_1	Maximum number of checks to do in the G step
p_1	Actual number of checks done in the G step
P_2	Default number of checks to do in the I step
p_2	Actual number of checks done in the I step

PDB	Protein Data Bank
PMF	Potentials of Mean Force
Pr	Probability
Q	Buffer Size
QM	Quantum Mechanics
QSAR	Quantitative Structure-Activity Relationship
R	Set of all rotamers/protonation state conformations
r	Pearson's correlation coefficient
RdRp	RNA-dependent RNA polymerase
RMSD	Root Mean Square Distance
ROC	Receiver operating characteristic
Ro3	Rule of three
S	Maximum number of search steps to undergo
s	Number of taken search steps so far
SASA	Solvent Accessible Surface Area
S_{MAX}	Maximum cell size (number of points a cell can contain)
SoF	Sum of proximity relative Frequencies
ss RNA	Single-stranded RNA
T	Total number of ligands
t	Time
$T_{0\rightarrow1000}$	Time taken to do steps $S_{0\rightarrow1000}$
T_H	Overhead Time
T_O	Overall Time
Ts	Search Time
VdW	Van der Waal's force
W_i	Cell width in dimension *i* (x, y, or z)
W_{MIN}	Minimum cell width in any dimension
WNV	West Nile Virus

Abstract

In a world where pathogenic bacteria, viruses, as well as cancer develop resistance to drugs on faster pace than discovering new ones, researchers bear the heaviest weight to design new drugs to overcome such phenomenon of drug resistance. Drug design is one of the most challenging tasks in computational and structural biology, which aims at developing new drugs or enhancing currently known drugs against certain diseases based on the knowledge of a biological target.

This thesis is about accelerating drug design work flow, through accurate acceleration of molecular docking tools, proper selection of candidates for Multiple Receptors Conformation (MRC) docking, and molecular dynamics (MD) simulation.

In this work, I first developed QuickVina 2, a fast, accurate, and reliable molecular docking tool that depends on the powerful scoring function of AutoDock Vina and accelerated search of QuickVina. QuickVina 2 was tested against the 195 protein-ligand complexes of the core set of PDBbind 2014, using default exhaustiveness level of 8. It successfully attained up to 20.49-fold acceleration over Vina with tendency for higher acceleration when the number of dimensions/variables increases. Meanwhile, 70% of its predicted modes were equal to or better than original Vina in terms of binding energy. The remaining 30% had average Energy difference only 0.58 Kcal/mol. The Pearson's correlation coefficient (r) between AutoDock Vina's and QuickVina 2's binding energy was 0.967 for the first predicted mode and 0.911 for the sum of all predicted modes. QuickVina 2 was found to be more accurate than GOLD 5.2 and is only slightly less accurate than Dock 6.6.

QuickVina 2 was employed to propose drug fragments for Dengue Virus Non-Structure Protein 5 (DENV-NS5), and the result was compared to AutoDock Vina result as a measure of double confirmation. Both QuickVina 2 and AutoDock Vina detected the same 13 fragments with slight differences in their estimated binding energies while QuickVina 2 detected three additional fragments. Two of the fragments were subjected to MD simulations for in silico validation. The simulation results suggest that the proposed ligands are plausible and could be considered for further computational

and experimental validation, as well as lead optimization. The work also involved refining the selection criteria of receptor conformation candidates that undergo MRC docking, in order to ensure diversity and increase sensitivity (decrease false negative rate) of detection.

QuickVina 2 was taken then to another dimension by enabling it to search wide search spaces, after introducing inter-process spatio-temporal integration between the searching threads to communicate their collective wisdom. That work resulted in the release of QuickVina-W, a tool suitable for Blind Docking. QuickVina-W explores four folds the number of points that Vina explores, in a more efficient way. It proved to be faster than QuickVina 2 (with average and maximum normalized overall time accelerations of 3.60 and 34.33 folds in relation to Vina versus 1.98 and 18.02 respectively), yet better than AutoDock Vina in terms of binding energy (78% of predictions with binding energy better than or equal to Vina) and RMSD (Root Mean Square Distance) to experimental data (with success rate of 72% by QuickVina-W versus 63% by Vina). It was based on the observation that the Average Sum of Proximity relative Frequencies (ASoF) of searching threads is ever increasing with search progression, and on the theory that allowing a searching thread to communicate with other nearby threads to make use of their wisdom, would increase the speed and sensitivity of that searching thread, in a way relevant to the increasing ASoF. This work monitored the ASoF and proved its direct relation to decision taking increased speed and accuracy which are reflected in turn on the search process.

Preface

Problem and Research Objectives

Today's world pathogenic bacteria, viruses and cancer develop resistance to drugs on faster pace than discovering new ones. To overcome such phenomenon of drug resistance, it is a necessity to accelerate the drug design process and to eliminate the false positive drug candidates. Previously, the drug design process was master-pieced by experts, where chemists design the drug manually. With complexity of biochemical systems, it becomes impossible for experts to develop and then deposit their domain knowledge to have a prefect design in a reasonable time; due to the curse of dimensionality.

To automate such process, computer-aided drug design field comes to exist. However, the state of the arts methods in this field involves computationally intensive and resources demanding processes. Therefore, the current computer-aided design approaches suffer from both low yield and high false positive rate together. Subsequently, it becomes the main interest for this thesis to accelerate the computer-aided structure-based drug design process in a high-quality way. Quality here refers to 1) decreasing both false positive and false negative results; 2) finding better results among different alternatives. This thesis focuses on the drug-receptor interaction modeling on two levels: protein-ligand docking and molecular dynamics simulation.

Contribution

The contributions done in this work can be listed in the following points:

- Replacing the inaccurate check condition in QuickVina which adapts the first-order-necessity-check with the more robust one which considers the first-order-consistency-check heuristic as well.

- Introducing a new, fast, and accurate molecular docking tool, QuickVina 2 that works well in practical environment of low computational resources., while maintaining significantly higher accuracy compared to the previous docking tool "QuickVina"
- Introducing an inter-process spatio-temporal integration that enhance both, the accuracy and the speed of the decision-making process across different searching threads.
- Formulating the relation between Average Sum of Proximity Relative Frequency (ASoF) and history guided search speed and accuracy.
- Introducing QuickVina-W as a blind docking tool, which enables docking a ligand to the whole surface of a protein without any prior knowledge of the target pocket.
- Refining the selection criteria of target receptor conformations used in multiple receptor conformation docking method that searches for a "true binder", to decrease its false negatives.
- Proposing 16 fragments for Dengue virus.
- Providing a showcase for QuickVina 2.

Publications and Presentations related to this thesis

Conferences

- **"Recent developments in drug design workflow management systems"**, Amr ALHOSSARY, KWOH Chee Keong, MU Yuguang. Oral presentation in "CDAMIES 2018".
- "**Proposing Drug Fragments for Dengue Virus NS5 Protein**". Amr ALHOSSARY, Yaw AWUNI, Chee-Keong KWOH, and Yuguang MU. Oral presentation in "GIW/Bioinfo 2017".

Articles

- "**Fast, Accurate, and Reliable Molecular Docking with QuickVina 2**". Alhossary, A., Handoko, S. D., Mu, Y., & Kwoh, C.K. (2015). Bioinformatics (2015) 31 (13): 2214-2216.
- "**Protein-Ligand Blind Docking Using QuickVina-W With Inter-Process Spatio-Temporal Integration**". Nafisa M. HASSAN +, Amr A. ALHOSSARY +, Yuguang MU, and Chee-Keong KWOH. Scientific Reports 7(1) (2017). DOI: 10.1038/s41598-017-15571-7. (Equally contributing 1st author)

- **"Molecular Dynamics and Simulation"**,
  Liangzhen Zheng, Amr A. ALHOSSARY, Chee-Keong Kwoh, Yuguang Mu, in *Encyclopedia of Bioinformatics and Computational Biology*. 2019. p. 550-566.
- "**Proposing Drug Fragments for Dengue Virus NS5 Protein**".
  Amr ALHOSSARY, Yaw AWUNI, Chee-Keong KWOH, and Yuguang MU. Journal of Bioinformatics and Computational Biology, (2018) 16 (3).

Thesis Synopsis

The thesis contains six chapters: A short summary and significant findings for each chapter is briefly summarized as follows.

- **Chapter 1:** Introduction to drug design

This chapter provides an overview on computer-aided drug design process and its different steps.

- **Chapter 2:** Practical and accurate acceleration of AutoDock Vina

This chapter presents a novel molecular docking tool *"QuickVina 2"* where the docking process is more accurate given low levels of exhaustiveness compared to QuickVina. As such, it is useful to accurately accelerate protein-ligand docking in small known search box.

- **Chapter 3:** Refining selection methodology before MRC docking to find a binder for Dengue Virus NS5 protein

This chapter applies *QuickVina 2* for virtual screening of drug fragments against dengue virus non-structural protein 5 (DENV-NS5). The chapter also describes how to decrease the false negative rate in the multiple receptor conformation step through proper selection of the receptor conformations that go for multiple receptor conformation docking. The chapter proposes 16 drug fragments for DENV-NS5.

- **Chapter 4:** Blind Docking with inter-process spatio-temporal integration

One problem of the newly designed drugs is the off target where other receptors or other cavities within the same receptor may fit the designed ligand better than the target cavity. Blind docking, the

mechanism by which the ligand is docked against the whole receptor surface, could be utilized to avoid such cases and to ensure that the ligand would bind to the target cavity most of the time.

This chapter takes docking to a new dimension, where a ligand is docked against the whole surface of the target protein without prior knowledge about the target cavity or without any preference to a specific cavity if such knowledge exists. In this chapter, QuickVina 2 was further developed into QuickVina-W after adding the interprocess spatio-temporal integration between the searching threads.

- **Chapter 5:** Future work

This chapter talks about future directions this work can go through. First I talk about extending QuickVina-W to distributed system memory architecture. Then, about applying molecular dynamics concepts in multiple level coarse graining molecular dynamics simulation to allow for the study of macromolecular allosteric modifications or large conformational changes (e.g. protein folding / unfolding), using a faster method than the molecular dynamics

- **Chapter 6:** Conclusion

This chapter includes a final summary of the thesis and its contributions.

Chapter 1 Introduction to drug design

Drug design is one of the most challenging tasks in computational and structural biology. It is both a science and art, in the sense that it employs physical principals as well as scientists' own intuition to select and apply best suitable available methods. It is an inventive process, which aims at developing new drugs or enhancing currently known drugs against certain diseases based on the knowledge of a biological target [1]. This chapter introduces the principals and main components of the computer-aided drug design, the different approaches that handle the process, and the most famous tools commonly used in every step of that process.

1.1. Drug design principals

There are two principles widely adapted in drug design, which allow a drug to function: 1) "***inhibiting a pathological pathway"*** by inhibiting the drug target molecule, and/or to 2) "***restoring the normal pathway"*** by promoting specific normal pathway targets that were suppressed in the pathological condition. Although the term ***"Drug Design"*** is commonly used, this term is not 100% accurate. The more accurate term is ***"ligand design"*** (i.e., design of a small molecule that binds tightly to its target) [2].

1.2. Approaches for Drug Design

There are two main approaches for Drug Design. They are "Ligand-Based Drug Design" and "Structure-Based Drug Design".

1.2.1. Ligand-Based Drug Design

This approach requires prior knowledge about ligands known to interact with the target of interest. From several interacting molecules, a model of the receptor can be built, and the minimal necessary structural characteristics a molecule must possess in order to bind to the target (the pharmacophore) [3, 4] can be identified. A Quantitative Structure-Activity Relationship (QSAR) between properties of

molecules (e.g. number of Carbon atoms in a branch) and their experimentally determined biological activity can be derived as well. Then all these relationships can be used for predicting what the properties of a new molecule would be.

1.2.2. Structure-Based Drug Design

This approach, on the other hand, requires prior knowledge of the 3D structure of the active site on a target molecule. Such knowledge can be obtained experimentally via X-Ray crystallography or Nuclear Magnetic Resonance (NMR) spectroscopy [5]. That 3D structure can be also predicted using homology modelling on experimentally determined structure of a related molecule.

Having both the 3D structure of the receptor from experiment or homology modelling and the structural dynamics and electronic properties of the ligand from calculations, two general strategies of structure-based drug design can be identified. The first strategy is “de novo” drug design, where a drug is built de novo from groups, molecular fragments, or individual atoms in an iterative process. The second strategy is “virtual screening” of large drug or drug-fragment databases to find suitable ligands for a specific target. The main advantage of de novo strategy is the potential to find new molecules, although they may be hard to synthesize. In contrast, the main advantage of the virtual screening strategy is the chemical feasibility and commercial availability of the proposed products, in contrast to the molecules proposed by completely de novo synthesis process.

1.2.3. Putting all pieces together

Before any drug design process can be initiated, it is a necessity to identify a target molecule to be addressed, through a systematic study of the pathway. Only then, the drug design process can be initiated.

The design workflow is the outcome of combination of docking, linking, and growing of fragments, MD Simulation, and lead optimization. A hybrid approach that combines both the knowledge of the receptor structure from X-Ray crystallography or Nuclear Magnetic Resonance

(Structure-based) from one side and the prior information about ligand interactions and pharmacophore (ligand based) from the other side can be also applied, as shown in Figure 1-1 [6].

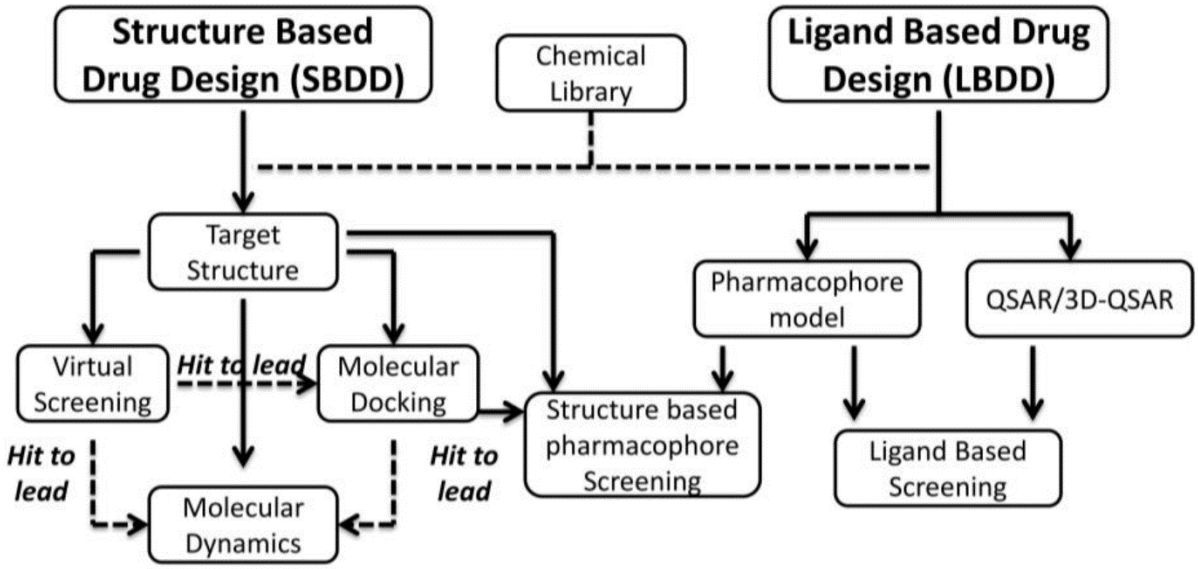


Figure 1-1 Hybrid approach of drug design
Adapted from Cozza, G., The Development of CK2 Inhibitors: From Traditional Pharmacology to in Silico Rational Drug Design. Pharmaceuticals (Basel), 2017. 10(1).

The drug design process occurs in an iterative manner, which starts usually with screening for suitable fragments and then use the found fragments as seeds in the next iteration of de novo process [7, 8]. Please refer to section 1.3 for illustration.

1.3. Computer-Aided Structure-Guided Drug design workflow

The general workflow is summarized in Figure 1-2 [9], illustrated using standard flowchart shapes: Processes are represented as rectangles, divisible processes are shown as double bounded rectangles, data in grey: intermediate data is shown as gray parallelogram, final data (documents) shown as gray document shape. Flow control shapes are shown in blue: decision is Blue diamond and start/end in blue ellipsoid.

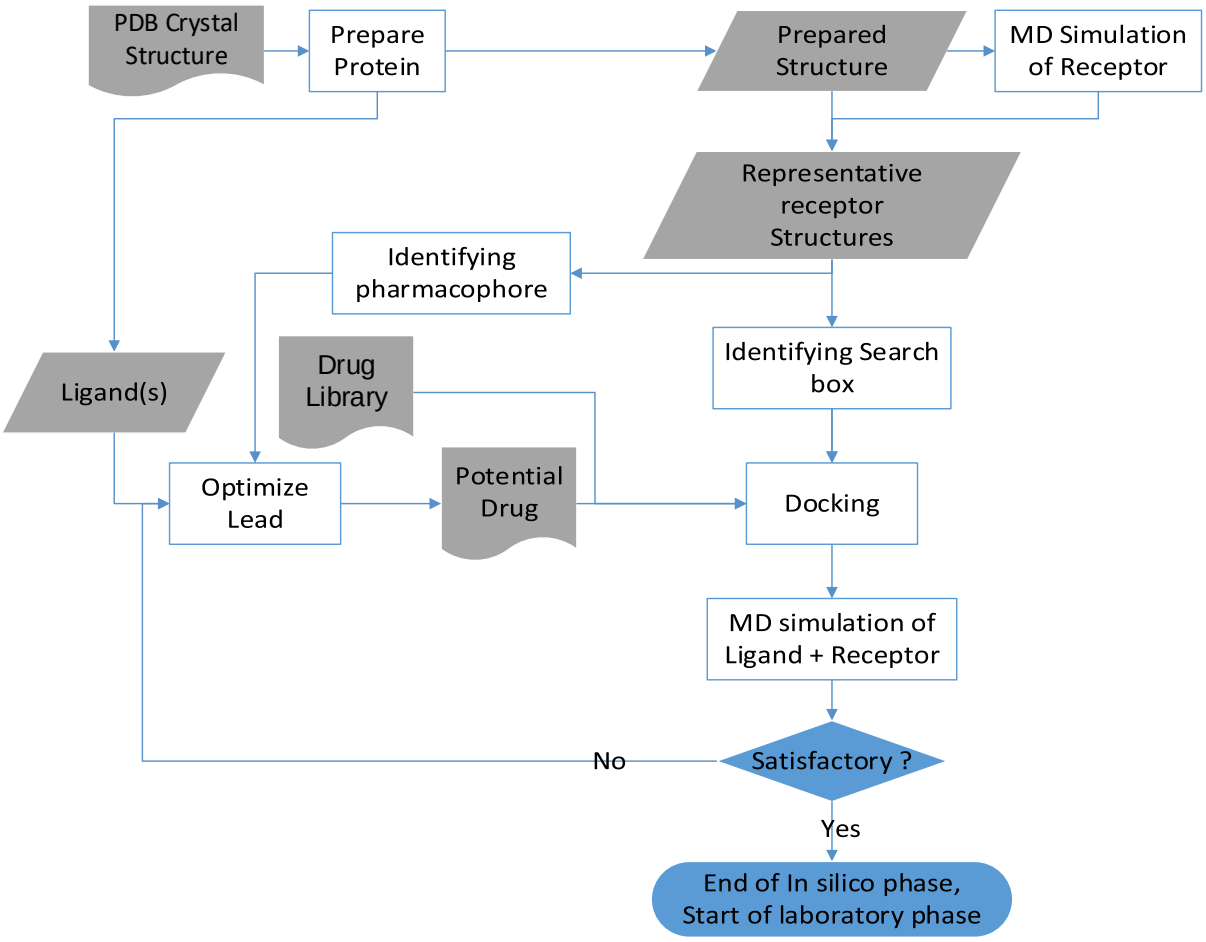


Figure 1-2 Computer-Aided Structure-Guided Drug Design Workflow

The workflow starts with the crystal structure of the target protein, which may or may not contain bound ligand(s). If a bound ligand is present, it may be composed of one single molecule or of smaller fragments.

First step (preprocessing) is protein preparation, to model missing atoms and extract present ligands. Then, molecular dynamics (MD) simulations of the receptor alone (without the ligand) are used to predict any other representative structures of the receptor, in addition to the crystal structure, to account for the flexibility of the protein. Based on the receptor structure, the pharmacophore and the search space can be identified. The pharmacophore is used with the help of any found ligands to develop candidate molecules. These potential molecules are then docked against the target to estimate their affinities and are ranked according to the docking scores they receive. Potential drugs

with highest scores are then subjected to another round of MD simulation with the receptor to simulate their binding behavior. If they show satisfactory behavior (e.g. in terms of association time), they are accepted and are sent for experimental validation, otherwise they are used as leads or seeds for a new cycle of lead optimization, docking, and MD simulation. The iterative process continues until satisfactory results are found.

1.3.1. Protein Preparation

Protein structures presented in PDB format are usually not complete. They can be missing some atoms, residues, or even complete loops. Hydrogen atoms are usually missing because they are beyond the precision of current crystallographic techniques. Most of other missing heavy atoms are from areas of lack of spatial certainty. Therefore, protein structures in PDB usually need pre-processing to prepare for the next steps. Pre-processing includes:

1) Adding missing atoms/residues.

2) Protonation state prediction. States and positions of Hydrogen atoms are easily predictable, except for Histidine, whose protonation state depends on its microenvironment.

It is important to be mindful of the protein's oligomeric state to avoid wasting the effort in searching for putative drugs that can tackle protein surface patches, while they are not accessible as they are buried in the biologically active complex.

1.3.2. Identifying Pharmacophore

Pharmacophore is the minimal necessary structural characteristics a molecule must possess in order to bind to its target. In this step, researches identify (or predict) the ligand atom types needed to interact with receptor along with their relative positions. This process is highlighted in Figure 1-3.

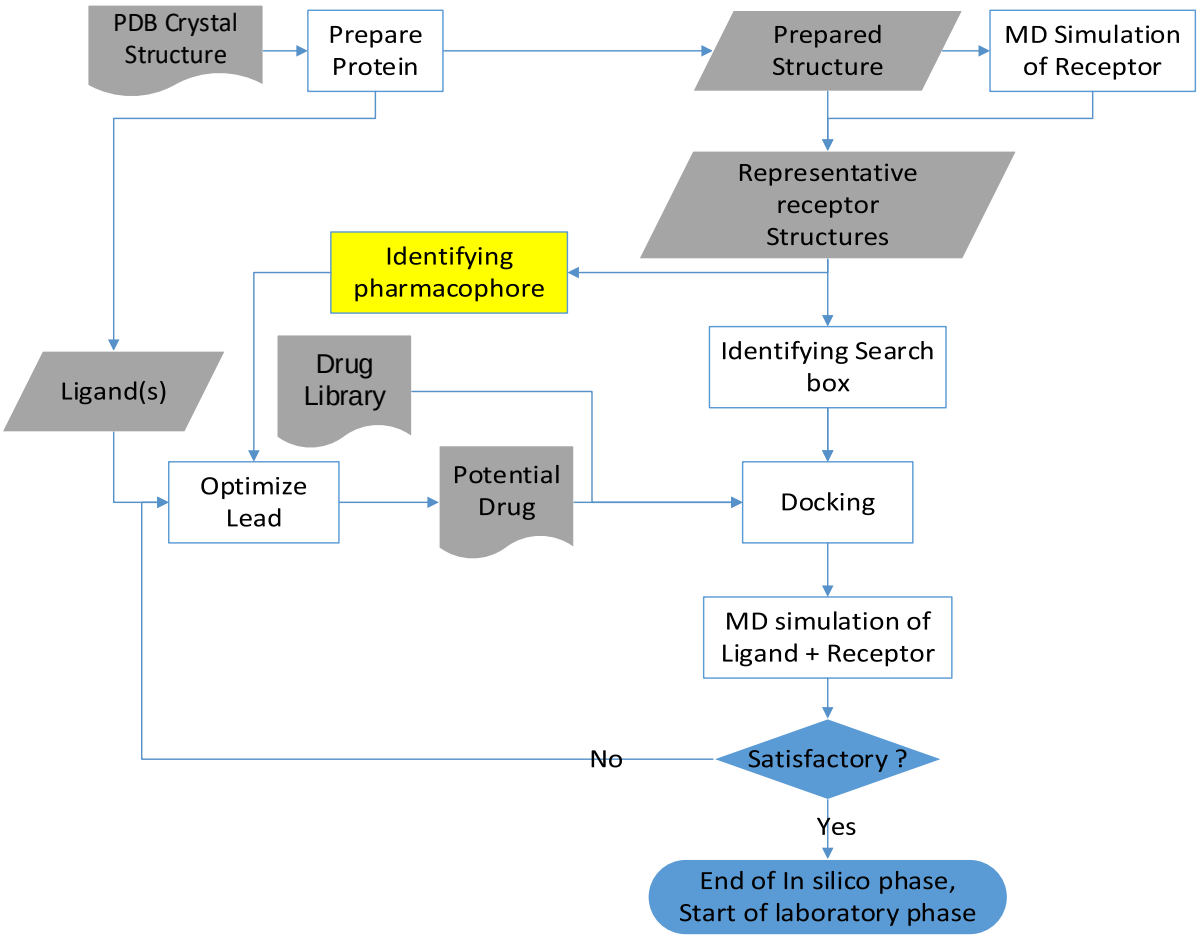


Figure 1-3 Identifying Pharmacophore in the detailed workflow

To identify (or predict) the ligand atom types needed to interact with a receptor along with their relative positions, both the ligand and the receptor atoms need to be classified into:

1) Hydrophobic atom: All carbons in hydrocarbon chains or in aromatic groups.
2) H-bond donor: Oxygen and nitrogen atoms bonded to hydrogen atom(s).
3) H-bond acceptor: Oxygen and sp^2 or sp hybridized nitrogen atoms with lone electron pair(s).
4) Polar atom: Oxygen and nitrogen atoms that are neither H-bond donor nor H-bond acceptor, Sulphur, Phosphorus, halogen, metal, and carbon atoms bonded to heteroatom(s).

Virtual atomic probes of the four types scan the ligand binding region, in order to predict which chemical fragments can be put into which corresponding spots in that region [10-12].

1.3.3. Identifying Search space

This step lies in the position highlighted in Yellow in Figure 1-4. It aims at predicting where hypothetically a drug could be introduced, so that it binds efficiently and with a high affinity. First, the Solvent Accessible Surface Area (SASA) needs to be identified, then any present cavities (also known as pockets), and search space boundaries can be identified.

There are generally two methods to identify Solvent Accessible Surface Area (SASA) and any existent cavities: Geometric and Energy calculations. The geometry-based methods are either grid-based or grid-free.

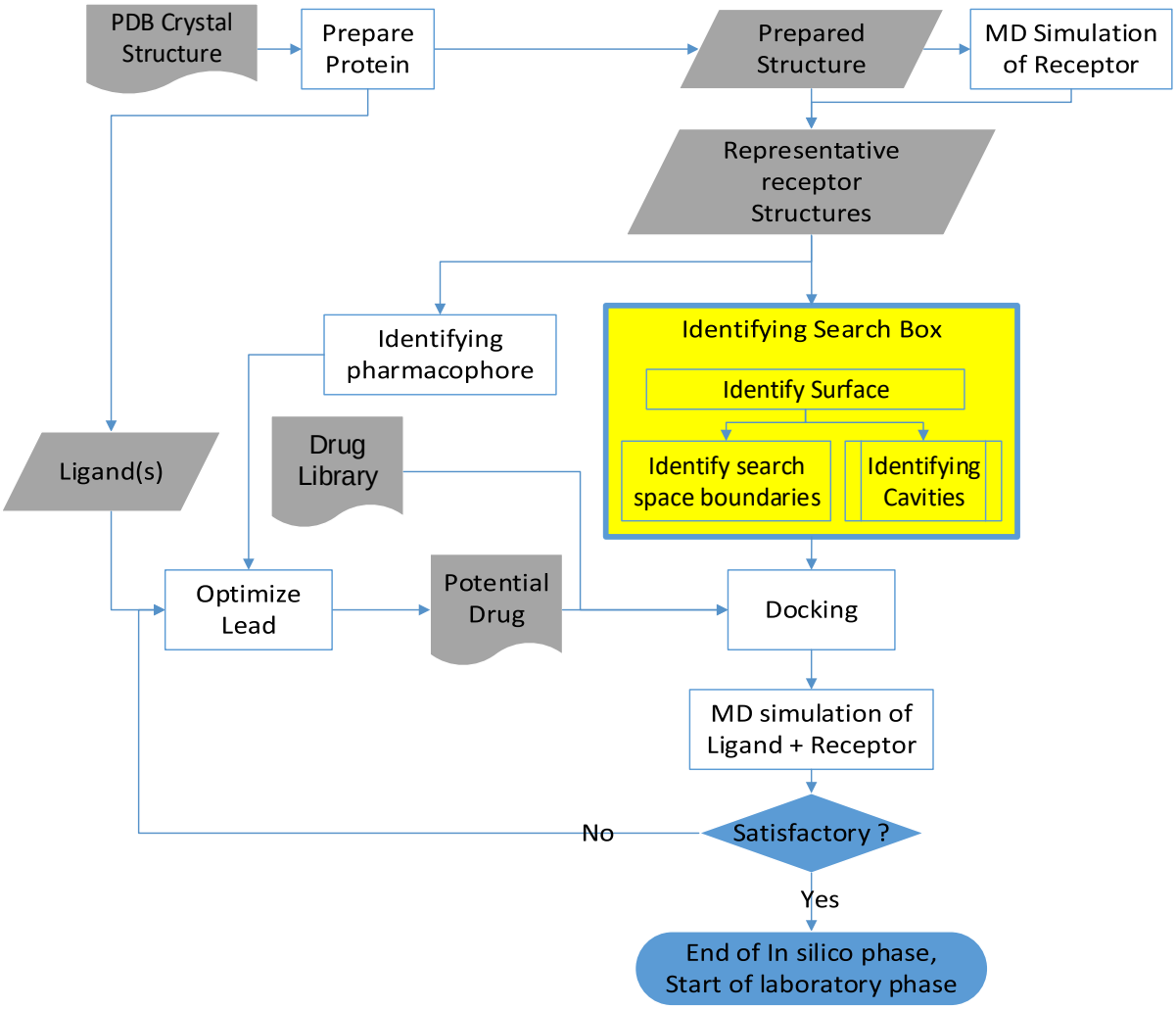


Figure 1-4 Identifying Searchbox position in the workflow

Grid-based approaches search within a 3D grid covering the protein for points with specific criteria like searching for surface-solvent-surface events. Examples of software that employ this approach are LIGSITE [13] and POCKET [14]. On the other hand, most grid-free approaches tend to probe the surface and find the largest sphere (or largest cluster of small spheres) not containing any atoms inside, and to aggregate smaller regions into bigger ones. Examples of software packages that adapt this tactic include SURFNET [15], and FPocket [16, 17]. Another new and simple yet efficient grid-free approach provides a quantitative method of likelihood that certain atom/residue is important for cavity by simply calculating the burial of atoms within a protein. It is implemented in DEPTH [18].

1.3.4. Molecular Dynamics Simulation

Molecular Dynamics (MD) simulation is a computer simulation of movements (kinetics) and interactions of atoms and molecules for a certain period of time. Molecular dynamics simulation is present in two positions in the workflow. They are highlighted in Yellow in Figure 1-5. The first position is the MD simulation of the receptor alone. Because the PDB structure is a single static conformation while the protein is a dynamic object, molecular dynamics (MD) simulations are applied to find other representative conformations of the receptor that may exist in addition to the crystal structure present in PDB. The second position is the MD simulation of the ligand-receptor complex, to study the ligand binding behavior. Some of the most common MD simulation packages are GROMACS [19, 20], AMBER [21], NAMD [22], LAMMPS [23], ACEMD [24], and ADUN [25].

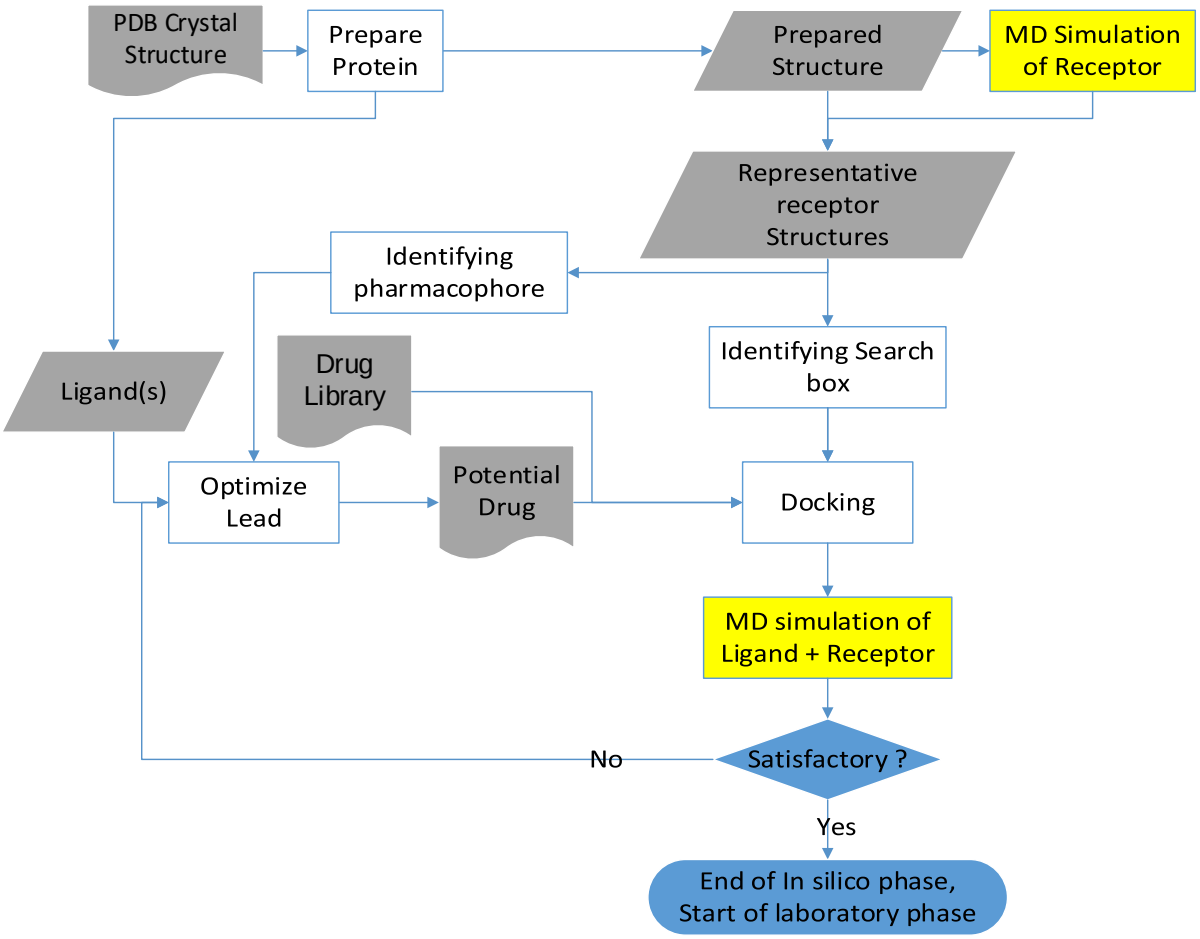


Figure 1-5 MD simulation in the workflow

Most of the interactions modelled in normal MD simulation are the non-covalent interactions, while covalent interactions, in terms of making and breaking covalent bonds, can be modelled using quantum mechanics (QM) simulation instead. The general aim of MD Simulation is to predict how a system would behave in a certain environment, using (usually) deterministic methods based on potential energy calculation iteratively on repetitive time step that has to be small enough (i.e. smaller than the fastest vibrational frequency in the system), to avoid discretization errors. The potential energy is dependent on the positions of all the particles of the system. Because it is too complicated to be solved analytically, it is usually solved numerically.

Systems that have energy barriers that separate regions of configuration space are liable to suffer from poor sampling. *Umbrella sampling* is meant to bridge that gap, in which a potential that is

not affected by the energy barrier influence is chosen, instead of the standard Boltzmann weighting used in Monte Carlo sampling.

1.3.4.1. Numerical Methods

Numerical algorithms were developed to model the equations of motion. Verlet algorithm [26] was one of the earliest algorithms used for modeling. It depends on relative conformations "r" and acceleration "a" at a certain point of time "t" or a previous point "t- δt", more precisely they depend on r(t), r(t- δt), and a(t), to produce a deterministic simulation trajectory. It is noteworthy that the velocity does not appear in Verlet's equations, although it is important for determination of the kinetic energy at a certain point in time. Calculating velocity and kinetic energy at certain time would add more overhead calculations. Several varieties of modifications were later developed on the original Verlet's algorithm to provide more views and more efficient calculations. They include Leapfrog algorithm (simplified Verlet) [27], Velocity Verlet [28], and Beeman's algorithm [29]. Yet, the most common disadvantage of all these methods is that they depend on the interaction between all atoms in the system, which leads to high exponential order of calculation time complexity of the system.

1.3.4.2. Limitations of classical MD Simulation methods

Current all-atom molecular modeling technique enables biological systems description with details limited to simulation times and system sizes less than 1000 ns and 10 nm respectively. On the other side, some biological processes, (e.g. protein folding), occur on time and length scales much larger than these limits, which yield them unmodelable.

In addition, size scaling is highly computationally expensive, due to large number of additional atoms (either solvent or biological material) added to the system, as well as the interactions between them which are scaled up by $O(n^2)$.

These limitations are dealt with using several strategies, including implicit solvent, reduced pairwise non-bonded calculations, Coarse Graining, and Multiscale Coarse Graining. They are

described below. Also there are several enhancements in the Force Fields used in MD simulation described in section 1.3.4.6 (Force Fields used in MD Simulation).

1.3.4.3. Exhaustive calculations vs. reduced calculations

The most CPU-intensive task in MD simulation is the evaluation of non-covalent component of energy potential. This is because the non-covalent component is non-local. That leads to interacting between all the particles in the system. If all pair-wise electrostatic and van der Waals interactions are considered, the computational load in this case is scaled by $O(n^2)$. It can be reduced if we consider methods that depend on Particle Mesh (e.g. Particle mesh Ewald method [30] and Particle-Particle-Particle Mesh (P^3M) method [31]) which reduce the cost to $O(n/\log(n))$, or simply by applying some cutoff techniques which reduce the cost to $O(n)$. Cutoff can be either by truncation, switch, or shift formulations. The simplest is truncation, while the most stable is shifting potentials.

N.B.: it is important to differentiate here between two concepts:

1) Exhaustive calculations, where "all atoms" are included in the calculation. That means no cutoff is considered (mentioned in this section).
2) "Atomistic modelling" (or "all-atoms model"), which means the simulation is carried on at atomic level, not on coarse grain (e.g. group, residue, etc.) level described in section 1.3.4.5 Coarse Graining (CG).

1.3.4.4. Explicit Solvent vs. Implicit Solvent

To simulate a system including a solvent, there are two types of approaches to model the solvent: the explicit solvent model, which represents every particle of the solvent (e.g. TIP3P, SPC/E and SPC-f water models), and the implicit solvent model, which employs the mean force of the solvent to model its effect. The Explicit solvent model is computationally much more expensive than the implicit model (roughly ten times more particles), but on the other hand, it secures modeling the fine details of kinetics of the solvent.

1.3.4.5. Coarse Graining (CG)

Coarse Graining is developed for bridging the "time-scale" and "length-scale" gaps between computational and experimental methods. The general contract of all CG methods is representing a system by a reduced number of degrees of freedom (DOFs) and fine interaction details, through mapping from atomically detailed configuration into a CG configuration called super atom [32]. That reduces the system size about **ten** folds and allows time steps up to **25–50** fs [33]. This way, the simulation would go faster and require less computational resources than the all-atom representation while achieving an increase of orders of magnitude in the simulated time and length scales.

The mapping process depends on the size and complexity of the biomolecules. Small and low details molecules mapping is straightforward while the mapping process of large and complex molecules is a challenging one. There are two general CG approaches: the residue-based CG [34], and the shape-based CG [35]. Zhang et al. designed a method called "essential dynamics coarse-graining (ED-CG)" to define CG sites from protein primary sequence that reflect the essential dynamics characterized by PCA of an atomistic trajectory [32].

1.3.4.5.1 Multiscale Coarse-Graining (MS-CG)

The multiscale coarse-graining method (MS-CG, also known as "force matching") is a method of decreasing the computational load of simulating a system by "optimizing a CG potential to reproduce many-body potentials of mean force (PMF) calculated from atomistic configurations" initially for nonbonded potentials and then for bonded interactions [36].

Instead of the commonly used two-body approximation, Larini et al. in 2010 proved that using explicit three-body potentials is superior to the two-body approximation [37].

LY Lu and G. Voth in 2009 implemented B-spline basis function in the MS-CG method, which shows dramatic reduction of memory requirements and increase in computational efficiency of the MS-CG calculation. Their work showed that the MS-CG force field can approximate the many-body PMF in the coarse-grained coordinates [38]. Then in 2011, they showed that MS-CG method can also

be used to analyze the CG interactions from atomistic MD trajectories via PMF calculations and its performance is comparable to various free energy computation methods [39]. The distribution function defined in the coarse-grained potential could be reproduced by iteratively applying the MS-CG algorithm [36]. It has been proven that both of the two common methods ("post-processing method" and "fixed part method" [40, 41]) can similarly make MS-CG potentials which are limited in accuracy only by the incompleteness of the basis set, usable [42].

Finally, it is worth noticing that it is necessary to intervene in the variational calculation in some way, in case we need to produce accurate MS-CG potential in both high and average potential energy regions [42].

1.3.4.6. Force Fields used in MD Simulation

One essential requirement for any simulation package is the definition of a potential function, which describes how the simulated particles in the system would interact. Such function is known as the Force Field. Force Fields are usually empirical. They usually employ preset bonding arrangements, which makes them capable of modeling structural and conformational changes but not chemical reactions (bond breaking and reformation) explicitly. Most famous classical force fields used for modeling of **Macromolecules** are GROMOS (Groningen Molecular Simulation) [43, 44], AMBER (Assisted Model Building with Energy Refinement) [45], and CHARMM (Chemistry at HARvard Macromolecular Mechanics) [46]. CHARMM can also be used for **small molecules**. OPLS (Optimized Potentials for Liquid Simulations) [47, 48] is used to model **liquids** and it can as well be used in **Energy minimization**.

1.3.4.6.1 Pair-potentials vs. many-body potentials

Force field potential function can be either pair-potential function, where the force is calculated as a sum of pairwise interactions between particles (e.g. Lennard-Jones potential of Van der Waals force and Born (ionic) model of the ionic lattice), or many-body potentials, where three or more particles

interact with each other. Examples of the later are Tersoff potential [49], embedded-atom method (EAM) [50], and Tight-Binding Second Moment Approximation (TBSMA) [51] potentials.

1.3.4.6.2 Static partial charges vs. Polarizable potentials

Most force fields (especially experimental Force fields) model the effect of polarizability as static partial charges, while new techniques model polarizability as fluctuating charges, electronic structural theory, induced dipole, distributed multipoles, point charges, and Bond Polarization Theory (BPT). Increased accuracy was achieved using fluctuating charges in water molecules [52] and with some promising results for proteins [53].

1.3.4.6.3 Coarse-Grained Force Fields

Coarse-Grained Force Fields can be based on generic atom types (e.g. MARTINI Force Field), or on secondary structure (e.g. VAMM).

MARTINI [54, 55] is a very common Coarse-Grained Force Field, based on having four categories of beads (Q (charged), P (polar), N (nonpolar) and C (apolar)) with several subtypes up to a total of 20 bead types. On the other hand, VAMM (Virtual Atom Molecular Mechanics) [56] is a "knowledge-based" force field developed to model large scale conformational transitions based on the virtual interactions of C-alpha atoms and on secondary structure and residue specific contact features.

1.3.4.6.4 Finite Element Method

Finite element method is a common method of numerical solution of complex problems including force and heat acting on irregular structures. It is mostly used for structural mechanics applications. In this method, the system is modeled as a set of finite elements of appropriate properties (e.g. stiffness, toughness, etc.), interconnected at points called nodes.

Although this method is famous for use in civil applications to solve displacement versus boundary and load conditions, it can be applied on smaller scale objects with high efficiency in terms of speed and accuracy. For instance, in 1999 O'Brien et al. simulated different fractured objects of

different toughness property after collisions, using continuously remeshing technique and it was comparative to the experimental results [57].

Even on the molecular levels there have been promising trails to combine FEM and MD simulation to model silicon [58] and laser induced pressure wave propagation [59]. In 2013, Lee at al. published a multiscale modeling technique for bridging molecular dynamics with finite element method, based on weighted average momentum principle. Their work was applied on 2-D problems, but to the best of our knowledge, no 3D applications are available so far [60].

1.3.4.7. Clustering the resulting trajectories

The output of MD simulation process is a trajectory of states (frames). In several cases – especially for docking purpose- they need then to be clustered into a relatively small number of groups along with representative conformation for every group.

1.3.5. Lead Optimization

This step enhances an already existing drug (a lead drug), through adjusting some chemical and pharmacokinetic properties (e.g. solubility), aiming at augmenting / optimizing some desired features (e.g. affinity and specificity) and suppressing some other undesired or harmful features (e.g. toxicity). In addition, this step aims at increasing the bioavailability of the proposed ligand. The details of the lead optimization step are highlighted in Figure 1-6.

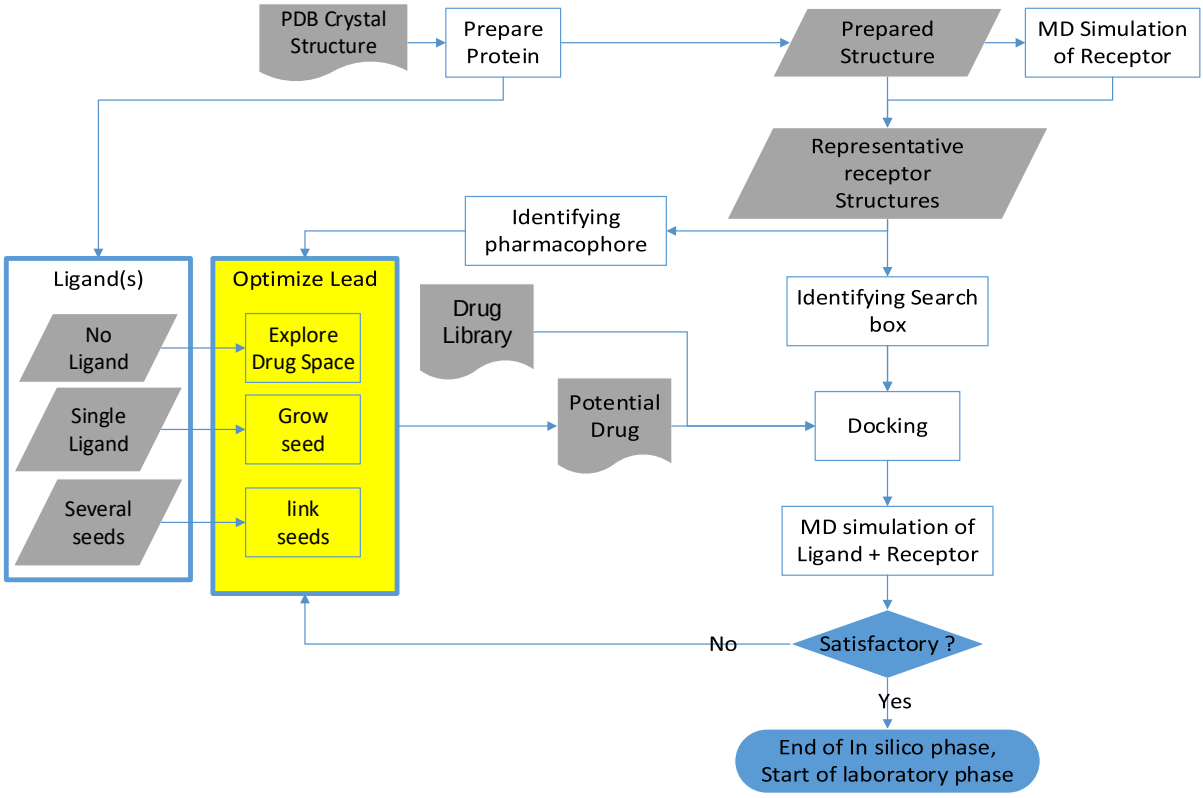


Figure 1-6 Lead optimization position in the workflow

The usual case is having only one particle ligand (seed) at a time, upon which the process starts. This is referred to as “Growing the seed”. However, sometimes more than one seed exist together (more than one particle in the pocket simultaneously), in such a case, we would need - also - to join them together in one new molecule. This is referred to as “Linking the seeds”. Generally, both growing and linking strategies are combined to make the process more robust.

In the unfortunate case of not having any available seeds at all, the whole drug space would have to be explored to search for any suitable ligand. In this case, a logic start point is putting a hypothetical seed in the form of H or C atom in the target pocket and using it to start searching for any suitable drug.

Ligand design tools employ different combinations of the approaches indicated above. A single tool may use one or more of these approaches. For example, PRO-LIGAND [61] and SPROUT [62] use “growing” and “linking” approaches. In addition to using growing and linking approaches,

LigBuilder [63, 64] use “mutation” approach as well. While Autogrow [65] uses “growing”, “docking” and “evolutionary techniques”.

1.3.6. Molecular Docking / Virtual Screening

A ligand library is a large library of small chemical compounds (fragments); each fragment can be tested to fit in a part of the pocket. Chemical supplies companies usually provide their collections in the form of digital structure files (e.g. PDB, MOL2) for testing *in silico,* in a process known as “Molecular Docking”, before clients get to purchase their needed chemicals to test them experimentally. This form of libraries is known as a “Virtual Library”.

Molecular docking process position in the workflow is shown in Figure 1-7. It aims at performing *in silico* prediction of the modes and affinities of non-covalent binding between the target receptor and the small ligand under examination. Sometimes, instead of starting with a single virtual molecule, a virtual library is scanned to search for the best ligand-receptor interaction mode. This process is known as “virtual screening”. “Virtual Screening” is defined as “automatically evaluating very large libraries of compounds using computer programs” [66].

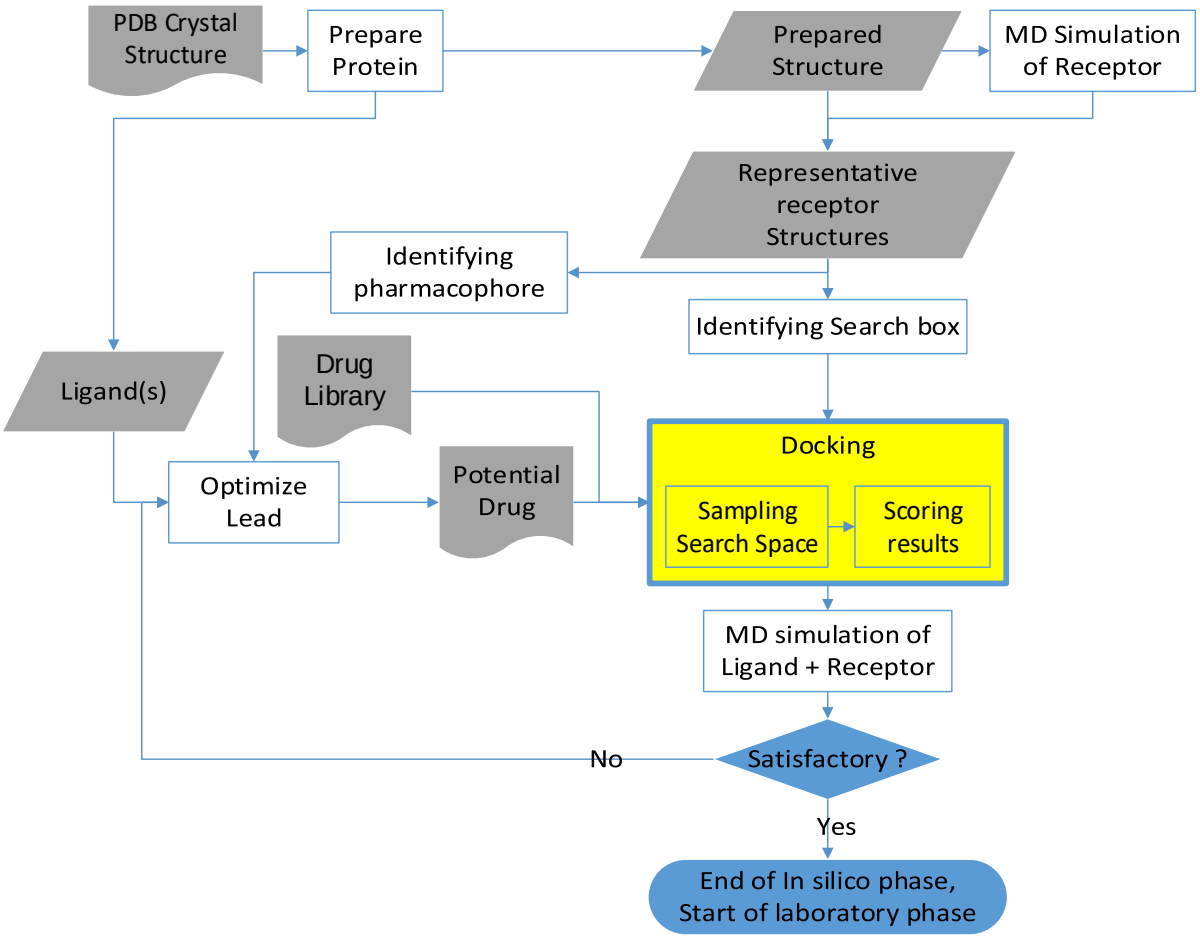


Figure 1-7 Docking in the workflow

The docking process includes two main components: 1) sampling the search space to find plausible conformations of the complex, and 2) scoring those plausible conformations to rank them based on estimated relative binding affinities [67].

Cross et al. performed a comparison between several docking programs, (namely UCSF Dock [68-73], FlexX [74], GLIDE [75, 76], ICM [77], PhDOCK [78], and Surflex [79]), to measure both docking poses and virtual screening accuracy against a collection of 68 diverse, high-resolution X-ray complexes. The result was that ICM, GLIDE, and Surflex generated ligand poses closer to experimental poses more frequently than the others did, based on the mean Receiver operating characteristic (ROC), Area Under Curve (AUC) and ROC enrichment values, obtained for the 40 protein targets in the Directory of Useful Decoys (DUD) [80, 81]. The study showed also that GLIDE and Surflex outperformed the other docking programs for virtual screening [82]. In a recent study, Wang et al. performed a

comprehensive evaluation of ten famous currently available docking programs, including five commercial and five academic programs. Wang et al. studied their accuracies of binding pose prediction (sampling power) and their binding affinity estimation (scoring power) and concluded that AutoDock Vina [83] has the highest scoring power among them [84].

1.3.6.1. Sampling Search Space

The search space is the combined space that spans over all possibilities a receptor-ligand complex can have. That includes all the degrees of freedom of ligand translation in the three dimensions and rotation around the three-dimensional axes, in addition to the torsion angles of all the flexible bonds that may exist within the ligand or within the receptor itself. Enumerating the search space is the process of searching for all poses that a receptor-ligand complex can have. Every "possible solution" can be conceptualized as a "**point** in the conformation space". Because the size of the solution space of any real-world docking problem is too large to be enumerated exhaustively, enumerating the search space is an unpractically time-consuming, computationally intensive, process. Evaluation of the objective function being optimized (i.e. the scoring function) is usually the bottleneck of any docking or virtual screening process.

Several techniques are employed to decrease that computational load, by splitting the objective function into several terms and pre-calculating them, to decrease the calculations time (e.g. grid pre-calculation), and/or by using a more efficient stochastic sampling strategy (e.g. genetic algorithms, Monte Carlo, and ensemble docking) that limit the effort exerted on function evaluation to important areas only. Ensemble Docking for instance solves the challenge of model selection, using a methodology that avoids the computationally expensive process of *sequential* docking of ligands into multiple protein structures.

Glide [75, 76] and eHits [85, 86] are examples of the docking tools that use exhaustive search, On the other hand, GOLD [87, 88], AutoDock [89-91], FlipDock [92] and FITTED [93] are examples of the docking tools that use genetic algorithm, while AutoDock Vina [83], ParDOCK [94], Rosetta Ligand

[95], and GlamDock [96] use Monte Carlo algorithm. GOLD [87, 88] in addition to Genetic Algorithm, uses ensemble docking for flexible docking.

1.3.6.2. Scoring Results

The aim of this integral step is to define a single one-dimensional measure to rank and compare the ligands according to their predicted binding affinity, over different poses, and hence be able to select the most suitable ligands. The scoring function assigns a relative value, which usually reflects an approximation of the physical forces, which contribute to the binding energy, to every pose. However, all scoring functions are just approximations of the real values. They are affected most of the times by several factors. For instance, the scores predicted by Vina family are affected by the molecule size (i.e. the bigger the molecule, the higher the score, even if it does not possess high affinity).

1.4. Challenges and limitations in Drug Design domain

In this section, I will focus only on the challenges related to drug-receptor interaction.

1.4.1. Molecular Docking

Docking is based on two main steps (search space sampling and scoring) and additional considerations:

1.4.1.1. Search space sampling

It is a computationally intensive and time-consuming task for current computers. The computational load increases exponentially with increased degrees of freedom in the ligand and/or the receptor. Although exhaustive search is replaced in most of the currently available tools with modern intelligent search techniques that consider ligand flexibility, more research and development for the docking domain is still required with state of the art problems, especially:

- Modelling receptor flexibility. As the curse of dimensionality suggests, the load increases exponentially with the increased number of degrees of freedom (DOFs). With receptor flexibility, every single side chain change includes several additional degrees of freedom, and with main

chain flexibility, every change involves several changes in several side chains, which in turn would lead to interacting, highly dimensional search optimization problem.

- Blind docking. In addition to high dimensionality problems mentioned earlier, blind docking is an example of a high dynamic range problem, where the whole surface of a receptor is searched for suitable docking poses without prior knowledge or preference to any specific pocket.

1.4.1.2. Scoring

Scoring is usually based on an approximation functions, meant for dimensionality reduction into a single dimensional value that can be used to relatively order the poses found during the search space sampling step. The (approximate) scoring function is commonly affected by several factors. The molecular size (in terms of number of heavy atoms) is one of them. One other important challenge is whether - and how - to incorporate data from different predicted poses into a single scoring value or to be confined with the best predicted pose only.

1.4.1.2.1 Multiple scoring operations:

- Multiple Receptor Conformation (MRC) Docking is when several conformations are known to exist for the same receptor. It is always more efficient to cache and reutilize whatever common calculations that are common between different conformations than to recalculate everything from scratch. Additionally, not all of the existing conformations can bind to a given ligand, due to mechanical hindering or any other reason. These non-binding conformations are considered “noisy”. They are not important, and they don’t benefit – and even may harm - the MRC docking process. It is also useful to filter out such noisy conformations before starting the MRC process. In Chapter 3 [Refining selection methodology before MRC docking to find a binder for Dengue Virus NS5 protein], I present an example for this case.
- Virtual Screening (VS), on the other hand, is when the same receptor has multiple candidate ligands to be docked against. A virtual drug library may be so big and diverse. In addition, ligands can be exceptionally large, with a big number of degrees of freedom. This situation raises the need

for even more efficient sampling algorithms, hopefully which can cache, hash, and recall any calculated *common subdomains* among several *diverse* ligands.

1.4.2. Molecular dynamics simulation

MD simulation is the most computationally intensive step in the whole workflow. It is an $O(N^2)$ problem where N is the total number of system atoms [97]. This is particularly slow in 2 cases:

- When the system has a lot of solvent molecules surrounding a (huge) receptor. That implies having an extremely big number of solvent molecules filling the simulation box.
- Where the receptor undergoes large allosteric modifications, e.g. unfolding. That implies having a large simulation box that can accommodate such large conformational changes, as well as running for a long simulation time, enough for such modifications to take place.

Coarse-grain modelling technique is used to decrease the computational load by modelling several atoms or groups together as a single super atom (coarse-grain). However, coarse-grains suffer from loss of accuracy and fine details in addition to difficulty to model side interactions as well as lack of transferability. Multiscale Coarse Graining aims at overcoming the accuracy limit, but it still suffers from the problems of difficult transferability and side interactions [97].

1.5. Conclusion

There are two approaches for drug design: ligand-based drug design and structure-based drug design. Below are summaries of the famous tools and challenges of computer aided drug design.

1.5.1. Summary of tools used in every step

Table 1-1 summarize the tools and techniques, used in the whole computer-aided drug design process.

Table 1-1 Summary of tools and techniques used in Drug Design

Step	Common tools and techniques
Identifying Search space	LIGSITE [13], POCKET [14], SURFNET [15], FPocket [16, 17], and DEPTH [18].
Molecular Dynamics Simulation packages	GROMACS [19, 20], AMBER [21], NAMD [22], LAMMPS [23], ACEMD [24], and ADUN [25]
MD Simulation Force Fields	OPLS [47, 48], GROMOS [43, 44], CHARMM [46], AMBER [45], MARTINI [54, 55], and VAMM [56].
Lead Optimization	PRO-LIGAND [61], SPROUT [62], LigBuilder [63, 64] , and Autogrow[65].
Molecular Docking	UCSF Dock [68-73], FlexX [74], Glide [75, 76], ICM [77], PhDOCK [78], Surflex [79], AutoDock Vina [83], eHits [85, 86], GOLD [87, 88], AutoDock [89-91], FlipDock [92], FITTED [93], ParDOCK [94], Rosetta Ligand [95], and GlamDock [96].

1.5.2. Summary of current challenges and addressed problems

Some of the current problems in Drug Design domain are summarized as follows:

Molecular Docking faces challenges in: 1) High dimensionality, when docking ligands with many degrees of freedom or modeling receptor flexibility, 2) modeling interfacing solvent molecules, 3) modeling wide dynamic range of design variables as in blind docking, and 4) defining an accurate and representative scoring function.

MD Simulation runtime depends heavily on the size of the simulation box and the number of simulated atoms. Simulations are limited in time scale and size scale. Coarse-graining MD simulation suffers from loss of details, difficulty to model side interactions, and lack of transferability. Multiscale coarse graining overcomes the accuracy limit only.

1.5.3. Publications in this chapter

Conferences

- **"Recent developments in drug design workflow management systems"**, Amr ALHOSSARY, KWOH Chee Keong, MU Yuguang. Oral presentation in "CDAMIES 2018".

Articles

- **"Molecular Dynamics and Simulation"**, Liangzhen Zheng, Amr A. ALHOSSARY, Chee-Keong Kwoh, Yuguang Mu, in *Encyclopedia of Bioinformatics and Computational Biology*. 2019. p. 550-566.

Chapter 2 Practical and accurate acceleration of AutoDock Vina

As discussed earlier, search space sampling is a computationally intensive problem. On the other hand, drug fragment databases are growing rapidly, along with the growing need for screening them, through means of efficient Molecular Docking. Several approaches are still being developed to search the conformation space effectively. In this chapter, I will introduce the work done to enhance the search-space sampling algorithm in the molecular docking tool QuickVina 2 as a practical molecular docking tool that accurately accelerates AutoDock Vina.

AutoDock Vina is a widely used protein-ligand docking tool, published in 2009 from the Scripps research Institute with moderate speed. It was implemented in C++, and it depended on the famous C++ library "BOOST". Handoko et al. improved this tool and published "QuickVina". QuickVina performed well when it was tested in exceptionally high levels of exhaustiveness, while its accuracy is compromised in normal default low exhaustiveness levels that are used daily.

In this chapter, I will present how the accuracy of the search algorithm was improved, while retaining speed advantage, in low exhaustiveness levels. My work in both this chapter and Chapter 4 was focused on the sampling algorithm, and I did not touch the scoring function. This work was published as the new docking tool "QuickVina 2" [98].

2.1. Background

2.1.1. Molecular Docking and QuickVina

Molecular docking aims at performing *in silico* prediction of the modes and affinities of non-covalent binding between a pair of molecules. Oftentimes, the molecules consist of a macromolecule (the receptor) and a micromolecule (the ligand). The first step of that *in silico* prediction process is sampling the search space, in terms of ligand translation, rotation, and torsions of flexible bonds that may exist within it or within the receptor. Molecular docking finds its practical application in the screening of

huge virtual libraries of small molecules against proteins of biological importance to predict the most favored modes of interaction. For this reason, molecular docking is gaining more and more attention nowadays in the field of drug design.

The search for the most favored mode of interaction is a computationally intensive process. This process mathematically represents a complex optimization problem, aiming at finding a docked conformation with the minimum energy solution, which includes both components: Intramolecular energy and intermolecular energy. It is generally considered a NP-Hard problem [99, 100]. It can be formally defined as follows:

"Let A and B be the ligand and protein molecule, respectively. Let f be a scoring (energy) function that ranks solutions with regards to binding energy, and let C be the conformational (decision) search space of all possible conformations (docking solutions) between A and B. Then, the docking problem is an optimization problem aimed at finding a $\vec{x} \in C$ *satisfying* $f(\vec{x}) \leq f(\vec{y})\ \forall y \in C$ *(minimization)"* [99] .

The search space that includes all possible ligand-receptor complex conformations is enormous. Therefore, a brute-force approach of solving is not an option. Since 1980s, several heuristics were applied to such a problem. That includes Simulated Annealing, Tabu search, molecular dynamics (MD), dead-end elimination (DEE) combined with A$*$ search, and finally evolutionary algorithms (EAs). All implementations of the molecular docking need to balance the tradeoff between the search complexity (in terms of CPU time) and the search accuracy (in terms of globally lowest energy score and highest affinity) [101].

AutoDock Vina (also referred to as Vina) is a famous and widely-used molecular docking tool released in 2009. It uses a powerful hybrid scoring function (empirical + knowledge-based), as its objective function [83], and employs an evolutionary search algorithm that combines both local search and global optimizer. Subsequently, QuickVina (referred to as QVina 1 in equations, figures and their captions) enhances the computation time of the original Vina by means of a heuristic that prevents unnecessary local searches from being carried out. To account for the fact that QuickVina 1 may

actually miss some important local searches, QuickVina 1 authors have specified a higher exhaustiveness level of the search as compared to that of the original Vina [102].

In this work, an additional modification was made to QuickVina. Hereinafter, I will refer to as QuickVina 2 (shortened to as QVina 2 in equations, figures and their captions). QuickVina 2 features improved robustness compared to QuickVina 1 while keeping the exhaustiveness level of the search similar to that of the original Vina. By means of some enhanced heuristics, QuickVina 2 should not miss any important local search while still preventing unnecessary local searches from being carried out. In other words, QuickVina 2 inherits both the speed of QuickVina 1 and the robustness of the original Vina.

2.1.2. Scalability and considerations to take

Scalability describes the ability of a program to effectively benefit from additional resources. To assess scalability, there are generally two notions: 1) hard scalability, which is how the solution time varies with the number of processors for a fixed *total* problem size, and 2) weak scalability, which is how the solution time varies with the number of processors for a fixed problem size *per processor*.

2.1.2.1. Amdahl Law

One important thing should be put into consideration is the portion of the code that can be parallelized. It is more important to work on making the code scalable than to add more nodes to the system. Amdahl's law (or Amdahl argument) [103] is used in parallel computing to predict the theoretical speedup in the latency of the execution of a task at fixed workload for a system, using multiple processors. It states that "the maximum speedup (the maximum number of processors which can be used effectively) is the inverse of the fraction of time the task must proceed on a single thread".

If α is the serial portion of the target code and p is number of processors that will run the code, The speedup according to Amdahl law can be formulated as $\frac{1}{\alpha+\frac{1-\alpha}{p}}$. Figure 2-1 shows how

problem solving codes with different parallelizable portions would upscale with increased number of processors.

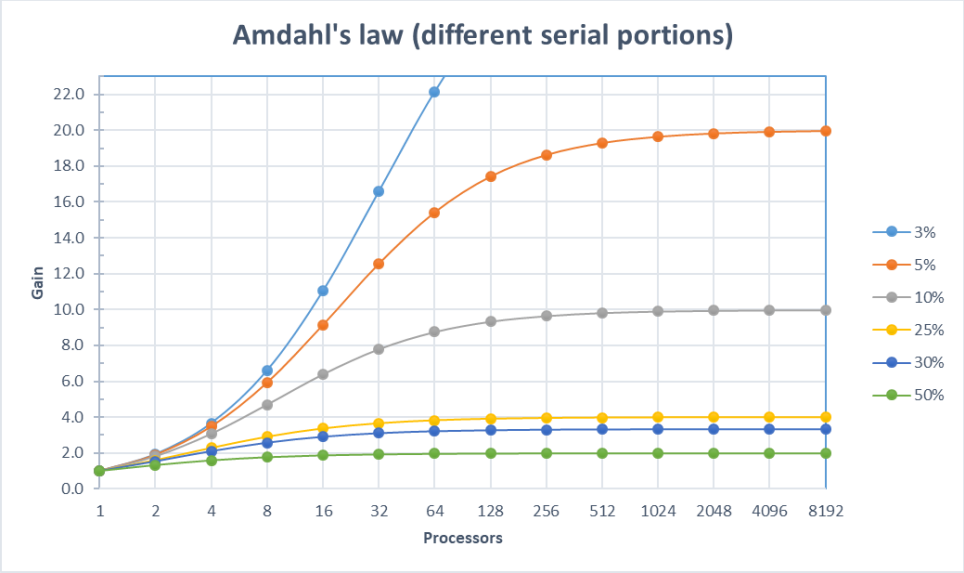


Figure 2-1 Amdahl Law for different portions of code that can not be paralellizable
The percentage of parallel portion of the task is the most controlling factor. Increasing the number of processors would reach an asymptotic upper limit, while increasing the parallelizable portion of code shifts the speedup curve upwards. In theory, if all the code is parallelizable, there will be no upper limit.

For example: suppose that only 70% of the code can be parallelized (α = 0.3), substituting the values of p with the values 8 and 16, the speedup would be 2.58 and 2.91 respectively. Doubling the number of processors increased the performance by 12.7% (almost 1/8), while the same resource upgrade would raise the speed up from 6.61 to 11.03 respectively, which adds more 66.9% (more than 2/3) if the serial portion is decreased to 3%. If the serial portion drops to 0.0, doubling the processors would in theory increase the performance by 100%.

2.2. Methods

2.2.1. Search space Sampling Algorithm in Vina

Every possible solution can be conceptualized as a "**point** in the search space". In Vina, the search space is the combination of the three dimensions of translation, the three axes of rotation, and any

torsion angles that exist in the ligand or in the receptor flexible side chains. Therefore, the degrees of freedom (DOFs) equal 6+ number of torsion angles. For example, when the ligand has 4 rotatable bonds, there will be 4+6 = 10 DOFs.

Vina explores the search space using a combination of global and local optimization. The global optimization is a form of Markov chain of modified Monte Carlo algorithm with restart, while the local optimization is in the form of nonlinear unconstrained iterative optimization using the famous Broyden-Fletcher-Goldfarb-Shannon (BFGS) method [104]. This is in line with the spirit of Memetic Algorithm (MA)[105].

The modified MC (Monte Carlo) algorithm differs from the ordinary MC algorithm in performing a local search (local optimization) on the new neighbor which results from MC randomization process, and checking the local optimization result for acceptance according to metropolis criterion, instead of checking the new random neighbor itself directly. *We* call that performed local optimization an "*essential* local optimization". If the result is accepted, a series of local searches, using the BFGS method is performed, in order to find a local minimum in this region, then the global optimizer will restart search in a new region [102].

Given sufficient number of iterations in local search, it is expected to find a local minimum in the current region, and combining both local and global searches, it is expected to cover several regions all over the search space. Hence, it is highly probable that the global minimum is one of those found minima.

2.2.2. QuickVina 1 basis of optimizing local search frequency

Local search is the most time-consuming part of Vina processing. Therefore, in a previous work done by Handoko et al., to accelerate Vina [102], the interest was to allow the application to perform local search on some of the points only, which are the promising ones, and to prevent others from undergoing unnecessary local search optimization. Their work resulted in "Quick Vina" (referred to in the rest of this chapter as "Quick Vina 1"). QuickVina 1 restricted the application local search to those

docking conformation candidates deemed to be significant by the ***first-order-necessary-condition*** heuristics.

According to that heuristic, if there are two points which have all partial derivatives of the scoring function with respect to all design variables with opposite signs to their counterpart derivatives in the other point, that guarantees that there is at least one stationary point in between them. Therefore, by keeping track of the previously visited points in a database, for every new potential point, when it is proposed, it is first checked a small number (P) of checks against the closest (P) visited points in the database. In case there exists at least one, which has its partial derivatives of the scoring function with respect to all design variables with opposite signs to their counterpart derivatives in the new point, that guarantees that there is at least one stationary point in between them. Therefore, this new point is considered a promising point and consequently it is allowed to go for local search and is added to the database to be used later.

For example, Figure 2-2 represents a hypothetical energy function of one dimension, with its derivatives in several points. Let us suppose that the database contains points A, B, and D, then point C is the new potential point. In this case, point C is accepted, because point D (which happened to have the closest Euclidian distance to it), has an opposite derivative sign. On the other hand, if only point B exists in the database, and point A is the new point, it is not accepted, because it has the same derivative sign as B.

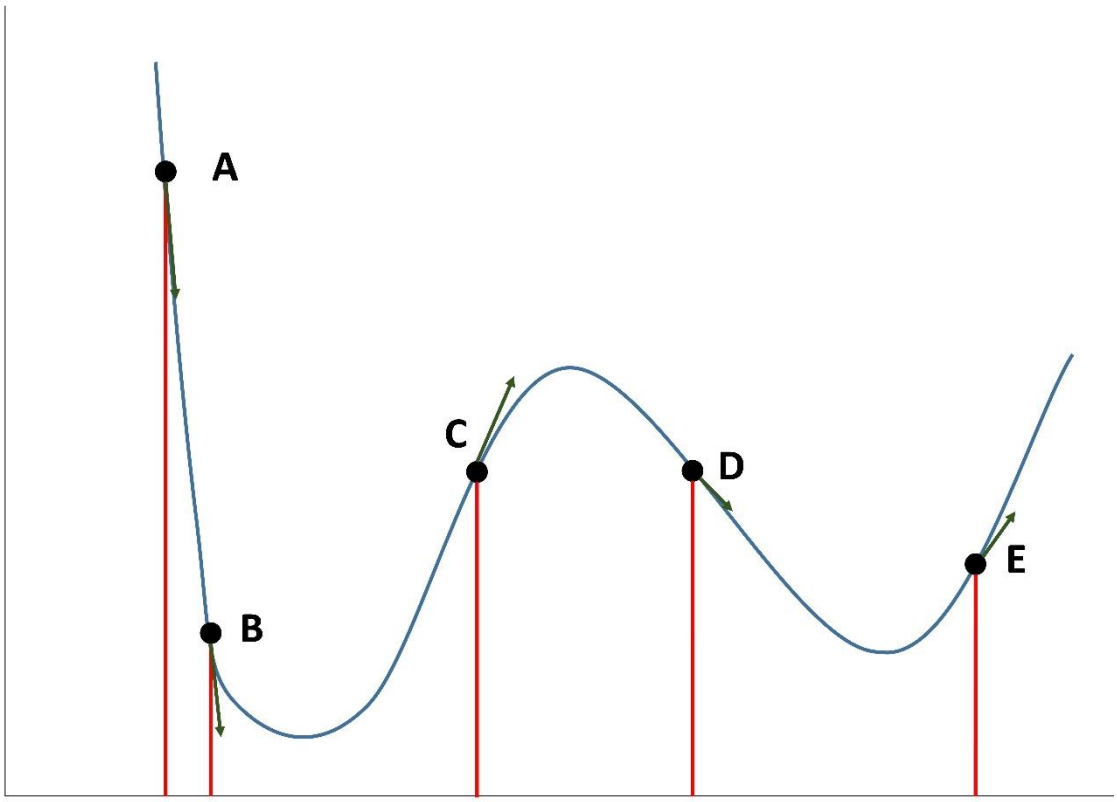


Figure 2-2 Sample 1D function with representative points
Function derivatives represented as green arrows, while values are represented as red columns.

2.2.3. Quick Vina 1 implementation

In QuickVina 1, the database of visited points was implemented as a circular buffer of pointers to "point" elements of size Q = 10N, where N is the number of design variables (degrees of freedom). That means any new element e to be added after 10 N entries were stored, would overwrite the oldest stored element in the database. Every element is represented as a vector of floats, that hold X values, as well as two "long long" integer variables (64 bits) that hold together the objective function partial derivatives signs in a compressed way. One of them holds the derivate signs as a positive/negative binary value, while the other holds flags for Zero as a special case, distinct from both positive and negative signs.

A circular buffer is composed of 1) a list (to store the points and track the size) and 2) an integer p that points to the position (index) of the next point to store in (the tail). If p goes past last element, it is set to 0.

```
p = (p + 1) % (get_maxSize());
```

The database is initially empty (head and tail both are 0). Then, with every addition operation, the list size keeps growing, up to the maximum allowed size. Then no more growth is allowed and only overriding is allowed.

```
list[p] = e;
```

The database supports appending and retrieval operations only. Every addition (appending) operation cost will be of order O (1). Initializing the list size to the max size from the beginning, ensures there is no allocation/reallocation overhead in the growth stage.

Setting P to 2N, every potential point is tested against the stored closest 2N points in Euclidian distance to this new point. Should any of them have all partial derivatives with either opposite signs to the new one or zeroes (i.e. already stationary in this dimension at this point), the potential point passes the test.

2.2.4. Limitation of QuickVina check

Suppose the database contains only point D, and then point B is newly introduced. Because there is no entry with opposite derivative in the database, it will be rejected, although at least one stationary point already exists in between.

This condition is infrequent in high exhaustiveness levels. However, it is not so rare in low levels around the default level. Please note that the default exhaustiveness level is eight, while QuickVina 1 was tested using exhaustiveness level 128 (i.e.64 folds of the default).

2.2.5. Increasing robustness of QuickVina check

To minimize the occurrence frequency of this situation, and consequently to minimize the number of necessary local searches that are missed, a novel ***first-order-consistency-check*** heuristic.

According to that heuristic, if there are two points which have all partial derivatives of the scoring function with respect to all design variables with same signs as their counterpart derivatives in the other point, it is guaranteed that there are at least two stationary points in between them, if the gradient (the difference) between the points values is in the opposite direction of their derivatives for all design variables. That is to say, instead of considering the partial derivative signs only, a combined condition of the function derivatives and the value together is used[106]. This method should be applicable to any protein-ligand combination under investigation.

To illustrate the idea, let us consider the previous scenario in Figure 2-2 (the database contains only point D, and point B is newly introduced), according to the QuickVina 1 criterion, it will fail the test because the derivatives have the same sign (both are going down). However, considering that the function value at point D is higher than that of point B, it is guaranteed that there is at least one stationary point in between. (There are actually at least two stationary points: one is lower that B and the other is higher than D). The same situation holds true if B is in the database and D is newly introduced. The opposite is true also for points with positive derivatives, (e.g. points C and E in Figure 2-2). This check does not compromise the performance but it ensure more robustness. Figure 4-1(middle) and Figure 4-2 (middle) – respectively - show the flow chart and pseudo code of the algorithm used in QuickVina 1 (same as QuickVina 2).

2.2.6. General Mathematical Model

To generalize the mathematical condition, it can be said that a point (i) is accepted if there exists a point (j) of the nearest 2N points in the database with either

$$\boldsymbol{sign}\left(\frac{\delta f}{\delta x_n}(\boldsymbol{i})\right) \cdot \boldsymbol{sign}\left(\frac{\delta f}{\delta x_n}(\boldsymbol{j})\right) \leq \mathbf{0} \qquad \text{(2-1)},$$

where $\frac{\delta f}{\delta x_n}(i)$ is partial derivative of the function (f) in relation to variable (x_n) at point (i), or with

$$sign\left(\frac{\delta f}{\delta x_n}(i)\right) . sign(\{f(i) - f(j)\} . \{i - j\}) \leq 0 \quad \text{(2-2)},$$

2.2.7. Implementing the more robust check

Minimal interference was intended throughout the work. To implement the new check with the least possible modification of the available code, only one floating-point value (C++ float variable) that holds the function overall score, was added to the database element class. The new condition is checked only in case the old condition fails.

2.3. Validation

QuickVina 2 was validated using the "Core set" of the PDBbind 2014, the same dataset used to test original Vina and QuickVina. PDBbind is a collection of biomolecular complexes in the Protein Data Bank (PDB), said to provide a "comprehensive collection of experimentally measured binding affinity data", subdivided into several sets. The core set is considered a "high-quality benchmark for evaluating various docking/scoring methods" [107-109].

The performance and output of the three Vina family tools (Vina, Quick Vina 1, and Quick Vina 2) were compared. In addition, QuickVina 2 was compared to two famous docking tools, namely UCSF Dock 6.6 [68-71]and GOLD 5.2 [87].

2.3.1. Dataset

Renxiao Wang's group at the Shanghai Institute of Organic Chemistry maintains PDBbind. Release 2013 of PDBbind (*previous release*) was based on PDB snapshot released on 1/1/2013, which contains 87,085 experimentally determined structures (theoretical models are not considered). This database snapshot is automatically screened to identify 38,918 valid complexes in four categories: protein-small ligand, nucleic acid-small ligand, protein-nucleic acid, and protein-protein complexes. They are then

manually reviewed to select those, which have experimentally determined binding affinity data (K_d, K_i or IC_{50}). They are 10,776 complexes, referred to as “General set”. Some quality filters related to the X-Ray itself, were then applied, to select 2,959 protein-ligand complexes as the “refined set”. Finally, the “core set” is formed by systematically sampling then clustering the “refined set”, based on protein sequence similarity with 90% cutoff. Clusters that have five members at least, and some other selection criteria regarding the binding constant range, are selected. To ensure diversity, three members represent each cluster: The ones with highest and lowest binding constant, and one middle binding constant member.

Although the maintaining group released a 2014 version of the dataset, they did not update the “core set”, but instead used the 2013 version collection. That is to say, the 2013 version is the latest version currently available of PDBbind “core set”.

2.3.2. Testing protocol

The same testing methodology used in original Vina publication [83] was exactly followed.

For the Vina family tools, a separate parameter file was generated per ligand-receptor complex, containing the two input file names, search space definition parameters, a unique random seed, explicit setting of number of CPUs (set to 8), and explicit setting of exhaustiveness. The details of the exhaustiveness parameter will be described in section 2.3.4 [Exhaustiveness]. The (same) parameter file was passed to Vina, QuickVina 1, and QuickVina 2, to ensure that the running circumstances are uniform across the three tools under comparison.

For the other two tools (Dock and GOLD), the goal was to define a search space that is 1) identical to the search space box used for Vina family and 2) unbiased to a certain pocket within the cavity.

2.3.2.1. Search space box

Vina family search space was identified as the minimal containing parallelepiped, aligned with the axes. Its size was increased by 10 Å keeping the ligand in its center, and then another 5 Å was added to one side of every dimension, to decentralize the search box on the experimental ligand. Finally, dimensions that are less than 22.5 Å, are symmetrically increased up to this value, by adding half of the difference symmetrically to each side while keeping the box decentralized.

Dock search space was prepared to be identical to the above-mentioned space. Because it is not a standard procedure in Dock to select spheres based on coordinates, all contact spheres were generated, and then an in-house developed code was used to generate a selection box and to select those spheres whose centers lie within the search box boundaries.

Similarly, because GOLD does not recognize box-shaped search spaces, the search space of each of the 195 complexes was identified, in the first attempt, as a sphere, centered at its corresponding Vina family search space center and with radius equal to the half of the largest dimension of that corresponding search space. Later this definition had to be changed; because it led to docking difficulty. Therefore, the search space was redefined as the list of ***receptor*** atoms whose centers lie within the corresponding Vina family search space and allowing five Angstroms margin.

2.3.3. Preparing dataset

2.3.3.1. Preparing for Vina family (Vina, QuickVina 1 & 2)

All ligands and receptors were converted to PDBQT format, using AutoDock Tools prepare_ligand4.py and prepare_receptor4.py scripts respectively, allowing rotation of all bonds in ligands. All ligands were randomized using Vina specific parameter “randomize-only”, as a preprocessing step.

2.3.3.2. Preparing for Dock and GOLD

The same randomized ligands obtained above were used as inputs for Dock and GOLD – after converting them into MOL2 format, and their respective outputs were then converted back into

PDBQT format to calculate the RMSD as described in [Appendix A: RMSD Calculation Method as described by AutoDock Vina team].

- Preparing for docking using Dock 6.6

The preparation process started with preparing for docking using Dock 6.6, employing the "Anchor and Grow" algorithm. However, there was a difficulty dealing with an application, which depends on some modules dated back to 1970s. The list of obstacles is as follows:

a- In the spheres generation step, the module named "sphgen", crashed frequently with relatively big molecules. In addition, it could not handle more than 99999 spheres. After some search, a contributed application (called "sphgen_cpp") that handles these limitations was found.

b- As indicated earlier, an in-house code had then to be written to select those whose centers lie within the search space, and to generate a selection box.

c- In the docking step. Using "dock6" application, four (4) of the receptor-ligand complexes failed to find a suitable conformation, using the same parameter values of the application tutorial. The problem continued to happen, even after relaxing the parameters: min_anchor_size to 6, pruning_clustering_cutoff to 25, and max_orientations to 50000. Only after communicating with dock fans mail group, disabling the bump filter and changing pruning_clustering_cutoff and pruning_conformer_score_cutoff parameters to 100 and 100.0 respectively, the docking can be accomplished.

d- In addition, in the same step, five (5) complexes crashed with an error message about a probably too big grid to fit in memory. By debugging the software, it was found that it is not related to Grid at all, but that there is a **limitation**, which caused the application to crash when trying to dock complexes with big receptor sites/ligands.

That limitation was because the application used to allocate residual matrix with size [(num_spheres * num_centers)^2]. Above certain number of nodes (i.e. of spheres-centres combinations), that amount of memory cannot be allocated.

That limitation was relaxed by modifying the code to dynamically allocate the matrix as a shredded two-dimensional array (an array of arrays) of size [(num_spheres * num_centers)] each. This way, the application could allocate the same needed amount on smaller chunks, summing up to extremely bigger amounts of memory up to the system limits. For example, one of the instances that could run after the batch, consumed **53 (*fifty-three*) Gigabytes** of memory safely. The software batch was sent to the Dock development team to include it in a subsequent batch/release.

- Preparing and docking using GOLD 5.2

Later, the process moved to preparing for docking using GOLD 5.2, trying to stick to the default values as much as possible. Phosphokinase default mode was used and Solvent Accessible Surface Area (SASA) search filtering was enabled. Because GOLD does not recognize box-shaped search spaces, the search space of each of the 195 complexes was identified as a sphere, centered at its corresponding Vina family search space center and with radius equal to the half of the largest dimension of that corresponding search space.

a- Employing these settings, 35 complexes failed to find any suitable conformations, until the SASA search filtering was disabled.

b- After disabling the SASA search, 5 of the complexes failed to dock altogether and 2 (two) complexes continued to search for more than 5 days until the process was killed.

c- The process was started over after redefining the search space as the list of ***receptor*** atoms whose centers lie within the corresponding Vina family search space and allowing five Angstroms margin.

d- Using this search space definition and enabling SASA filtering again, all the complexes could produce output, except two. Then the SASA filtering was disabled for these two complexes and they could produce results.

2.3.4. Exhaustiveness

One of the important parameters that affect Vina Performance is called “Exhaustiveness”. It controls how thoroughly the program searches the conformational space for the minima. It is claimed that increasing the exhaustiveness would “increase the time linearly and decrease the probability of not finding the minimum exponentially” The default exhaustiveness value is eight. The experiment was repeated 3 times with three identical sets of data, but with different exhaustiveness levels, explicitly set to 4, 8 and 32.

2.3.5. Testing platforms

The three Vina tools were compiled on the same Linux machine. They were tested on the same CentOS Linux release 6.0 x86_64, running on a server with 8 “Intel® Xeon® CPU X7560 2.27GHz” processors (8 processors * 8 cores each * 2 hardware threads per core = 128 logical CPUs), and 98 GB RAM.

The same machine was used to test Dock, noting that Dock 6 works only in a *pseudo parallel* mode and cannot make use of the multicore feature of the testing machine. Meanwhile GOLD run on another separate machine.

2.3.6. Unifying the data formats

To perform the additional comparison (to GOLD and UCSF Dock), the (randomized) PDBQT ligands were converted back to PDB/MOL2 using Antechamber and Chimera [110]. The Gasteiger method [111] was employed for assigning partial charges to ligands atoms.

2.4. Results and discussion

The docking results were collected, in the form of docking time, list of predicted modes, along with their corresponding calculated docking scores. In the coming sections, the quality of prediction of Quick Vina in relation to Vina will be discussed, and then the comparison between their accelerations.

2.4.1. Quality

The two main measures of quality of prediction are the Binding Energy (the more the negative energy the better prediction), and the RMSD to experimentally proven results (the closer the distance the better the prediction). From inspecting the results, QuickVina 2 shows much better prediction quality, in terms of both Binding Energy and RMSD to experimental data.

The first measure of prediction quality (binding energy) is shown in Figure 2-3. The binding energies of the first predicted modes in QuickVina 1 and 2 were plotted on the vertical axis, against their corresponding modes in original Vina on the horizontal axis. It shows that the binding Energies of QuickVina 2 are almost identical to those of Vina. The Pearson's correlation coefficient (r) for the two pairs was 0.967 between Vina and QuickVina 2, while it was only 0.780 between Vina and QuickVina 1. The same observation was also observed for the sums of all predicted modes where the respective values were 0.912 and 0.637 (data is not shown).

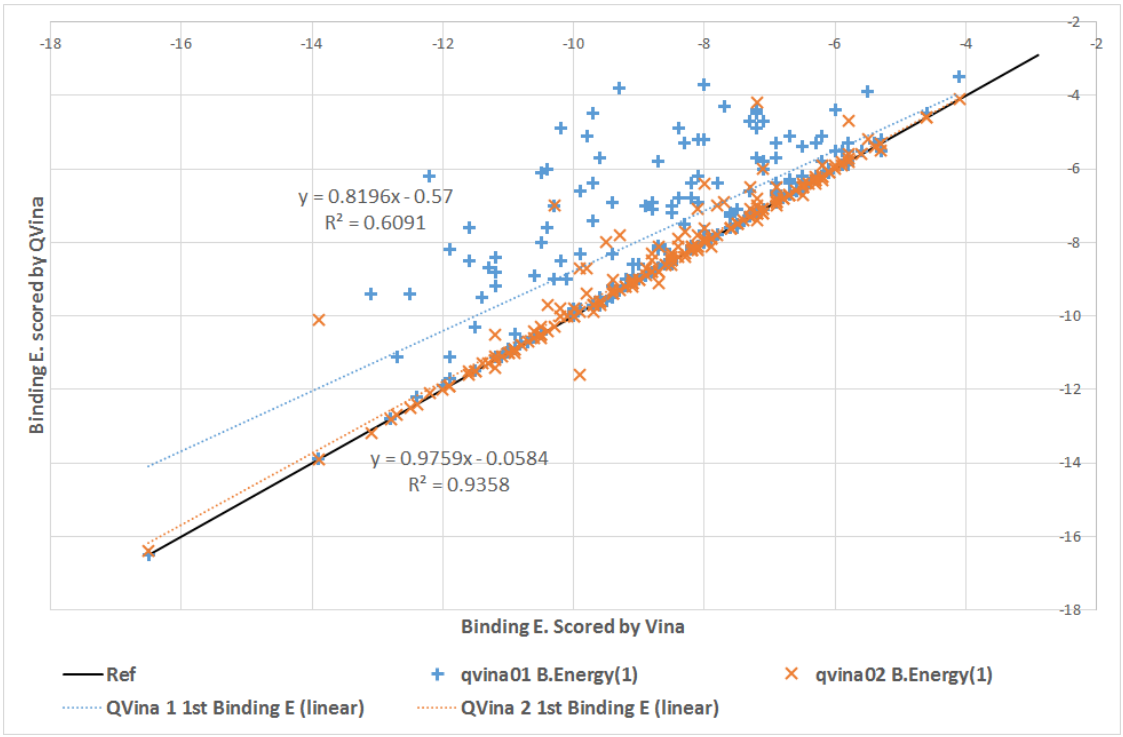


Figure 2-3 Binding Energies Scored by Vina, Qvina 1 & 2
Modes predicted by QVina 2 (orange x), have almost the same binding Energy as that of original Vina (Black diagonal reference line), while those predicted by QVina 1 (light blue +), have more scattered and less negative scored binding Energy.

Figure 2-4 shows percentages of better, equal, and worse predicted results (in relation to Vina), for both QuickVina 1 and 2. It shows that 70% of first predicted modes from QuickVina 2 are equal to or better than Vina predicted modes, in terms of Binding Energy. Even the remaining 30% have an average Energy difference as small as 0.58 Kcal/mol. On the contrary, the ratio is almost reversed for QuickVina 1: Only 37% are as well as or better than the Original Vina, and the remaining 63% have an average E difference as far as 1.56 Kcal/mol.

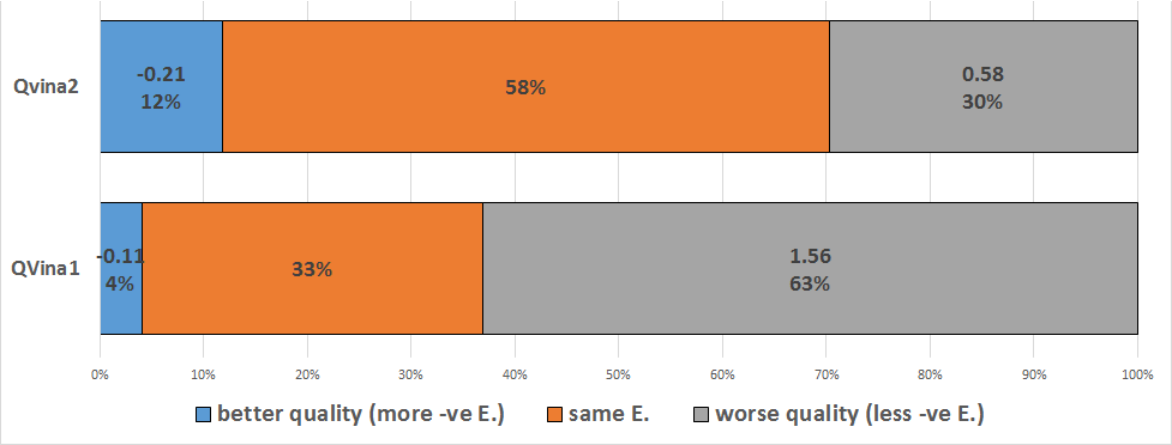


Figure 2-4 First predicted mode quality (in terms of binding energy), in relation to Vina
Percentage of predictions with the same Binding Energies is shown in the middle part, while those with better Energy (more negative) are shown to the left (along with the average of energy difference from Vina, and those with worse Energy (less negative) are shown to the right along with their average energy difference.

The second measure of quality of prediction is how close it is to the experimentally provided data. Dock and GOLD were compared to the Vina family only in RMSD to experimental data. They cannot be compared in terms of binding energy because the three packages employ three different scoring functions. AutoDock Vina team described a way to calculate the RMSD between experimental and predicted results, which "takes into account symmetry, partial symmetry (e.g., symmetry within a rotatable branch) and near symmetry, in a simple heuristic way". It is described in [Appendix A: RMSD Calculation Method as described by AutoDock Vina team].

Figure 2-5 shows the relation between the RMSD between the experimentally found modes and those predicted by Quick Vina 2 on the horizontal axis, and the RMSD between the experimental modes and modes predicted by other tools on the vertical axis. Vina 1.1.2 is shown as Light Blue x, GOLD 5.2 as Orange +, and Dock 6.6 as Black -. QuickVina 1 is not shown for figure simplicity. However, it is included in the full numbers table below. The original Vina team specified 2Å – as a common practice - as the binary threshold that distinguishes successful predictions from unsuccessful ones. Following the same convention, one threshold was set for QuickVina 2 predictions (Red vertical line) and another one for the other tools predictions (Red horizontal line). It is noticed that QuickVina 2 has

the highest agreement with Vina as indicated by having most of Vina points are on the reference line (as close to the experimental predictions as the original Vina).

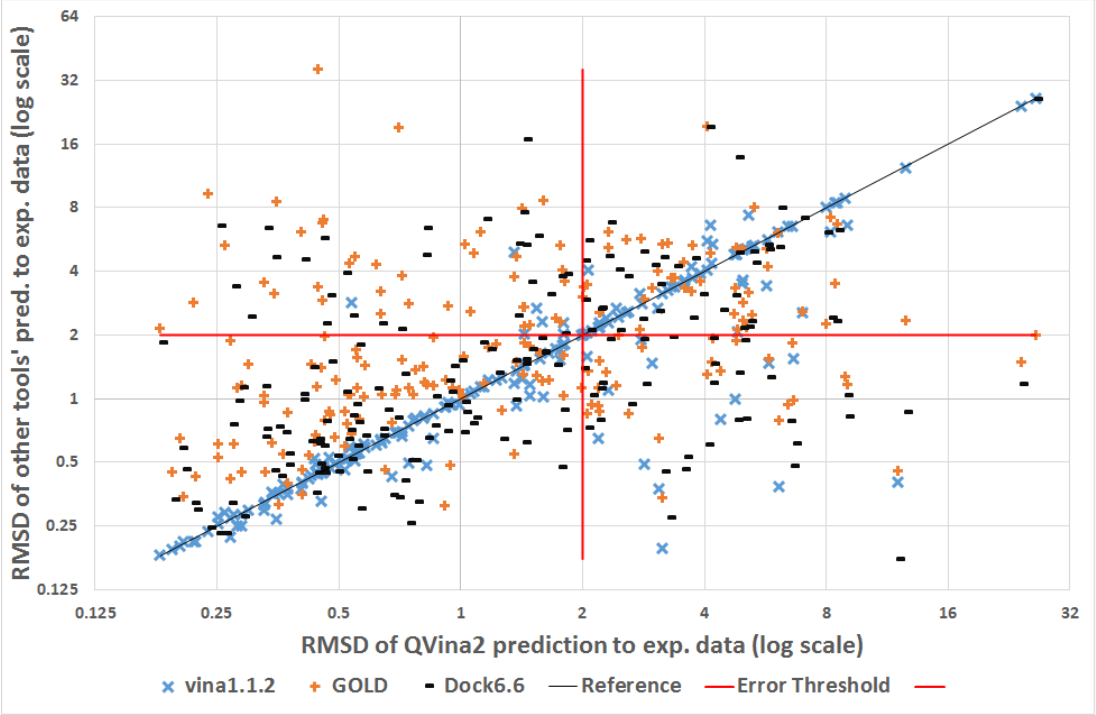


Figure 2-5 RMSD to Experimental Data of Vina, GOLD, and Dock versus QVina 2
RMSD from Vina 1.1.2 (Light Blue x), GOLD (Orange +), and Dock 6 (Black -) on the vertical axis versus RMSD between Quick Vina 2 and experimental data on the horizontal axis (on log scale). Two 2 Å error thresholds (Red lines), split the graph into four quadrants: succeeded by both (lower left), failed by both (upper right), succeeded by QVina 2 only (upper left), and failed by QVina only (lower right).

The detailed numbers of success/fail predictions of Figure 2-5 is shown in Table 2-1. The table shows that QuickVina 2 has the highest *consistency* with Vina among the other docking tools. On the accuracy against the experimental conformations, Dock 6.6 is slightly more accurate than QuickVina 2 at the expense of long computation time. Dock 6.6 attained 127 successes while QuickVina 2 achieved 123. Meanwhile, GOLD 5.2 only had 111 successes.

Table 2-1 Successes/Fails of QVina2 vs other tools (exhaustiveness = 8)

		Vina		GOLD		Dock		Qvina1	
		Success	Fail	Success	Fail	Success	Fail	Success	Fail
QVina2	Success	116	14	84	39	96	27	79	44
	Fail	7	58	27	45	31	41	9	63

Figure 2-6 represents evolution of the success rates of the three Vina family tools among different exhaustiveness levels, from 4 (lower), then 8 (middle), through 32 (upper), in the form of SUCCESS/FAIL pairs triplets.

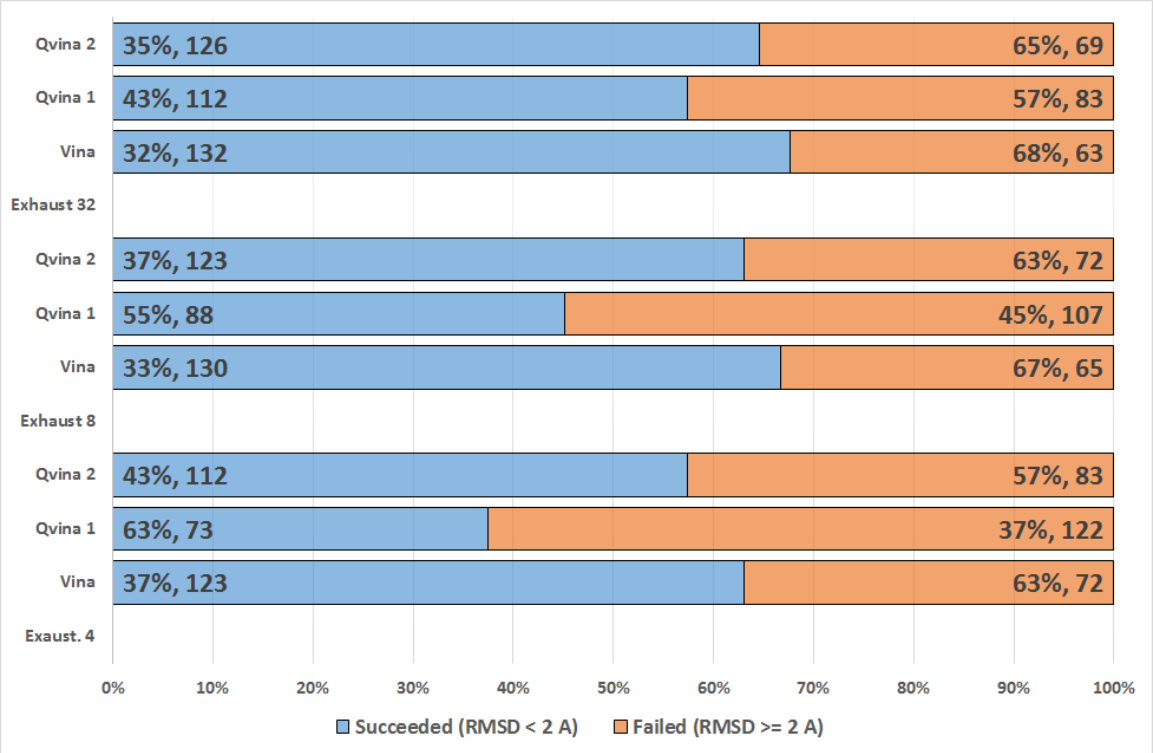


Figure 2-6 Evolution of success rate among different low exhaustiveness levels.
Exhaustiveness value equals 4 in lower group, 8 in middle group, and 32 in upper group. In every group, results from original Vina, QVina 1, and QVina 2 are shown as percentage and number of complexes that either passed or failed, with 2Å cutoff.

The full numerical series is in Table 2-2. The data is divided into sets of quartets (SUCCESS by both, by Vina only, by QuickVina Only, FAILED by both). Each quartet is ordered in a form that matches its corresponding four quadrants in Figure 2-5. It is noteworthy that in higher exhaustiveness levels, the number of succeeded predictions increases, and number of failed predictions decreases consequently out of the (fixed) total number.

Table 2-2 Evolution of Success/Fail fractions

QuickVina \ Vina		Exhaustiveness = 4		Exhaustiveness = 8		Exhaustiveness=32	
		Success	Fail	Success	Fail	Success	Fail
QVina 1	Fail	57	65	48	59	29	54
	Success	66	7	82	6	103	9
QVina 2	Fail	18	65	14	58	9	60
	Success	105	7	116	7	123	3
Success rate inc. %		**59%**	**0%**	**41%**	**17%**	**19%**	**-67%**

Every quartet is a self-contained set, containing the amount of results prediction by Vina (V) and one of the QVina versions (QVina 1 and 2), using a certain exhaustiveness value. The lower row represents the percentage of increase of successful prediction by QuickVina 2 over QuickVina 1.

Figure 2-7 shows the relation between the RMSD to experimental data and its corresponding binding energy of the first mode predicted by each of the three tools, along with their trend lines, with exhaustiveness level = 8. The slopes of the three lines show that the closer the distance to the experimental data, the more the stable complex. This is intuitively logic, because the experimental data is the already proven stable structure. In the figure, it is noticed that Vina and QuickVina 2 trend lines are very close to each other, while QuickVina 1 trend line is slightly elevated and has more slope. That means QuickVina 2 predictions tend to have equal binding energies to their Vina counterparts, while QuickVina 1 tends to predict less stable (less negative binding energies) structures.

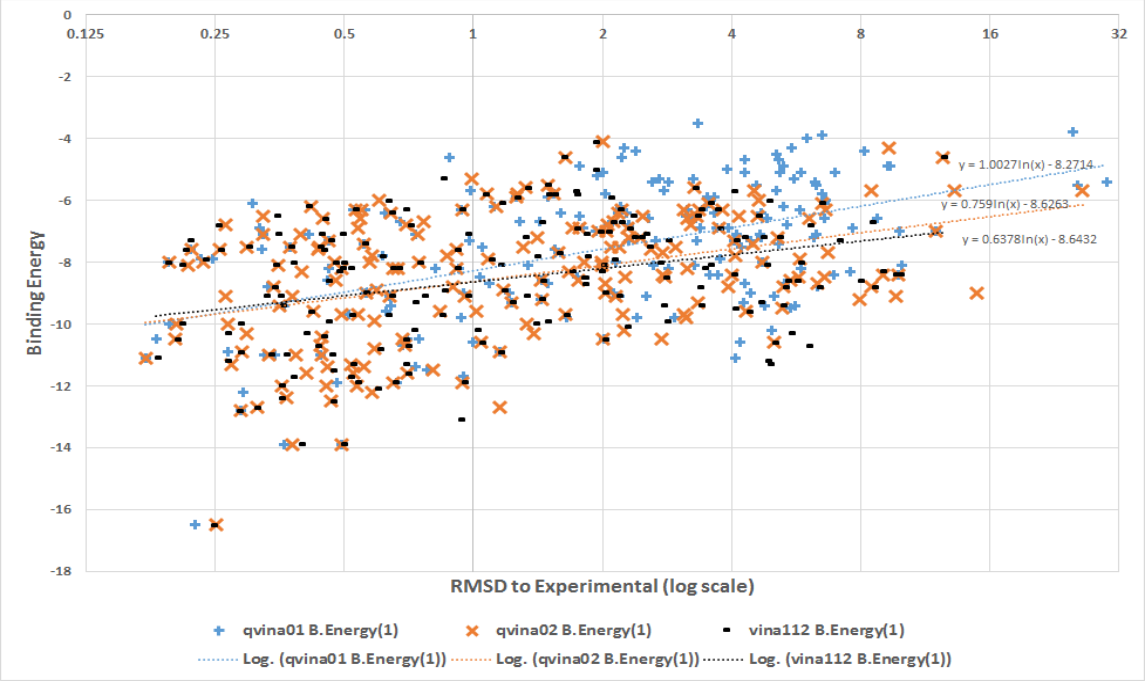


Figure 2-7 RMSD to Experimental data vs. Binding Energy, of 1st predicted mode
Relation between RMSD to experimental data on horizontal axis (log scale) and binding energy on vertical axis. Vina (Black -) and QVina 2 (Orange x). Trend lines are very close, while QVina 1 (Light Blue +) trend line has a higher, slightly separate slope.

2.4.2. Speed and Acceleration

The running time of the three tools in the form of absolute duration, then in the form of Search-time acceleration rate for Quick Vina 1 and 2 against original Vina are shown in Figure 2-8. Search time acceleration is defined as the processing time of Vina divided by its corresponding time of Quick Vina, excluding overhead time of both pre- and post- processing. Vina family speeds couldn't be compared to the two additional tools, because the three groups were run on two different machines with three different configurations. The most important temporal statistics are presented in Table 2-3.

Figure 2-8 shows that Vina processing time increases exponentially in relation to number of heavy atoms as in a), and in a rate of second order in relation to active bonds as in b), while QuickVina corresponding processing time increases linearly with a very gentle slope (QuickVina 2 processing time slope is 0.37). That makes QuickVina acceleration obvious over Vina. It is notable in Figure 2-8 (b) and (c) that the acceleration rate in QuickVina 2 is less than that of QuickVina 1, which is -of course- expected; because it performs more local searches than QuickVina 1. However, the acceleration tends

to increase in a power of 2 with increased number of heavy atoms/Active bonds. Figure 2-8 (c) (Acceleration vs. No. of Heavy Atoms) shows better fitting than Figure 2-8 (d) (Acceleration vs. No. of Active Bonds).

It is expected that the slower rate of acceleration on the left parts of the graph(s) be due to the overhead of forking and joining threads as well as filling the entries database first before taking any decision.

It is notable also that the rightmost parts of the curves converge together where the acceleration rates have very close values for both QuickVina 1 and 2. For example, 1ZEA, which has 70 atoms and 32 bonds, has acceleration rates of 22.28 and 20.49 for QuickVina 1 and 2 respectively. Similarly, 3AG9, which has 67 atoms and 41 bonds, has acceleration rate of 18.95 for QVina2 and 19.75 for QuickVina 1. Actually, an important observation to consider is that the speed up reaches levels comparable to those of QuickVina 1, because the main need for sophisticated -yet efficient- algorithm is when the number of dimensions and the complexity of the optimization problem increases.

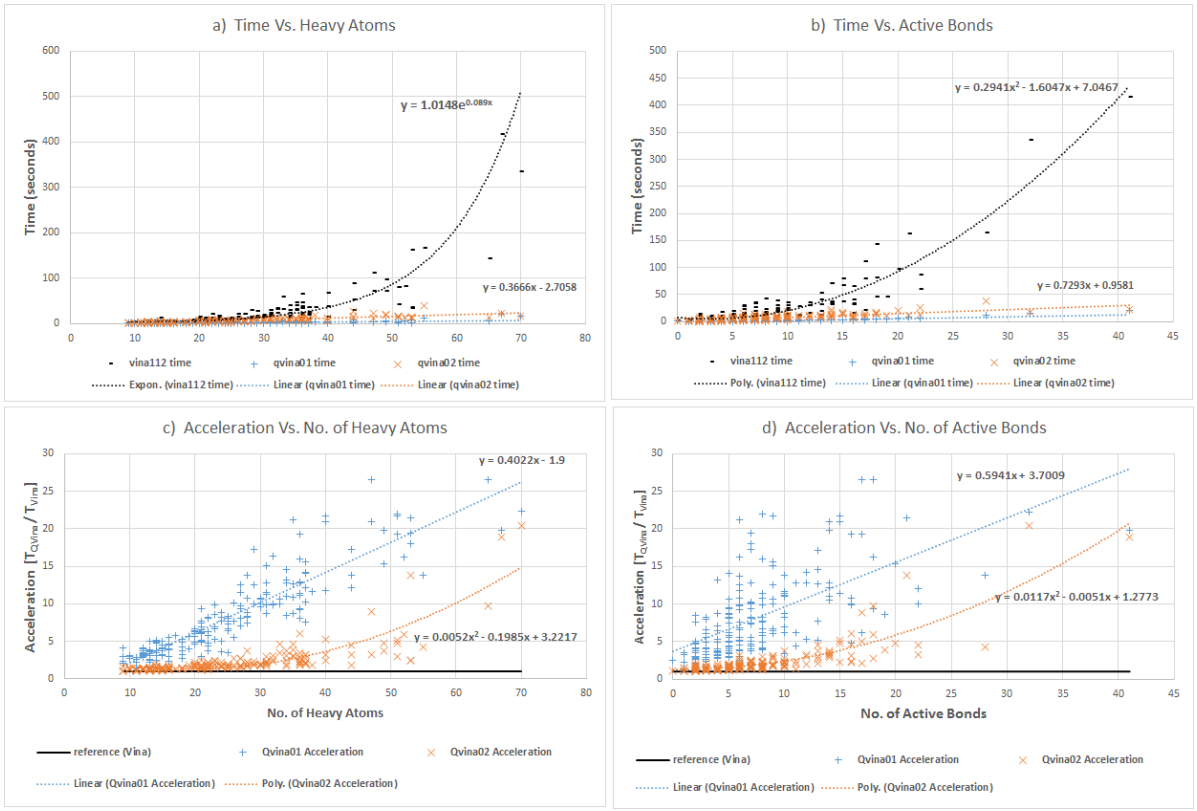


Figure 2-8 Temporal measures
Running time (in seconds) versus a) number of heavy atoms and b) number of active bonds. Acceleration [in terms of running time of Original Vina / running time of Quick Vina] is plotted versus c) number of heavy atoms and d) number of active bonds. In all the four, Original Vina is plotted in Black, QuickVina 1 in (light blue +) and QuickVina 2 in (Orange x).

Figure 2-9 is a close up view of Figure 2-8 (d) for the accelerations attained by QuickVina 1 and QuickVina 2 over the original Vina under different numbers of active rotatable bonds in the range of [0, 10] rotatable bonds, in the log scale.

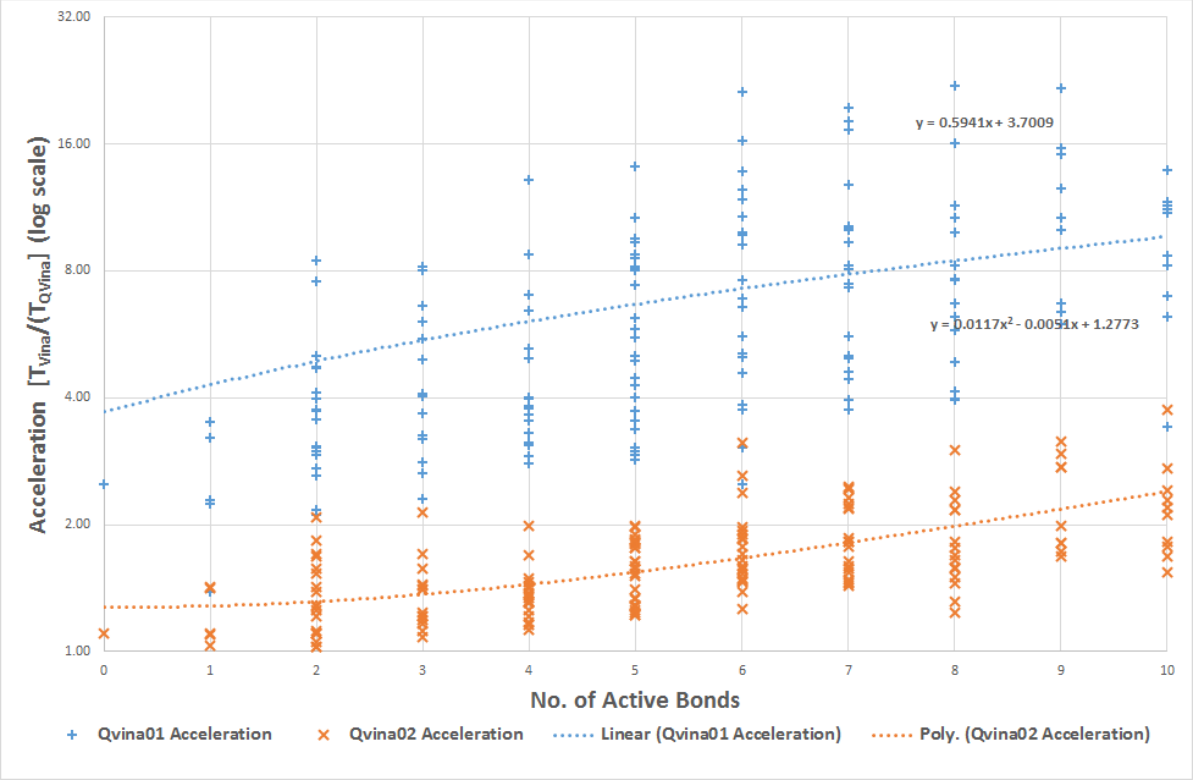


Figure 2-9 Close up view on acceleration vs No. Active Bonds
Zoomed view in the range [0-10] of Figure 2-8 (d) [Acceleration versus active bonds]. The trend lines are drawn according to the full dataset. QVina2 trend shows quadratic uprising tendency with increased number of active bonds.

The full list of running times absolute values and accelerations is presented in tabular form in Table 2-3.

Table 2-3 Runtime and Acceleration statistics

	Running Time (in sec.)			Acceleration	
	Vina	QuickVina 1	QuickVina 2	QuickVina 1	QuickVina 2
Min	0.99	0.45	0.95	1.39	1.02
Average	21.55	1.86	6.55	8.26	2.30
Max	415.62	21.04	38.96	26.61	20.49

2.4.3. A visual example of prediction quality

A good example of the quality of prediction of QuickVina 2 is 3PWW complex, shown in Figure 2-10. It has 20 active bonds and 49 heavy atoms. Although it was accelerated in a lower rate than QuickVina 1 (4.74 versus 15.36 times), QuickVina 2 succeeded with RMSD of 1.436, while Vina, GOLD, Dock, and QuickVina 1 (not shown) all failed with respective RMSD values of 2.032, 2.711, 5.352, and 6.067). In

Figure 2-10, it is easy to notice the closeness between QuickVina 2 prediction in Red and the experimental data in Cyan.

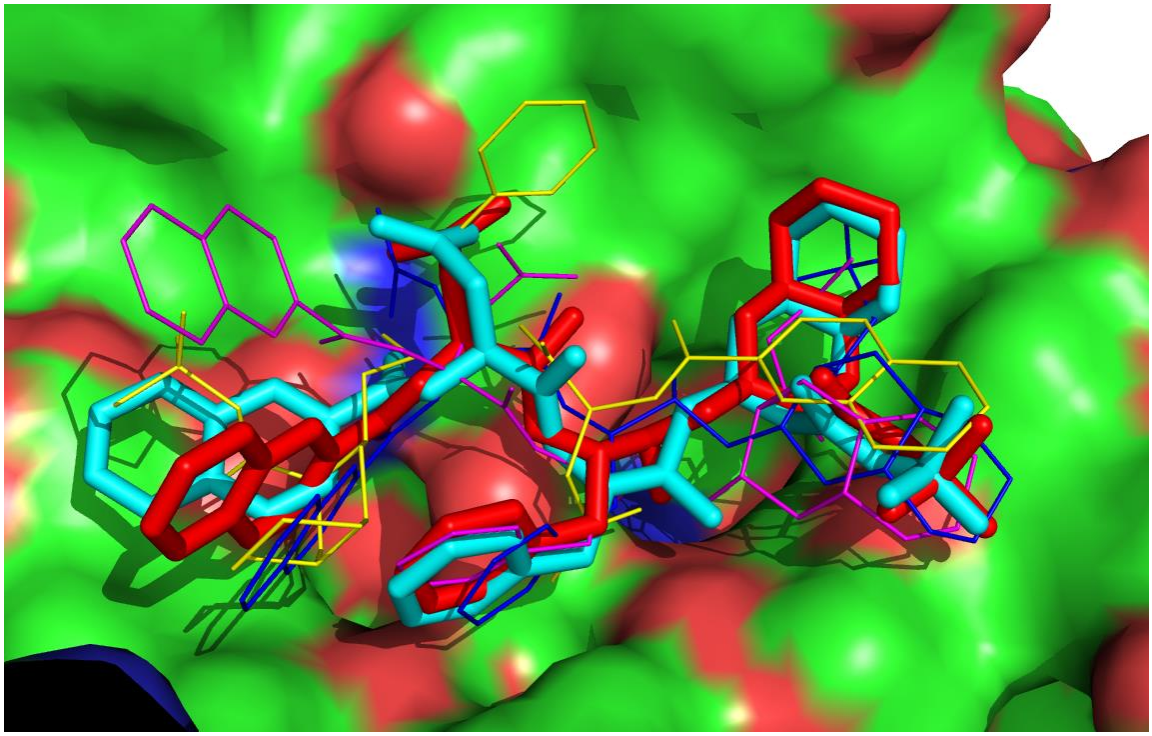


Figure 2-10 Experimental, Vina, QVina 1 and 2 first mode of 3PWW
Experimental (Cyan sticks), and QVina 2 (Red Sticks) are close to each other (except for the terminal double ring), while GOLD (Magenta lines) and Vina (Blue lines) conformations have flipped upper right and lower right branches and relatively distant left branch, and Dock (Yellow lines) has a totally flipped conformation. Part of the receptor was removed to show the ligands.

2.5. Conclusion

QuickVina 2 is a fast and accurate molecular docking tool. It was developed to overcome a limitation of QuickVina that rendered it unpractical as a docking tool in regular exhaustiveness values. Tested against 195 protein–ligand complexes that compose the core set of the 2014 release of the PDBbind using default exhaustiveness level of 8, QuickVina 2 successfully attained up to 20.49-fold acceleration over Vina. QuickVina 2 showed tendency for higher acceleration when the number of dimensions/variables increases. Meanwhile, 70% of predicted modes were equal to or better than original Vina, in terms of binding energy. The remaining 30% had average Energy difference only 0.58 Kcal/mol. The Pearson's correlation coefficient (r) between Vina's and QuickVina 2's binding energy

was 0.967 for the first predicted mode and 0.911 for the sum of all predicted modes. It is also witnessed that QuickVina 2 is more accurate than GOLD 5.2 and is only slightly less accurate than Dock 6.6. It paved the way for some high-throughput and sufficiently accurate virtual screening of molecular libraries. This in turn brings great value to the fragment-based computer-aided drug design.

2.5.1. Contribution in this chapter

- Replacing the inaccurate check condition in QuickVina which is based on the first order *first-order-necessity-check* with the more robust one which considers the *first-order-consistency-check* heuristic as well.
- Introducing QuickVina 2 as a fast and accurate molecular docking tool that works well in practically low computational resources., which restores QuickVina lost accuracy

2.5.2. Publications in this chapter

- "**Fast, Accurate, and Reliable Molecular Docking with QuickVina 2**".
  Alhossary, A., Handoko, S. D., Mu, Y., & Kwoh, C.K. (2015). Bioinformatics (2015) 31 (13): 2214-2216.

Chapter 3 Refining selection methodology before MRC docking to find a binder for Dengue Virus NS5 protein

In this chapter, I used QuickVina 2, to propose 16 drug fragments as potential inhibitor of the dengue Virus (DENV) Non-Structural Protein 5 (NS5). In addition, in this chapter, I show an example of how to choose the representative structures, upon which to do Multiple Receptor Conformation (MRC) docking, in the MRC docking step; in order to avoid false negatives and to ensure the diversity of results.

3.1. Project Motivation

The current presented drug design workflow was applied on the Dengue Virus because of two reasons: First is that in spite of the worldwide interest with the study of dengue fever, there is no known drug for dengue fever yet, and all the available treatment is symptomatic. The second reason is that there is currently ongoing laboratory research on this virus to discover more knowledge about its proteome and functioning. Consequently, this work would be just up to date with the latest discoveries.

3.2. Background

3.2.1. Disease

Dengue fever is a febrile illness, widely distributed in tropical and subtropical areas (Figure 3-2), transmitted by the bite of an infected female *Aedes aegypti* mosquito [112, 113]. Figure 3-1 shows the *Aedes aegypti* mosquito. *Aedes albopictus* is also a secondary vector of dengue fever that has spread from Asia to cold areas in North America and Europe due to its ability to tolerate and adapt to below-freezing temperatures [112, 113].

Figure 3-1 Aedes aegypti *mosquito*

The lethal form of Dengue fever is known as “Severe dengue”. It has become a leading cause of hospitalization and death among children of most Asian and Latin American Countries [112]. The WHO estimates that over 40% of the world's population, that is about 2.5 billion people, are at risk of dengue fever. It also estimates 50–100 million dengue infections worldwide every year. There is still no specific treatment for dengue fever [112].

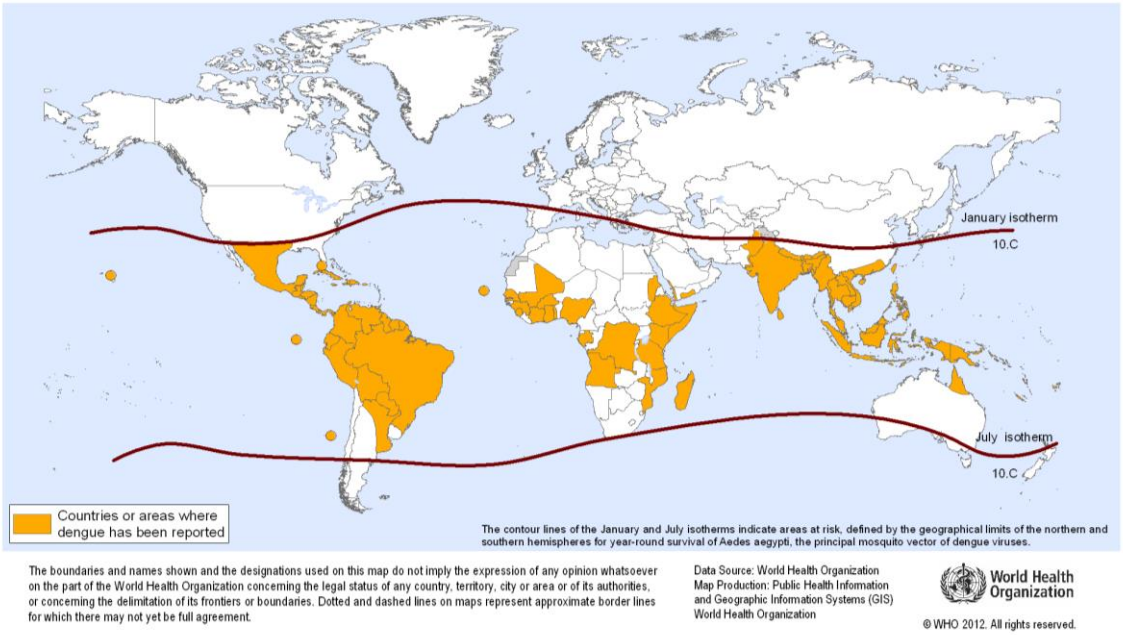


Figure 3-2 Countries or areas at risk of Dengue Fever (source: WHO)

3.2.2. Pathogen

The causative agent of Dengue fever is the Dengue Virus (DENV), which belongs to genus Flavivirus in the family Flaviviridae. It has four distinct serotypes (DEN-1, DEN-2, DEN-3 and DEN-4). The serotypes are closely related: They share about 65% of their genome. Once infected with one serotype, the patient develops a lifelong immunity against that serotype but not against the other three. A close successive infection with two different serotypes would impair the immune response in the second instance and may predispose to severe dengue through a mechanism known as antibody-dependent enhancement [114, 115].

3.2.3. Viral genome

The viral genome is composed of one positive-sense single-stranded (ss) RNA of about 11Kb. It contains type-1 cap-structure ($^{7Me}GpppA_{2'OMe}$) at the 5' end and no poly (rA) tail at the 3' end [116]. The viral genome is replicated by the virus' own RdRp enzyme.

The viral genome is translated into a single polyprotein. This polyprotein is further processed, co- and post-translationally, by both the host's proteases and the virus' two-component protease "NS2B-NS3pro". The products are three structural proteins (C, prM, and E), and seven non-structural proteins (NS1, NS2A, NS2B, NS3, NS4A, NS4B, and NS5) [117].

3.2.3.1. NS5 RdRp

NS5 is the largest (900 residues) and most conserved protein of the viral proteome. It has a Methyltransferase (MT) domain at its N-terminus (residues 1-296), and an RNA-dependent RNA polymerase (RdRp) at its C-terminus (residues 320–900) [118].

The RdRp, like all other flavivirus RdRp enzymes, has the *right-hand* configuration (Figure 3-3), which is composed of thumb, palm and fingers domains. This RdRp contains also two loops that protrude from the fingers subdomain and connect it to the thumb domain. They are called "finger tips", and may play a role in the global conformational change of the RdRp structure during the

initiation stage by transmitting the conformational changes that occur in the fingers to the thumb subdomains [119].

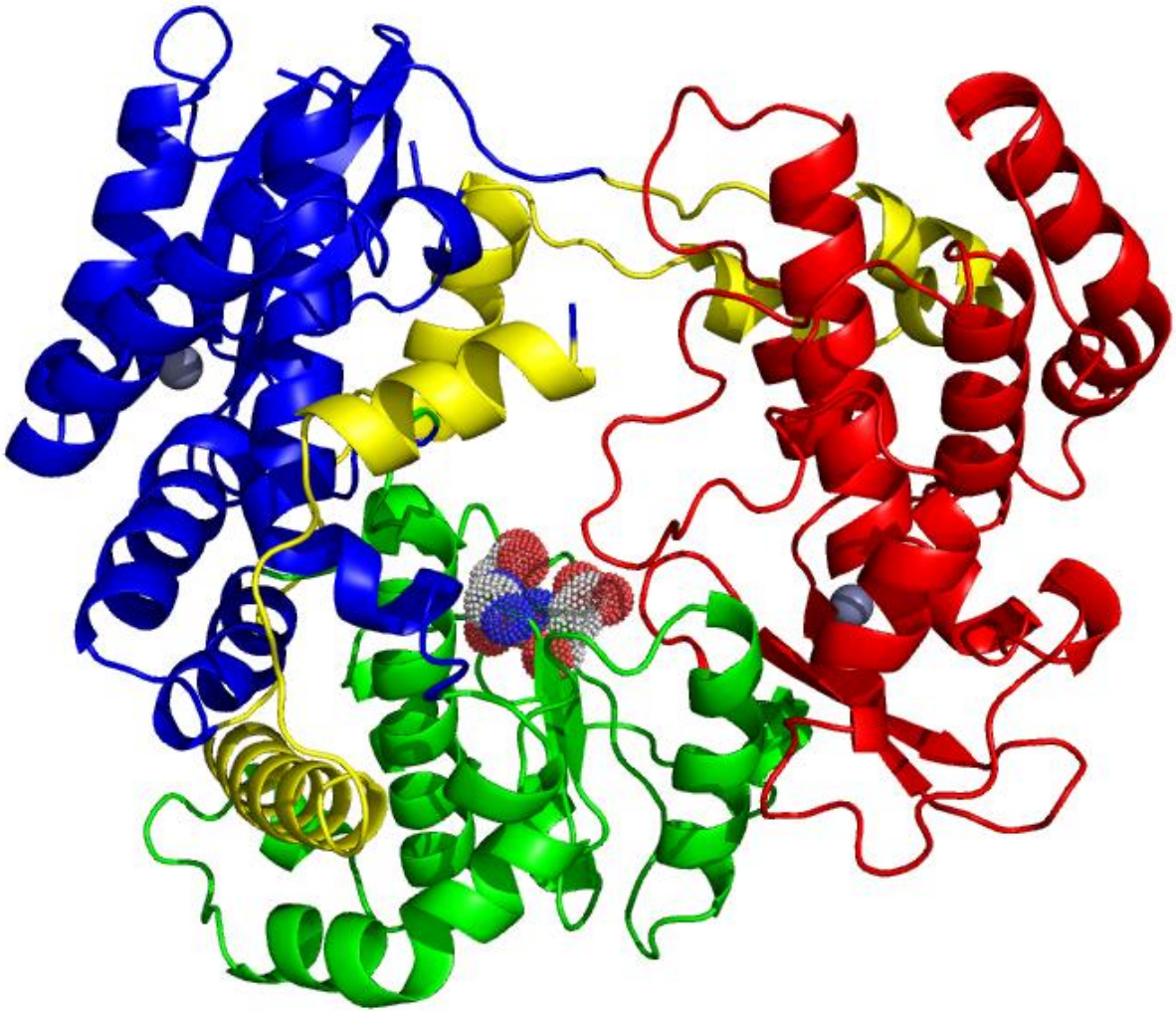


Figure 3-3 NS5 RdRp Domains
NS5 RdRp shows the right-hand configuration: Thumb domain (red), Fingers (blue), Palm (green). It shows also finger tips (yellow). Two Zinc atoms are shown as grey spheres. Also, the GDD motif characteristic for RNA polymerases is shown as dotted spheres.

3.2.4. In silico techniques for proposing drug fragments

Fragment based drug discovery (FBDD) offers an alternative approach to the traditional method of screening huge compound libraries, which often generate few hits [120], and many false positives. The most common computational techniques applied in FBDD include, fragment growing, fragment linking, and fragment based screening [121]. Fragment growing is an iterative process in which, at each step, a functional group or substituent is added to the fragment with the sole objective of

maximizing its interactions with the binding site residues [122]. A number of successes involving the fragment growing approach have been reported [123-127]. In fragment linking, separate fragments are linked to produce a more potent drug candidate [122]. As early as two decades ago, Hajduk, P. J. et al [128], linked together two different fragments to generate a much more potent compound. Fragment based screening involves screening a library of fragment compounds against the target to identify potential lead molecules. This approach which applies molecular docking has been successful in identifying new chemical entities for many biological targets [129].

3.3. Project objective

The principal objective of this study was to propose drug-like fragment compounds for the DENV using available computational Biology tools.

3.3.1. Protein selection

NS5 RdRp has a primer-independent activity. It uses the 5' ss RNA strand as a template to synthesize a 3' strand and then uses the 3' strand to synthesize 5' strands. It interacts simultaneously with 3' and 5' terminal regions of the template RNA [119]. The replication cycle is asymmetrical (the number of 5' strands produced is 10 folds the number of 3' strands) [130].

Because the viral RdRp enzyme does not exist in humans, and the polymerase activity is essential for viral replication, the viral RdRp is a good target for the development of drugs against Dengue virus [131].

3.3.2. Cavity Selection

Malet et al. [119], proposed two conserved cavities between DENV and WNV after structural analysis of flavivirus RdRp domains. Both are in the thumb subdomain of the RdRp and are called cavities A and B (Figure 3-4). They proposed these two cavities as potential targets for the development of small molecule inhibitors.

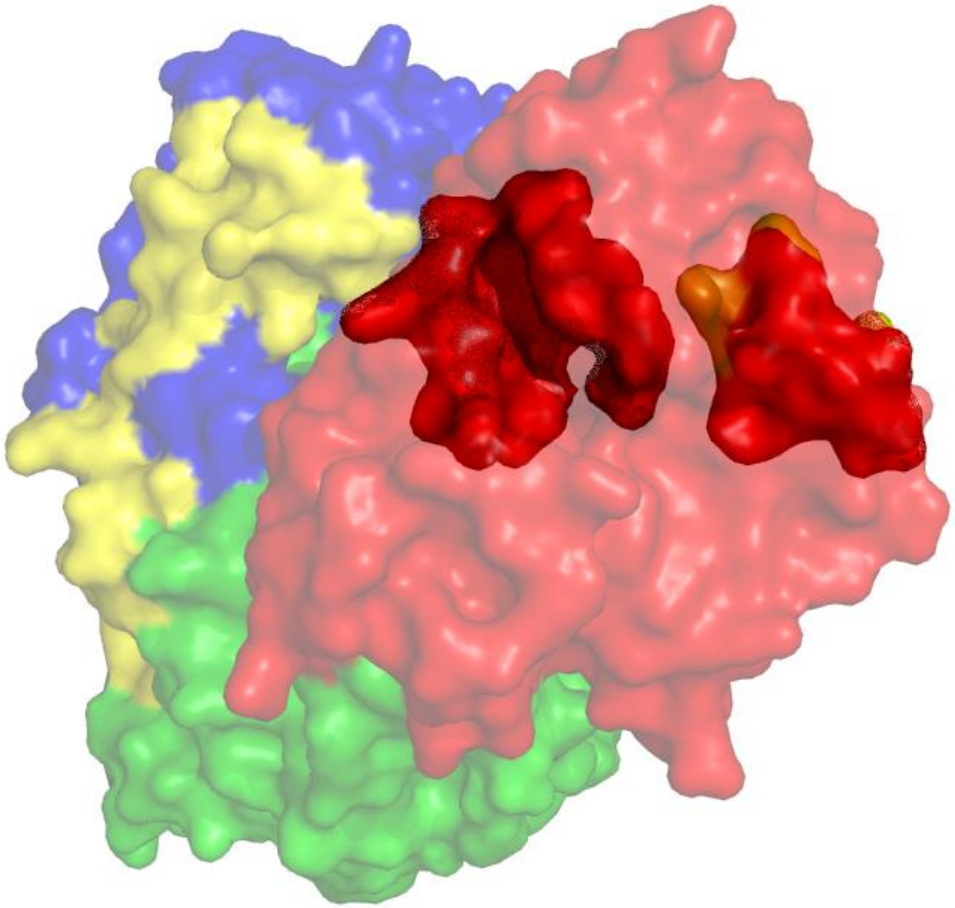


Figure 3-4 Cavities A and B
Cavity A (Left, highlighted Red) and cavity B (Right, highlighted Red and Yellow) shown within NS5 RdRp. All remaining parts are dimmed with color schemes the same as used in (Figure 3-3 NS5 RdRp Domains)

Zou et al. [118] then studied the biological relevance of cavities A and B using "mutagenesis" and "revertant analysis". They observed that out of the five mutations (L328A, K330A, K756A, W859A, and I863A) that proved to be relevant for the viral replication, only one (K756A) belongs to cavity A while the remaining four (L328A, K330A, W859A, and I863A) belong to cavity B. Three of them (L328A, W859A, and I863A) reduce the RNA synthesis on or before *de novo* initiation (not elongation). They also concluded that Lys-330 functions by interacting with the viral NS3 helicase domain. Thus, cavity B was recommended as a potential target for drugs against flavivirus RdRp. It was suggested that cavity B plays an essential non-enzymatic role in the process of DENV replication.

In the remaining sections of this chapter, I will be using DENV specific numbering, which is shifted by one residue from the West Nile Virus (WNV) numbering used above.

3.3.3. Closer view on cavity B

Cavity B is shown in Figure 3-5. The deepest part is shown in Yellow while its right and left boundaries are shown in Blue and Green respectively. LYS329 (shown in Red) is located at the entrance of the cavity and was observed in the simulations to flip in and out of the cavity. This phenomenon suggests that LYS329 could close and open the cavity to substrate or product.

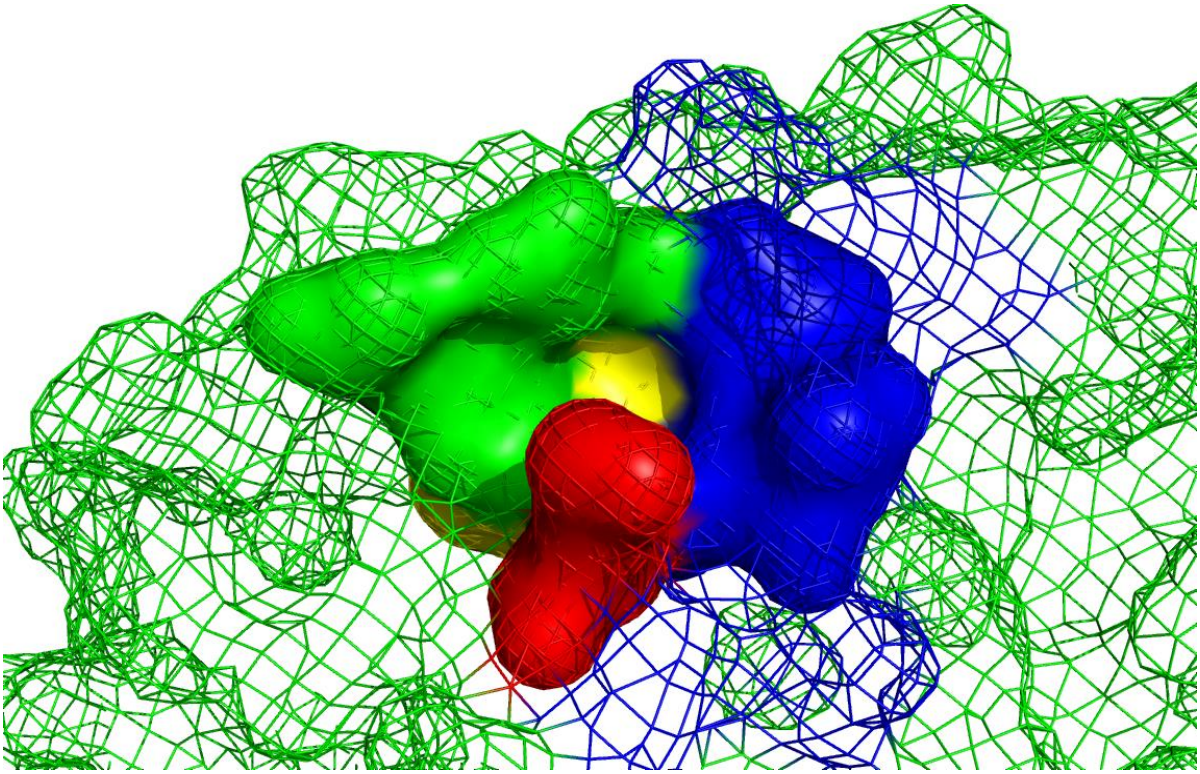


Figure 3-5 cavity B visually subdivided into areas deep, right, and left, and LYS 329
The deepest area of the cavity is colored in Yellow. Areas to the right and left are colored in Blue and Green. The cavity entrance is closed by LYS329 (in red).

In their publication, Zou et al [118], performed mutagenesis analysis on cavity B and showed that TRP859, located in the deepest part of the cavity, is the most important residue for viral replication.

3.4. Methods

3.4.1. Protein Preparation

The preparation started with the apo form of the DENV NS5 RdRp domain [PDB ID 4HHJ], whose missing atoms and loops were modelled in using SWISS-MODEL [132, 133]. The protonation states of histidine residues were determined using Chimera 10.2 [110].

3.4.2. Exploring the configuration space of the receptor by MD simulations

The intension was to -as much as possible- explore a large number of configurations of the receptor. As the crystal structure of the protein is a single snapshot of its dynamic spectrum, its plasticity was explored by performing MD simulations.

GROMACS (GROningen MAchine for Chemical Simulations) [19, 20] was used to simulate the protein. The apo form of the protein was solvated in an isotonic TIP3P water model, containing neutralizing Na^+ and Cl^- ions, in a cubic box using AMBER99SB-ILDN forcefield. Three trajectories, 40 nanoseconds each, were generated with different randomly assigned initial velocities. The initial 20 ns of each trajectory were omitted, and remaining parts were concatenated into a single 60 ns trajectory (F) consisting of 60,000 frames.

3.4.3. Selection of representative structures

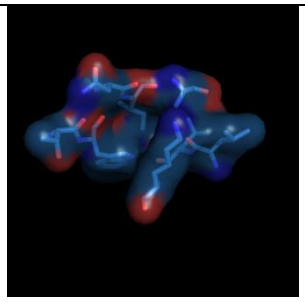
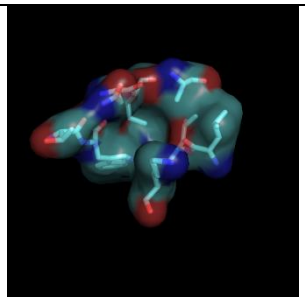
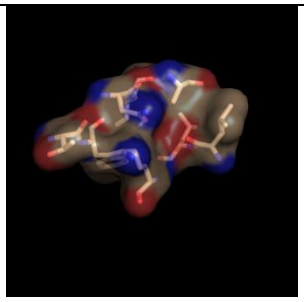
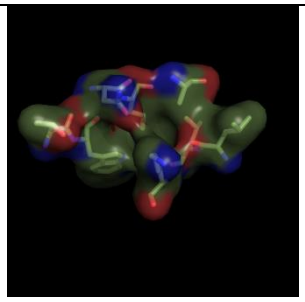
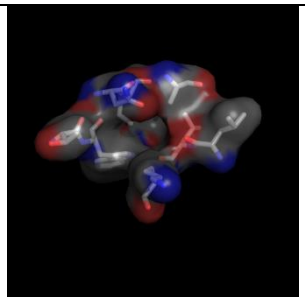
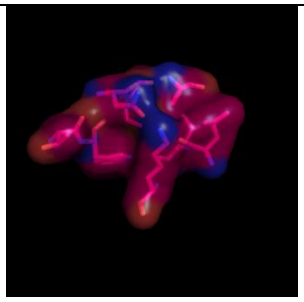
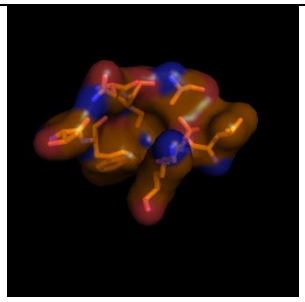
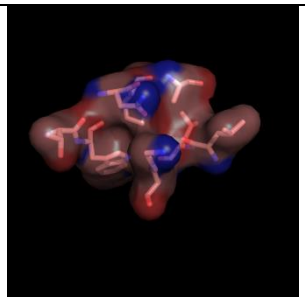
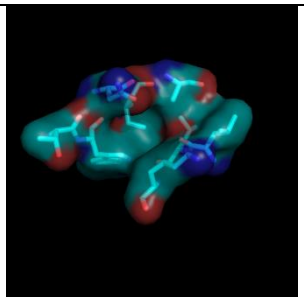
The multiple receptor conformation docking method described in Awuni and Mu paper [134] was extended here. In their method, three MD simulations Influenza A nucleoprotein (NP) were conducted. Then the initial parts of the trajectories were omitted, and the later parts were concatenated into one common trajectory. Then the receptor trajectory was clustered on the tail-loop binding pocket (the target binding site) for representative structures. All of those generated structures were directly used in the Multiple Receptor Conformation (MRC) docking exercises. Here, however, the binding sites of the representative structures are checked first for cavity boundaries, size and drugability prior to the docking process. Those that are not detectable, have too small size, or are not

expected to be drugable, are considered to be *noisy* and they were excluded, in order for decreasing the false negative rate, as described in the subsequent subsections.

3.4.3.1. Obtaining representative structures

On the basis of the Root Mean Square Distance(RMSD) of cavity B (as defined in [119]) heavy atoms, the combined trajectory was grouped into a set of clusters (C) using the GROMOS algorithm [135] and a cutoff of 0.1 nm. This algorithm orders the output clusters in non-increasing order of sizes. The middle structures of the top eight clusters (c_1-c_8), which contained 91.6% of the whole set were selected. Their respective cluster sizes are 33014, 7550, 4952, 3188, 2922, 2015, 1538, and 1317. These eight representative structures as well as the original crystal structure (c_0) were used for the next step of the process. Table 3-1 shows the nine conformations while Figure 3-6 shows the crystal structure (C_0) and two representative structures (C_7, C_8) overlapped.

Table 3-1 Cavity B in crystal structure (c0) and structues (c1-c8)

0 	1 	2 
3 	4 	5 
6 	7 	8 

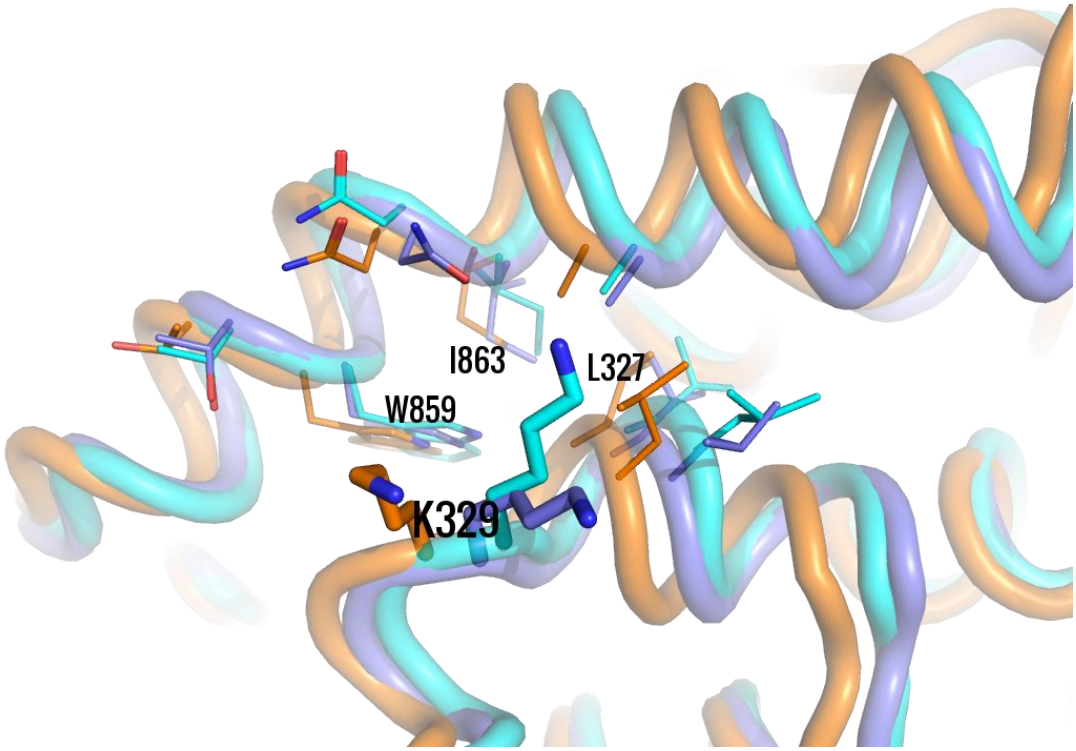


Figure 3-6 Cavity B in Crystal Structure (C0) and clusters (C7, C8)
Clusters C_7 (Orange) and C_8 (Cyan) are aligned to Crystal structure (Light Blue). They show the different confirmations of the cavity. LYS329 moves in different directions, opening and closing the cavity.

3.4.3.2. Filtering out noisy representative structures

The previous step produced nine conformations ($c_{0\text{-}8}$). However, not all of them are useful. By contrary, some of them are noisy and problematic. If those problematic conformations are kept, they would decrease the scores of important ligands, leading them to be incorrectly considered negatives (decreased sensitivity). The previously described method was extended by selecting only the conformations with predicted cavities that contain TRP859 residue mentioned earlier, to undergo further steps.

The nine conformations ($c_{0\text{-}8}$) were checked using the Fpocket program [16] for cavity boundaries, size and drugability. Drugability is a statistical measure in the domain [0, 1]. It represents the likelihood of a pocket to be drugable, where "0" means undrugable and "1" means very likely to be drugable. Values greater than or equal to 0.5 are considered likely drugable. The results are summarized in Table 3-2 below. The table shows that cluster c_6 was not detected at all, and that

conformations 1, 3, and 5 had detected cavities, however the deep area containing TRP859 was not detected those cavities. It also shows that the crystal structure (c_0) is detected as two separate cavities with the deep part connected to the left part. Structure c_7 passed the drugability threshold (0.58 > 0.5). c_0 has the biggest predicted cavity size of 756 Å^3, followed by c_7 with a cavity size of 733 Å^3. Therefore, the set of conformations 0, 2, 4, 7, and 8 was selected for virtual screening ($C' = \{c_0, c_2, c_4, c_7, c_8\}$) and c_7 was considered as the best candidate prototype for next steps of drug design and development.

Table 3-2 Summary of cavity B size and drugability score using FPocket

Structure / Measure	0	1	2	3	4	5	6	7	8
Right area Vol.	460						X		
Left area Vol.	756	492	566	491	267	411	X	733	496
Deep Area. Vol.		X		X		X	X		
Drugability Score	0.012		0.21		0.05			0.58	0.08

The structures 1, 3, 5, and 6 (shaded in gray), were not selected because they do not contain detected deep area. Structures 0, 2, 4, 7, and 8 were accepted for next step. Among them, structure 7 (highlighted in Green) was considered as their prototype.

3.4.4. Docking / Virtual Screening

The database used in this study is the Maybridge diversity Ro3 (rule of three) fragment library [87] which contains 2,783 small molecules (L). The library was prepared using Discovery Studio to produces all possible 4,511 restrained-rotation isomers (D and L rotamers)/protonation state conformations (R).

The virtual screening step was carried out using QuickVina 2, which was previously described in Chapter 2 [136]. The whole process is summarized in Figure 3-7.

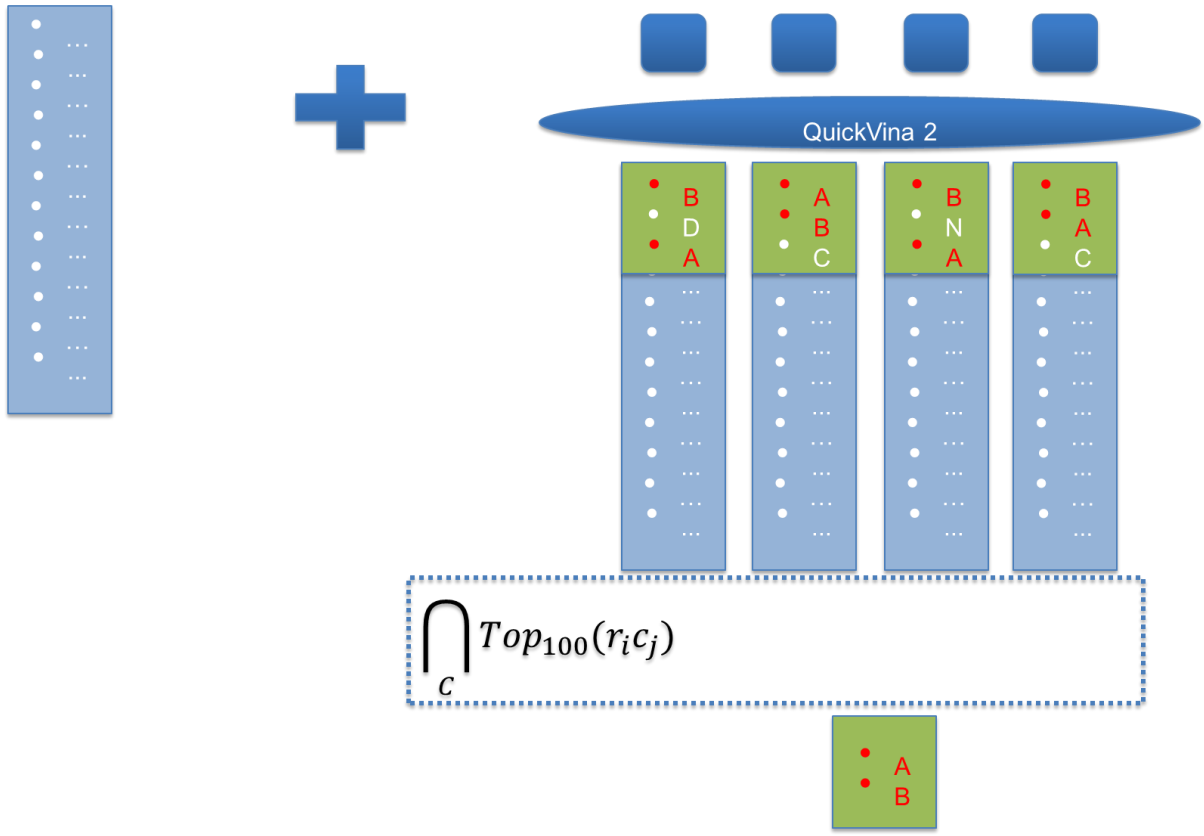


Figure 3-7 Multiple Receptor Docking process using QuickVina 2

After activating (enabling) all rotatable bonds of all rotamer / protonation state conformations ($r_i \in R$), the set *R* was docked against the full set of cluster representatives of the receptor ($c_j \in C'$). That gives a long list of results ($r_i c_j$). The set ($r_i c_j$) was ranked based on binding energy (E). E is calculated as the sum of Binding energy electrostatic force and Van der Waal's force (ES + VdW) components. Top H hits of the ranked lists are selected for next step, where the final set was the set of results that occur in all the selected top H sets ($\bigcap_{C_j} Top_H(r_i c_j)$) of all cluster representatives. H is usually set to 50/100/150/200. However, in this research, in order to study the progress of results, H started with low value as H= 50 and kept increasing in steps of 10, as described in the [Results and Discussion] section. Finally, setting H=100, the final set was ($\bigcap_C Top_{100}(r_i c_j)$). After the search was done using "QuickVina 2", it was repeated using "Vina", and both results were compared.

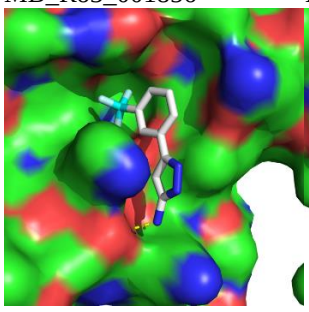
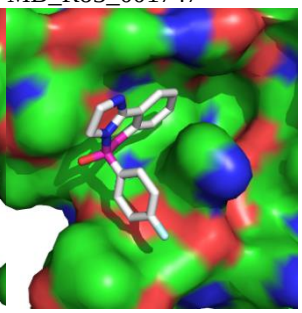
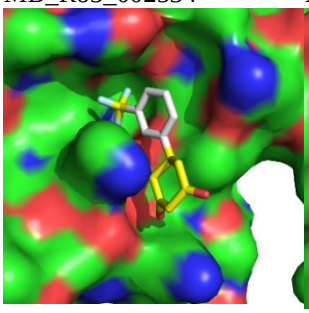
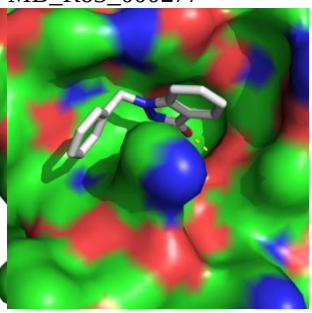
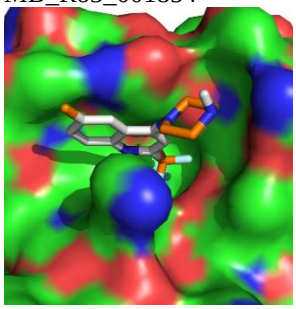
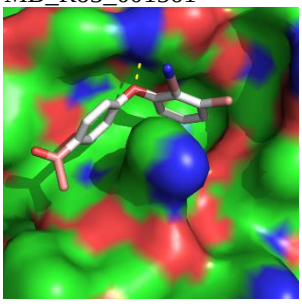
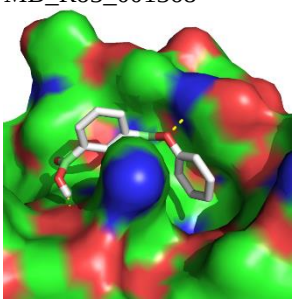
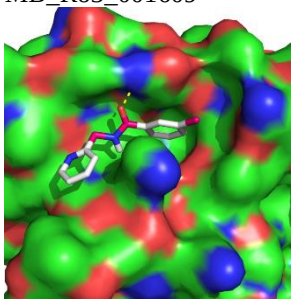
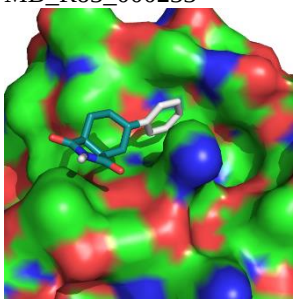
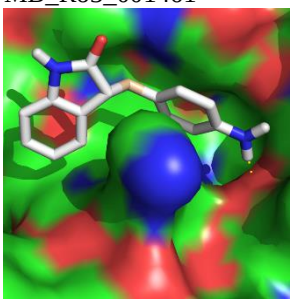
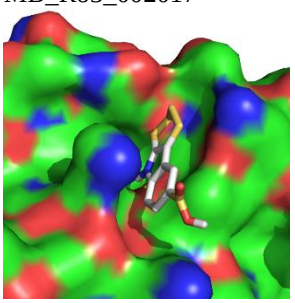
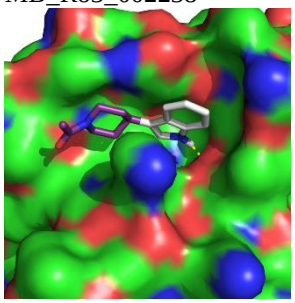
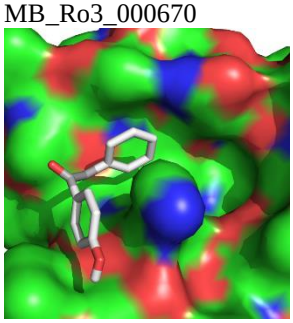
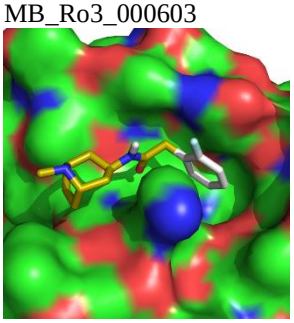
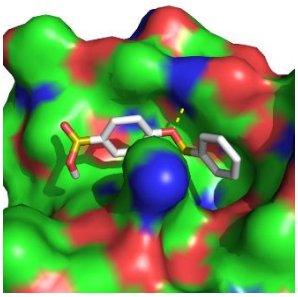
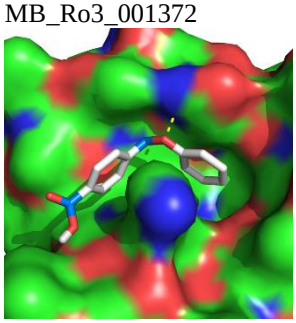
3.5. Results and Discussion

After virtually screening Maybridge, the intersection of the five ordered lists containing top 100 entries of every representative structure was compiled. The result contained 16 entries. They are listed in Table 3-3, ordered by average affinity. The process was repeated using AutoDock Vina 1.1.2. It was not surprising that Vina showed similar results, except that three hits were proposed by QuickVina 2 but not by AutoDock Vina. The three hits are highlighted in Yellow. Two hits were found to be flipped in order due to a change in the second decimal number. They are highlighted in Green. All the respective poses are shown in Table 3-4.

Table 3-3 listing of Selected ligands ordered by predicted affinities

Quick Vina 2			AutoDock Vina		
No	Ligand	Average affinity (Kcal/mol)	No	Ligand	Average affinity (Kcal/mol)
1	MB_Ro3_001856	-6.30	1	MB_Ro3_001856	-6.30
2	MB_Ro3_001747	-6.18	2	MB_Ro3_001747	-6.18
3	MB_Ro3_002534	-6.16	3	MB_Ro3_002534	-6.16
4	MB_Ro3_000277	-5.98	4	MB_Ro3_000277	-6.00
5	MB_Ro3_001561	-5.98	5	MB_Ro3_001561	-6.00
6	MB_Ro3_001854	-5.98	6	MB_Ro3_001854	-5.98
7	MB_Ro3_001368	-5.90	7	MB_Ro3_001368	-5.92
8	MB_Ro3_001609	-5.90	8	MB_Ro3_000233	-5.92
9	MB_Ro3_000233	-5.88	9	MB_Ro3_001609	-5.90
10	MB_Ro3_001461	-5.88			
11	MB_Ro3_002017	-5.84	10	MB_Ro3_002017	-5.82
12	MB_Ro3_002238	-5.8	11	MB_Ro3_002238	-5.80
13	MB_Ro3_000670	-5.80	12	MB_Ro3_000670	-5.80
14	MB_Ro3_000603	-5.80			
15	MB_Ro3_001839	-5.78			
16	MB_Ro3_001372	-5.76	13	MB_Ro3_001372	-5.76

Table 3-4 view of Selected ligands (as ordered on the right half of Table 3-3)

MB_Ro3_001856	MB_Ro3_001747	MB_Ro3_002534	MB_Ro3_000277
			
MB_Ro3_001854	MB_Ro3_001561	MB_Ro3_001368	MB_Ro3_001609
			
MB_Ro3_000233	MB_Ro3_001461	MB_Ro3_002017	MB_Ro3_002238
			
MB_Ro3_000670	MB_Ro3_000603	MB_Ro3_001839	MB_Ro3_001372
			

3.5.1. Selection of cutoff

Starting with H=50, which is a rather restrictive value of the cutoff for top H results, then using this value, to collect the common results among the top H values, only one result (ligand #2534) appeared to be common among all the nine available conformations (set C). Inspecting the docked result, Figure 3-8(A), it is noticeable that the ligand occupies the right and deep areas of cavity B, with its Trifluoromethyl group sifted in the deep area. This is fine; however, considering that this library is a (rule of three) library of drug (fragments), this is too little to fill the whole cavity.

On the other hand, applying the same value to the set of five selected conformations (set C`), five results are found. The five results include the same ligand from the counterpart set, another very closely related and overlapping ligand (#1856), as well as three ligands that occupy the left area of the cavity (277, 1561, and 1372) (Figure 3-8(B)).

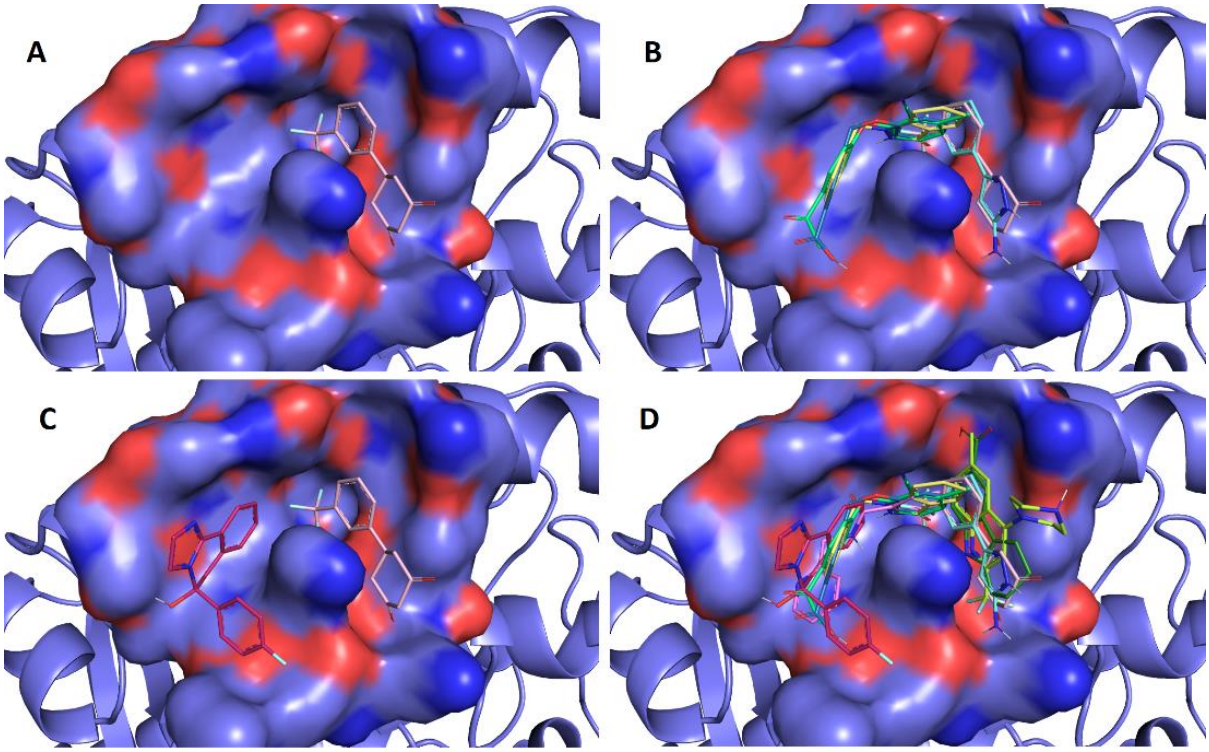


Figure 3-8 comparison between cutoff 50/60 among 5/9 structures sets
A) Cutoff 50 in 9 structures set: only one ligand selected. B) Cutoff 50 in 5 structures set: five ligands selected. C) Cutoff 60 in 9 structures set: only two ligands selected. D) Cutoff 60 in 5 structures set: nine ligands selected. The details of the ligands are listed in (Table 3-5 Order of result ligands among different cutofffs).

Upon inspecting the ligands 1856 and 2534, it is notable that they are closely related. Actually, the presence of both of them is important to enrich the diversity of options while missing one of them is a false negative. Relaxing the cutoff to 60, the set C produced only the two ligands shown in Figure 3-8(C) (1747, in addition to 2534 previously shown), while set C` produced 9 results (shown in Figure 3-8(D)). Relaxing the cutoff more, set C keeps producing only the same two ligands until the cutoff is relaxed to 90, while set C` results keep increasing (5, 9, 10, 12, …). Table 3-5 shows the progression of results of applying different cutoff values to both sets, C (with 9 conformations) and C` (with 5 conformations only).

To search for a candidate ligand, one usually starts with two reference values: one from a known positive control and the other one from a known negative control. Ten, one makes sure that the known positive and the known negative ligands were predicted as (true) positive and (true) negative respectively. The calculated value of the true negative binder can then be taken as the cutoff (exclusive), i.e. only what is above that value is selected. When there is no known negative control, one opts for two binders: One high positive control (with a high affinity) and one low positive control (with a low affinity) instead and consider the low positive one as the cutoff (inclusive), i.e. to select this ligand and all ligands that lay above it.

Unfortunately, one of the limitations of this system is that there is neither an experimental negative control nor a low positive control references. Therefore, the decision was to consider the lowest predicted ligand at the most restricted cutoff 50 (ligand # 1372) as our low positive, and to keep relaxing the cutoff, as long as it produces results **better** (more negative) than #1372 (the designated lower limit). Therefore, the final cutoff N was set to H=100.

Table 3-5 Order of result ligands among different cutofffs

	Order in 9 conformations							Order in 5 conformations						
	50	60	70	80	90	100	110	50	60	70	80	90	100	110
001856	-	-	-	-	3	3	3	1	1	1	1	1	1	1
001747	-	1	1	1	1	1	1	-	2	2	2	2	2	2
002534	1	2	2	2	2	2	2	2	3	3	3	3	3	3
000277	-	-	-	-	-	-	-	3	4	4	4	4	4	4
001854	-	-	-	-	-	-	-	-	5	5	5	5	5	5
001561	-	-	-	-	-	-	-	4	6	6	6	6	6	6
001609	-	-	-	-	-	-	-	-	7	7	7	7	7	7
001368	-	-	-	-	-	-	6	-	-	8	8	8	8	8
001461	-	-	-	-	6	7	8	-	-	-	9	9	9	9
000233	-	-	-	-	4	4	4	-	-	-	-	10	10	10
002017	-	-	-	-	-	5	5	-	8	9	10	11	11	11
000603	-	-	-	-	-	-	-	-	-	-	-	-	12	12
002199	-	-	-	-	-	-	9	-	-	-	-	-	-	13
000670	-	-	-	-	-	-	-	-	-	-	11	12	13	14
002238	-	-	-	-	5	6	7	-	-	-	-	13	14	15
001839	-	-	-	-	-	-	-	-	-	-	-	-	15	16
001372	-	-	-	-	-	-	-	5	9	10	12	14	16	17
001494	-	-	-	-	-	-	-	-	-	-	-	-	-	18
000511	-	-	-	-	-	-	-	-	-	-	-	-	-	19
Total	**1**	**2**	**2**	**2**	**6**	**7**	**9**	**5**	**9**	**10**	**12**	**14**	**16**	**19**

3.5.2. Shall all conformations be considered?

To answer this question, let us inspect the results in both cases: when we consider set C` (i.e. only the five receptor-conformations with detectable deep area), and set C (i.e. when considering all available conformations). Let us study a simple example in the current system. Ligand 1856 ranked number 1 when only the five conformations with detectable deep area in cavity B are considered. This rank started as early as cutoff 50, and it kept that position through all cutoff values. On the other hand, when all available conformations (including those noisy ones) were consider, that ligand did not appear on the results until the cutoff reached as high as 90. When it appeared there, it jumped to the top third rank and kept in that position.

Digging into the details, it was noticeable that ligand 1856 was already present in the top 50 in eight of the nine available conformations. It was only ranked 83rd (among top 90) when it was docked against conformation #6. It is noteworthy that this specific conformation is the noisiest one,

which does not feature any detectable cavity at all, because the cavity is obliterated by the movement of its edges as well as Lys 329 towards the opposite wall.

Figure 3-9(A) shows the docked position of ligand 1856 against conformation number 6 of cavity B. While Figure 3-9(B) shows the output of the same ligand docked against conformation 7 of the same cavity. The ligand is very marginal in Figure 3-9(A), while on the other hand, it shows perfect fit in Figure 3-9(B) with impacted head in the deep area of the cavity. Only because of that bad fit, the ligand did not appear in the results list when all of the available conformations were considered, although it was ranked number one when only drugable (non-noisy) structures were considered.

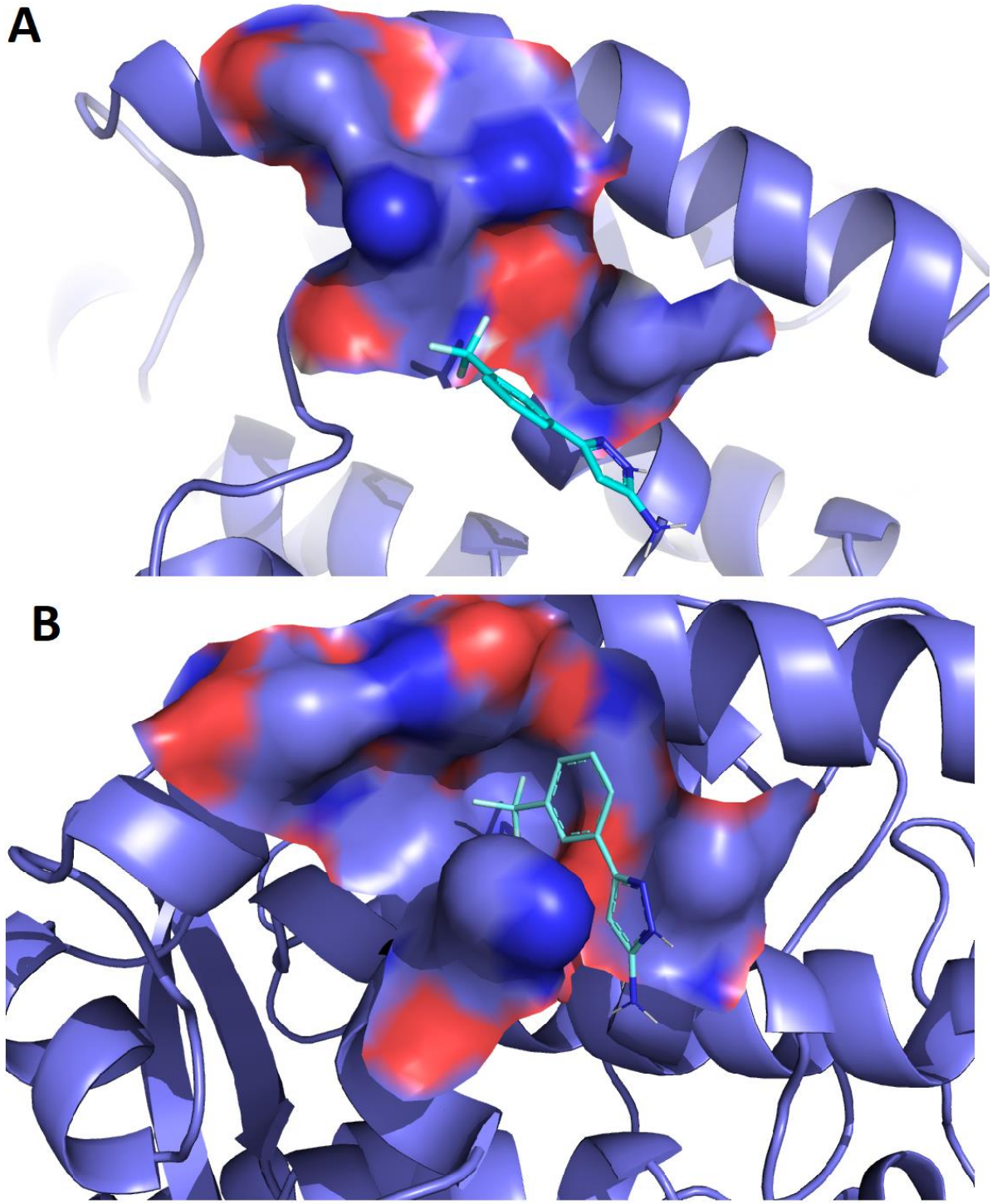


Figure 3-9 Ligand 1856 in conformations 6 and 7
Ligand 1856 in structure C6 (A) and in structure C7 (B). The cavity is shown as surface while the rest of the protein is shown as cartoon. A) The cavity is obliterated, and the ligand looks very marginal. B) The cavity is open, and the ligand is docked well in the right area with its Trifluoromethyl group impacted in the cavity deep area.

Therefor the answer of the question "shall all conformations be considered?" is No. We should consider only all drugable conformations, to enrich diversity, as showed in the previous section

[Selection of cutoff], and to decrease the false negative rate that may occur if extensive filtration criteria was applied as shown here. This answer is expected to be a general rule of thumb, not restricted to the DENV-NS5 RdRp (the current protein).

One final note remains: In their previous work, Awuni and Mu showed that considering multiple receptor conformations helps to reduce the false positive rate. The work done in this chapter, shows that considering only some of the available conformations help decrease the false negative rate (increase the sensitivity) that might occur when excess conformations are used for selection. However, although the method -even after the modification done here- is a versatile one, choosing which conformations to consider for multiple receptor conformation docking amongst all available conformations is still a challenging topic that has no general rule yet, and should be handled in a case-by-case manner.

3.5.3. Validation

The top-ranked ligand (ligand 1856) and one other ligand (ligand 233) ranked in the middle were further subjected to Molecular Dynamics simulation to study their binding stability. Each of the two ligands was simulated three times. Figure 3-10 shows the MD simulation results. The minimum distances of the ligand to the protein are collectively shown to the left. The RMSD of each ligand was calculated after least-square fitting the protein main chain and is shown to the right. For ligand1856, the minimum distance between the ligand and the receptor was stable at 2A in all of its three simulations, and the RMSD did not exceed 4.5 A. For ligand 233, although the RMSD exceeded 4A (up to 7A), the minimum distance to the receptor did not exceed 3A, which indicates that the ligand changed its binding pose but remained in contact with the protein. A snapshot of the pose of each ligand in the receptor is also shown.

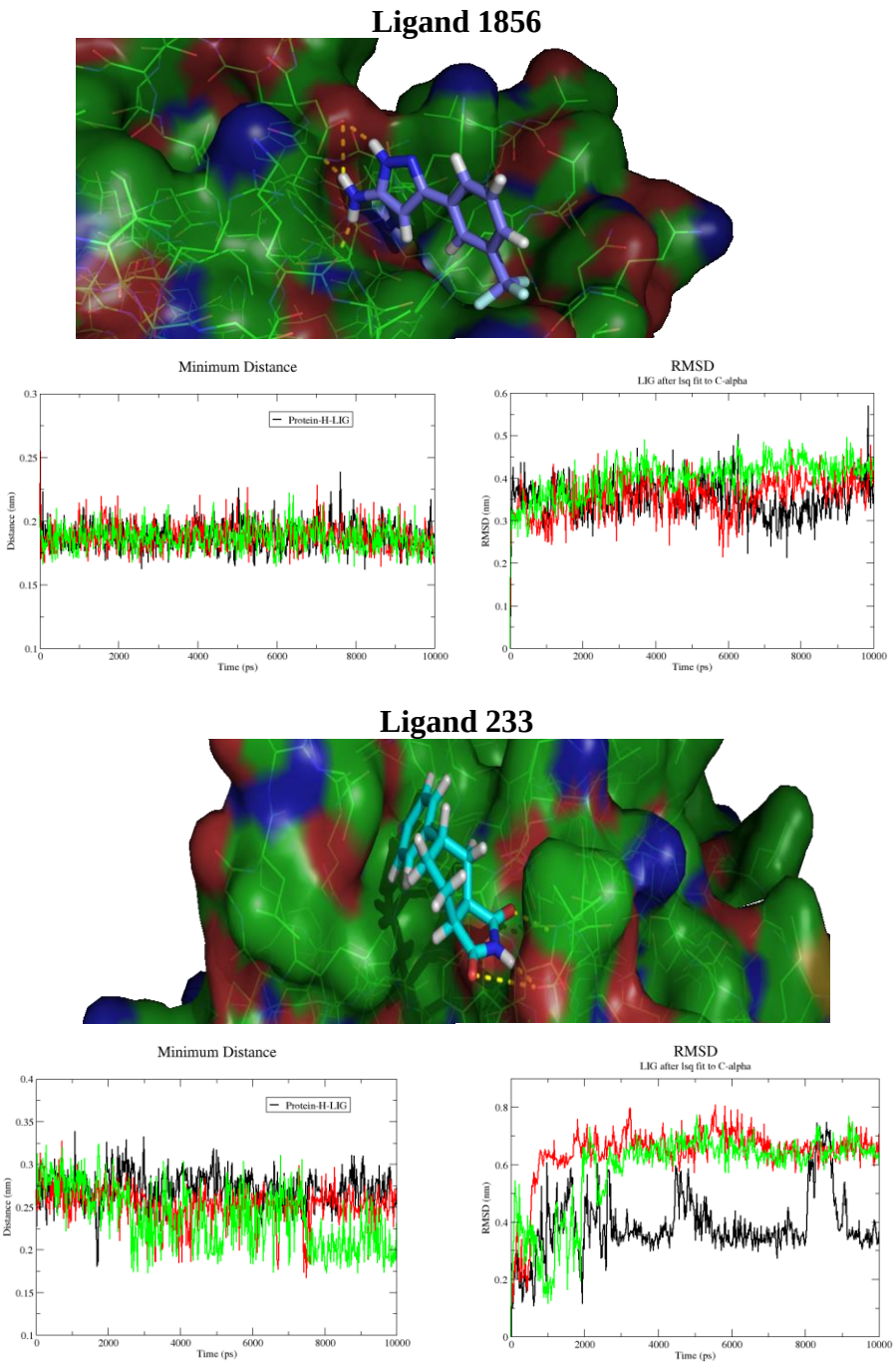


Figure 3-10 Simulation of Two proposed ligands
Three trajectories of two selected ligands MD Simulation results. Each ligand is simulated on three runs (black, red, and green respectively). The three minimal distances and the three RMSDs are shown (left and right).

3.5.4. Statistical significance of results

The search using this method is a sort selection without replacement. To calculate the statistical significance of the results, let's assume that the tool under the null hypothesis orders the ligands by random. In such a case, the probability of selecting a ligand on the basis of the success criterion of falling within the top 100 of a list is 100/T, where T is the total number of ligands. The probability of selecting any (specific) 16 ligands by random is then denoted by:

$$Pr\{S = 16\} = \frac{100}{T} * \frac{100}{T-1} * \ldots * \frac{100}{T-16-1}$$

With T=4511, the first term is 100/4511 ≈ 0.022168, and it keeps increasing in the subsequent terms, because the denominator keeps decreasing, up to ≈ 0.022242. The product of all terms is ≈ 3.4930E-27

The probability of reselecting the same 16 in 5 different lists (for 5 different clusters) by random is

$$Pr = \prod_{i=1}^{n} Pr_i^{S=16}$$

Therefore, the final probability product is ≈ 4.91E-424, which is highly unlikely under the null hypothesis (i.e. statistically significant).

3.5.5. Biological soundness of results

Sixteen ligand fragments were proposed. Visual inspection with pymol suggests that the binding modes of the molecules in Table 3-3 and Table 3-4 are reasonable, with hydrophobic parts sinking inside the pocket and hydrophilic/polar groups pointed towards the solvent.

The MD simulation results of the selected two ligands (ligands 1856 and 233) indicate that the minimum distances between both ligands and receptor are stable and maintained in a range of 0.2-0.25 nm. The RMSDs also show the tendency for stability except for one trajectory of ligand 233. These observations suggest that the proposed ligands are eligible for further experimental validation.

It is worth noting that the 16 proposed molecules are spread around the cavity sides. This presents the possibility of linking two or more of the fragments into one inhibitor with higher specificity and affinity. One example is illustrated in Figure 3-11, where MB_Ro3_001856 (in light blue) is shown on the right side of the cavity and MB_Ro3_001747 (in red) is on the left side.

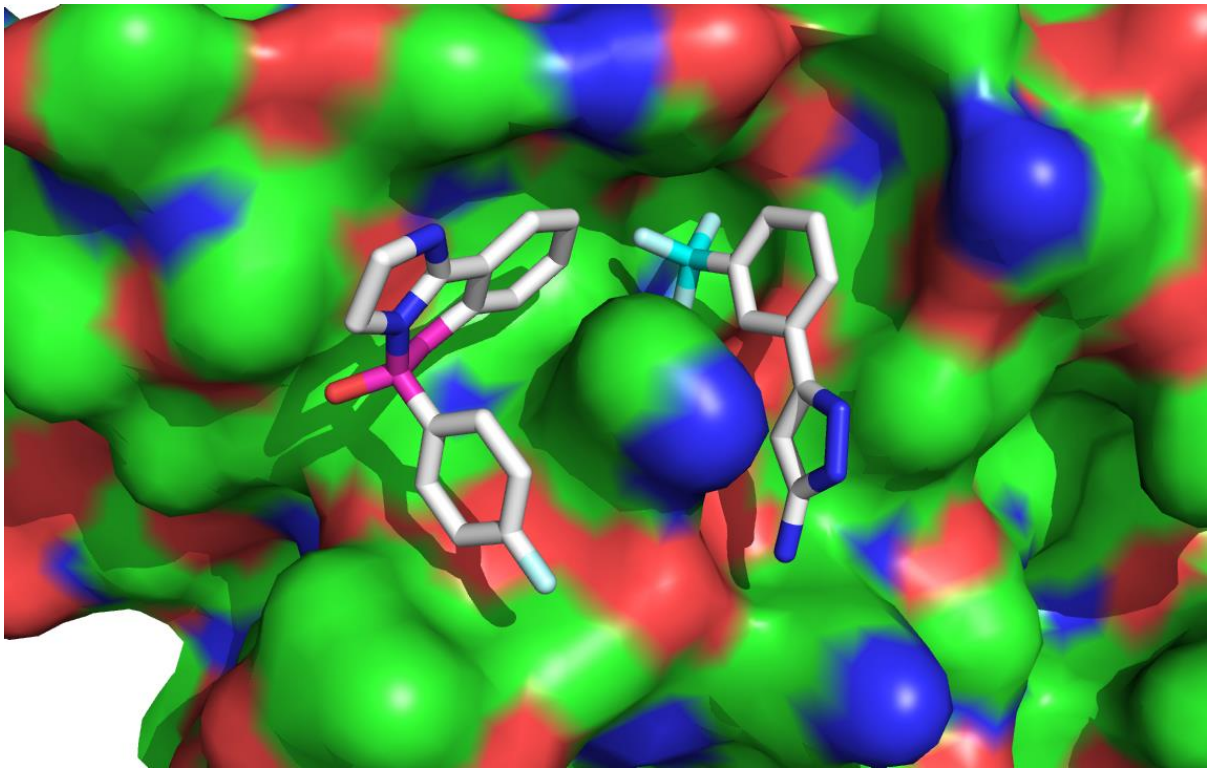


Figure 3-11 Two ligands on both sides of the cavity

It is also interesting to mention the similarity in the ligands. That makes identifying the pharmacophore, exchanging one alternative with the other, and flipping some functional groups from one ligand to the other a relatively easy task. Two examples are shown in Figure 3-12 and Figure 3-13.

Figure 3-12 shows overlapped ligands 1561 and 277. They share two benzene rings and one connecting atom in between. Ligand 277 has a hydrogen interaction with LEU61. Similarly, in Figure 3-13, ligands 1856 and 2534 are overlapped. It is easily noticeable that the CF_3 and adjacent Benzene ring are identical and that the difference exists only on the other side. This similarity enables substitution of atoms/ groups between the two alternatives. For example, the ligand 001856 contains a nitrogen atom which forms a hydrogen bond with D67 while 002534 does not.

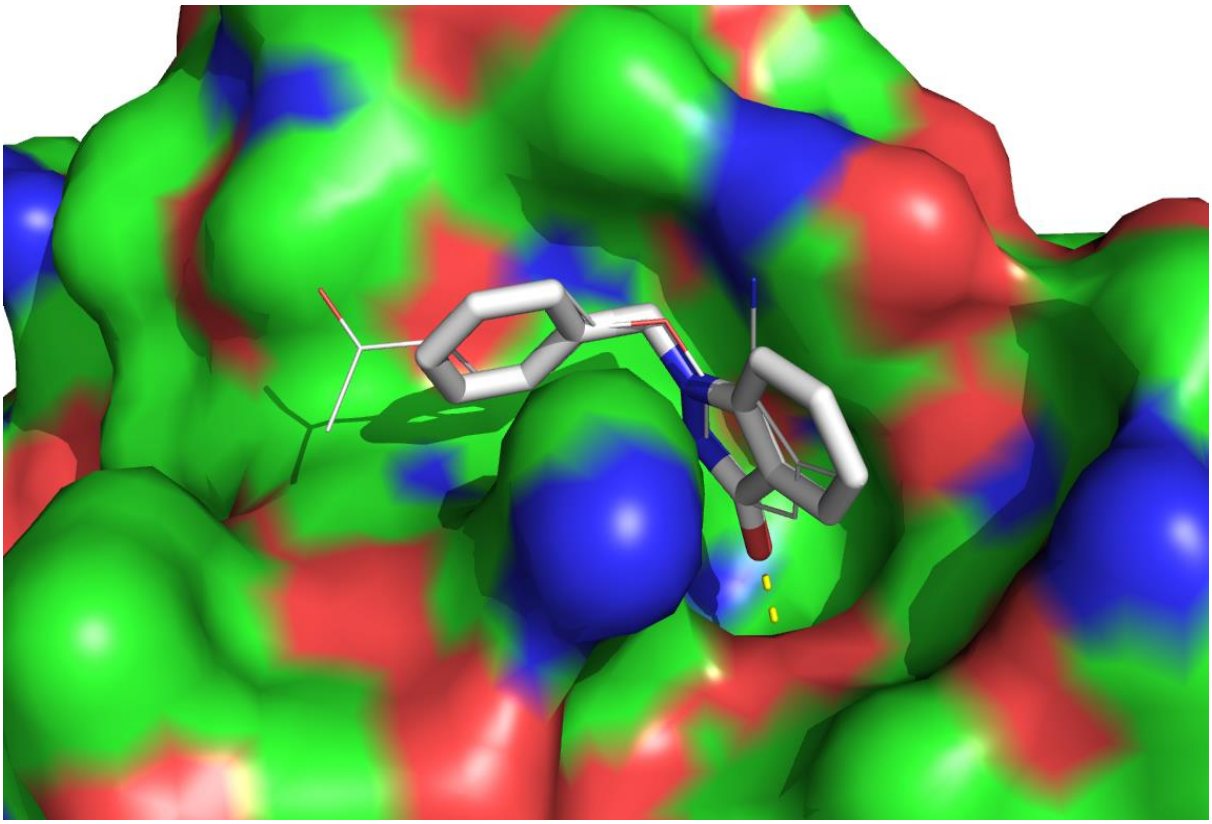


Figure 3-12 Ligands 1561 and 277 overlapped
Ligand 1561 (shown as sticks) and ligand 277 (shown as lines) have two overlapping rings and a connecting atom. Ligand 277 has a dual ring with a deep hydrogen interaction, while 1561 has only one ring and pointing out C≡N group.

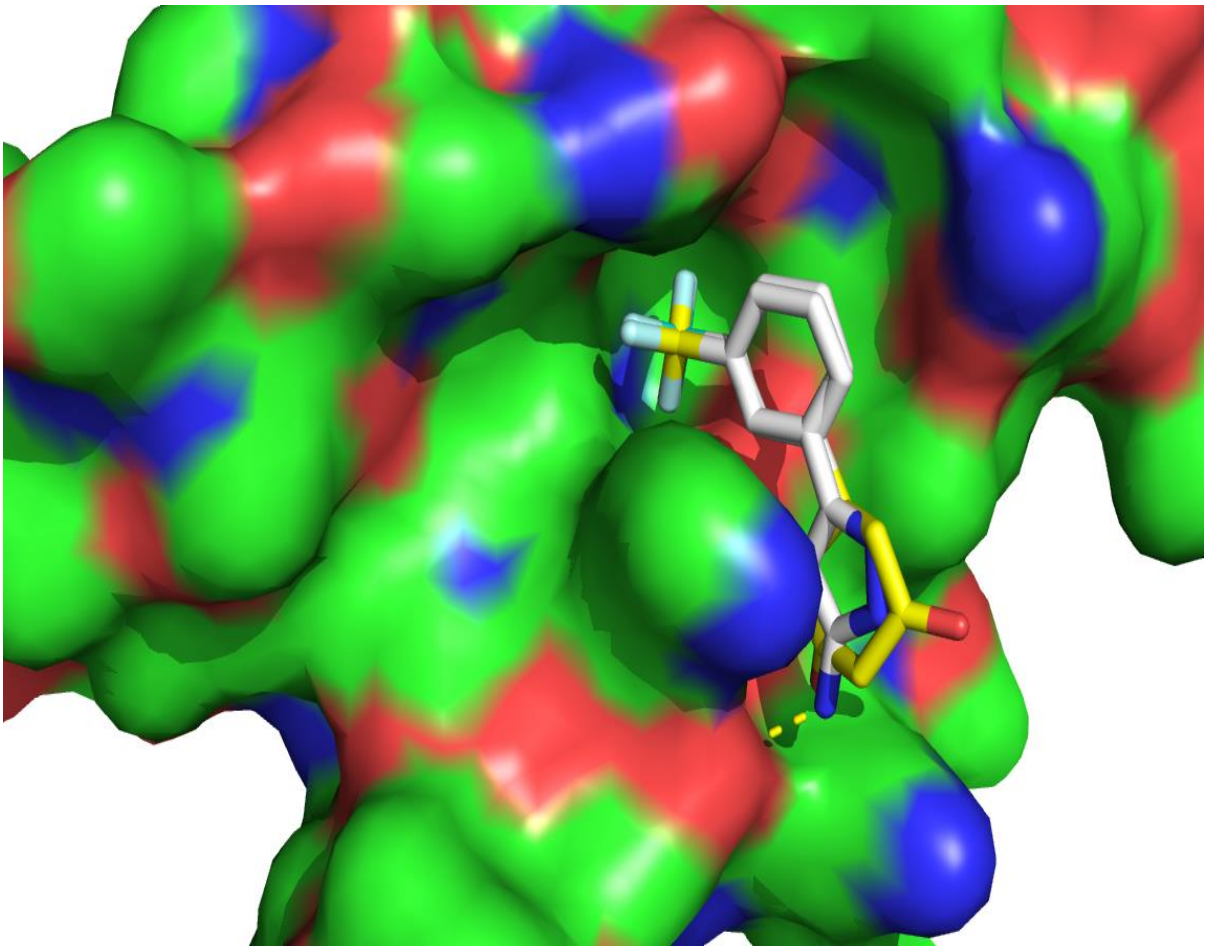


Figure 3-13 overlapping ligands 1856 and 2534
Ligands 1856 (in light Cyan) and 2534 (in Yellow) overlapped show identity in the CF_3 and adjacent Benzene ring, and diference in the free (more solvent exposed) ring.

3.6. Conclusion

In this chapter, drug design workflow similar to the one applied on the Influenza A nucleoprotein (NP) tail-loop binding pocket, described by Awuni and Mu was applied on Dengue Fever DENV3 NS5 protein and proposed 16 fragments. The methodology was extended here by selecting only some of the available conformations for multiple receptor conformation docking. The method can be applied to any small sized target pocket, not restricted to DENV-NS5 cavity B.

Both QuickVina 2 and AutoDock Vina detected the same 13 fragments with slight differences in their estimated Binding energies while QuickVina 2 detected three additional fragments.

The top-ranked fragment and one other fragment ranked in the middle were subjected to MD simulations for in silico validation. The simulation results suggest that the proposed ligands are plausible and could be considered for further computational and experimental validation, as well as lead optimization.

3.6.1. Contribution in this chapter

- Refining the selection criteria of target receptor conformations used in multiple receptor conformation docking method that searches for a "true binder", to decrease its false negatives.
- Proposing 16 fragments for Dengue virus.
- Providing a showcase for QuickVina 2.

3.6.2. Publications in this chapter

Conferences

- "**Proposing Drug Fragments for Dengue Virus NS5 Protein**".
  Amr ALHOSSARY, Yaw AWUNI, Chee-Keong KWOH, and Yuguang MU. Oral presentation in "GIW/Bioinfo 2017".

Articles

- "**Proposing Drug Fragments for Dengue Virus NS5 Protein**".
  Amr ALHOSSARY, Yaw AWUNI, Chee-Keong KWOH, and Yuguang MU. Journal of Bioinformatics and Computational Biology, (2018) 16 (3).

Chapter 4 Blind Docking with inter-process spatio-temporal integration

In this chapter, I will describe how I added the size dimension to QuickVina 2 to enable it to perform blind docking over the whole surface of a target protein. My work in both this chapter and Chapter 2 was focused on the sampling algorithm, while the scoring function was not modified.

4.1. Introduction

4.1.1. Virtual Screening

In the in-silico drug discovery domain, Virtual Screening is designed to produce and screen drug candidates more effectively than the physical assessment of thousands of diverse compounds a day, using high-throughput screening robotics, and thus increasing the rate of drug discovery while reducing the need for expensive laboratory work. Molecular docking is the core of virtual screening. It aims at the prediction of the modes and affinities of non-covalent binding between a pair of molecules. Oftentimes, the molecules consist of a macromolecule (the receptor) and a small molecule (the ligand). The multidimensional search space of the ligand includes the degrees of freedom of its translation, rotation, and torsions of the flexible bonds that may exist within it. Some packages consider flexibility in the receptor as well [83, 137, 138].

A successful docking application needs to have two pillars: 1) a method to explore the ligand-receptor conformation space for plausible poses [the search algorithm], and 2) a method to relatively order those plausible poses [the scoring function]. In a recent study, Wang et al. performed a comprehensive evaluation of ten famous currently available docking programs, including five commercial and five academic programs. Wang et al. studied their accuracies of binding pose prediction (sampling power) and their binding affinity estimation (scoring power) and concluded that AutoDock Vina [83] has the highest scoring power among them [84].

AutoDock Vina utilizes a powerful hybrid scoring function (empirical + knowledge-based) and employs an evolutionary search, for the minimum-energy docking conformations (solutions). The search process in Vina is performed as iterations of 1) global optimization in the form of modified Monte Carlo [139], performed on initial seeds (pseudorandom points), followed by 2) local optimization with BFGS method [104]. The modified Monte Carlo search is to perform a cycle of what *we* call an "essential" local optimization first before testing the proposed point according to the Metropolis acceptance criteria. The dimensions of the search space in Vina family include three translations and three rotations of the ligand (applied at its root), as well as the torsion angles of all active (rotatable) bonds within it or within the receptor. That is to say, the number of degrees of freedom (N) in Vina is 6+ number of rotatable bonds. Quick Vina [102] was developed to speed up Vina using heuristic to save local optimization by trying the potentially significant points only. A "potentially significant point" is a point that is expected to undergo optimization through a new pathway not explored by other points before. The technique is to check any provisional point against the search thread history of visited points and accept only the points where there is at least one (near) history point such that for each design variable pair (provisional point – history point variable pair), the partial derivatives of the scoring function with respect to the variable at both points either have opposite signs or one of them is zero. This means that accepted points are assured to be in a new unexplored energy well. QuickVina 1 uses less search time than the original Vina does, however it was designed to run on impractically big number of CPUs to overcome the high rate of false negatives. QuickVina 2 (also referred to as 'QVina 2' in equations, figures and their captions) [136] restored the lost accuracy of QuickVina (compared to Vina) by using a more robust test that considers the ***first-order-consistency-check***. Vina, QuickVina 1 and 2, depend on multiprocessing to achieve fast search, where several threads traverse the search space simultaneously.

4.1.2. Blind Docking

Blind Docking refers to docking a ligand to the whole surface of a protein without any prior knowledge of the target pocket. Blind docking involves several trials/runs and several energy calculations before

a favorable protein-ligand complex pose is found. However, the number of trials and energy evaluations necessary for a blind docking job is unknown. In their paper, Hetenyi, and Van Der Spoel recommended a number of trials to exceed 100 times, and at least 10 million energy evaluations per trial in case of flexible ligands [140].

When it comes to Blind Docking, most -if not all- of the famous (non-exhaustive) docking tools are quite limited. That is because the stochastic nature of search for a fixed number of steps makes it unlikely to sample the whole energy landscape surface thoroughly enough to find all the important poses. Researchers usually mitigate this issue by either reducing the search complexity (by splitting the docking box into multiple boxes [141, 142] or sacrificing the flexibility of some parts of the ligand [143]), or repeating the search several times using different seeds [144], and in both cases, they later merge the results together manually.

4.1.3. Interprocess Communication

Interprocess Communication – or inter-process communication- (IPC), is how an Operating System allows the running processes to share their data and exchange messages between them.

4.1.3.1. Processes and threads

A “thread of execution” is the smallest sequence of programmed instructions that can be managed independently by a scheduler, which is typically a part of the operating system [145]. This can be ordered over several levels (Processes, Kernel threads, User threads, and Fibers). Multiple threads usually exist as multiple instances of the same code (sharing their address space) and manipulate the same data, as subsets within the same process. Multiple processes – on the other hand- are usually independent, from different code segments (not sharing their address space), and consequently carrying more state information.

The implementation of hierarchy of levels of threads of execution (mainly between threads and processes) varies among operating systems. Some systems (e.g. Windows NT and OS/2) are said to have “cheap threads and expensive processes”; while other operating systems do not show such

great difference except for the cost of address space switch. In Linux for instance, despite the fact that threads are grouped together in front of the user's eye, they are separate processes from the kernel perspective. After initializing the memory pages, the kernel has no idea, and it doesn't really care what they are: They are scheduled independently, and they have nothing in common other than some overlapping memory pages.

4.1.3.2. Interprocess versus interthread communication

Processes, being separate independent in origin, can interact only through system-provided inter-process communication mechanisms, while threads, being of a homogenous origin, can communicate directly as well as indirectly, using system-provided methods.

4.1.3.3. Approaches

Processes usually communicate using some of these forms

1) File / Memory Mapped File: Files are records, generated by a file server and persisted on the hard disk. They can be accessed concurrently by multiple processes. Memory mapped files are files which are mapped to the RAM. Therefore, they can be addressed through memory addresses rather than writing to streams.
2) Signals: system type messages generated by one process and aims at controlling another one. For example, a web browser user interface may ask the downloading thread to start or stop downloading some media file.
3) Pipe / Named Pipe: A pipe is a unidirectional producer-consumer data channel. It can be used to stream data from the producer (the writing) side to the consumer (the reading) side. A named pipe is a pipe that is implemented through file system rather than streams.
4) Socket: Server/client model of interaction, using networking interface. Suitable also for communication between different nodes in the same network.
5) Message queue: Data stream that allows several processes to read and write to the same stream.

6) Message passing: Message passing allows multiple programs to communicate using message queues and/or non-OS managed channels. Message Passing Interface (MPI) standard is a famous example.
7) Shared memory: Same memory block (usually of the free heap) can be accessed by two or more processes. Careful synchronization should be considered to avoid race conditions.

4.1.3.4. Synchronization

Working in a multithreaded environment, there is no guarantee that the read/write operations will take place in a non-destructive way, nor that the messages will reach their destinations in the same order of sequence of transmission from either a single source or multiple concurrent sources. Synchronization is the method of ensuring sufficient control on messages flow.

All methods of concurrency have common objectives: 1) ensuring sequence, 2) eliminating redundancy, 3) ensuring data integrity and avoiding destructive writing (race condition), and 4) avoiding dead locks. Most common methods of synchronization are: 1) Semaphores, 2) Mutex Locks, 3) Barriers, and 4) Monitors [146].

4.1.4. This work

This chapter shows that enabling inter-process communication would enhance both the accuracy and the speed of the decision-making process, and that such enhancement is directly proportional to the amount of the accumulated common wisdom, among all the threads. Based on that, a new docking tool, QuickVina-W (referred to as QVina-W in equations, figures and their captions) was introduced. It is suitable for blind docking, eliminating the need to run the docking tool several times or to split the docking box and then to aggregate the search results.

4.2. Theory

4.2.1. Hypothesis

The philosophy behind QuickVina[102, 136] is to optimize local searches (the most time-consuming search step) only to potentially significant points, by means of keeping track of the visited points in the search history and examining every new potential point against up to P history points before it is accepted and allowed to undergo local optimization. This was perfect for relatively small search spaces. However, it is quite limited for large-sized search space, because the search threads are diluted over the huge search volume, and hence inefficient sampling takes place.

The core of enabling wide docking box search is substituting the first few checks against a thread history with checks on a high-quality collection of common history points. That is to say, the number of checks (P) that a potential point undergoes in QuickVina is split into two steps: The first step, Global step (G), is to check a small number P_1 (<< P) of high quality points from all available threads history. The second step, Individual step (I), is the normal QuickVina 2 check against thread's individual history points P_2 (= P - P_1). This way,

- Having history from *other* threads allows making use of other threads experience and make decisions in *already explored* energy landscape areas, while having history from an *individual* same thread allows making decisions in *virgin* areas.
- For a ***significant*** point, starting with P_1 check decreases the number of checks needed before accepting the point (increased decision-making speed).
- For an ***insignificant*** potential point on the other hand, having the number of high quality checks, P_1 kept to a considerably small value, leaving *large enough* number of history points to check in P_2 before rejecting a point, ensures *confidence* that this rejection is not due to a false negative, i.e. there is no compromise in search accuracy / ability to find minima. Going through a full set of P checks then rejecting a potential insignificant point is faster than sending

it to a set of unnecessary iterations of local optimization, the most time-consuming step of the search (increased search speed).

- The more the time to pass, the more the high-quality points to accumulate in the global history, the faster (and the more accurate) decision to take in the G stage. This will end up with each thread being either thoroughly exploring unexplored areas or just traversing explored areas.
- The increased speed allowed scanning more points in the same period of time, which increases the overall search accuracy of the tool.

Figure 4-1 shows the flow chart of the three tools (Vina, QuickVina 2, and QuickVina-W).

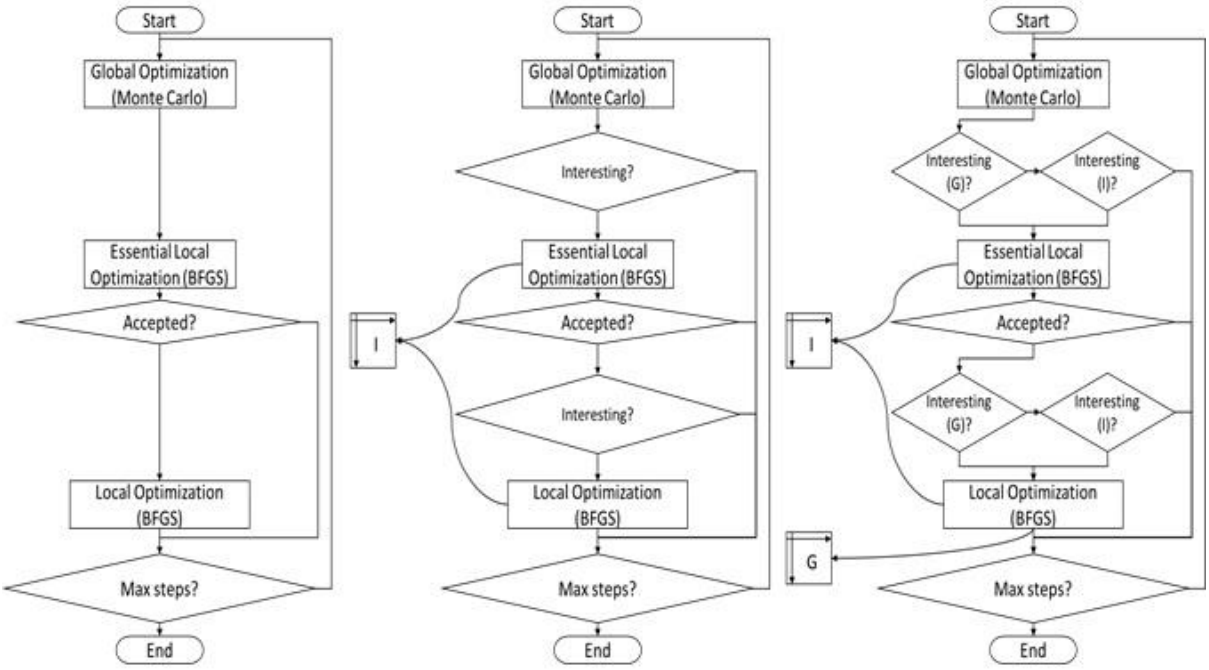


Figure 4-1 Flowchart of Vina, QVina (1/2), and QVina-W

The pseudocodes of Vina, QuickVina 2 and QuickVina-W are shown in Figure 4-2.

```
Listing 1: Optimization in Vina
Function Search(initial point)
Input:  An initial point to seed the seach
Output: Last point in the stochastic optimization search
Effect: Adding some solutions to output vector

FOR STEP = 1 TO Max_Steps
    CALL global optimization on point X
    CALL BFGS for Essential Local Optimization
    IF X2 is accepted by metropolis check
        CALL BFGS for Essential Local Optimization
        If X3 is better than previous point
            ADD X3 to output
        END IF
    END IF
END FOR
```

```
Listing 2: Optimization in QuickVina 2
Function Search(initial point)
Input:  An initial point to seed the seach
Output: Last point in the stochastic optimization search
Effect: 1) Adding some solutions to output vector
        2) Adding all accepted pathway to individual
           buffer

FOR STEP = 1 TO Max_Steps
    CALL global optimization on point X
    IF X1 NOT interesting according to I check
        CONTINUE
    END IF
    CALL BFGS for Essential Local Optimization
    ADD path X1-X2 to INDIVIDUAL History
    IF X2 is accepted by metropolis check
        IF X2 NOT interesting according to I Check
            CONTINUE
        END IF
        CALL BFGS for Essential Local Optimization
        ADD path X2-X3 to N History
        If X3 is better than previous point
            ADD X3 to output
        END IF
    END IF
END FOR
```

```
Listing 3: Optimization in QuickVina-W
Function Search(initial point)
Input:  An initial point to seed the seach
Output: Last point in the stochastic optimization search
Effect: 1) Adding some solutions to output vector
        2) Adding all accepted pathway to individual buffer
        3) Adding last local optimization point to global
           buffer

FOR STEP = 1 TO Max_Steps
    CALL global optimization on point X
    IF X1 NOT interesting according to G check
    AND X1 NOT interesting according to I check
        CONTINUE
    END IF
    CALL BFGS for Essential Local Optimization
    ADD path X1-X2 to INDIVIDUAL History
    IF X2 is accepted by metropolis check
        IF X2 NOT interesting according. to G Check
        AND X2 NOT interesting according to I Check
            CONTINUE
        END IF
        CALL BFGS for Essential Local Optimization
        ADD path X2-X3 to N History
        ADD X3 to G History
        If X3 is better than previous point
            ADD X3 to output
        END IF
    END IF
END FOR
```

Figure 4-2 Pseudocode of Vina, QVina (1/2), and QVina-W
Pseudocode of Vina, QVina 2, and QVina-W shown from left to right respectively

To illustrate the theory; consider a search space with (n) number of exploring threads, the relative frequency (Fr) of the head of thread i to pass in proximity to any point of the history of thread j at time $t \neq 0$ is

$$Fr(i,j,t) = \frac{\sum_{j=0}^{t-1} C(x_i, x_j)}{t \neq 0}, \tag{4-1}$$

$$C(x_i, x_j) = \begin{cases} 1; & d(x_i, x_j) \leq D \\ 0; & otherwise \end{cases} \tag{4-2}$$

where $d(x_i, x_j)$ is the Euclidian distance between points x_i and x_j, and D is a predetermined cutoff radius. The sum of proximity relative Frequencies (SoF) of thread i head to pass near to the history of any other thread j at time t

$$SoF_{it} = \sum_{j \neq i} Fr(i, j, t) \tag{4-3}$$

is a real number ∈ [0, *n*]. It is important to note that *SoF* can exceed 1.0 (because a thread may pass near to more than one other thread simultaneously). This is particularly common in two cases: 1) near local minima [in pockets], where several threads tend to converge, and 2) progressively towards the end of the search, as all the threads tend to cover the entire search space extensively. The Average *SoF* (*ASoF*) at any time *t* is

$$ASoF_t = \frac{\sum_i^n SoF_{it}}{n}. \tag{4-4}$$

It is expected that tracking *ASoF* over *t* ∈ [0, T] would show a progressing trend.

Next, consider the theoretical 2D search space shown in Figure 4-3 with three threads exploring it (A, B, and C) and a local minimum in the lower right corner. At *t*=1, none of the threads are near to the history of any other threads yet, so the $ASoF_1$ = 0. When *t*=2, A_2 was close to the history of C_1 so A_2 gets score 1 and the $ASoF_2$ would equal (1/2 + 0 + 0)/3 =0.167. When *t*=3, A_3 passed by the history of B_2, and B_3 passed by C_1, so $AsoF_3$ = (1/3 + 1/3 + 0)/3 = 0.222. When *t*=4, A_4 was already descending in the well, near to B_2 again (from the other side) and got a score of 1; B_4 was near to both A_1 and C_0, thus got a score of 2; while C_4 was close to B_0, getting another score of 1. Therefore, $ASoF_4$= (1/4 + 2/4 + 1/4)/3 = 0.333. Consequently, the progress of the *ASoF* would be (0, 0.167, 0.222, 0.333), which increases with time. The hypothesis behind this research is that the increase in *ASoF* is associated with increased speed and accuracy of decision making, as will be elaborated using an example later.

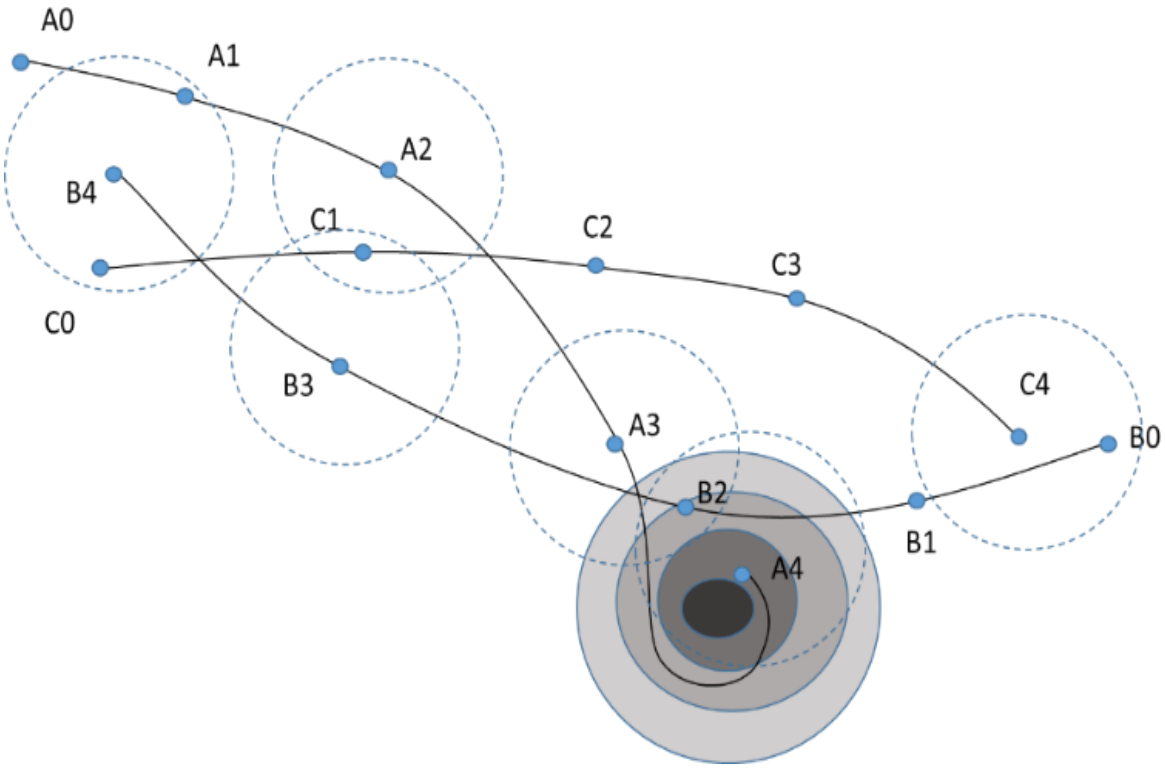


Figure 4-3 Illustration of progression of threads proximity in 2D
A, B, and C represent the searching threads while 0, 1, 2, 3, and 4 refer to the time point at which the thread is. The dark area at the lower right corner represents a local minimum.

4.2.2. High-quality history points

An essential question here is which points from the history of all threads should be kept. The decision was made to use the output of last iteration of local optimization (also known as end-points or stop points). Typically, these points had undergone up to 300 cycles of essential local optimization then up to other 300 cycles of local optimization (refer to Figure 4-18). Therefore, each of these points is the lowest along the direction of gradient of up to 600 other points. Being the lowest implies being the nearest to the local minima (if any) in this area.

4.2.1. Proving the hypothesis

To prove our theory, a blind docking search was run on every complex of the PDBbind Core dataset using 64 threads, and the average sum of relative frequencies of any one of the running threads to fall in close proximity of 5 Angstrom to the history "foot print" of any of the other 63 threads (ASoF) was counted. In Figure 4-4, one example complex from the data set is studied (namely PDB ID 10GS). Figure

4-4(I) shows the progression of that ASoF in the first 340 time frames (each defined as 1000 steps of local/essential-local searches running concurrently), while Figure 4-4(II) shows snapshots of the search history at selected points on the graph shown in Figure 4-4(I).

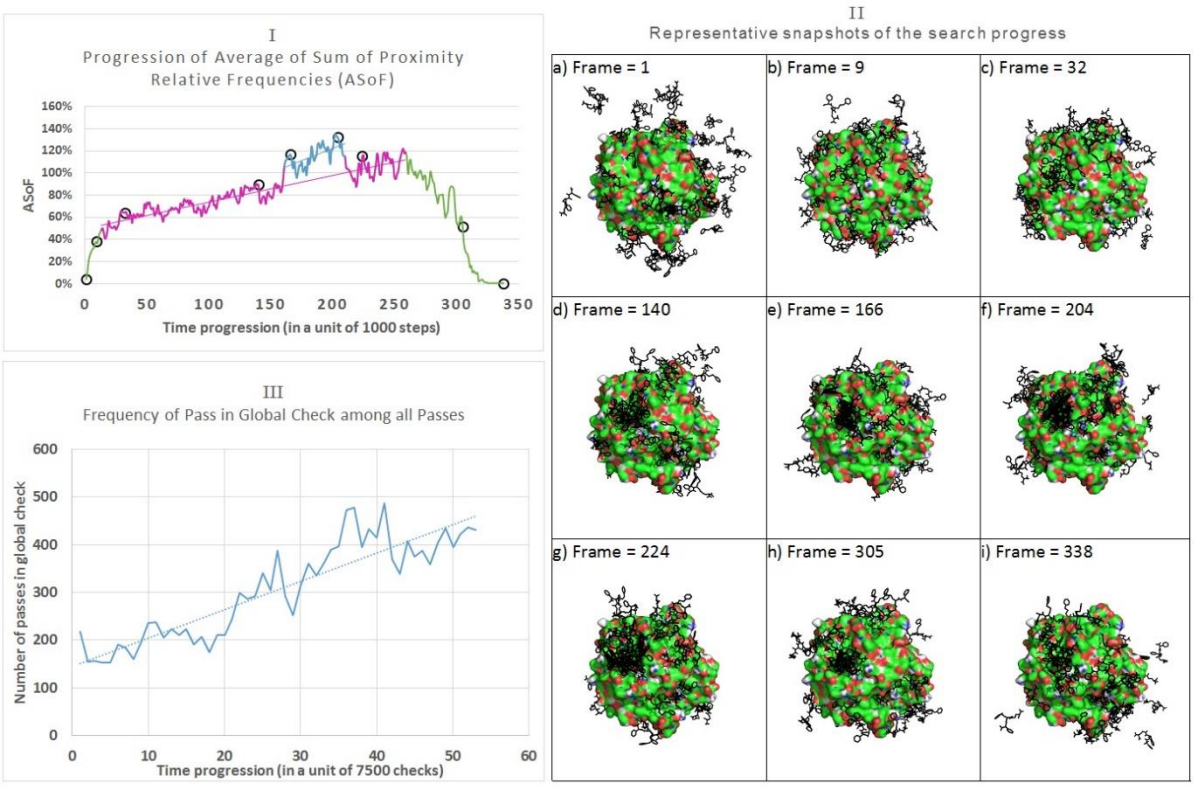


Figure 4-4 Study of search progress of 10GS
I) Progression of Average of Sum of proximity relative Frequencies (ASoF) of the first 340 time frames. The numbers are averaged over periods of 1000 steps each. The curve shows an initial sharp rising segment and a terminal falling segment (frames 1-12 and 260-340 respectively, both colored in green). Between these segments, a rising trend (frames 13-259, colored in magenta) appears with a slightly shifted small segment (frames 161-210, colored in blue).To investigate and explain the reason of this curve's features, some representative points are marked with black circles (frames 1, 9, 32, 140, 166, 204, 224, 305, and 338), and their snapshots are shown in (II). (II) Snapshots of the search progress of ligand-protein complex PDB ID 10GS, using 64 threads. The ligands representing the searching threads are in black. (III) Progression of Frequency of G test pass among all passes. Every point is the sum of G pass among 2000 passes (either G or I pass) from all threads.

Figure 4-4(I) shows in the initial loading phase (the corresponding left green segment) that the searching threads start uniformly distributed over the whole search space with a zero ASoF between threads, as seen on Figure 4-4(II)a. Then rapidly most of them settle on the surface of the receptor, and consequently their ASoF starts to rise. The second phase (frames 13-259) shows that the search threads explore the whole accessible search space. It shows a steady increase in the ASoF. That goes along with the steady increase in the aggregation of threads, as seen in Figure 4-4(II)c-g (roughly

proportional to the number of black dots in clustered structures). One unique feature of this figure was the transient curve-shift seen in frames 161-210 (the blue-colored segment in the middle of the magenta-colored one). By inspecting the search history, the reason of this shift was found to be that there was a transient thread (congestion) in one of the cavities. This congestion started, evolved, then resolved. This incident is manifest as thread concentration spikes from d (frame 140) to e (frame 166), then the continued increase in f (frame 204) then decrease again in g (frame 224) to a value lower than e but higher than d. The final phase (the right green segment, frames 260-340) is associated with decline in the ASoF. In this phase, the running threads finish one by one and the density of active threads tends to decline, as seen in Figure 4-4(II)h. It is noteworthy that the successful threads finish their searches and end earlier, while by the end of the search time, the remaining threads are those that tend to fail in finding suitable solutions and lost their way and tend to explore areas in the search space too far from the receptor surface, as in Figure 4-4(II)i.

With increased relative frequency of proximity, it is expected that it is associated with an increased rate of passing of the QuickVina acceptance check through the global check G. The progression of Success in that stage among every 7500 passed checks from all concurrent threads, is shown in Figure 4-4(III). The figure shows a rising trend. There are two positive expected effects on the increased frequency of G step passes: increased speed and accuracy.

Since the global stage (G) precedes the individual stage (I) and the number of all the performed steps in G stage represents a small portion of the total checks ($p_1 \leq P_1 << P$), it is expected that the more accepted points from the global stage, the less the total checks become, hence less time is required for decision making. Additionally, as more history is accumulated in the global buffer, the search keeps becoming faster towards the end than it was in the beginning. That effect is already found in Figure 4-5.

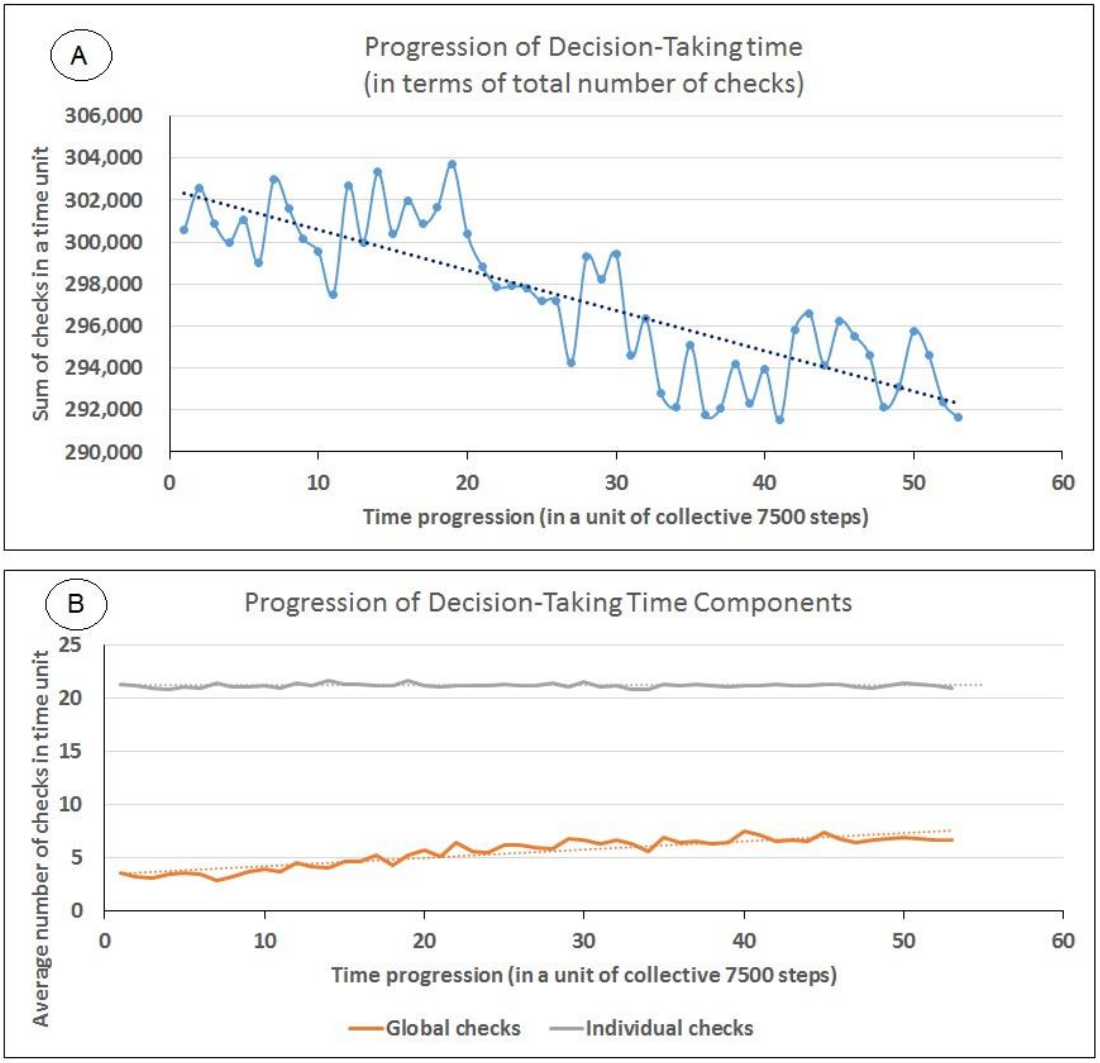


Figure 4-5 Analysis of decision-making process
A) Progression of decision time, in terms of number of checks taken to pass a potentially significant point is the sum of checks done in 7500 passed tests. B) Fractional analysis of decision taking time components. The frequency of success in the Global stage increases with time, while the frequency of success in the Individual stage is stationary.

In Figure 4-5(A), the progression of sum of checks needed to pass 7500 tests is shown. It is obvious that the number of needed checks per test decreases, which means a significant point is accepted faster to undergo local optimization. To ensure that the declining curve is merely due to the effect of the increased frequency of the G component only, the different fractions of components participating in the decision taking time were plotted in Figure 4-5(B). The curve shows that the (G) component is increasing while the (I) component is stationary.

To show the effect of increased search speed on the search time, Figure 4-6 shows the (normalized) search time trend achieved by QuickVina-W for different numbers of steps S (X1, X2, and X4), in relation to the number of heavy atoms. The figure shows that the (normalized) search time decreases with increasing number of steps S, and that decrease occurs in an increasing rate with increasing S.

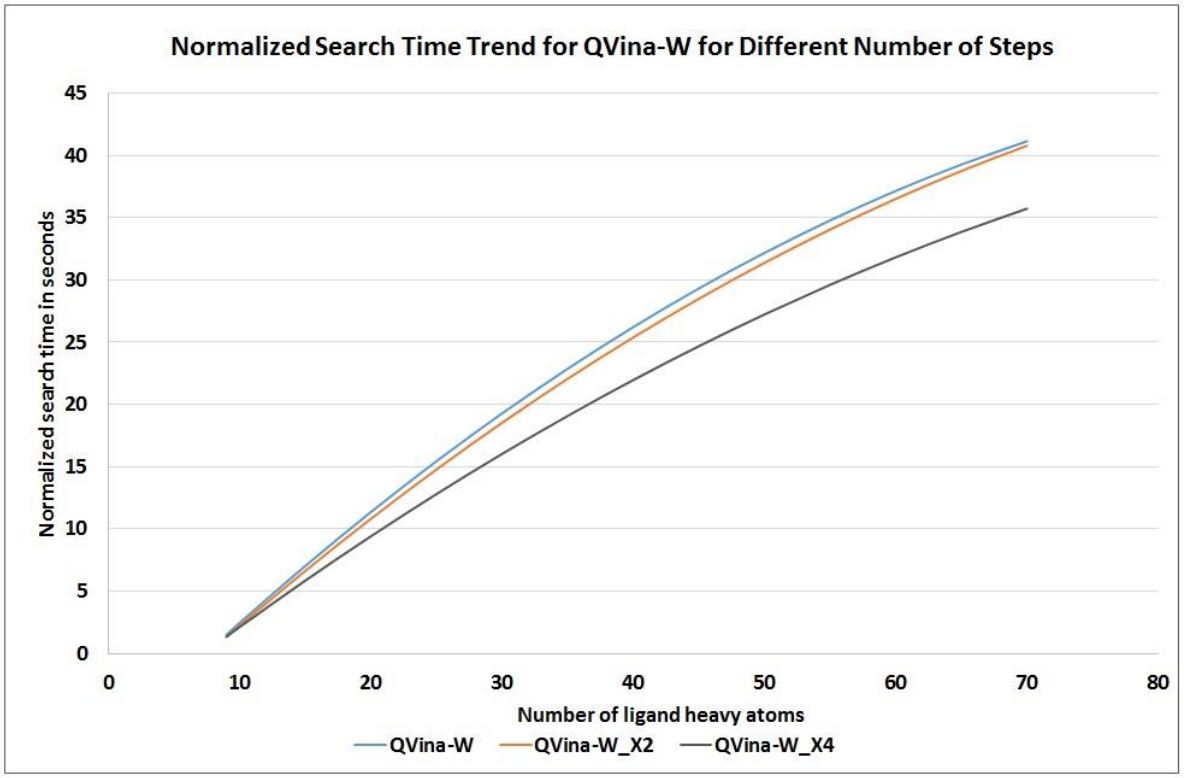


Figure 4-6 Normalized Search Time Trend for QVina-W for steps X1, X2, and X4
The curve shift from middle to lower curves is much bigger than the shift from the upper to the middle curves, indicating that the search speed keeps increasing as .the search progresses, in an increasing rate.

The second effect of increased ASoF is the decision accuracy in terms of sensitivity. Considering QuickVina acceptance check as ultimate positive condition, the sensitivity of the G phase successes can be defined as a rate of acceptance by G step divided by rate of total acceptance (G + I). Again, Figure 4-7 shows a rising trend denoting increasing sensitivity.

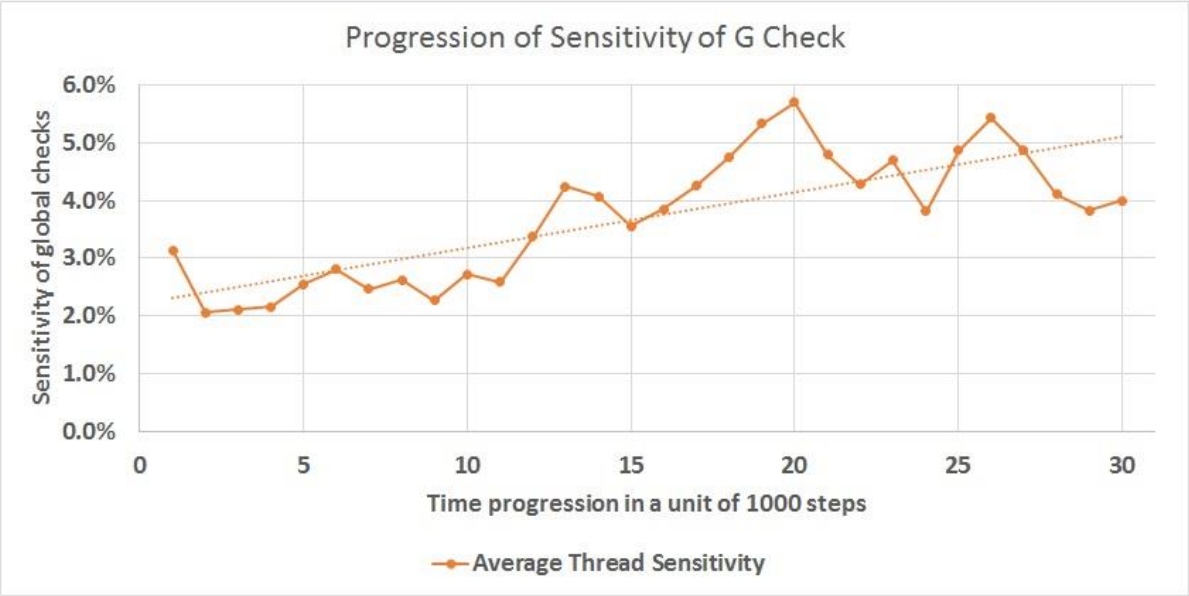


Figure 4-7 Progression of G check Sensitivity
The sensitivity of the global checks is shown in relation to time in terms of a unit of 1000 steps of the search process

Finally, to prove the relation between the ASoF and sensitivity, both the ASoF from Figure 4-4(I) and average sensitivity from Figure 4-7 are plotted together in Figure 4-8. The graph shows a definite relation between them with Correlation Coefficient r= 0.862, which proves our theory.

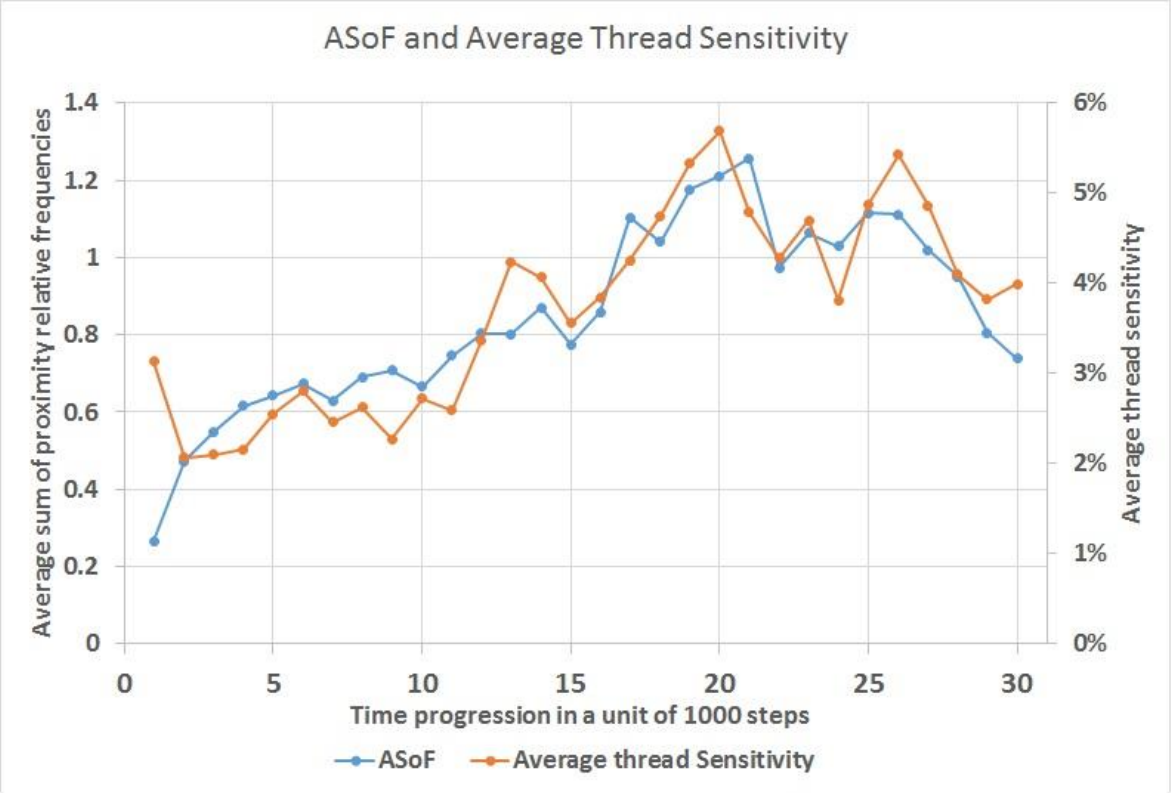


Figure 4-8 Relation between ASoF and Average thread Sensitivity
A plot of both ASOF and average sensitivity in relation to time in term of a unit of 1000 steps. It shows the definite relation between them. with Correlation Coefficient r= 0.862.

4.3. Methods

4.3.1. Data

The theory was tested on the core set of PDBbind 2015, which includes representative 195 protein-ligand complexes [147-149]. MGLTools [89, 150] were used to prepare the ligands through command line, by converting all ligands and receptor files into PDBQT, and enabled all rotatable bonds (backbone phi and psi, amide, and guanidinium). To avoid any possibility of bias, the ligands were then randomized using vina "--randomize_only" parameter, to generate random starting respective poses different from the experimental ligands. Moreover, for every complex, the same stochastic search seed was used for both the small and large search spaces and for all the tested configurations.

QuickVina-W was validated (in its final configuration) by virtually screening the 54520 structures from Maybridge library screening collection against the crystal structure of influenza A H1N1 Nucleoprotein (NP) chain A monomer, obtained from protein data bank (PDB ID: 2IQH), comparing the results to those obtained from Vina on the same structure.

4.3.1.1. Search space

There are two search space settings: one for searching a certain pocket (referred to as small search space hereinafter) and the other for searching the whole receptor surface without any preference to any pocket (referred to as large search space hereinafter).

The small search space is similar to that defined in Vina, QuickVina 1 and QuickVina 2 [83, 102, 136] as described in [section 2.3.2.1 Search space box]. For each of the 195 complexes of the test set, the search space is defined as the minimal rectangular parallelepiped, aligned with the coordinate system that includes the docked ligand, plus added 5 Å in the 3D (three dimensions: X, Y, and Z). Additional 5 Å were added randomly to either sides in each dimension, to decentralize the search space over the target pocket. If the search space in any dimension was less than 22.5 Å, it was uniformly increased to this value to ensure the search space allowed the ligand to rotate. For the influenza A NP, the T-loop binding site used in Awuni et al. [134] was used.

The large search space (for both the PDBbind core set and Influenza NP) was defined, by determining the largest dimension of the ligand, and adding its value to the protein at both directions in each of the three dimensions, following the recommendation that the ligand should be allowed to rotate in the search space [89]. No other measures to decentralize the search space were taken; because it is already centered on the protein center of geometry, while the target pocket is somewhere on the protein surface.

4.3.2. Procedure overview

After preparing the benchmarking dataset, QuickVina performance on the dataset was profiled using different configurations of internal tool parameters on small search space, in order to select the best

candidate configuration. Afterwards, that configuration was projected to large search space, where the inter-process communication was applied, and its performance was profiled. Then the maximum number of steps was kept increasing, until it reached four folds of the number of steps the original Vina undergoes. More procedure steps are explained in detail in the next sections.

4.3.3. Development of QuickVina-W

4.3.3.1. Profiling different configurations of QuickVina 2 on small search space

Vina has a parameter called "exhaustiveness" that controls how comprehensive its search is. The more the exhaustiveness, the less the probability a good result is missed. Throughout the profiling, the value of exhaustiveness was kept equal to the number of CPUs used. The C++ code of QuickVina 2 was changed to test different configurations. These configurations include maximum number of checks (P) mentioned earlier and buffer size (Q). Different combinations of configurations {(P, Q)}, as well as exhaustiveness level (E) were tested. Configurations are {(P, Q) | P ∈ {0.5N, N, 2N, 4N, 6N, 8N} AND Q ∈ {N, 2N, 5N, 10N, 20N, 40N} AND P ≤ Q}. The value N is the number of DOFs (equals six + number of rotatable bonds as mentioned in section 2.2.1 [Search space Sampling Algorithm in Vina]). Combinations are in the form {((P, Q), E) | E ∈ {8, 16, 32}}. For example, suppose that QuickVina 2 is run with configuration (P=4N, Q=5N) and exhaustiveness (E=16). When the ligand has 4 bonds, there will be 4+6 = 10 degrees of freedom (N), so the maximum number of checks would be the nearest 40 (4N) points among the latest 50 (5N), for each of the 16 threads exploring the search space. Running the tool with different configurations to explore different dimensions was automated using a BASH script. Then collecting the data and generating several graphs thereof was done using "Origin C" programming language inside Origin [151].

Figure 4-9 shows the success rate of different settings using exhaustiveness 16. (Since, Q >= P, it was necessary to fill the missing entries in the graph, by duplicating the last entries in each line where P > Q, in order to get clearer figures).

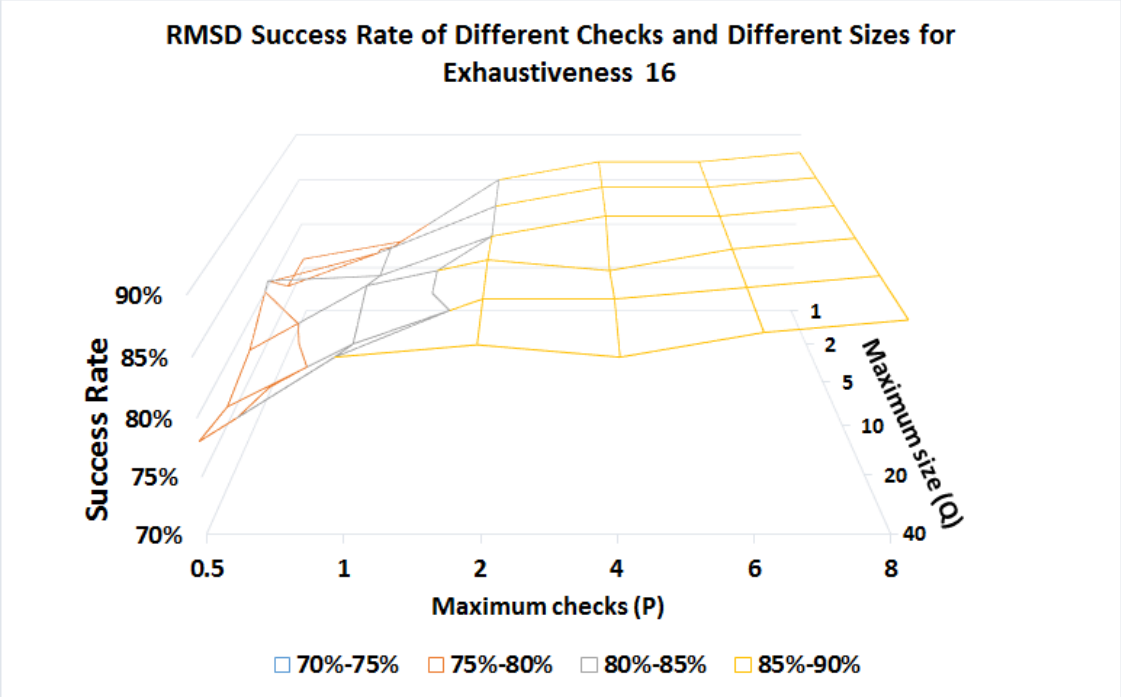


Figure 4-9 RMSD success rate of different checks and different sizes for exhaustiveness 16

The several experiments done showed that the maximum checks of 2N is the junction between high-slope and plateau sections of the surface, and 4N is a safe margin, after which no significant improvement occurs despite of changing the maximum size and exhaustiveness. Consequently, 4N was used as the maximum check, and more investigations were made to determine the size to be combined with it, in order to give the best results in terms of acceleration, binding energy and RMSD. The settings of maximum checks (P) of 4N and a maximum size (Q) of 5N was the configuration with the best results obtained as will be discussed later.

In order to find the best Q, Figure 4-10 shows the search time acceleration of 4N checks along different sizes as calculated in equation(4-5). Buffer size of 5N shows the highest acceleration results

among all sizes and shows that with increasing the buffer size, the speed decreases, because it implies more time for sorting and selecting more history points.

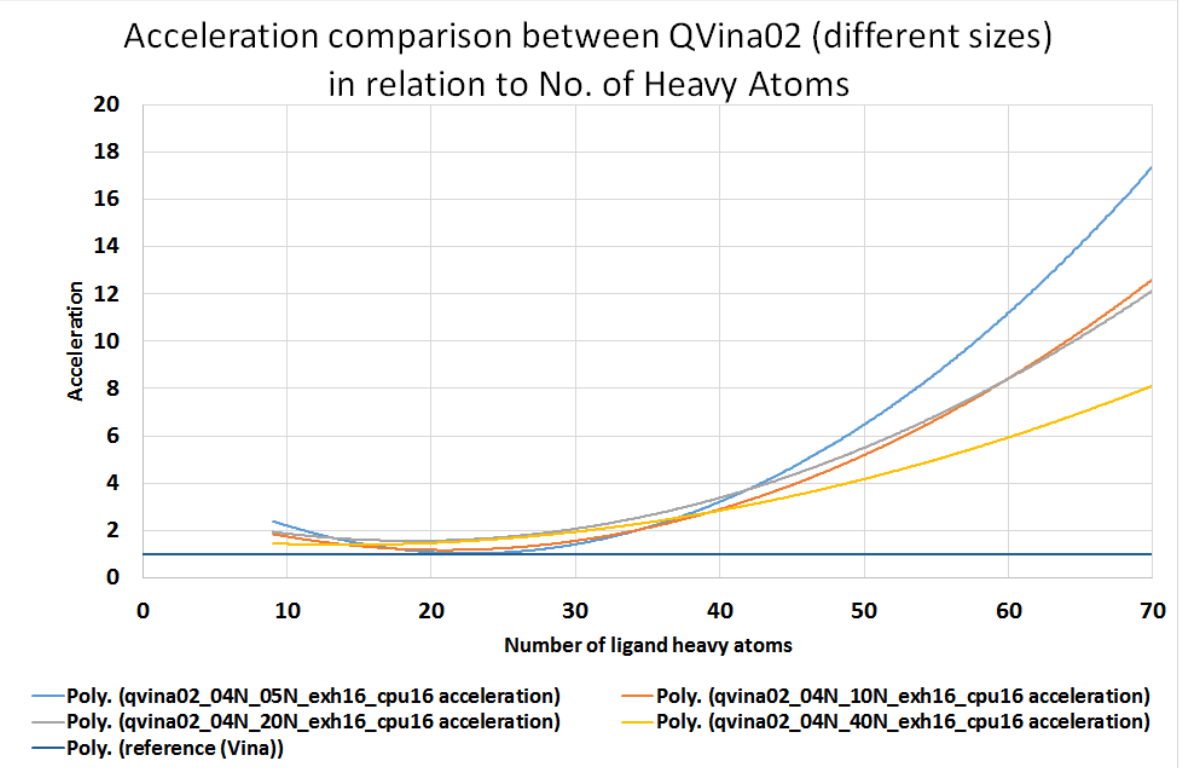


Figure 4-10 Acceleration of QuickVina 2 for checks of 4N with different sizes
Acceleration of QuickVina 2 for checks of 4N along different sizes, in relation to the number of heavy atoms using exhaustiveness 16 (Q ∈{5, 10, 20, 40})

Although the setting of 4N_5N did not show the best results in terms of binding energy as seen in Figure 4-11, the average of "worse" binding energy differences was the lowest among all of them except for the 4N_40N, (which is by far the slowest). In addition, the RMSD of size of 5N was the best among other sizes with checks of 4N as shown in Figure 4-12. Consequently, the good acceleration, the relatively low binding energy difference average, and the RMSD of 4N_5N was good enough to build upon for the next steps. The notations of the legend for the above-mentioned figures are as follows: Vina is the original Vina 1.1.2 results; Qvina02_P_Q refers to the previous QuickVina 2 before adding the global buffer with maximum checks (P), and maximum sizes (Q), the exhaustiveness (16) and number of used CPU (16) are coded as well.

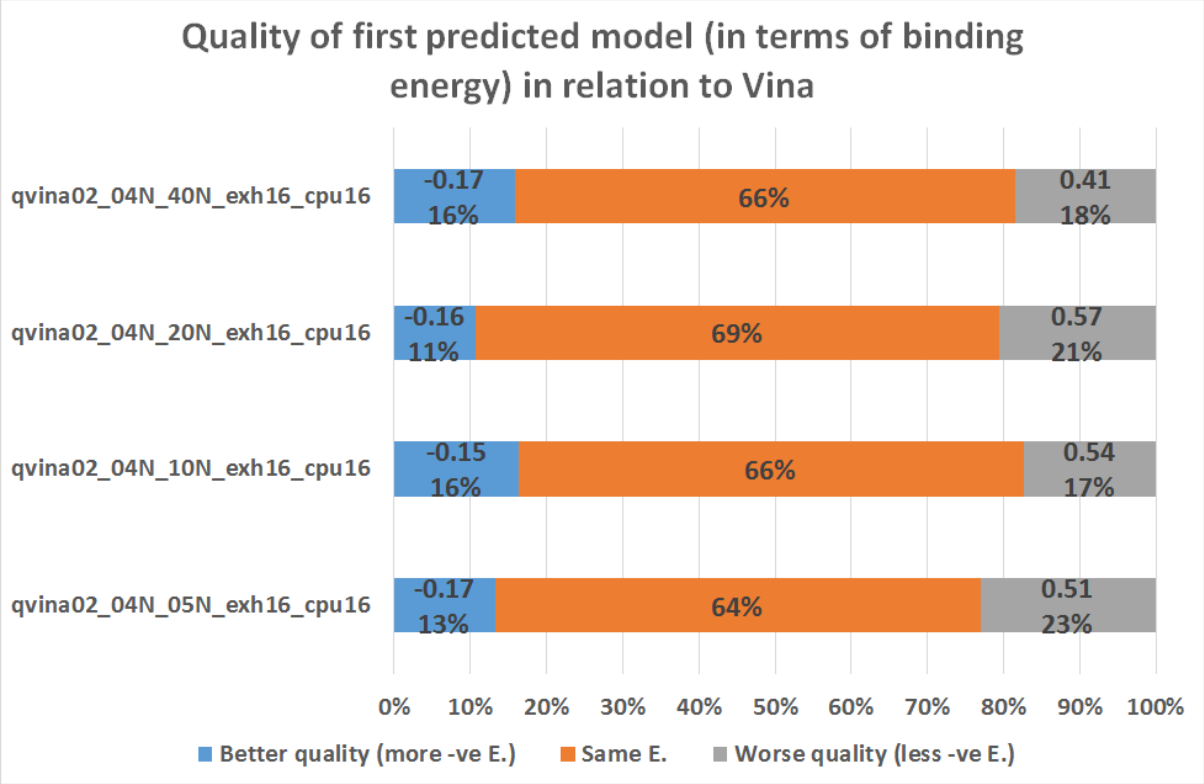


Figure 4-11 Binding Energy of checks of 4N (small search space)
Binding Energy of checks of 4N along different sizes of QuickVina 2 for small search space using exhaustiveness 16.

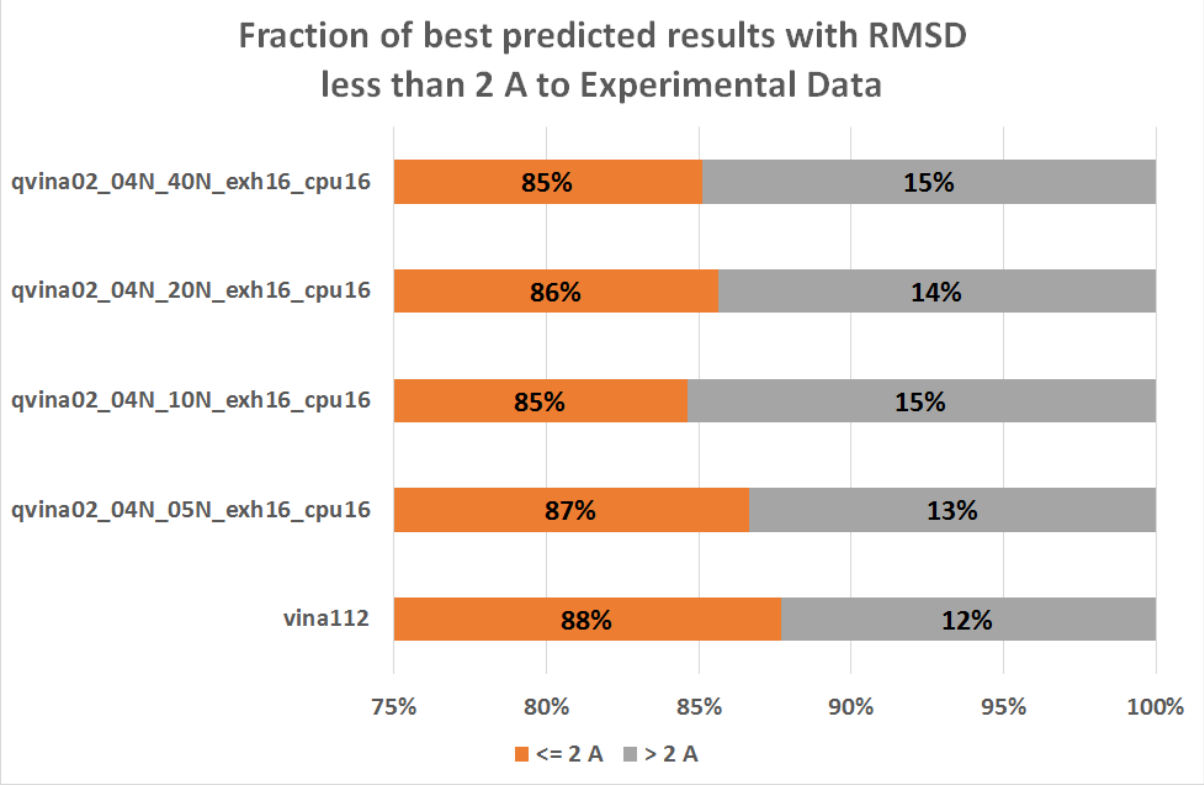


Figure 4-12 RMSD of any (best) predicted results to experimental data in small search space
RMSD of any predicted result to experimental data for QuickVina 2 of checks 4N and different sizes for exhaustiveness 16. Success is determined as distance <=2 Angstroms, and failure as distance>= 2 Angstroms.

After small search space profiling, the configuration that showed the best results was selected to project it to the large search space. The selected configuration was P=4N, Q=5N. Exhaustiveness value of 64 was used in the large search space.

4.3.3.2. Adding the Hybrid Buffer and profiling on large search space

The maximum number of checks and the buffer size were kept to 4N and 5N respectively. QuickVina 2 code was modified to add a global buffer in addition to the individual buffer for every thread. The global and individual buffers were referred to as the **hybrid** buffer, which means for a thread to check whether a potential point is significant or not, it would check the nearest points in the **global** buffer first. If not detected as significant, or if there are not enough near points in the global history to decide, then the thread searches the history of its own **individual** buffer next.

The two parameters of the new check were then profiled: the proximity cutoff radius (D) and the maximum number of checks allowed from the global buffer ($P_1 << P$). The proximity cutoff is calculated as the Euclidian distance in the three dimensions.

The number of checks *actually performed* from the global buffer is ($p_1 \leq P_1$). After p_1 checks are done, the rest of P checks (p_2) are taken from the threads own individual buffer. Please note that $p_2 = P - p_1$ (not $p_2 = P - P_1$). In both steps, the history points to be checked are **ordered** from nearer to farther according to their Euclidian distance to the potential point in all N dimensions. To extend the previous example with selected configuration (P=4N, Q=5N) and a ligand with four rotatable bonds, the total maximum checks to be done (P) is 4N (i.e. 40 checks). Now, if the maximum allowed checks from the global buffer P_1 is N (i.e. 10) and none of them passed the test, then the remaining 30 checks are completed from the individual thread buffer. If there are only 8 points in the history of the global buffer within the cutoff D, then the individual checks p_2 will be 32.

All the coming figures hereafter are studied on exhaustiveness of 64 and 64 CPUs as well. Figure 4-13 shows the acceleration of different configurations of QuickVina 2 after adding the octree. The notations of the legend are as follows: QVina02-octree_RP$_1$_PQ represents the updated QuickVina

2 after adding the global buffer, P maximum checks, and Q maximum sizes plus R and P_1 referring to the cutoff radius in Angstroms, and the portion of maximum checks allowed from the global buffer respectively.

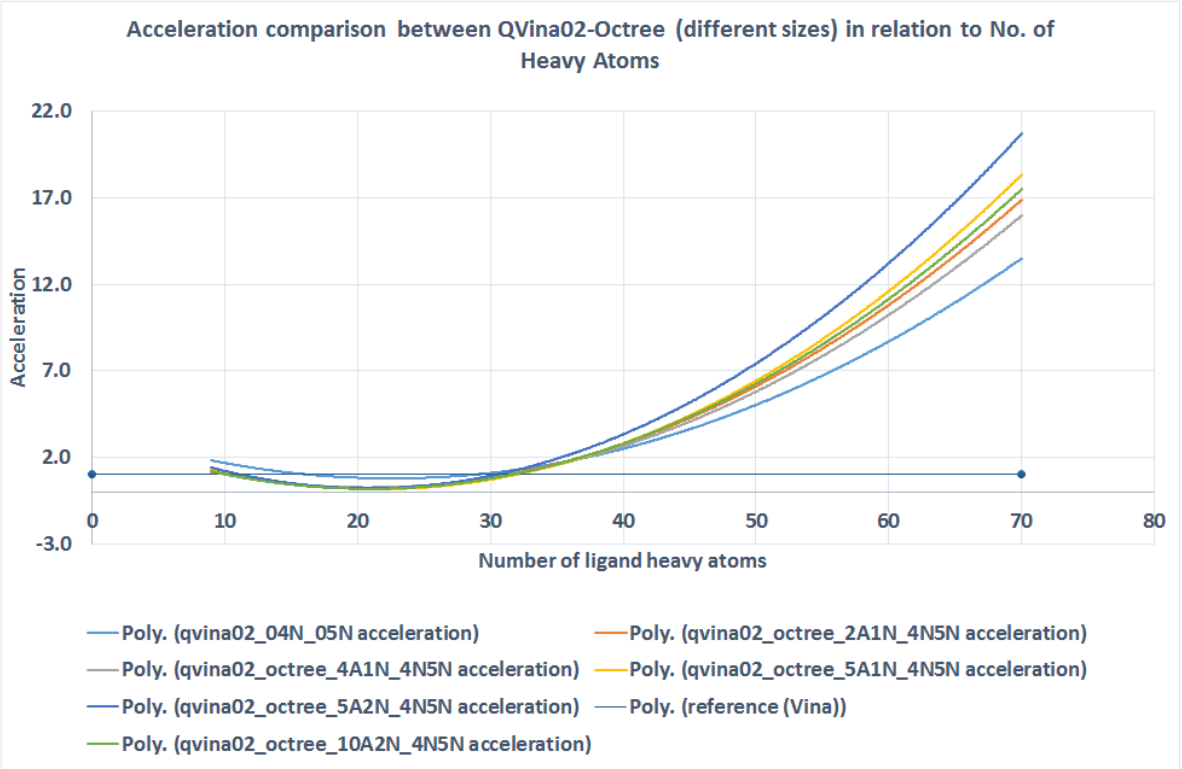


Figure 4-13 Acceleration comparison between different configurations of QVina-W in relation to No. of Heavy Atoms Acceleration between different configurations of QVina-W in relation to the number of heavy atoms. Qvina02_4N5N is old QuickVina 2 before adding global buffer. The notation QVina02-octree_DP1_PQ refers to: qvina02_octree: QuickVina after adding the global buffer, D: the cutoff radius in Angstroms, P1: the number of allowed checks from the global buffer, P: the total checks from both buffers, and Q: the size of the individual buffer.

The accelerations of all QuickVina-W configurations are better than QuickVina 2 accelerations as shown in Figure 4-13. As a matter of fact, the quality of the prediction is more important than the speed. Therefore, although the radius of 5Å and 2N as maximum checks of the global buffer seems the fastest, the configuration of a cutoff radius of 5Å and 1N of the global buffer was then chosen because it showed better results regarding both the binding energy and RMSD.

The quality of prediction is shown in the next two figures. In terms of binding energy, Figure 4-14 illustrates the percentage of the first models that show better quality (more negative energy), the same binding energy, and worse binding energy, in comparison to the original Vina.

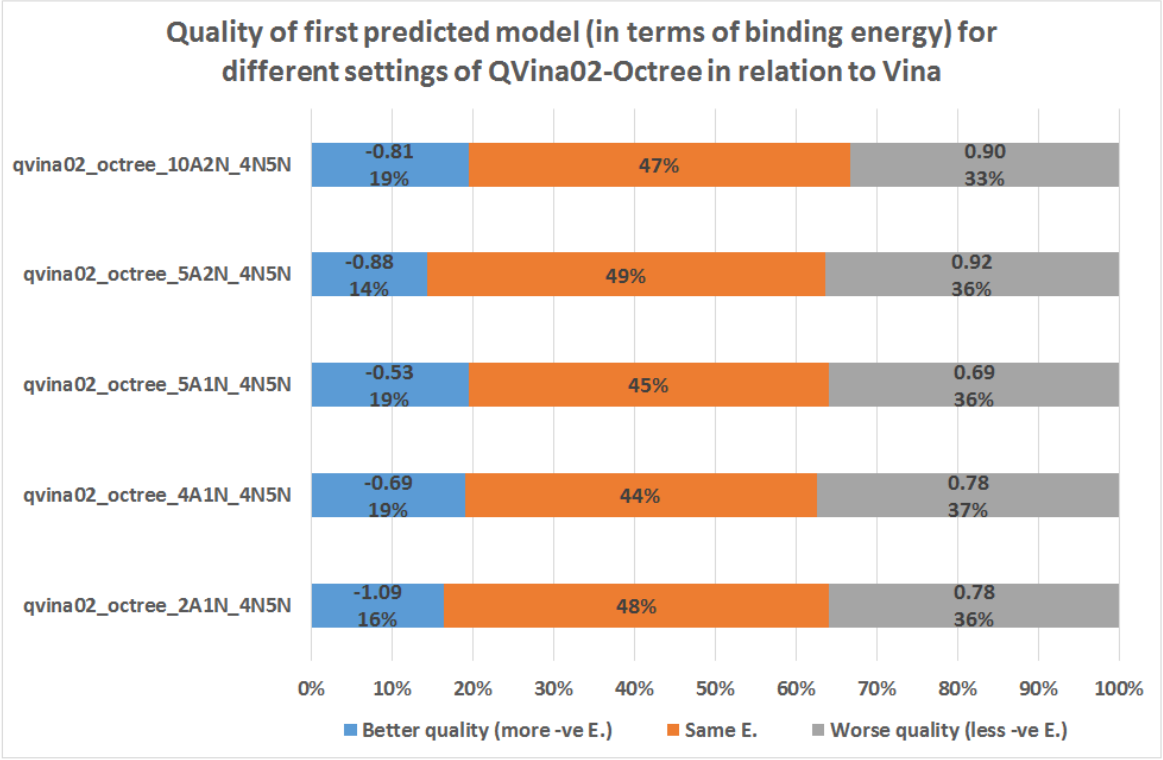


Figure 4-14 Quality of first predicted model (in terms of binding energy) for different configurations of QVina-W in relation to Vina
Binding energy of the first predicted model for different settings of qvina with octree (QVina-W). Averages of binding energy differences in both better and worse quality are given above the percentage

In spite of the good results (in terms of binding energy) obtained from the configuration of 10Å, and 2N checks (33% only worse binding energy), it did not show similar good ranking regarding the RMSD calculations as illustrated in Figure 4-15. The configurations 2A1N, 5A2N, and 5A1N share next lowest value (36% worse binding energy). Similarly, configuration of 5Å2N was the best regarding the acceleration, yet the configuration of 5Å1N is better from the binding energy view as it showed the highest portion of better quality (19%), as opposed to 14% and 16% for 5Å2N and 2A1N respectively. On the other side, Figure 4-15 shows that it (5Å1N) scored the best RMSD among all other settings, with success of 63% similar to the original Vina. Consequently, the settings were

finalized to be 5Å for the radius, 1N allowed checks from the global buffer, and 4N checks, and 5N size respectively.

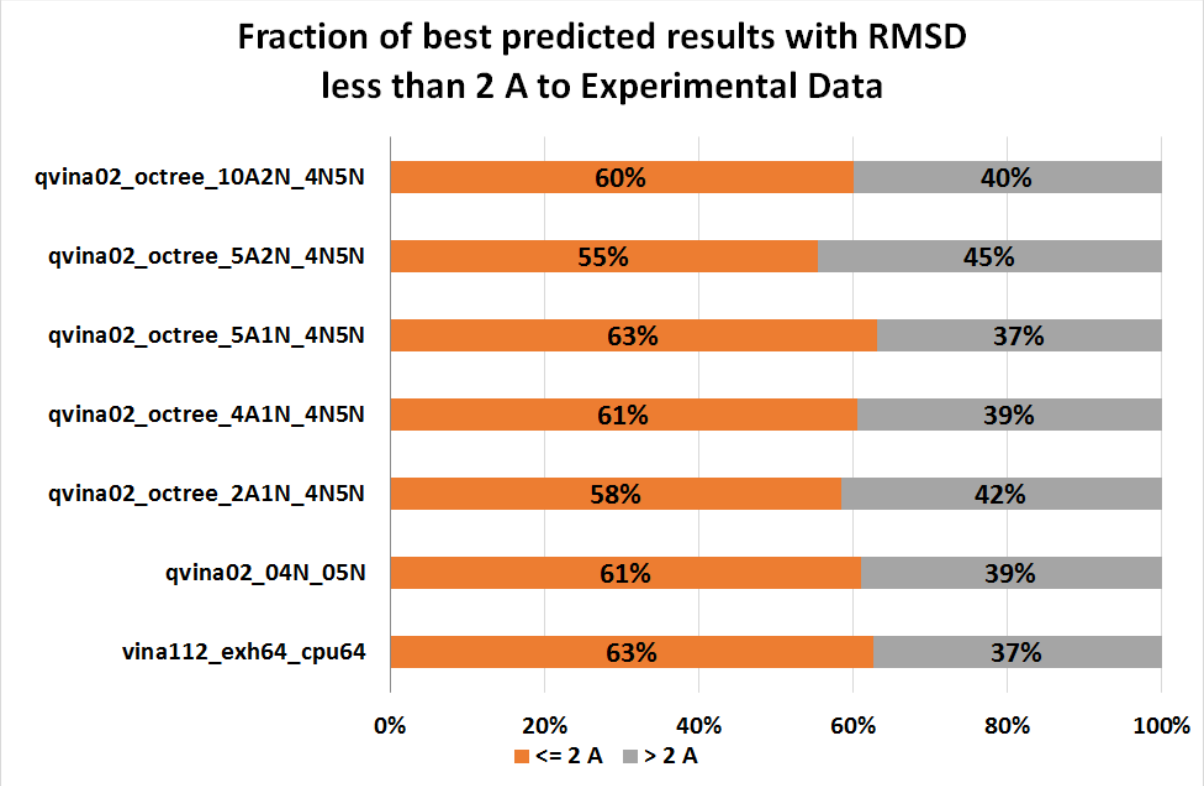


Figure 4-15 Fraction of any (best) predicted result with RMSD not more than 2 A to Experimental Data for different configurations of QVina-W
Fraction of any predicted result with RMSD < =2A in relation to experimental data for different settings of QVina-W (radius, and maximum checks allowed from global buffer).

Therefore, the best configuration was that with P_1 = 1N, R=5Å, P=4N, and Q=5N as shown in the above figures.

4.3.3.3. Increasing the maximum number of steps

Applying the hybrid buffer boosted a leap of speed in QuickVina 2 without loss of search accuracy. That performance leap was further built on to improve the accuracy by increasing the maximum number of steps (S) of optimization iterations Vina undergoes. S is determined as a function of the ligand number of movable atoms and rotatable bonds. At a first sight, it seems that increasing S would increase the total duration of the search and would slow the searching speed. However, as shown earlier, the more steps taken so far ($s \leq S$), the higher the probability to pass nearby high-quality history

points, and consequently, the faster (and the more accurate) the decision-making will be. That is to say, provided $T_{0\rightarrow1000}$ is the time taken to do steps $S_{0\rightarrow1000}$, if S is doubled, then $T_{0\rightarrow2000} < 2 * T_{0\rightarrow1000}$ as shown later in the results.

The configuration with the best results so far (P_1 = 1N, R=5Å, P=4N, and Q=5N) was elected and the number of steps S was kept increasing, expecting the accuracy to increase, and keeping in mind to preserve the speed faster than – or at least comparable to- that of Vina, until it reached up to 4 folds the number of steps.

Figure 4-16 shows the acceleration of previous published QuickVina 2 and QuickVina 2 with octree against the number of ligand heavy atoms. As expected, the acceleration of quadrupled QuickVina 2 with octree is less than that of the doubled and even the previous QuickVina 2, since it undergoes much more steps.

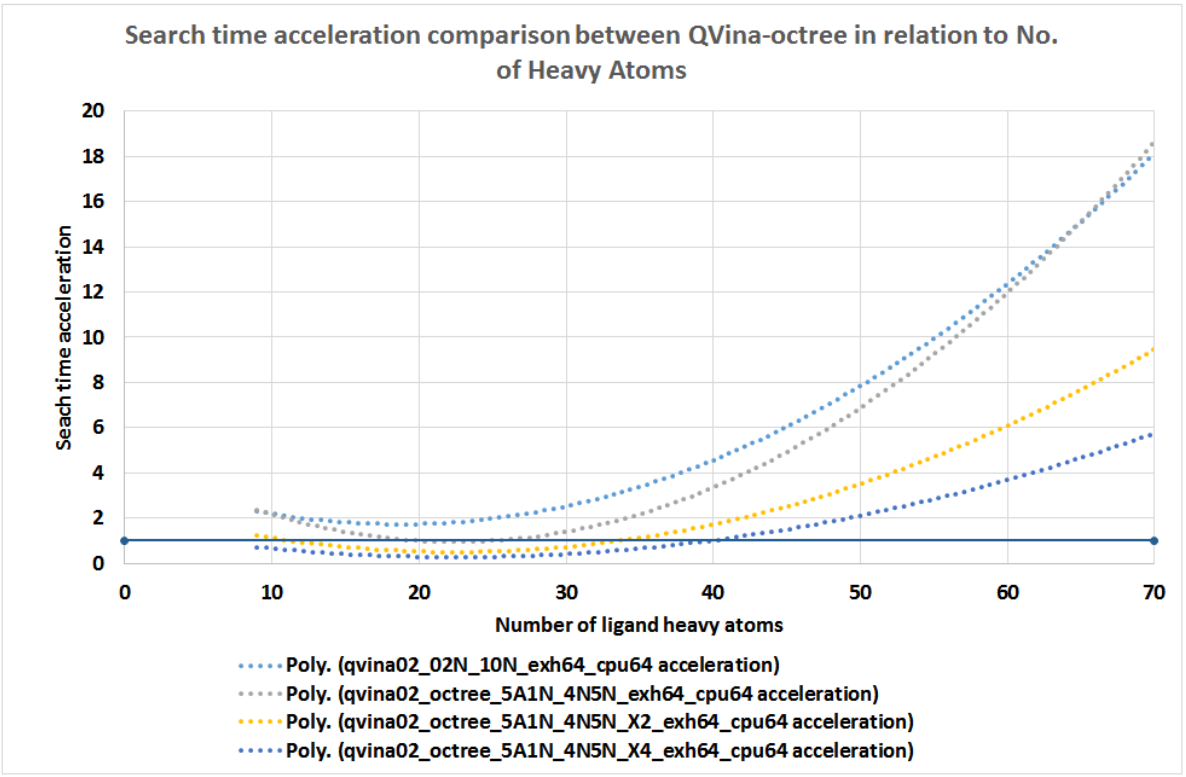


Figure 4-16 Search time acceleration comparison between different maximum steps of QVina-W in relation to No. of Heavy Atoms

The calculation in equation (4-7) yields the rising trend in Figure 4-17 that shows highest acceleration obtained by the quadrupled configuration. The search time component of the overall acceleration is discussed in the "Search progressions" section.

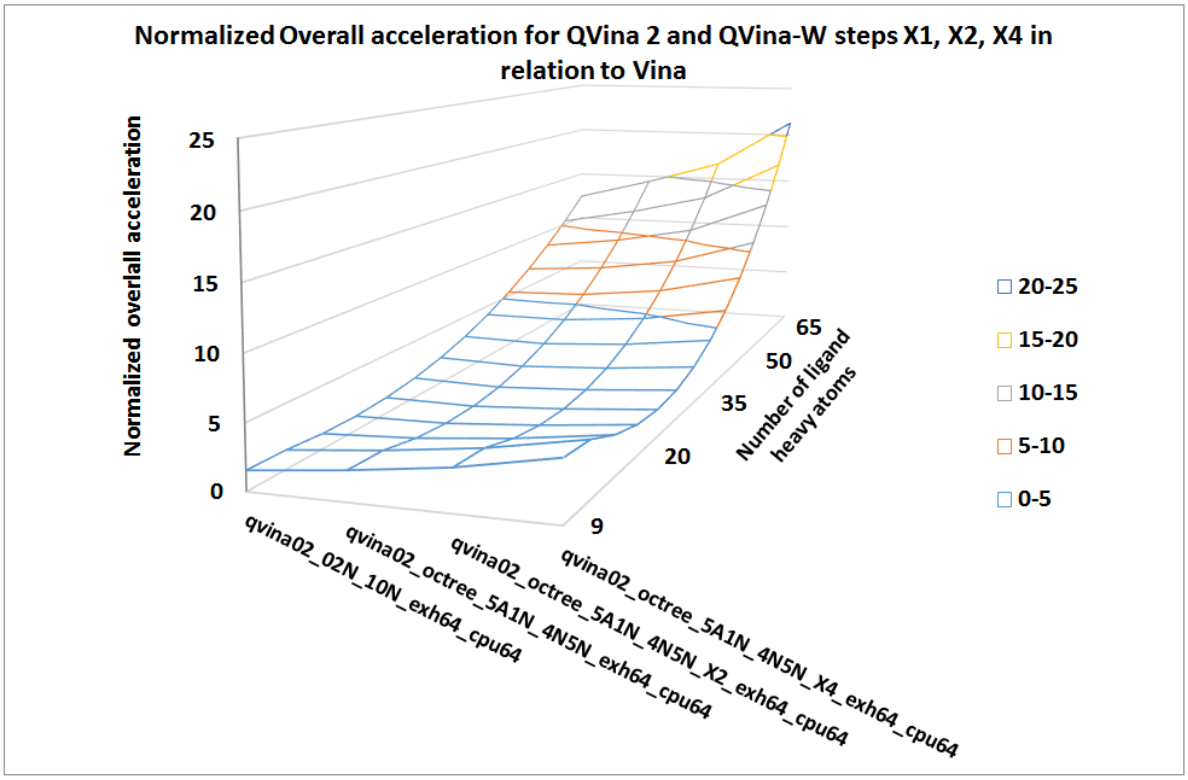


Figure 4-17 Normalized Overall acceleration for QVina 2 and QVina-W steps X1, X2, X4 in relation to Vina

4.3.3.4. Parallelizing the preparation stage

To ensure that the new tool is efficient for whole protein surface sampling, the tool overhead time was decreased by adding multithreading to its preprocessing stage, when it prepares the ligand, receptor, and scoring grid.

4.3.3.5. Implementation Details

AutoDock Vina was implemented in C++, and consequently all QuickVina tools were. Multithreading in AutoDock Vina is implemented using the famous cross platform library BOOST threads, in which thread creation follows the RAII design pattern where they are activated upon creation and owned by the parent process until they finish execution, when they are terminated. Despite the fact that threads

are grouped together in front of the user's eye, they are separate processes from the kernel perspective: They are scheduled independently, and they have nothing in common other than some overlapping memory pages. Linux implements threads as separate processes with different process ID (PID) but same Thread Group ID (TGID), which is reported to the user to give him the impression of the same process, while actually they are not.

All angles are measured in radians and normalized to the range [-2π, 2π]. Translation distance is measured in Angstrom.

The newly added global history buffer class responsible for the inter-process communication is a single object that implements the "singleton" design pattern. The buffer size might increase without limits. However, as long as it would hold a relatively small number of points, the size is not an issue. The global buffer is implemented as an octree (octal tree) of history points. The octree root is a cell that spans over the whole search space; and the history points are distributed in the octree according to their spatial distribution in the three dimensions. The choice of the Octree data structure to store the history points from all threads is related to the fact that blind docking is a [spatially-non-focused search]. Therefore, injecting spatial orientation to enforce spatio-temporal integration necessitates choosing a data structure that best performs in relation to the 3D position, which is the octree.

Figure 4-18 shows an overview of the whole tool after putting all the parts together. While the searching threads explore the search space, all history points are continuously added to the circular buffer, while end points are stored in the Octree according to their three-dimensional position. The local minima are added to the results vector.

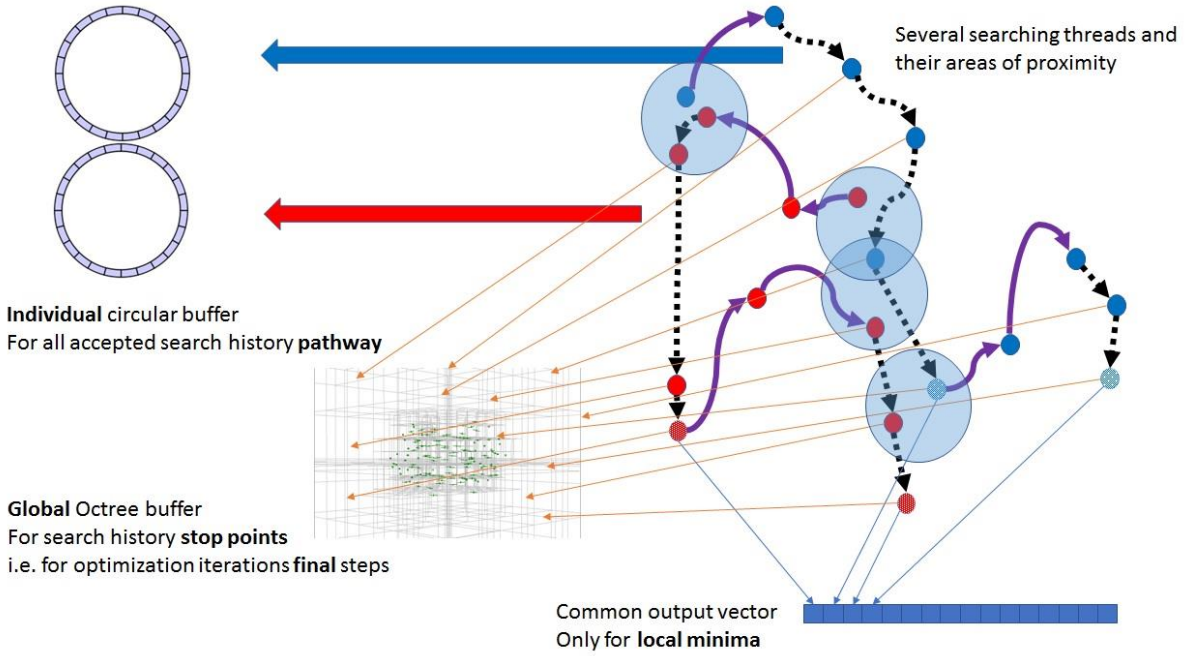


Figure 4-18 Overview of the whole tool with both types of buffers
Searching threads are shown in violet solid arrows (for Monte Carlo optimization) and black dotted arrows (for BFGS optimization). End (stop) points are shown as small red/blue spheres. Areas where threads come close the each other are shown as light blue spheres. All history points are continuously added to the circular buffer, while end points are stored in the Octree according to their three-dimensional position. Finally, the local minima are added to the results vector.

With a fixed proximity cutoff, the tree traversal and processing time increases with increasing the limit of maximum number of contents a node possesses, because that increased limit implies more unnecessary tests. While on the other hand, decreasing the maximum cell (node) content limit in a recursive binary search causes longer processing time, because it implies deeper recursive search overhead. That means one must balance between the depth and the breadth of the search.

This tradeoff was managed by having two limits, one on the minimum cell size width ($W_{MIN,}$ given an arbitrary value of 0.1 Angstrom) and another one on the maximum number of points a cell can contain ($S_{MAX,}$ given the value 8); and giving W_{MIN} a higher priority over S_{MAX}. This way, every leaf node can accept up to S_{MAX}. Every time a new point is added to a full (containing S_{MAX} points) leaf node, this node is converted into an internal node and is split into eight leaf children, unless the node is too small to be divided (i.e. unless each cell dimension W_i in the 3D is less than W_{MIN}). In such a case, the node will not be divided. Instead, the new point will just be added, and the node will contain more

than the default S_{MAX} capacity. This decision was made because that condition usually occurs around the local minima, where nodes tend to accumulate very close to each other (note that binding pockets are global attractors). In this case, 1) Searching such area will be slow, because it will go so deep (down to 14 levels according to our primary experiments), and the cell width might fall beyond the capacity of C++ float type precision. 2) Most –if not all– of the adjacent nodes should be considered for checking and consequently there is no need for recursive calls overhead.

When a new potential point is proposed, the currently held points in the tree are **filtered** according to the Euclidian distance in the first three dimensions **(**i.e. **in the spatial distance**) to the new point. Then, those which are within the cutoff are **ordered** according to the Euclidian distance **(in all dimensions)** to the new point from nearer to farther. Points that pass the local optimization step are added to the octree (the history buffer). Lastly, navigating the tree and reading node contents are not synchronized, while adding contents to a node and splitting it are synchronized using C++09 Shared Mutex.

Traversing and reading the tree (non-blocking) showed no bottle necks. Adding to an existing node is still not blocking. Adding to a full node, leaves other threads which have already enter the node, but it blocks (synchronizes access of) other new entering threads until the target node is split into 8. According to our experiments, the performance is not affected with upscaling number of threads, as long as they do not exceed number of available cores.

4.3.4. Collecting Results to Analyze

For the PDBbind dataset, the output data was collected as PDBQT files, and the gained results were compared to the experimental data. OriginLab was used to facilitate studying the several dimensions of the data; (i.e. checks, sizes, and exhaustiveness), in order to compare the results from the combination of configurations.

Several performance measurements of the docking process were studied, namely Acceleration, Binding Energy, and Root Mean Square Distance (RMSD).

- The acceleration was calculated by dividing the time consumed by Vina by the time of QuickVina 2 or QuickVina-W of the studied configuration. The components of time a tool takes are:

    1) The search time (T_S), which is the time taken purely to search for probable solutions. Search time acceleration (a_{Ts}) is calculated as

    $$a_{Ts} = \frac{Ts_{Vina}}{Ts_{QVina}} , \quad (4\text{-}5)$$

    where Ts_{Vina} and Ts_{QVina} refer to the search time of Vina and QuickVina respectively.

    2) The overhead time (T_H), which is the time necessary to load the input files, prepare for the run, and write output files. All the versions before parallelizing the preparation step share almost identical set of values for overhead time, and all versions after parallelization have another almost identical set.

    3) The overall time (T_O): all the clock time taken by the tool process run from start to finish, which equals the sum of the previous two times. The overall-time acceleration (a_{To}) is calculated as

    $$a_{To} = \frac{To_{Vina}}{To_{QVina}} , \quad (4\text{-}6)$$

    where To_{Vina} and To_{QVina} refer to the overall time of Vina and QuickVina respectively.

- The binding energy was studied by calculating the energy of corresponding models of QuickVina 2 and QuickVina-W configuration(s) against that of Vina for the same complex.
- For the RMSD measure, a prediction was considered to be successful if the RMSD of the predicted pose (as calculated in [83, 102, 136] and explained in [Appendix A: RMSD Calculation Method as described by AutoDock Vina team]) with respect to the experimental structure is less than 2Å. The percentage of complexes with successful RMSD was calculated over 195 total complexes of the PDBbind database.

In addition, all the history points from all the threads were outputted into separate files to allow retrospective analysis. The number of passes/fails in QuickVina 2 acceptance check per both the global buffer and the individual buffer was counted, along with the number of checks per every passed test (no need to count the failed tests, because it is already known that it is the maximum). The progression of the ratio between success in global check and success in individual check was monitored.

4.4. Results and Discussion

4.4.1. Increasing maximum steps

The effect of increasing the maximum steps in the best configuration (5Å1N_4N5N) was tested after changing the maximum steps to be doubled (x2) and quadrupled (x4). It is important to emphasize here that increasing the number of steps increases the *search accuracy* (defined as the ability of the search engine to sample lower minima of the energy landscape, missing less minima otherwise identified by more exhaustive and expansive searches), but not the *scoring function accuracy* (defined as how closely the scoring function describes the reality, by having a global minimum near the known correct answer and creating an energetic gap between this solution and other potential solutions).

In term of the binding energy illustrated in Figure 4-19, there is an obvious enhancement trend in QuickVina 2, QuickVina-W configurations X1, X2, and X4. The later shows 78% of the predictions with binding energies better than or equal to the original Vina 1.1.2, out of which 26% are better with average 0.81 KCal/mol more negative.

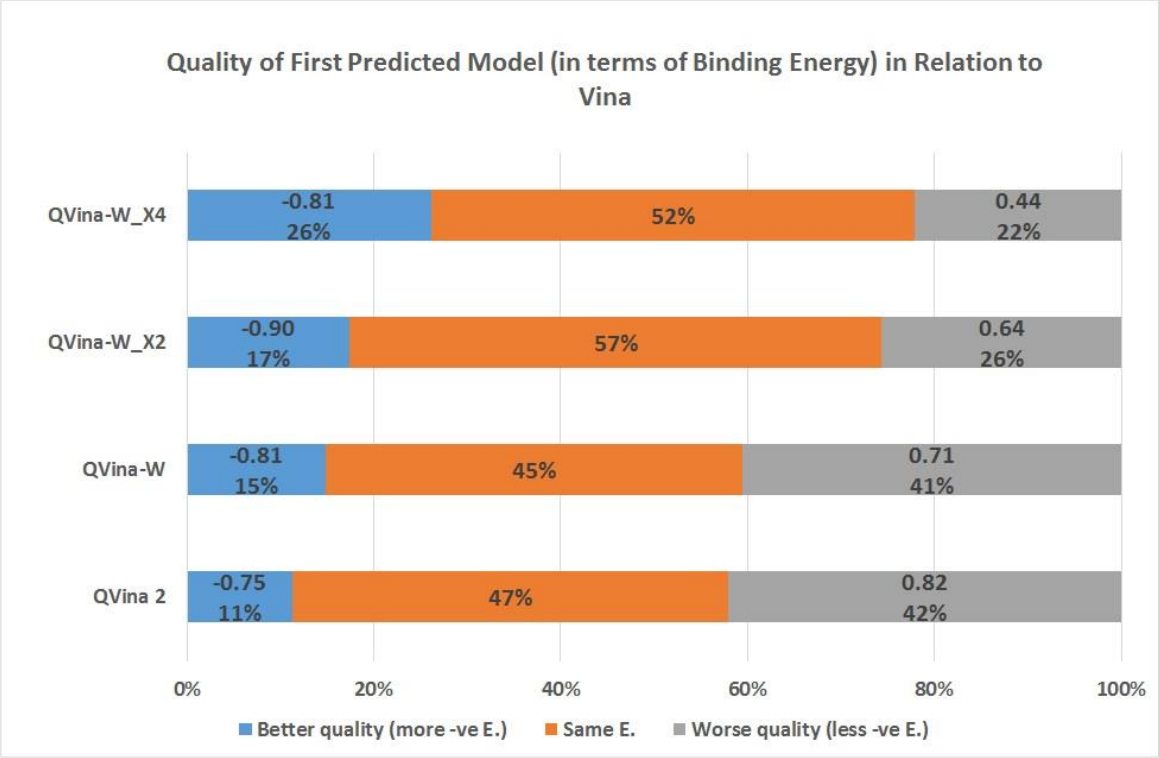


Figure 4-19 Quality of first predicted model (in terms of binding energy) of different steps of QVina-W and the previously published version of QVina, in relation to Vina
Binding energy of the first predicted model of the previously published version of QVina(QVina 2), and QVina-W with different steps compared to Vina. The decimal numbers on both sides are the average Binding Energy difference

Similarly, Figure 4-20(A) shows a clear RMSD success enhancement of QuickVina-W over the original Vina 1.1.2, with 46% success in the **first** predicted mode, versus 39% for the Vina predictions. Similarly, Figure 4-20(B) shows that the quadrupled number of steps is by far much better than Vina considering **any (best)** predicted mode (72% for the new QuickVina versus 63% for the original Vina).

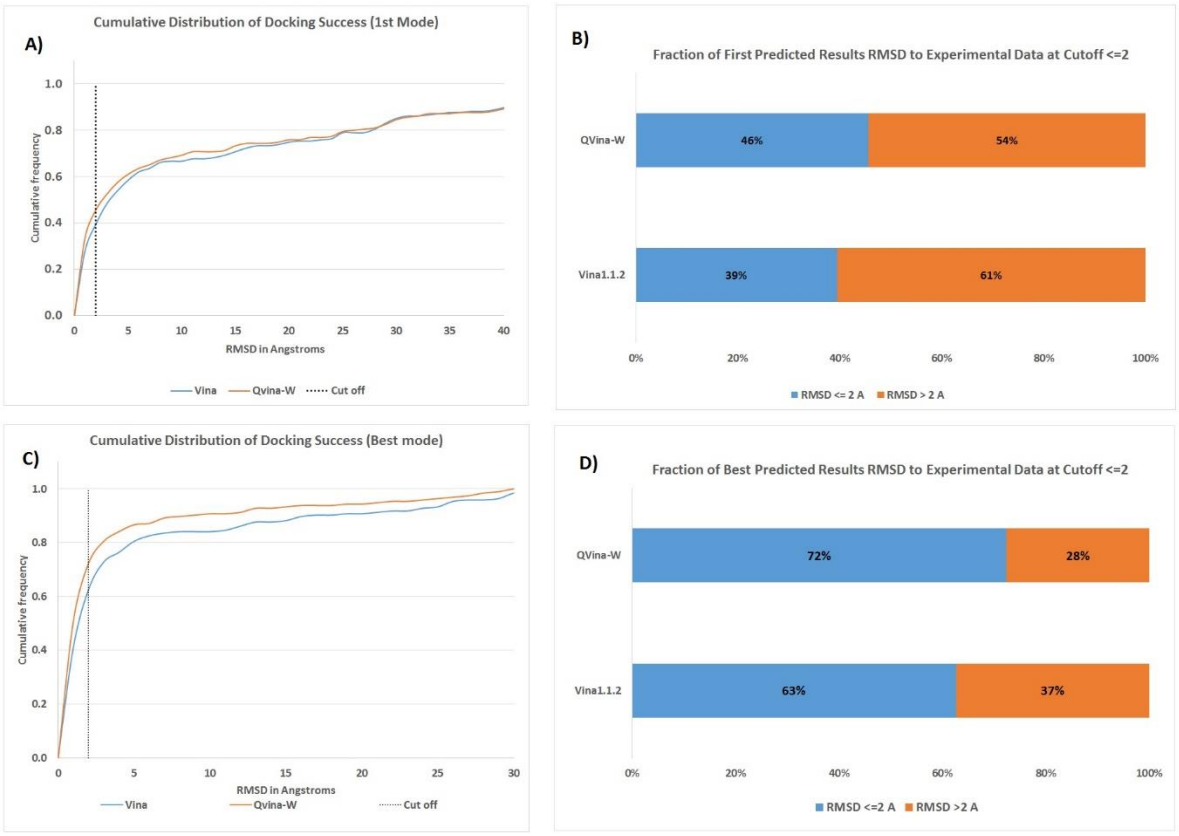
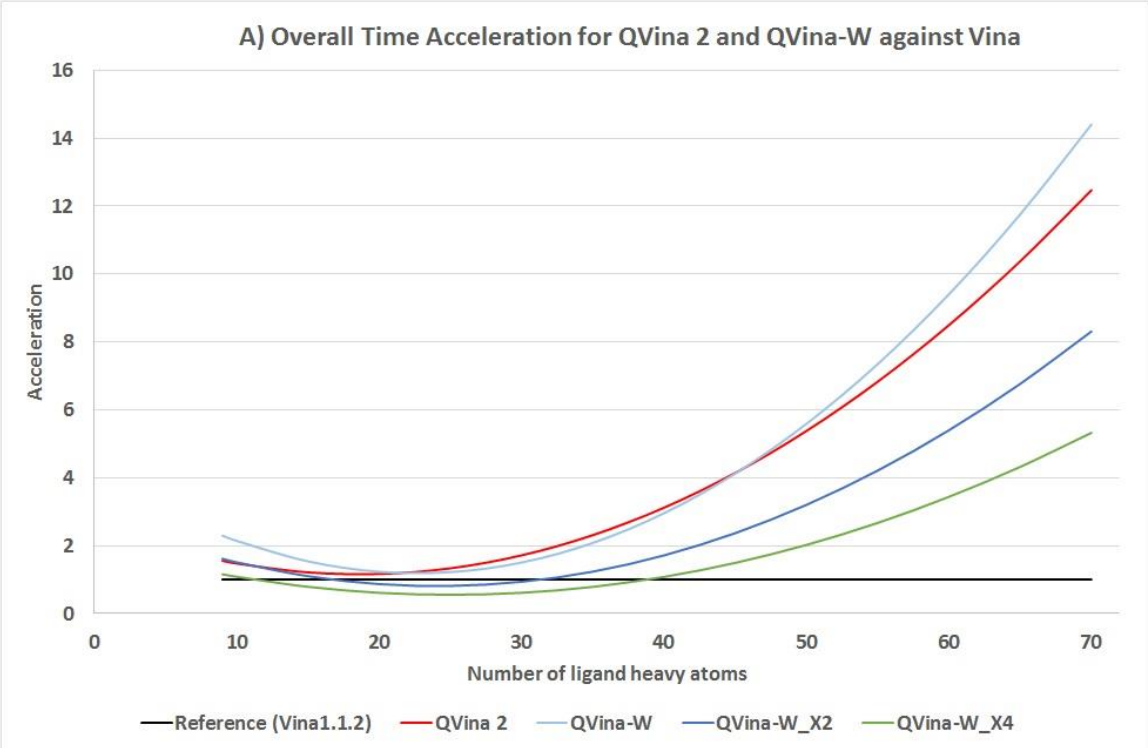


Figure 4-20 RMSD of Vina, QVina 2, and QVina-W
Relative frequency of successes using RMSD to experimental data for both Vina and QVina-W: A) RMSD distribution of the first mode. B) First mode Success at 2.0 A. C) RMSD distribution of the best mode. D) Best Mode Success at 2.0 A.

It is here worth mentioning that for blind docking experiments over the whole receptors surface, it is useful to consider all predicted modes, not the first one only.

4.4.2. Overall acceleration after multithreading the preparation overhead

Since the overhead time was accelerated as well, it is more legitimate to calculate the acceleration based on the overall time rather than the search time only. Figure 4-21(A) shows a crude calculation of the overall-time acceleration of previous QuickVina 2 and QuickVina-W steps X1, X2, and X4 against Vina. The overall-time acceleration, as calculated in equation 4-6, shows that the quadrupled QuickVina-W still has the superiority over Vina when the ligand contains a number ≤ 11 or ≥ 39 heavy atoms.



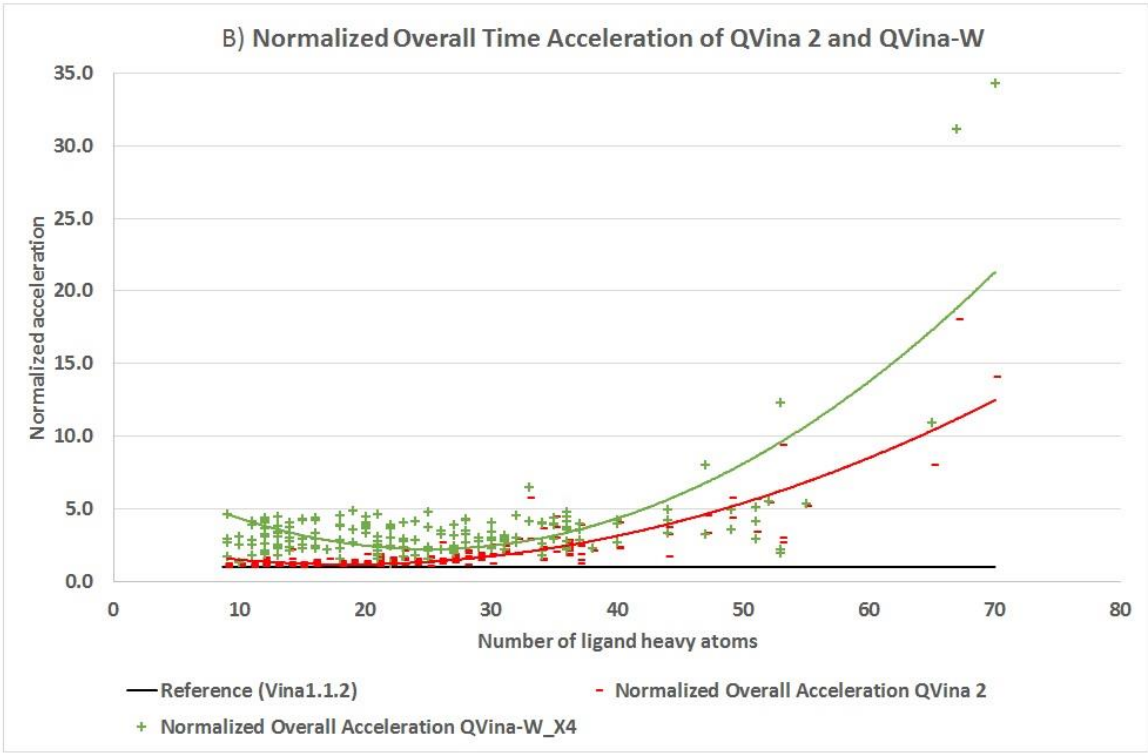


Figure 4-21 Acceleration of QVina 2 and QVina-W (different steps) against Vina
A) Overall time acceleration for QVina 2 and QVina-W steps X1, X2, X4 against Vina. B) Normalized overall time acceleration for QVina 2 and QVina-W

Finally, considering that a single run of the QuickVina-W is actually a double or quadrable run without having to repeat the overhead time, the acceleration calculation can be normalized by means of dividing the QuickVina time by its base (B ∈ {1, 2, 4}). This way, the calculation would be

$$normalized\ a_{To} = \frac{To_{Vina} * B}{To_{QVina}}\ , \tag{4-7}$$

where B determines the abovementioned QuickVina base. The exact normalized acceleration values are displayed in Figure 4-21(B) and summarized in Table *4-1*. They show that the quadrupled configuration is superior to the previous QuickVina 2 with a maximum normalized acceleration of 34.33 and average 3.60 folds versus respective values of 18.02 and 1.98 for the pervious QuickVina 2. It is noteworthy that the normalized acceleration average is only less than a factor of 2, because there is a direct relation between the ligand size (number of heavy atoms) and the gained acceleration, but the dataset ligand sizes are not evenly distributed. The histogram of number of heavy atoms, as seen in Figure 4-22, is right skewed, with 58.5% of the set having <=25 heavy atoms.

Table 4-1 Normalized overall time acceleration values (in relation to Vina)

Normalized overall time acceleration	Average	Maximum
QuickVina 2	1.98	18.02
QuickVina-W	3.60	34.33

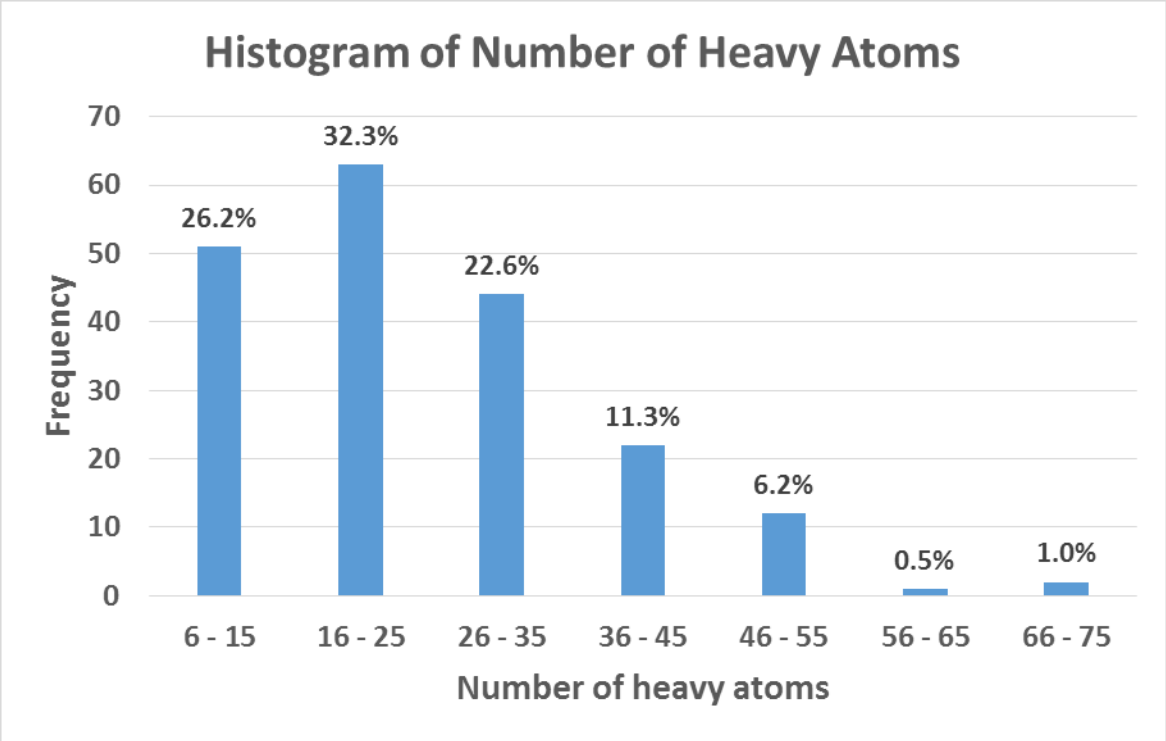


Figure 4-22 Histogram of frequency of number of heavy atoms for the PDBbind dataset
Obviously, 58.5% of the sample has less than or equal to 25 heavy atoms.

4.4.3. Application example of QuickVina-W

Using MayBridge dataset of 54520 molecules, and comparing the results with those of Vina, the difference between predicted binding energy by Vina and QuickVina-W was studied as shown in Figure 4-23. The difference between Vina and QuickVina-W predictions were quantified in the form of an integer number $\in$ [-9, 9]. QuickVina-W better prediction was indicated by a positive sign and Vina better predictions by a negative sign (9 means all QuickVina-W predictions are better than all Vina predictions, -9 means all are worse, 0 means all are identical). We care mainly about the Binding energy of first different results because the later the order of a pose in the list of results, the less important this result is (likely to be insignificant). This index is calculated by counting the difference between indices of first different modes, ignoring identical values and considering repeats. For instance, if QuickVina-W predictions binding energies (BEs) were (-7.8, -7.6, …) and Vina BEs were (-7.6, …) the first different difference is 1 because there is one 7.8 present in Vina side. Also if QuickVina-W predictions scored (-8.0, -8.0, -7.5, …) and Vina Scored (-8.0, -8.0, -8.0 …), after ignoring the first

two identical -8.0 from both sides, the first different difference is -1. Finally if QuickVina-W predictions scored (-8.1, -8.0, -7.6, …) and Vina Scored (-8.1, -8.0, -7.8, -7.8, -7.8,-7.5, …), the first different difference is -7.6 versus three repetitions of -7.8 results scored by Vina, so the first different difference would be -3.

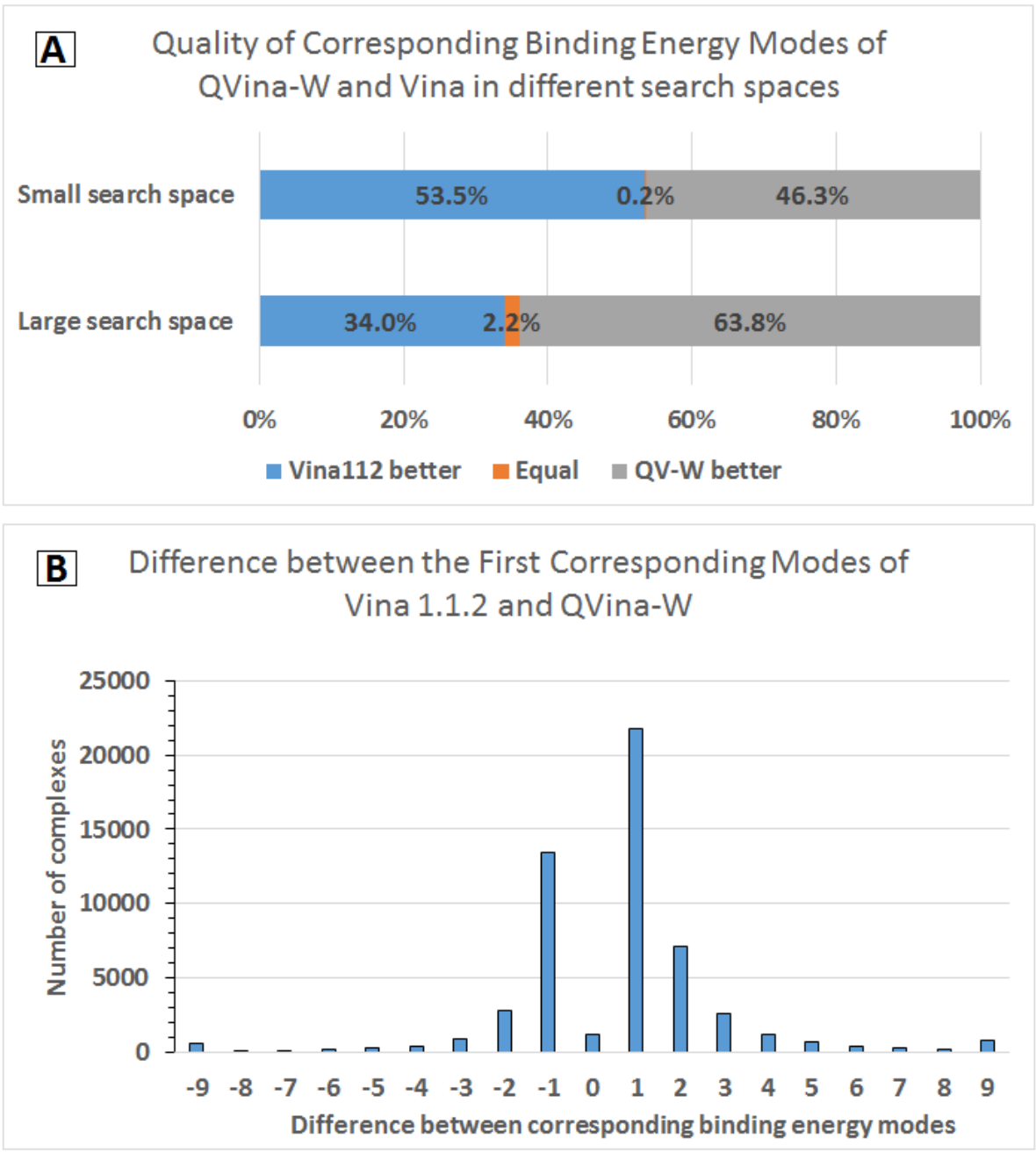


Figure 4-23 Comparison between binding energies of Vina 1.1.2 and QVina-W using the "first different difference" notation

Figure 4-23(A) shows that QuickVina-W prediction is comparable to Vina in small search space (when the target is small enough for the search to converge around the target) QuickVina-W is found to score 46.5% of predictions better than or equal to Vina. However, when the search space is large enough, this number rises to 66%, indicating that QuickVina-W search is be superior to Vina. Figure 4-23(B) shows detailed distribution of index of first different difference in the range [-9, 9].

Figure 4-24 shows a comparison plot between the binding energies of Vina 1.1.2 and QuickVina-W per ligand. The sums of the corresponding modes are shown in Figure 4-24(A). Corresponding means if for example, Vina returned 5 results and QuickVina-W returned 9 results, it would not be fair to compare the sum of all modes from Vina to those from QuickVina. Instead, the first 5 poses only were sum up from QuickVina-W and compared to all 5 poses from Vina. Figure 4-24(B) is a zoom in the range [-80, -70]. It shows that most of QuickVina-W predictions give more negative binding energies for the same ligand than Vina predictions.

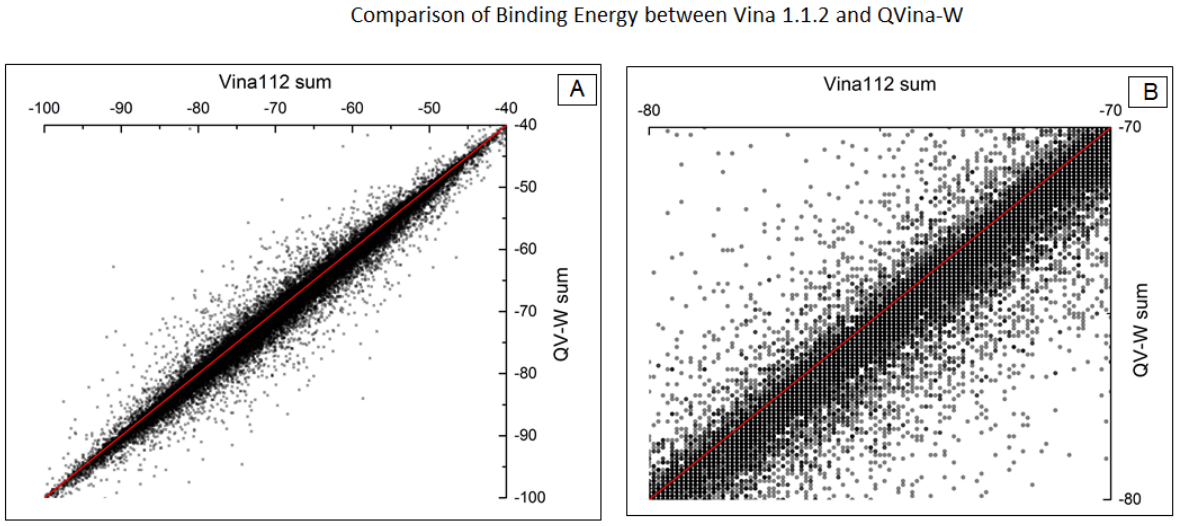


Figure 4-24 Comparison of Binding Energy between Vina 1.1.2 and QVina-W
Most of the QVina-W predictions tend to be of lower energies than those of Vina.

From all the results shown above, it can be concluded that the latest configuration of QuickVina with global buffer, explores four folds points in the search space than Vina and previous QuickVina 2, which obtained better results than Vina, yet in faster time compared to QuickVina 2. It

shows better results in terms of both Binding Energy and RMSD. It is faster than Vina in a crude comparison when the ligand heavy atoms are ≤11 or ≥39; and faster than both Vina and QuickVina 2 in a normalized acceleration where it scored 34.33-fold maximal acceleration and 3.60 folds average acceleration over Vina 1.1.2. The final configuration is QuickVina with circular individual buffer of size 5N, maximum checks 4N and Octree global buffer with cutoff of 5 Angstrom and maximum checks of 1N, where N is number of degrees of freedom.

This tool is released under the name “QuickVina-W” (QVina-W), which refers to the ability to work in (wide) search space. It is suitable for blind docking with its proven high accuracy and high speed.[152]

Finally, Figure 4-25 shows the big picture illustrating the position of all Vina family (Vina, QuickVina 1, QuickVina 2, and QuickVina-W) in relation to speed, accuracy, and size. Vina was developed with a high accuracy and rather high speed. QuickVina 1 superseded Vina in terms of speed, but it was mild in terms of accuracy. QuickVina 2 restored accuracy on the expense of little speed (still faster than Vina). Then QuickVina-W supersedes the three of them in terms of both speed and accuracy, and in addition, it lies in a higher level in the size dimension.

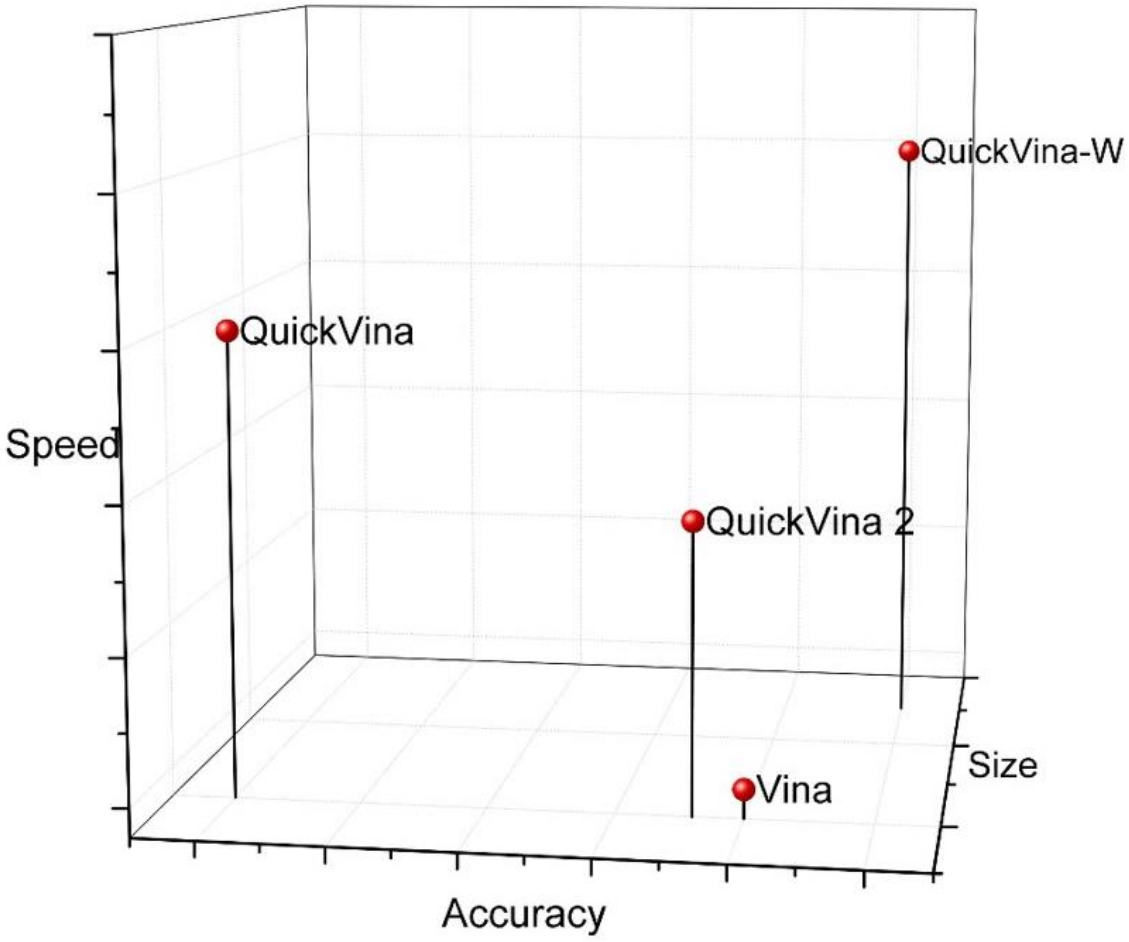


Figure 4-25 Full picture of all Vina family suit of docking tools

4.5. Conclusion

This work presents QuickVina-W, a new docking tool particularly useful for wide search space, especially for blind docking, that utilizes the powerful scoring function of AutoDock Vina, the accelerated search of QuickVina 2, and adds thorough search for wide search space. It is based on the observation that allowing a searching thread to communicate with other nearby threads to make use of their wisdom, would increase the speed and sensitivity of that searching thread. This communication was allowed by means of a global buffer that keeps high quality search history points from all threads.

To test for the hypothesis, the search process was analyzed to trace the Average Sum of Proximity relative Frequencies (ASoF) among all searching threads, along with its effect on the speed and sensitivity of decision taking, as well as the effect on increasing number of search steps on the search speed and accuracy. That proved the direct relation between the length of the search and ASoF, which is reflected on the search speed and accuracy, and that in turn implies higher probability for better results. QuickVina-W makes use of the acceleration and explores four folds the number of points that Vina used to explore in a more efficient way. In addition, the preparation overhead was multithreaded, which adds more to the overall time acceleration.

QuickVina-W proved to be faster than QuickVina 2 (with average and maximum normalized overall time accelerations of 3.60 and 34.33 folds in relation to Vina versus 1.98 and 18.02 respectively), yet better than Vina in terms of Binding Energy (78% of predictions with binding energy better than or equal to Vina) and RMSD (with success rate of 72% by QuickVina-W versus 63% by Vina).

4.5.1. Contribution in this chapter

- Introducing inter-process spatiotemporal integration between searching threads.
- Introducing QuickVina-W as a blind docking tool.
- Describing the relation between Average Sum of Proximity Relative Frequency (ASoF) and history guided search speed and accuracy.

4.5.2. Publications in this chapter

- "**Protein-Ligand Blind Docking Using QuickVina-W With Inter-Process Spatio-Temporal Integration**".
  Nafisa M. HASSAN +, Amr A. ALHOSSARY +, Yuguang MU, and Chee-Keong KWOH. Scientific Reports 7(1) (2017). DOI: 10.1038/s41598-017-15571-7. (Equally contributing 1st author)

Chapter 5 Future work

In this chapter, I shed the light on two directions to further continue on my previous work. In section 5.1, I will talk about how to extend QuickVina-W to distributed architecture. Then in section 5.2, I talk about applying it in Multilevel Coarse Graining.

5.1. Direction I: Extending QuickVina-W to distributed-memory architectures

Distributed memory architectures are becoming popular nowadays. To extend the method described in chapter 4 from shared memory system to a distributed memory system, we have three tasks to be fulfilled using the following approaches / alternatives:

1) Distributing the octree among the nodes:
    a. Each node has its full copy of the tree, that is synchronized with all other nodes' copies over intervals, or
    b. Each node keeps track of a part of the tree (according the geometric location it is responsible for). This alternative implies 2b as well.
2) Spreading threads over nodes:
    a. Confining each thread to a node regardless to the thread's current geometric location, or
    b. Dedicating each area of the search space to a node, that is responsible for performing the search in this area and keeping track of the history data, regardless to the number of threads that may traverse this area simultaneously.
3) We –obviously- have to distribute the tree and searching threads in the beginning, follow up the progress throughout the search, and collect the results in the end.

A naïve developer will go for 1b (and consequently 2b): When a thread enters exits an area, its current details will be passed to the node responsible for the new area, which in turn would create

a new local thread and loads it with data. However, this approach has a major drawback: During the search process, several threads would be attracted to binding pockets. Which leads to congestion of threads in one area and sparsity in other areas. In addition to the overhead of creating new threads whenever a new thread enters the zone.

A well versed developer -instead- would give each node an equal number of threads that does not vary in the runtime (alternatives1a and 2a). That will ensure unanimously equally distributed workload among the nodes all the time. The next question would be: "When to synchronize the tree?" A first guess answer is when a node is split into 8 children. This timing implies the least number of contained points per (new) node, which means the least likelihood to clash if two nodes are split simultaneously.

Any available MPI library (e.g. OpenMPI) can be used to facilitate communication among nodes in a distributed memory environment.

5.2. Direction II: Application of docking concepts in Multilevel coarse-graining

The thesis is about accelerating drug design workflow with focus on drug-receptor interaction modeling. So far, we have made some achievements towards this purpose in the molecular docking step, which is the static form of interaction modeling. This chapter presents how we can accelerate the Molecular Dynamics (MD) Simulation step, which is the dynamic form of interaction modeling, by incorporating the concepts formulated in this work into MD simulation packages to allow them to do Multilevel coarse graining.

5.2.1. Background

5.2.1.1. MD simulation

Molecular dynamics simulation is a numeric tool based on the Newton's equations of motion for N-body system simulations by iteratively updating particle forces and system potential energies

composed by inter-atomic/intra-atomic potentials or mechanical force fields. During MD simulations, coordinates, velocities, forces and potential energies are calculated every iterative step, whose temporal gap between each other is called “time step”. The smaller the time step, the more accurate but more computationally intensive the simulation is [97]. Detailed description of MD simulation is presented in section 1.3.4 [Molecular Dynamics Simulation]. I will not repeat them here to avoid redundancy.

5.2.1.2. Limitations of current MD simulation tools

Modeling the biological processes of protein folding/unfolding suffer from both time scale and size scale limitations. The time scale limitation is because these transformations usually occur over periods of time in order of seconds, while modeling the highest frequency movements (hydrogen atom oscillations) requires using a time step in order of femtoseconds. The size scale limitation is attributed to the fact that such large allosteric transformations require huge simulation boxes, to ensure there will be no clashes during the transformation process. More details are presented in section 1.3.4.2 [Limitations of classical MD Simulation methods].

5.2.1.3. Coarse graining MD simulation

Coarse-graining is a molecular dynamics technique that aims at reducing the computational cost by replacing a group of atoms (fine grains) with a single super atom (coarse-grain), which represents the collective effect of all original atoms. This process necessitates the presence of three mechanisms: 1) a mechanism to map a set of fine grains to a coarse-grain, 2) a mechanism to determine the position of each of the newly introduced coarse grains, and 3) a mechanism to derive a new modified force field that suits the newly introduced coarse grains. Limitations of Coarse Grained MD simulation are mentioned previously in section 1.4.2 [Molecular dynamics simulation].

5.2.2. Too detail or not too detail? That is the question.

The ultimate question that needs to be addressed before designing any coarse grained simulation is how much detail is needed in that simulation. There is always a tradeoff between having a detailed

representation and an efficient simulation. A detailed system is more accurate but more resource consuming, in terms of time and computational power. On the other hand, a less detailed system would require less resources, but it might lose some of the details of interest.

5.2.2.1. Multilevel Coarse Graining is the answer

A system of hierarchal coarse graining abstractions of simulation can escape the tradeoff. The aim of such hierarchal system is to model the important details and to hide/bypass other less important details in a dynamic way. That is to say, for every level of size and time scale, there is a set of forces or interaction (Van der Waals interactions, hydrogen bonding, Electrostatic interactions, torque, Brownian movement, etc.) that prevail.

Different chemical groups can be modeled in a higher abstraction level, (e.g. a coarse grain), then functional groups, domains, subunits, etc. Abstraction keeps evolving as necessary. Every higher level of abstraction can interact on a wider size scale using a longer time step. On the other hand, when two bodies come to proximity, their interactions are modeled using a more detailed method, (i.e. lower abstraction level).

5.2.3. Challenges facing Multilevel Coarse Graining

In addition to the need to achieve the abovementioned coarse graining requirements, (Multilevel Coarse Graining) MLCG needs other requirements, which are challenges in their nature.

5.2.3.1. Logically dividing a particle / joining different particles in run time

First problem: How to define the level of abstraction is a challenge. There is still no universal clue on where to stop abstraction or division. Measures like flexibility, hydrophilicity, how exposed, and how deep (how much distance to SASA) a domain is, can be probable measures to determine when to stop.

Second problem: To the best of my knowledge, all coarse graining implementations are static, i.e. they don't change in run time. Instead, the coarse beads are defined once in the beginning, and the configurations stays till the simulation ends. One possible approach is to run several small

simulations, then reevaluating the abstraction status on intervals. If two particles are found to be close enough to each other, we can consider joining them. Similarly, if a particle is found to be affected by opposing exerted forces, we should consider splitting that particle. In general, overcoming that challenge, needs either a simulation package that accepts several IDs for the same atom (according to the assigned particle), or non-consecutive atom IDs.

5.2.3.1.1 Finite Element Method

Finite element methods (FEM) are a group of modeling methods, commonly used in structural mechanics. In this method, the system is modeled as a set of finite elements of appropriate properties (e.g. stiffness, toughness, etc.), interconnected at points called nodes. A macromolecular system can be modeled as a set of connected rigid elements. FEM can model the deformation that occurs upon exerting forces on rigid bodies, as well as progression of energy through them.

5.2.3.2. Hybrid time step

Synchronizing how different particles of different sizes behave over different time scales is a challenge. The rule of thumb here is to consider any particle of lower abstraction level or whose movements are of high frequency as a rigid body, while it is moving within the higher level particle.

One approach might be to run the whole system for one interval using higher abstraction associated with longer time step, then going for a second interval with a lower abstraction associated with shorter time step, and so on.

Another approach is to isolate some of the particles (the higher order ones), simulate their interaction, as if the lower order ones are not there. Then to project the effects of the coarse grained motions on the lower order ones. This approach is more complicated, but can lead to realistic results in a relatively shorter time.

Third approach is to consider time-unbound simulations. Monte Carlo (MC) and molecular docking are two examples of this approach. Both MC and Docking have the advantage of passing

energy barriers. Docking has an additional advantage of fast convergence, while MC simulation has the advantage of producing a progressive trajectory.

5.2.3.3. Solvent effect

Modeling the solvent effect explicit cause several problems in simulation. In MD simulation, solvent molecules are the usually most prevalent molecules in cubical simulation boxes. Although the thin layer of solvent surrounding interacting molecules are useful to model any biological phenomena, the rest of molecules have very little information to add to the simulation. Moreover, their collective effect can be simply modeled as additional parameters of conductivity, permeability, and molecule diffusion. In MC simulation, having the explicit solvent molecules pumping into a moving particle, leads to high failure rate due to clashing.

For a coarse grained system, it's better to avoid explicit solvent modeling and to adhere to implicit solvent effect instead. For a multilevel coarse graining, it is possible to add the solvent particles while the system is being run on the all-atomic level, and to remove them back and keep their effect in higher levels.

5.2.4. How can my previous work fit in MLCG

Rigidifying a body (Ignoring its high frequency movements), enables applying the docking search algorithms used to predict plausible poses on it. Domains of the protein can be treated the same way ligands are treated in a blind docking process, applying the concepts and techniques developed in QuickVina-W, described in Chapter 4 [Blind Docking with inter-process spatio-temporal integration], as Domain-Domain docking can help finding simulation metastable states.

Chapter 6 Conclusion

This thesis describes the work done towards accelerating the bottlenecks of computer aided structure guided drug design.

In chapter 1, I supplied a background introduction about the computer-Aided structure-guided drug design workflow and a literature review on the recent advances in that domain.

In chapter 2, I described the work done to accelerate the sampling algorithm of the molecular docking tool AutoDock Vina, which resulted in the release of "QuickVina 2", a fast, accurate, and reliable docking tool that depends on the high scoring power of Vina and the advanced sampling algorithm of QuickVina, without compromise in the search accuracy. It was tested against the core set of BDBbind 2014 using the default exhaustiveness and showed that 70% of the results have predicted binding energies equal or better than Vina. The remaining 30% had average Energy difference only 0.58 Kcal/mol. The Pearson's correlation coefficient (r) between Vina's and QuickVina 2's binding energy was 0.967 for the first predicted mode and 0.911 for the sum of all predicted modes. QuickVina 2 is more accurate than GOLD 5.2 and is only slightly less accurate than Dock 6.6.

In chapter 3, I described the work done to propose 16 drug fragments for Dengue Virus non-structure protein 5 (DENV-NS5). To achieve that goal, the selection criteria of receptors used in the multiple receptor docking process was revised. This work involves also comparing "QuickVina 2" output to AutoDock Vina. The work highlights the importance of selecting only the drugable conformations and filtering out the noisy ones, to ensure high diversity and low false negative rates. Both QuickVina 2 and AutoDock Vina detected the same 13 fragments with slight differences in their estimated Binding energies while QuickVina 2 detected three additional fragments. That double ensured that QuickVina 2 has a comparable performance in relation to AutoDock Vina.

In Chapter 4, I described the work done to enable QuickVina to do blind docking. This work resulted in the development of QuickVina-W, another new docking tool for searching wide search

boxes, which is particularly useful for blind docking. QuickVina-W explores four folds the number of points Vina explores, in a higher speed. That speed even keeps increasing as the search progresses. When tested against BDBbind dataset, QuickVina-W proved to be faster than QuickVina 2 (with average and maximum normalized overall time accelerations of 3.60 and 34.33 folds in relation to Vina versus 1.98 and 18.02 respectively), yet better than AutoDock Vina in terms of Binding Energy (78% of predictions with binding energy better than or equal to Vina) and RMSD (with success rate of 72% by QuickVina-W versus 63% by Vina). QuickVina-W is based on is the observation that allowing a searching thread to communicate with other nearby threads to make use of their wisdom, would increase the speed and sensitivity of that searching thread. The research done in this chapter also proved the direct relation between the length of the search and increasing Average Sum of proximity relative Frequency (ASoF) of searching threads, which -if used - is reflected on the search speed and accuracy.

In chapter 5, I gave a brief about future directions of work that can be taken to accelerate the drug design workflow. First by making use of increasingly being available distributed memory supercomputers, then, using the work done in this thesis, to accelerate molecular dynamics simulation, using a hierarchal system of multilevel coarse-graining / rigid body simulation abstractions.

Appendices

Appendix A: RMSD Calculation Method as described by AutoDock Vina team

AutoDock Vina team described a way to calculate the RMSD between experimental and predicted results, which "takes into account symmetry, partial symmetry (e.g., symmetry within a rotatable branch) and near symmetry, in a simple heuristic way" [83]. It is defined as

$$\boldsymbol{RMSD_{ab} = \max(RMSD'_{ab}, RMSD'_{ba})} \quad \textbf{(A-1)},$$

$$\boldsymbol{RMSD'_{ab} = \sqrt{\frac{1}{N}\sum_i min_j\ r_{ij}^2}} \quad \textbf{(A-2)},$$

where the sum is over all heavy atoms N, and minimum is over all atoms in b with same atom type as atom i of structure a [83].

References

1. Krogsgaard-Larsen, P., K. Strømgaard, and U. Madsen, *Textbook of drug design and discovery*. 4th ed. 2010, Boca Raton, FL: CRC Press/Taylor & Francis. xv, 460 p.
2. Tollenaere, J.P., *The role of structure-based ligand design and molecular modelling in drug discovery.* Pharm World Sci, 1996. **18**(2): p. 56-62.
3. Langer, T. and R.D. Hoffmann, *Pharmacophores and pharmacophore searches [electronic resource] / edited by Thierry Langer and Rémy D. Hoffmann*. Methods and principles in medicinal chemistry: 32. 2006: Weinheim : Wiley-VCH ; [Chichester : John Wiley, distributor], c2006.
4. Güner, O.F., *Pharmacophore perception, development, and use in drug design / edited by Osman F. Güner*. IUL biotechnology series. 2000: LaJolla, CA : International University Line, c2000.
5. Tari, L.W., *Structure-Based Drug Discovery [electronic resource] / edited by Leslie W. Tari*. Methods in Molecular Biology, Methods and Protocols: 841. 2012: Totowa, NJ : Humana Press, 2012.
6. Cozza, G., *The Development of CK2 Inhibitors: From Traditional Pharmacology to in Silico Rational Drug Design.* Pharmaceuticals (Basel), 2017. **10**(1).
7. Verlinde, C.L.M.J. and W.G.J. Hol, *Structure-based drug design: progress, results and challenges.* Structure, 1994. **2**(7): p. 577-587.
8. Schneider, G. and U. Fechner, *Computer-based de novo design of drug-like molecules.* Nat Rev Drug Discov, 2005. **4**(8): p. 649-63.
9. ALHOSSARY, A.K., Chee Keong; MU, Yuguang, *Recent developments in drug design workflow management systems*, in *CDAMIES 2018*. 2018: India.
10. Kier, L.B., *Molecular orbital calculation of preferred conformations of acetylcholine, muscarine, and muscarone.* Mol Pharmacol, 1967. **3**(5): p. 487-94.
11. Kier, L.B., *Molecular orbital theory in drug research*. 1971: Academic Press.
12. Drie, J.H.V., *Monty Kier and the Origin of the Pharmacophore Concept.* Internet Electronic Journal of Molecular Design - IEJMD, 2007. **6**: p. 271-279.
13. Huang, B.D. and M. Schroeder, *LIGSITE(csc): predicting ligand binding sites using the Connolly surface and degree of conservation.* Bmc Structural Biology, 2006. **6**.
14. Levitt, D.G. and L.J. Banaszak, *POCKET: A computer graphies method for identifying and displaying protein cavities and their surrounding amino acids.* Journal of Molecular Graphics, 1992. **10**(4): p. 229-234.
15. Laskowski, R.A., *Surfnet - a Program for Visualizing Molecular-Surfaces, Cavities, and Intermolecular Interactions.* Journal of Molecular Graphics, 1995. **13**(5): p. 323-&.
16. Le Guilloux, V., P. Schmidtke, and P. Tuffery, *Fpocket: An open source platform for ligand pocket detection.* Bmc Bioinformatics, 2009. **10**.
17. Schmidtke, P., et al., *fpocket: online tools for protein ensemble pocket detection and tracking.* Nucleic Acids Research, 2010. **38**: p. W582-W589.
18. Tan, K.P., R. Varadarajan, and M.S. Madhusudhan, *DEPTH: a web server to compute depth and predict small-molecule binding cavities in proteins.* Nucleic Acids Res, 2011. **39**(Web Server issue): p. W242-8.
19. Van Der Spoel, D., et al., *GROMACS: fast, flexible, and free.* J Comput Chem, 2005. **26**(16): p. 1701-18.
20. Hess, B., et al., *GROMACS 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation.* Journal of Chemical Theory and Computation, 2008. **4**(3): p. 435-447.
21. Cornell, W.D., et al., *A second generation force field for the simulation of proteins, nucleic acids, and organic molecules (vol 117, pg 5179, 1995).* Journal of the American Chemical Society, 1996. **118**(9): p. 2309-2309.

22. Phillips, J.C., et al., *Scalable molecular dynamics with NAMD.* Journal of Computational Chemistry, 2005. **26**(16): p. 1781-1802.
23. Plimpton, S., *Fast Parallel Algorithms for Short-Range Molecular-Dynamics.* Journal of Computational Physics, 1995. **117**(1): p. 1-19.
24. Harvey, M.J., G. Giupponi, and G. De Fabritiis, *ACEMD: Accelerating Biomolecular Dynamics in the Microsecond Time Scale.* Journal of Chemical Theory and Computation, 2009. **5**(6): p. 1632-1639.
25. Johnston, M.A., I.F. Galvan, and J. Villa-Freixa, *Framework-based design of a new all-purpose molecular simulation application: The Adun simulator.* Journal of Computational Chemistry, 2005. **26**(15): p. 1647-1659.
26. Verlet, L., *Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules.* Physical Review, 1967. **159**(1): p. 98-103.
27. Hockney, R.W. and J.W. Eastwood, *Computer Simulation Using Particles*. 1981: McGraw-Hill.
28. Swope, W.C., et al., *A computer simulation method for the calculation of equilibrium constants for the formation of physical clusters of molecules: Application to small water clusters.* J. Chem. Phys., 1982. **76**(1): p. 637-649.
29. Beeman, D., *Some multistep methods for use in molecular dynamics calculations.* Journal of Computational Physics, 1976. **20**(2): p. 130-139.
30. Toukmaji, A.Y. and J.A. Board Jr, *Ewald summation techniques in perspective: a survey.* Computer Physics Communications, 1996. **95**(2–3): p. 73-92.
31. Eastwood, J.W., R.W. Hockney, and D.N. Lawrence, *P3M3DP-the three-dimensional periodic particle-particle/particle-mesh program.* Computer Physics Communications, 1984. **35**(0): p. C-618-C-619.
32. Zhang, Z.Y., et al., *A Systematic Methodology for Defining Coarse-Grained Sites in Large Biomolecules.* Biophysical Journal, 2008. **95**(11): p. 5073-5083.
33. Shih, A.Y., et al., *Coarse Grained Protein–Lipid Model with Application to Lipoprotein Particles†.* The Journal of Physical Chemistry B, 2006. **110**(8): p. 3674-3684.
34. Shih, A.Y., et al., *Assembly of lipoprotein particles revealed by coarse-grained molecular dynamics simulations.* Journal of Structural Biology, 2007. **157**(3): p. 579-592.
35. Arkhipov, A., Y. Yin, and K. Schulten, *Four-scale description of membrane sculpting by BAR domains.* Biophys J, 2008. **95**(6): p. 2806-21.
36. Lu, L., J.F. Dama, and G.A. Voth, *Fitting coarse-grained distribution functions through an iterative force-matching method.* J Chem Phys, 2013. **139**(12): p. 121906.
37. Larini, L., L. Lu, and G.A. Voth, *The multiscale coarse-graining method. VI. Implementation of three-body coarse-grained potentials.* The Journal of Chemical Physics, 2010. **132**(16): p. 164107.
38. Lu, L.Y. and G.A. Voth, *Systematic Coarse-graining of a Multicomponent Lipid Bilayer.* Journal of Physical Chemistry B, 2009. **113**(5): p. 1501-1510.
39. Lu, L. and G.A. Voth, *The multiscale coarse-graining method. VII. Free energy decomposition of coarse-grained effective potentials.* The Journal of Chemical Physics, 2011. **134**(22): p. 224107.
40. Das, A. and H.C. Andersen, *The multiscale coarse-graining method. III. A test of pairwise additivity of the coarse-grained potential and of new basis functions for the variational calculation.* J Chem Phys, 2009. **131**(3): p. 034102.
41. Noid, W.G., et al., *The multiscale coarse-graining method. II. Numerical implementation for coarse-grained molecular models.* J Chem Phys, 2008. **128**(24): p. 244115.
42. Das, A., et al., *The multiscale coarse-graining method. X. Improved algorithms for constructing coarse-grained potentials for molecular systems.* J Chem Phys, 2012. **136**(19): p. 194115.
43. Christen, M., et al., *The GROMOS software for biomolecular simulation GROMOS05, Journal of Computational Chemistry Volume 26, Issue 16.* Journal of Computational Chemistry, 2005. **26**(16): p. 1719-1751.

44. van Gunsteren, W.F., et al., *Biomolecular Simulation: The {GROMOS96} manual and userguide*. 1996: Hochschuleverlag AG an der ETH Zürich.
45. Cornell, W.D., et al., *A Second Generation Force Field for the Simulation of Proteins, Nucleic Acids, and Organic Molecules.* Journal of the American Chemical Society, 1995. **117**(19): p. 5179-5197.
46. Brooks, B.R., et al., *CHARMM A program for macromolecular energy, minimization, and dynamics calculations, Journal of Computational Chemistry Volume 4, Issue 2.* Journal of Computational Chemistry, 1983. **4**(2): p. 187-217.
47. Jorgensen, W.L., D.S. Maxwell, and J. TiradoRives, *Development and testing of the OPLS all-atom force field on conformational energetics and properties of organic liquids.* Journal of the American Chemical Society, 1996. **118**(45): p. 11225-11236.
48. Jorgensen, W.L. and J. Tirado-Rives, *The OPLS [optimized potentials for liquid simulations] potential functions for proteins, energy minimizations for crystals of cyclic peptides and crambin.* Journal of the American Chemical Society, 1988. **110**(6): p. 1657-1666.
49. Tersoff, J., *Modeling Solid-State Chemistry - Interatomic Potentials for Multicomponent Systems.* Physical Review B, 1989. **39**(8): p. 5566-5568.
50. Daw, M.S., S.M. Foiles, and M.I. Baskes, *The Embedded-Atom Method - a Review of Theory and Applications.* Materials Science Reports, 1993. **9**(7-8): p. 251-310.
51. Cleri, F. and V. Rosato, *Tight-Binding Potentials for Transition-Metals and Alloys.* Physical Review B, 1993. **48**(1): p. 22-33.
52. Lamoureux, G., et al., *A polarizable model of water for molecular dynamics simulations of biomolecules.* Chemical Physics Letters, 2006. **418**(1-3): p. 245-249.
53. Patel, S., A.D. Mackerell, Jr., and C.L. Brooks, 3rd, *CHARMM fluctuating charge force field for proteins: II protein/solvent properties from molecular dynamics simulations using a nonadditive electrostatic model.* J Comput Chem, 2004. **25**(12): p. 1504-14.
54. Marrink, S.J., A.H. de Vries, and A.E. Mark, *Coarse grained model for semiquantitative lipid simulations.* Journal of Physical Chemistry B, 2004. **108**(2): p. 750-760.
55. Marrink, S.J., et al., *The MARTINI force field: coarse grained model for biomolecular simulations.* J Phys Chem B, 2007. **111**(27): p. 7812-24.
56. Korkut, A. and W.A. Hendrickson, *A force field for virtual atom molecular mechanics of proteins.* Proc Natl Acad Sci U S A, 2009. **106**(37): p. 15667-72.
57. O'Brien, J.F. and J.K. Hodgins, *Graphical modeling and animation of brittle fracture.* Siggraph 99 Conference Proceedings, 1999: p. 137-146.
58. Izumi, S., T. Kawakami, and S. Sakai. *A FEM-MD combination method for silicon*. in *Simulation of Semiconductor Processes and Devices, 1999. SISPAD '99. 1999 International Conference on*. 1999.
59. Smirnova, J.A., L.V. Zhigilei, and B.J. Garrison, *A combined molecular dynamics and finite element method technique applied to laser induced pressure wave propagation.* Computer Physics Communications, 1999. **118**(1): p. 11-16.
60. Lee, Y.C. and C. Basaran, *A multiscale modeling technique for bridging molecular dynamics with finite element method.* Journal of Computational Physics, 2013. **253**: p. 64-85.
61. Clark, D.E., et al., *PRO-LIGAND: an approach to de novo molecular design. 1. Application to the design of organic molecules.* J Comput Aided Mol Des, 1995. **9**(1): p. 13-32.
62. Gillet, V., et al., *SPROUT: a program for structure generation.* J Comput Aided Mol Des, 1993. **7**(2): p. 127-53.
63. Wang, R.X., Y. Gao, and L.H. Lai, *LigBuilder: A multi-purpose program for structure-based drug design.* Journal of Molecular Modeling, 2000. **6**(7-8): p. 498-516.
64. Yuan, Y., J. Pei, and L. Lai, *LigBuilder 2: a practical de novo drug design approach.* J Chem Inf Model, 2011. **51**(5): p. 1083-91.
65. Durrant, J.D., R.E. Amaro, and J.A. McCammon, *AutoGrow: a novel algorithm for protein inhibitor design.* Chem Biol Drug Des, 2009. **73**(2): p. 168-78.

66. Walters, W.P., M.T. Stahl, and M.A. Murcko, *Virtual screening - an overview.* Drug Discovery Today, 1998. **3**(4): p. 160-178.
67. Biesiada, J., et al., *Survey of public domain software for docking simulations and virtual screening.* Hum Genomics, 2011. **5**(5): p. 497-505.
68. Brozell, S.R., et al., *Evaluation of DOCK 6 as a pose generation and database enrichment tool.* J Comput Aided Mol Des, 2012. **26**(6): p. 749-73.
69. Brozell, S.R. and D.A. Case, *Advanced scoring functions in DOCK 6.* Abstracts of Papers of the American Chemical Society, 2011. **241**.
70. Lang, P.T., et al., *DOCK 6: combining techniques to model RNA-small molecule complexes.* RNA, 2009. **15**(6): p. 1219-30.
71. Kuntz, I.D., et al., *A geometric approach to macromolecule-ligand interactions.* J Mol Biol, 1982. **161**(2): p. 269-88.
72. Shoichet, B.K., D.L. Bodian, and I.D. Kuntz, *Molecular Docking Using Shape Descriptors.* Journal of Computational Chemistry, 1992. **13**(3): p. 380-397.
73. Meng, E.C., B.K. Shoichet, and I.D. Kuntz, *Automated Docking with Grid-Based Energy Evaluation.* Journal of Computational Chemistry, 1992. **13**(4): p. 505-524.
74. Kramer, B., M. Rarey, and T. Lengauer, *Evaluation of the FLEXX incremental construction algorithm for protein-ligand docking.* Proteins, 1999. **37**(2): p. 228-41.
75. Friesner, R.A., et al., *Glide: A new approach for rapid, accurate docking and scoring. 1. Method and assessment of docking accuracy.* Journal of Medicinal Chemistry, 2004. **47**(7): p. 1739-1749.
76. Halgren, T.A., et al., *Glide: a new approach for rapid, accurate docking and scoring. 2. Enrichment factors in database screening.* J Med Chem, 2004. **47**(7): p. 1750-9.
77. Neves, M.A., M. Totrov, and R. Abagyan, *Docking and scoring with ICM: the benchmarking results and strategies for improvement.* J Comput Aided Mol Des, 2012. **26**(6): p. 675-86.
78. Joseph-McCarthy, D., et al., *Pharmacophore-based molecular docking to account for ligand flexibility.* Proteins, 2003. **51**(2): p. 172-88.
79. Jain, A.N., *Surflex: Fully automatic flexible molecular docking using a molecular similarity-based search engine.* Journal of Medicinal Chemistry, 2003. **46**(4): p. 499-511.
80. Huang, N., B.K. Shoichet, and J.J. Irwin, *Benchmarking sets for molecular docking.* J Med Chem, 2006. **49**(23): p. 6789-801.
81. Irwin, J.J., *Community benchmarks for virtual screening.* J Comput Aided Mol Des, 2008. **22**(3-4): p. 193-9.
82. Cross, J.B., et al., *Comparison of several molecular docking programs: pose prediction and virtual screening accuracy.* J Chem Inf Model, 2009. **49**(6): p. 1455-74.
83. Trott, O. 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, 2010. **31**(2): p. 455-61.
84. Wang, Z., et al., *Comprehensive evaluation of ten docking programs on a diverse set of protein-ligand complexes: the prediction accuracy of sampling power and scoring power.* Phys Chem Chem Phys, 2016. **18**(18): p. 12964-75.
85. Zsoldos, Z., et al., *eHiTS: a new fast, exhaustive flexible ligand docking system.* J Mol Graph Model, 2007. **26**(1): p. 198-212.
86. Zsoldos, Z., et al., *eHiTS: an innovative approach to the docking and scoring function problems.* Curr Protein Pept Sci, 2006. **7**(5): p. 421-35.
87. Jones, G., et al., *Development and validation of a genetic algorithm for flexible docking.* Journal of Molecular Biology, 1997. **267**(3): p. 727-748.
88. Verdonk, M.L., et al., *Improved protein-ligand docking using GOLD.* Proteins-Structure Function and Genetics, 2003. **52**(4): p. 609-623.
89. Morris, G.M., et al., *AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility.* J Comput Chem, 2009. **30**(16): p. 2785-91.

90. Morris, G.M., et al., *Distributed automated docking of flexible ligands to proteins: parallel applications of AutoDock 2.4.* J Comput Aided Mol Des, 1996. **10**(4): p. 293-304.
91. Goodsell, D.S., G.M. Morris, and A.J. Olson, *Automated docking of flexible ligands: applications of AutoDock.* J Mol Recognit, 1996. **9**(1): p. 1-5.
92. Zhao, Y. and M.F. Sanner, *FLIPDock: docking flexible ligands into flexible receptors.* Proteins, 2007. **68**(3): p. 726-37.
93. Corbeil, C.R., P. Englebienne, and N. Moitessier, *Docking ligands into flexible and solvated macromolecules. 1. Development and validation of FITTED 1.0.* J Chem Inf Model, 2007. **47**(2): p. 435-49.
94. Gupta, A., et al., *ParDOCK: an all atom energy based Monte Carlo docking protocol for protein-ligand complexes.* Protein Pept Lett, 2007. **14**(7): p. 632-46.
95. Leaver-Fay, A., et al., *ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules.* Methods Enzymol, 2011. **487**: p. 545-74.
96. Tietze, S. and J. Apostolakis, *GlamDock: development and validation of a new docking tool on several thousand protein-ligand complexes.* J Chem Inf Model, 2007. **47**(4): p. 1657-72.
97. Zheng, L., et al., *Molecular Dynamics and Simulation*, in *Encyclopedia of Bioinformatics and Computational Biology*. 2019. p. 550-566.
98. Alhossary, A., et al., *Fast, accurate, and reliable molecular docking with QuickVina 2.* Bioinformatics, 2015. **31**(13): p. 2214-2216.
99. Garcia-Godoy, M.J., et al., *Solving molecular docking problems with multi-objective metaheuristics.* Molecules, 2015. **20**(6): p. 10154-83.
100. Huey, R., et al., *A semiempirical free energy force field with charge-based desolvation.* J Comput Chem, 2007. **28**(6): p. 1145-52.
101. Thomsen, R., *Flexible ligand docking using evolutionary algorithms: investigating the effects of variation operators and local search hybrids.* Biosystems, 2003. **72**(1-2): p. 57-73.
102. Handoko, S.D., et al., *QuickVina: accelerating AutoDock Vina using gradient-based heuristics for global optimization.* IEEE/ACM Trans Comput Biol Bioinform, 2012. **9**(5): p. 1266-72.
103. Rodgers, D.P., *Improvements in multiprocessor system design.* SIGARCH Comput. Archit. News, 1985. **13**(3): p. 225-231.
104. Nocedal, J., S.J. Wright, and SpringerLink (Online service). *Numerical Optimization*. Second Edition. ed. Springer Series in Operations Research and Financial Engineering,. 2006, New York, NY: Springer New York. XXII, 664 s. 85 illus.
105. Ong, Y.S. and A.J. Keane, *Meta-Lamarckian Learning in Memetic Algorithms.* IEEE Transactions on Evolutionary Computation, 2004. **8**(2): p. 99-110.
106. ALHOSSARY, A., et al., *Fast, Accurate, and Reliable Molecular Docking with QuickVina 2.* Bioinformatics, 2015.
107. Cheng, T., et al., *Comparative assessment of scoring functions on a diverse test set.* J Chem Inf Model, 2009. **49**(4): p. 1079-93.
108. Wang, R.X., et al., *The PDBbind database: collection of binding affinities for protein-ligand complexes with known three-dimensional structures.* J Med Chem, 2004. **47**(12): p. 2977-2980.
109. Wang, R.X., et al., *The PDBbind database: Methodologies and updates.* J Med Chem, 2005. **48**(12): p. 4111-4119.
110. Pettersen, E.F., et al., *UCSF Chimera--a visualization system for exploratory research and analysis.* J Comput Chem, 2004. **25**(13): p. 1605-12.
111. Gasteiger, J. and M. Marsili, *A new model for calculating atomic charges in molecules.* Tetrahedron Letters, 1978. **19**(34): p. 3181-3184.
112. (WHO), W.H.O. *Dengue and severe dengue Fact sheet*. 9/7/2014]; Available from: http://www.who.int/mediacentre/factsheets/fs117/en/index.html.
113. Romi, R., F. Severini, and L. Toma, *Cold acclimation and overwintering of female Aedes albopictus in Roma.* J Am Mosq Control Assoc, 2006. **22**(1): p. 149-51.

114. Dejnirattisai, W., et al., *Cross-reacting antibodies enhance dengue virus infection in humans.* Science, 2010. **328**(5979): p. 745-8.
115. Halstead, S.B., *Dengue hemorrhagic fever: two infections and antibody dependent enhancement, a brief history and personal memoir.* Rev Cubana Med Trop, 2002. **54**(3): p. 171-9.
116. Mandl, C.W., C. Kunz, and F.X. Heinz, *Presence of poly(A) in a flavivirus: Significant differences between the 3' noncoding regions of the genomic RNAs of tick-borne encephalitis virus strains.* Journal of Virology, 1991. **65**(8): p. 4070-4077.
117. Lescar, J., et al., *Towards the design of antiviral inhibitors against flaviviruses: the case for the multifunctional NS3 protein from Dengue virus as a target.* Antiviral Res, 2008. **80**(2): p. 94-101.
118. Zou, G., et al., *Functional analysis of two cavities in flavivirus NS5 polymerase.* J Biol Chem, 2011. **286**(16): p. 14362-72.
119. Malet, H., et al., *The flavivirus polymerase as a target for drug discovery.* Antiviral Res, 2008. **80**(1): p. 23-35.
120. Barker, A., et al., *Expanding medicinal chemistry space.* Drug Discov Today, 2013. **18**(5-6): p. 298-304.
121. Kumar, A., A. Voet, and K.Y. Zhang, *Fragment based drug design: from experimental to computational approaches.* Curr Med Chem, 2012. **19**(30): p. 5128-47.
122. de Kloe, G.E., et al., *Transforming fragments into candidates: small becomes big in medicinal chemistry.* Drug Discovery Today, 2009. **14**(13-14): p. 630-646.
123. Murray, C.W. and D.C. Rees, *The rise of fragment-based drug discovery.* Nature Chemistry, 2009. **1**(3): p. 187-192.
124. Hoffer, L., J.P. Renaud, and D. Horvath, *Fragment-Based Drug Design: Computational and Experimental State of the Art.* Combinatorial Chemistry & High Throughput Screening, 2011. **14**(6): p. 500-520.
125. Erlanson, D.A., *Introduction to Fragment-Based Drug Discovery.* Fragment-Based Drug Discovery and X-Ray Crystallography, 2012. **317**: p. 1-32.
126. Congreve, M., et al., *Recent developments in fragment-based drug discovery.* Journal of Medicinal Chemistry, 2008. **51**(13): p. 3661-3680.
127. Chessari, G. and A.J. Woodhead, *From fragment to clinical candidate-a historical perspective.* Drug Discovery Today, 2009. **14**(13-14): p. 668-675.
128. Hajduk, P.J., et al., *Discovery of potent nonpeptide inhibitors of stromelysin using SAR by NMR.* Journal of the American Chemical Society, 1997. **119**(25): p. 5818-5827.
129. Kolb, P., et al., *Docking and chemoinformatic screens for new ligands and targets.* Current Opinion in Biotechnology, 2009. **20**(4): p. 429-436.
130. Chu, P.W.G. and E.G. Westaway, *Replication strategy of Kunjin Virus: Evidence for recycling role of replicative form RNA as template in semiconservative and asymmetric replication.* Virology, 1985. **140**(1): p. 68-79.
131. Cerutti, H. and J.A. Casas-Mollano, *On the origin and functions of RNA-mediated silencing: from protists to man.* Curr Genet, 2006. **50**(2): p. 81-99.
132. Schwede, T., et al., *SWISS-MODEL: An automated protein homology-modeling server.* Nucleic Acids Res, 2003. **31**(13): p. 3381-5.
133. Arnold, K., et al., *The SWISS-MODEL workspace: a web-based environment for protein structure homology modelling.* Bioinformatics, 2006. **22**(2): p. 195-201.
134. Awuni, Y. and Y.G. Mu, *Reduction of False Positives in Structure-Based Virtual Screening When Receptor Plasticity Is Considered.* Molecules, 2015. **20**(3): p. 5152-5164.
135. Daura, X., et al., *Peptide Folding: When Simulation Meets Experiment.* Angewandte Chemie International Edition, 1999. **38**(1-2): p. 236-240.
136. Alhossary, A., et al., *Fast, accurate, and reliable molecular docking with QuickVina 2.* Bioinformatics, 2015. **31**(13): p. 2214-6.

137. Shin, W.H. and C. Seok, *GalaxyDock: protein-ligand docking with flexible protein side-chains.* J Chem Inf Model, 2012. **52**(12): p. 3225-32.
138. Ravindranath, P.A., et al., *AutoDockFR: Advances in Protein-Ligand Docking with Explicitly Specified Binding Site Flexibility.* PLoS Comput Biol, 2015. **11**(12): p. e1004586.
139. Metropolis, N. and S. Ulam, *The Monte Carlo Method.* Journal of the American Statistical Association, 1949. **44**(247): p. 335-341.
140. Hetenyi, C. and D. van der Spoel, *Efficient docking of peptides to proteins without prior knowledge of the binding site.* Protein Sci, 2002. **11**(7): p. 1729-37.
141. Schrödinger. *Glide Knowledge Base*. 2016 December 2, 2016 [cited 2017 18/7/2017]; Available from: https://www.schrodinger.com/kb/599.
142. Bioinformatics and High Performance Computing (BIO-HPC) Research group. *Achilles Blind Docking Server*. 2017 [cited 2017 18/07/2017]; Achilles Blind Docking Server at the Universidad Católica San Antonio de Murcia (UCAM)]. Available from: https://bio-hpc.ucam.edu/achilles/.
143. Dhanik, A., J.S. McMurray, and L.E. Kavraki, *DINC: a new AutoDock-based protocol for docking large ligands.* BMC Struct Biol, 2013. **13 Suppl 1**: p. S11.
144. Schellhammer, I. and M. Rarey, *FlexX-Scan: fast, structure-based virtual screening.* Proteins, 2004. **57**(3): p. 504-17.
145. Lamport, L., *How to Make a Multiprocessor Computer That Correctly Executes Multiprocess Programs.* Ieee Transactions on Computers, 1979. **28**(9): p. 690-691.
146. Silberschatz, A.G., Greg; Galvin, Peter, *Operating System Concepts*. 10 ed. 2018.
147. Liu, Z., et al., *PDB-wide collection of binding data: current status of the PDBbind database.* Bioinformatics, 2015. **31**(3): p. 405-12.
148. Wang, R.X., et al., *The PDBbind database: Methodologies and updates.* Journal of Medicinal Chemistry, 2005. **48**(12): p. 4111-4119.
149. Wang, R.X., et al., *The PDBbind database: Collection of binding affinities for protein-ligand complexes with known three-dimensional structures.* Journal of Medicinal Chemistry, 2004. **47**(12): p. 2977-2980.
150. Sanner, M.F., *Python: a programming language for software integration and development.* J Mol Graph Model, 1999. **17**(1): p. 57-61.
151. *Origin*. OriginLab: Northampton MA.
152. Hassan, N.M., et al., *Protein-Ligand Blind Docking Using QuickVina-W With Inter-Process Spatio-Temporal Integration.* Scientific Reports, 2017. **7**(1): p. 15451.