PONTIFICAL CATHOLIC UNIVERSITY OF RIO GRANDE DO SUL
FACULTY OF INFORMATICS
COMPUTER SCIENCE GRADUATE PROGRAM

An Effective Method to Optimize Docking-Based Virtual Screening in a Clustered Fully-Flexible Receptor Model Deployed on Cloud Platforms

Renata De Paris

Thesis presented as partial requirement for obtaining a Ph. D. degree in Computer Science at Pontifical Catholic University of Rio Grande do Sul.

Advisor: Prof. Duncan Dubugras Alcoba Ruiz
Co-Advisor: Prof. Osmar Norberto de Souza

Porto Alegre
2017

Ficha Catalográfica

D278a De Paris, Renata

An Effective Method to Optimize Docking-Based Virtual Screening in a Clustered Fully-Flexible Receptor Model Deployed on Cloud Platforms / Renata De Paris . – 2016.
167 f.
Tese (Doutorado) – Programa de Pós-Graduação em Ciência da Computação, PUCRS.

Orientador: Prof. Dr. Duncan Dubugras Alcoba Ruiz.
Co-orientador: Prof. Dr. Osmar Norberto de Souza.

1. Scientific Workflow. 2. Cloud Computing. 3. Clustering of Molecular Dinamycs Trajectories. 4. Molecular Docking Simulations. 5. Fully-Flexible Receptor Model. I. Ruiz, Duncan Dubugras Alcoba. II. Norberto de Souza, Osmar. III. Título.

Elaborada pelo Sistema de Geração Automática de Ficha Catalográfica da PUCRS
com os dados fornecidos pelo(a) autor(a).

TERMO DE APRESENTAÇÃO DE TESE DE DOUTORADO

Tese intitulada "An Effective Method to Optimize Docking-Based Virtual Screening in a Clustered Fully-Flexible Receptor Model Deployed on Cloud Platforms" apresentada por Renata De Paris como parte dos requisitos para obtenção do grau de Doutora em Ciência da Computação, aprovada em 28 de outubro de 2016 pela Comissão Examinadora:

[illegible]

Dr. Duncan Dubugras Alcoba Ruiz — PPGCC/PUCRS
Orientador

[illegible]

Dr. Osmar Norberto de Souza — PPGCC/PUCRS
Coorientador

Tiago Ferreto

Dr. Tiago Coelho Ferreto — PPGCC/PUCRS

Laurent Emmanuel Dardenne

Dr. Laurent Emmanuel Dardenne — LNCC

[illegible]

Dr. André Carlos Ponce de Leon Ferreira de Carvalho — USP

Homologada em 09/03/2017, conforme Ata No. 02 pela Comissão Coordenadora.

[illegible]

Prof. Dr. Luiz Gustavo Leão Fernandes
Coordenador.

“It is not the strongest of the species that survive, but the one most responsive to change.”
(*Charles Robert Darwin*)

ACKNOWLEDGMENTS

First of all, many thanks to my advisor Prof. Duncan Ruiz, for his support, his guidance and encourage words gave me during these years of doctoral course. Thank you for trusting on my potential and giving me opportunities to my personal and professional growth.

Special thanks are due to my co-advisor Prof. Osmar Noberto de Souza for his patience in teaching on drug discovery, and to let me so fascinated by this research area. Thanks for turning a scary/tricky question into a simple answer.

Other special thanks are due to Prof. Paul Watson, whom, between 2015 and 2016, kindly received me in NCL Science Central, the Europe's science and technology hub from the School of Computing Science of Newcastle University, for his attention and enlightening discussions. I would like to acknowledge and thank Jacek Cala for his patience, his willingness to assist, mainly, his valuable suggestions and precious helps with e-Science Central. He was a person who played a indispensable role during the development of this study. I also thank Matthew Forshaw and Barry Hodgson to facilitate the access to the CIC Private Cloud, and Onnagh McGee to sort out problems on remote accessibility. I would also like to extend my thanks to the members of the Scalable Computing Group for their helpful discussions and suggestions .

A special thanks to my colleagues of GPIN-PUCRS whose help was essential to my growth during the research period. In particular, thanks to Christian Quevedo and Rodrigo Barros, whose their support, advices and reviews were essential for the work progress. I would also like to thank my closest friend and former GPIN's member Daiane Hemerich for her suggestions and reviews to this manuscript, as well as her emotional support.

I would also like to acknowledge the Computer Science Graduate Program at PUCRS and Hewlett Packard - PROFAC for the PhD scholarship, as well as the National Council for Scientific and Technological Development (CNPq-Brasil) for the Visitor PhD Student (Sandwich) scholarship.

Last, but not least, my most sincere thanks to those who are the basis of this study: my parents, Sérgio and Maria. Your examples of life and unconditional support in all its forms were essential to carrying out this study. Thanks to my brother Cléber, his wife Nadime, and my aunt Assunta for their encouraging words; as well as all my family, whose support and comprehension were essential for me to get to this stage of my journey. Thank you for every minute of concern, every word of support, every day missing me so far away.

A special thank to my husband Mateus for helping me survive all the stress and bad times during these four long years, and, mainly, for not letting me give up. Thank you ever so much for understanding my absence during many days: you weren't less important than my work!

AN EFFECTIVE METHOD TO OPTIMIZE DOCKING-BASED VIRTUAL SCREENING IN A CLUSTERED FULLY-FLEXIBLE RECEPTOR MODEL DEPLOYED ON CLOUD PLATFORMS

ABSTRACT

The use of conformations obtained from molecular dynamics trajectories in the molecular docking experiments is the most accurate approach to simulate the behavior of receptors and ligands in molecular environments. However, such simulations are computationally expensive and their execution may become an infeasible task due to the large number of structural information, typically considered to represent the explicit flexibility of receptors. In addition, the computational demand increases when Fully-Flexible Receptor (FFR) models are routinely applied for screening of large compounds libraries. This study presents a novel method to optimize docking-based virtual screening of FFR models by reducing the size of FFR models at docking runtime, and scaling docking workflow invocations out onto virtual machines from cloud platforms. For this purpose, we developed e-FReDock, a cloud-based scientific workflow that assists in faster high-throughput docking simulations of flexible receptors and ligands. e-FReDock is based on a free-parameter selective method to perform ensemble docking experiments with multiple ligands from a clustered FFR model. The e-FReDock input data was generated by applying six clustering methods for partitioning conformations with different features in their substrate-binding cavities, aiming at identifying groups of snapshots with favorable interactions for specific ligands at docking runtime. Experimental results show the high quality Reduced Fully-Flexible Receptor (RFFR) models achieved by e-FReDock in two distinct sets of analyses. The first analysis shows that e-FReDock is able to preserve the quality of the FFR model between 84.00% and 94.00%, while its dimensionality reduces on average 49.68%. The second analysis reports that resulting RFFR models are able to reach better docking results than those obtained from the rigid version of the FFR model in 97.00% of the ligands tested.

Keywords: Scientific Workflow, Cloud Computing, Clustering of MD Trajectories, Molecular Docking Simulations, Fully-Flexible Receptor Model.

UM MÉTODO EFETIVO PARA OTIMIZAR A TRIAGEM VIRTUAL BASEADA EM DOCAGEM DE UM MODELO DE RECEPTOR TOTALMENTE FLEXÍVEL AGRUPADO UTILIZANDO COMPUTAÇÃO EM NUVEM

RESUMO

O uso de conformações obtidas por trajetórias da dinâmica molecular nos experimentos de docagem molecular é a abordagem mais precisa para simular o comportamento de receptores e ligantes em ambientes moleculares. Entretanto, tais simulações exigem alto custo computacional e a sua completa execução pode se tornar uma tarefa impraticável devido ao vasto número de informações estruturais consideradas para representar a explícita flexibilidade de receptores. Além disso, o problema é ainda mais desafiante quando deseja-se utilizar modelos de receptores totalmente flexíveis (*Fully-Flexible Receptor* - FFR) para realizar a triagem virtual em bibliotecas de ligantes. Este estudo apresenta um método inovador para otimizar a triagem virtual baseada em docagem molecular de modelos FFR por meio da redução do número de experimentos de docagem e, da invocação escalar de workflows de docagem para máquinas virtuais de plataformas em nuvem. Para esse propósito, o workflow científico basedo em nuvem, chamado e-FReDock, foi desenvolvido para acelerar as simulações da docagem molecular em larga escala. e-FReDock é baseado em um método seletivo sem paramêtros para executar experimentos de docagem *ensemble* com múltiplos ligantes. Como dados de entrada do e-FReDock, aplicou-se seis métodos de agrupamento para particionar conformações com diferentes características estruturais no sítio de ligação da cavidade do substrato do receptor, visando identificar grupos de conformações favoráveis a interagir com específicos ligantes durante os experimentos de docagem. Os resultados mostram o elevado nível de qualidade obtido pelos modelos de receptores totalmente flexíveis reduzidos (*Reduced Fully-Flexible Receptor* - RFFR) ao final dos experimentos em dois conjuntos de análises. O primeiro mostra que e-FReDock é capaz de preservar a qualidade do modelo FFR entre 84,00% e 94,00%, enquanto a sua dimensionalidade reduz em uma média de 49,68%. O segundo relata que os modelos RFFR resultantes são capazes de melhorar os resultados de docagem molecular em 97,00% dos ligantes testados quando comparados com a versão rígida do modelo FFR.

Palavras-Chave: Workflow Científico, Computação em Nuvem, Agrupamento de Trajetórias da Dinâmica Molecular, Docagem Molecular, Modelo de Receptor Totalmente Flexível.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF ACRONYMS AND ABBREVIATIONS

3-D	Three-Dimensional
API	Application Program Interface
CIC	Cloud Innovation Centre
DAG	Directed Acyclic Graph
DCG	Directed Cyclic Graph
e-SC	e-Science Central
E-R	Entity-Relational
FEB	Free Energy of Binding
FFR	Fully-Flexible Receptor
FReMI	Flexible Receptor Middleware
HPC	High Performance Computing
IaaS	Infrastructure-as-a-Sevice
INH	Isoniazid
InhA	2-*trans*-enoil-ACP (CoA) reductase (E.C.1.3.1.9) from *Mycobacterium tuberculosis*
MD	Molecular Dynamics
MPI	Message Passing Interface
LGA	Lamarckian Genetic Algorithm
P-SaMI	Self-adaptive Multiple Instances Pattern
PaaS	Platform-as-a-Service
PDB	Protein Data Bank
OS	Operating System
QoS	Quality of Service
RDD	Rational Drug Design
RFFR	Reduced Fully-Flexible Receptor
RMSD	Root Means Square Deviation
SaaS	Software-as-a-Service
SLA	Service Level Agreement
SWfMS	Scientific Workflow Management Systems
SQD	Sum of the Quartile Differences
TB	Tuberculosis
THT	Trans-2-Hexadecenoyl-(N-Acetyl-Cysteamine)-Thioester

TCL – Triclosan

VM – Virtual Machine

wFReDoW – Web Flexible Receptor Docking Workflow

CONTENTS

1. INTRODUCTION

Demand for pharmaceutical industry in marketing newer and more efficient drugs has steadily grown with the first rough draft of human genome sequence in 2001 [GAG11]. Nevertheless, the development of new drugs is a very lengthy and time-consuming process. It also requires substantial investments in technology resources, such as computational power to store, manage, execute, and analyze simulations on protein-ligand interactions [MMM09, PMD+10]. Thus, new computational methods are needed to aid time reduction and to accurately investigate chemical and biological behaviors of ligands and receptors during the Rational Drug Design (RDD) process [Kun92, Kap08].

RDD is the process that applies multidisciplinary approaches to transform active biological compounds into suitable drugs [Kap08, MMM09]. Molecular docking simulations constitute the second step of the RDD. They identify and optimize drug candidates by analyzing and modeling molecular interactions between ligands or small molecules and target protein or receptor [Kap08]. Such simulations are performed by a docking software, such as AutoDock4.2 [MHL+09]. Each docking software has a search algorithm that generates a set of different binding modes of a protein-ligand complex, and a scoring function that can rank them, as well as estimating the best binding affinity by computing, among other values, the Free Energy of Binding (FEB).

Protein flexibility is a key issue in docking programs, and while these are capable of exploring the flexibility of ligands, the explicit treatment of the flexibility, for both protein and ligand, during their interactions, remains a challenging task [LBM+09, SRC+13]. This is due to the large number of degrees of freedom associated to the protein, which cannot be directly transferred from the docking methods used in the context of ligand flexibility [SRC+13]. Nevertheless, proteins are very versatile, and their flexibility cannot be a priori neglected since it plays an essential role in their structure and function [FLSM14, BRM15]. Buonfiglio et al. [BRM15] state that ignoring protein flexibility in docking experiments is indeed a potentially dangerous practice that most likely would result in false-negative outcomes.

To account for the dynamic behavior of proteins, we make use of an ensemble of conformations obtained from a Molecular Dynamics (MD) simulation [ABG06, CKS+08]. MD simulation is one of the most affordable and accurate methods for identifying alternative binding forms of proteins, making possible to understand from fast internal motions to slow conformational changes [ABG06]. The result of an MD simulation is a series of instant conformations of the protein along the simulation time scale. These conformations are also often called snapshots. Throughout this thesis, the term Fully-Flexible Receptor (FFR) model [MWRNdS11] is used to refer to the ensemble of snapshots that constitutes an MD trajectory. In this approach, each ligand is docked separately at each conformation of the FFR model [QDPRNdS14]. The use of FFR model is an appealing approach for improving the realism of proteins and ligands into flexible molecular environments [ABG06, EMBS14]. The major problem in using an ensemble of snapshots during docking experiments is that it becomes a limiting and costly task as the dimensionality of the FFR model increases. Several studies have attempted to deal with this virtual high-throughput screening; however, it remains a challenge in the present day [ABM08, SRC$^+$13, DPFNdSR13, QDPRNdS14, DPQRNdS15, FLSM14, ADK15, BRM15].

1.1 Problem Statement

There are certain issues associated with the use of multiple receptor MD conformations in routine of docking-based virtual screening. One of the main reasons behind this difficulty is the required time to perform virtual screening in an FFR model, which can have hundreds of thousands up to millions of conformations, against a database of small molecules. There is currently a large number of small molecules that have emerged in databases such as ZINC [ISM$^+$12], PubChem [WXS$^+$09] and GDB-17 [RvDBR12]. Another reason is that information provided by multiple protein conformations is often unworkable and, thus, a proper treatment must be done in order to exploit it effectively [BRM15]. Several attempts have been made to answer the following question:

How can the computational efficiency and prediction accuracy of molecular docking simulations, based on FFR models, be balanced to perform practical virtual screening on a large library of ligands?

Previous studies have reported efforts in strategically selecting a small number of MD conformations before starting the docking experiments [TSP$^+$14, DPQR$^+$15]. Landon et al. [LAB$^+$08] represented 90% of the conformational flexibility within ligand-biding site of a 160 ns MD trajectory of the H5N1 into reduced ensembles of 10 apo and 5 holo structures by using clustering algorithms. A more detailed analysis on finding small ensembles of representative MD conformations using different clustering algorithms was done by Torda and van Gunsteren [TvG94] and Shao et al. [STTC07]. To generate groups of representative MD conformations, they used pairwise Root Means Square Deviation (RMSD), the well-known measure of similarity for clustering MD trajectories. Our research group has also performed advances in this field, with a new measure of similarity introduced for clustering MD trajectories [DPQRNdS15]. We figured out that by making use of pairwise RMSD for all or part of the MD structures is not the most appropriate gauge to cluster conformations when the target protein has a plastic active site, since they are heavily influenced by changes that occur in parts of the structures [DPQRNdS15].

Recently, studies have shown an increased interest in reducing the number of MD structures during docking experiments [MSR$^+$11, DPFNdSR13, QDPRNdS14, HRFNdS15]. However, it raises another question:

How to select an accurate representation of the receptor and to generate a Reduced Fully-Flexible Receptor (RFFR) model that best complements the shape of a specific ligand?

Machado et al [MSR$^+$11] presented an approach to answer this question. They developed FReDoWs, a scientific workflow that automates docking experiments and performs selective molecular docking simulations by identifying the best FEB, which is achieved from exhaustive docking experiments between the FFR model and a ligand with similar structure. Despite the appropriate accuracy showed in their results, FReDoWs may become time-consuming as the number of dissimilar small molecules and receptor snapshots increase. Another way to answer the second question is to make use of the web Flexible Receptor Docking Workflow (wFReDoW) [DPFNdSR13]. wFReDoW was created in our research group and uses worker nodes from the Amazon Elastic Compute Cloud to reduce both the dimensionality of FFR models and the overall docking execution time by using a strategically approach to identify promising snapshots. De Paris et al. [DPFNdSR13] reported that wFReDoW was able to reduce from days to hours the overall time execution, maintaining over 95% of accuracy in a 3.1 ns MD trajectory of InhA. Nevertheless, wFReDoW has some limitations such as: (i) its input parameters should have accurate information by the expert domain on best and worst interaction energies produced by a FFR model and a ligand, and (ii) its deployment model does not allow to improve performance as the worker nodes increases since it is based on a Message Parsing Interface (MPI) cluster model.

1.1.1 Health and Socio-Economic Motivations

The FFR model employed in this study was generated from an MD simulation of the 2-*trans*-enoil-ACP (CoA) reductase (E.C.1.3.1.9) enzyme or InhA-NADH complex from *Mycobacterium tuberculosis* [DQB$^+$95]. InhA is part of the fatty acid biosynthesis system type II (FASII), and plays a role in the synthesis of mycolic acids, which are key components of the *Mycobacterium tuberculosis* cell wall. Inhibition of InhA by the drug isoniazid, for instance, kills the bacteria [DQB$^+$95]. The InhA enzyme is one of the best established and validated target for the development of anti-tuberculosis (anti-TB) agents [RVS$^+$99, HSN$^+$12]. Tuberculosis is a top killer infectious disease that affects people worldwide, particularly those living in low-and middle-income countries. TB decreases their capacity to work, adds treatment expenses, and exacerbates their poverty. It is estimated that TB alone causes near $ 12 billion to disappear from the global economy annually [Org15]. The most effective anti-TB drugs currently available for the treatment of tuberculosis are isoniazid [VWA$^+$06] and rifampicin [TIM$^+$93]. Although these drugs have been used since their discovery in 1952, studies indicate the growth of drug-resistant TB cases in the last decades [Org15]. According to the World Health Organization [Org15], 9.6 million people were diagnosed with the disease and 1.5 million died in 2014, and 480,000 people developed multidrug-resistant TB. In the face of these facts, new anti-TB drugs capable of providing a more efficient treatment of TB infection, as well as of drug-resistant TB, must be intensely investigated. Therefore, future research should attempt to find more powerful and selective inhibitors of the InhA enzyme.

1.2 Thesis Contributions

The main contribution of this thesis is a new and effective method to reduce the time for docking-based virtual screening of FFR models. This is achieved by discarding groups of unpromising snapshots for specific ligands, and scale docking experiments out onto cloud virtual machines. For this purpose, the scientific workflow called e-FReDock was built and deployed into cloud platforms for performing selective ensemble docking experiments.

As a result, we expect that the method developed will:

- Reduce the dimensionality of FFR models without losing biologically relevant information;
- Accelerate ensemble docking experiments by scaling receptor-ligand interactions out onto cloud Virtual Machines (VMs).

As a consequence, resulting RFFR models obtained by discarding non-promising snapshots from the original model, can be accurately shaped for a greater number of ligands, and the total time spent in the ensemble docking experiments can be considerably reduced.

1.2.1 Specific Contributions

The specific contributions of this thesis are:

- A new approach for clustering MD trajectories. We develop and validate a new approach for clustering MD trajectories using features from the substrate-binding cavity as similarity function. We show in Chapter 3 a detailed study on different clustering methods and similarity metrics (e.g. protein RMSD, cavity RMSD and substrate-binding cavity) for clustering MD trajectories with the aim of identifying an optimal clustering solution to be used as input to e-FReDock.
- A new method for optimizing docking-based virtual screening of FFR models based on the Self-adaptive Multiple Instances Pattern (P-SaMI) [HRFNdS15]. This method, which is presented in Chapter 3, uses the clustered FFR model to strategically discard

groups of unpromising snapshots of specific ligands at docking runtime. The output is a new RFFR model tailored for each ligand tested. One of the main advantages of this method is its ability of identifying batches with (un)promising snapshots and prioritizing them by using a set of metrics computed from docking results of processed snapshots, rather than using the original midpoint average between the worst and best FEB values, provided by a domain expert as proposed by [HRFNdS15].

- The e-FReDock cloud-based scientific workflow. This workflow manages data and service required to evaluate and discard unpromising snapshots at docking runtime. We show in Chapter 5 that e-FReDock has a set of components to execute docking experiments of FFR models, scale workflows out onto cloud VMs, and discards snapshots based on the best binding free energies for the ligand and snapshots already processed. This workflow was designed to run molecular docking simulations of FFR models based on the method proposed in Chapter 4.
- The use of MongoDB database model to store and manage e-FReDock data. We present in Chapter 5, Section 5.3.2, the Entity-Relational (E-R) based conceptual model to store control system and input/output docking data.
- The use of public and private cloud platforms to deploy e-FReDock. We present in Chapter 6, Section 6.2, the performance evaluation of e-FReDock on both, public and private cloud Virtual Machines (VMs), in order to select the most cost-effective cloud instance setting.

1.3 Thesis Overview

The remainder of this thesis is organized as follows:

- Chapter 2 introduces the major biological and computational concepts necessary for a better comprehension of this study. It starts with an overview of the molecular docking experiments in RDD process and the approaches currently used to consider the explicit flexibility of receptors, focusing on ensemble docking experiments, which is the approach used in this study. The FFR model used to conduct all experiments in this thesis is also described in this section. The last two sections describe the basic aspects of scientific workflows and cloud computing paradigm.
- Chapter 3 describes the methods used for clustering the snapshots of the FFR models and identifies an optimal solution to be used as input to e-FReDock. It also gives a comparison of the gains obtained by using the substrate-binding cavity features for clustering the MD trajectory and metrics of similarities.

- Chapter 4 specifies the method proposed to reduce the dimensionality of FFR models and points out a suitable parametrization for performing a set of empirical experiments.
- Chapter 5 presents the conceptual architecture designed for e-FReDock scientific workflow. It includes the representation of this architecture and the detailed specification of every block developed on e-Science Central (e-SC) [HWWC13], the workflow enactment system used to design e-FReDock. The ER based conceptual model to store docking data into NoSQL MongoDB database [Cho13] is also shown.
- Chapter 6 presents the results of this thesis. Two sets of experiments are reported in this section by using e-FReDock. The first set evaluates the performance of e-FReDock when it is executed in VMs from private and public cloud platforms. A small set of snapshots from the FFR model is used for performing molecular docking simulations with a small ligand to define a baseline cost in using Azure Dv2-series VMs [AZU16]. The second set of experiments reports the selective ensemble docking experiments performed by the e-FReDock deployed on cloud VMs using a set of 103 ligands, 14 from PDB [BWF$^+$00] and 89 from ZINC [ISM$^+$12] databases. In this set of experiments, two analyses are presented in order to assess the quality of results achieved from e-FReDock. First, the best FEB values from the RFFR model and the FFR model are compared for a set of 16 ligands. Second, the best FEB values obtained from the RFFR models and the rigid version of the FFR model are compared for all ligands tested.
- Chapter 7 reports related works. These works are associated with the approaches used to optimize docking-based virtual screening of FFR models.
- Chapter 8 summarizes the conclusions of this study which have led to this thesis. Additionally, it draws the limitations, gives directions for future works, and describes published papers.

2. BACKGROUND

This chapter introduces the major bioinformatics and computational concepts necessary for a better comprehension of this thesis. We starts with an overview of the Rational Drug Design (RDD) process focusing on molecular docking simulations, which consist the second step of RDD. After, we describe the most well-known alternative docking approaches accounting for receptor flexibility, including the approach adopted in this study, the ensemble docking, and the Fully-Flexible Receptor (FFR) model are the basis for all experiments described in this Thesis. This chapter closes presenting the main design options of scientific workflows and cloud computing used for developing the cloud-based environment, in order to optimize molecular docking simulations of fully flexible receptors.

2.1 The Rational Drug Design Process

Rational Drug Design [Kun92] is the technique used to transform biologically active compounds into suitable drugs, aiming to prevent and treat diseases. According to [MMM09], a promising practice to achieve the full benefit of this process is by properly combining the study and application of multidisciplinary approaches, such as computer science, mathematics, and computational chemistry. RDD is comprised of four steps [Kun92]:

1. Identification of the macromolecule of pharmacological importance or target receptor (protein, DNA, RNA or others). After identifying the disease and isolating a biological target, the domain expert performs analyses in the three-dimensional (3D) structure of the target receptor. The potential binding sites in the target receptor may be detected by computational analysis.

2. *In silico* experiments. A set of ligands is selected based on the potential binding sites found in the target receptor (step1). Such ligands are usually found in databases of small molecules like ZINC [ISM$^+$12]. In this process, different conformations and orientations assumed by a ligand in order to fit into a receptor binding site are simulated by a molecular docking software, such as AutoDock [MHL$^+$09].

3. Experimental tests. Ligands that successfully bind to the receptor binding site and are able to inhibit or enhance the pharmacological activities are synthesized and tested.

4. New drug evaluation. Based on the experimental results, the new drug is manufactured or the RDD process returns to step 1. In the last case, changes are made in the ligand search protocol.

The three important factors in this process are speed, cost, and quality. Kapetanovic [Kap08] states that computational and experimental techniques have important roles in drug discovery and development, and represent complementary approaches. Previous studies have indicated that a careful quality control on the early stages tends to increase the effectiveness of clinical trials [Kap08, PMD$^+$10, SBBW12]. Applying computational techniques in RDD process helps to minimize time and improve biological testing, while increasing predictability. Therefore, this study focuses on significantly contributing for improvement of the most relevant and costly step of the RDD process, the molecular docking simulations.

2.2 Molecular Docking Simulations

Docking techniques predict the correct conformation of a small-molecule ligand and its receptor by examining and modeling molecular interactions between them [ABG06, Kap08]. Typically, docking experiments aims to find lead compounds by screening a large database of small molecules with the intention of identifying favorable receptor-ligand interactions [WWF$^+$04]. Structure-based virtual screening, which aims at prioritizing small-molecule candidates by their affinity to the binding site based on 3D structures of known protein targets, is a well-established and widely used technique to perform high-throughput receptor-ligand docking.

Molecular docking simulations generate hundreds or even thousands possible poses that a ligand may fit within the protein binding site by using a docking software, such as AutoDock [MHL$^+$09], DOCK [LBM$^+$09], GOLD [JWG$^+$97] or Surflex-Dock [Jai07]. Over the past decades, considerable advances in computational power and the fast-growing number of novel drug candidates placed in libraries of small compounds have steadily increased the number of docking programs. All docking software generate and evaluate different ligands poses based on two basic methods: a search algorithm and a scoring function. The search algorithm explores the degrees of freedom of a protein-ligand complex by generating satisfactory enough samples of binding affinity predictions. The scoring function estimates the energy binding for a given binding mode in order to identify accurately the best receptor-ligand conformation [SRC$^+$13, GdMD14]. The Free Energy of Binding (FEB) is a commonly-used measure in scoring function. It estimates the energy for both the bound and unbound receptor-ligand states from a force field that allows incorporation of intermolecular energies [MHL$^+$09]. The Root Mean Square Deviation (RMSD) is also a metric used to evaluate docking results. RMSD indicates the level of pose correctness between a docking result and a molecule's previously recorded conformation [KDFB04]. This metric calculates the average distance between atoms of the docked ligand and the reference ligand, which is determined from a crystallographic structure or by domain expert assessments. An promising receptor-ligand interaction is usually identified when the docking software predicts the most negative

FEB values (kcal/mol) and an RMSD value (Å) close to zero if a reference ligand is used. Figure 2.1 illustrates the best TCL400 ligand pose after a molecular docking simulation and its reference ligand, extracted from the crystal structure of the InhA-NADH complex from *Mycobacterium tuberculosis* (PDB: 1P45) [KMA+03].

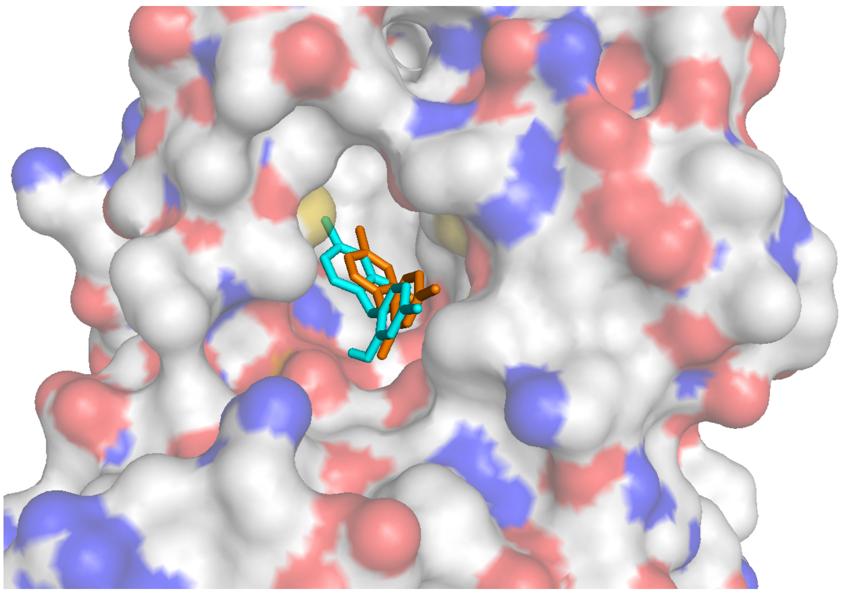


Figure 2.1 – Example of molecular docking process. The fragment of the protein binding site of the InhA structure (PDB:1P45) is depicted using a surface representation and atom type colors (carbon and hydrogen: light gray; nitrogen: blue; oxygen: red; sulfur: yellow). TCL ligand (TCL400) in its crystallographic reference and final docking positions are represented by sticks in orange and blue, respectively. Figure produced with PyMol package software [DeL16].

Despite having different purposes and requirements, most docking methods are capable of considering the flexibility of ligands accurately [JWG+97, Jai07, MHL+09, LBM+09]. However, the well-known drawback of docking programs is their lack of sensitivity to recognize slight conformational and structural changes in the target protein. The incorporation of protein flexibility in docking-based virtual screening is still a daunting task [ABM08, SRC+13, FLSM14, ADK15, BRM15]. The major challenge behind this difficulty is the high number of conformation changes that have to be considered to represent the explicit plasticity or flexibility during the ligands and receptors interactions [TK03, ABG06, Won08]. To obtain more realistic binding energies for full target flexibility, docking programs should accurately predict essential modes of backbone motion and improves scoring to enhance selectiveness [BZ10].

Buonfiglio et al. [BRM15] suggest that ignoring protein flexibility in virtual screening is indeed a potentially dangerous practice that most likely would result in false-negative outcomes as the protein conformation required for binding is missing. Conversely, the wide number of information required to account for at least part of the protein flexibility is a limiting factor due to computational cost, which increases as the level of accuracy rises.

2.3 Molecular Docking Approaches Accounting for Receptor Flexibility

According to Cozzini et al. [CKS$^+$08], proteins are inherently flexible systems and have an intrinsic ability to undergo functionally relevant conformation transitions under native state conditions on a wide range of scales, both in time and space. A number of different approaches, with their advantages and disadvantages, are currently been applied by various authors to treat the partial or total flexibility of receptor during docking procedures. Some approaches are classified by Teodoro et al. [TK03] into five main categories: soft docking [FRTA02], side-chain flexibility [Lea94], molecular relaxation [ACBI09], collective degrees of freedom [Gar92, ZS99] and ensemble docking [CKS$^+$08, TA08].

Soft docking is an implicit way to consider partial protein flexibility, where the ligand "penetrates" the protein surface to identify small and centralized changes that occur in flexible environment. Although speed and low cost are the main advantages, it is also considered the simplest method as the changes in protein conformation are minimal and, therefore, the level of false positives is increased [ABG06, ADK15].

The side-chain flexibility approach addresses conformational changes by selecting residues that are within the protein active site during the docking experiment or after, by considering the final ligand pose [ABG06]. This approach keeps the protein backbone fixed and explores side chain variations in the active site of the protein, based on a rotamer library. The rotamer library contains discrete predetermined conformations of the ligand and side chains that help the docking algorithm to predict low-energy conformations [Lea94]. This approach is useful when the structural and functional information of the receptor is known in advance. However, it may produce worse results than using a single structure due to the restricted flexibility between the ligand and protein side chains [ADK15].

Molecular relaxation process explores the flexibility of both ligand and receptor, by relaxing the protein backbone and side chain atoms nearby, using a rigid-body docking [HZ10]. Its first procedure is to place the ligand orientations/conformations in the rigid-body docking, and afterward the energy minimization of the formed protein-ligand complexes is done by using Monte Carlo methods or MD [HZ10, ADK15]. The advantage of the molecular relation is the capacity of finding novel conformations due to the inclusion of backbone flexibility and side-chain changes [HZ10]. However, the computational cost required by the scoring function to avoid improper backbone torsions and side chain atoms is considered one of the main limiting task to the relaxation process.

The collective degrees of freedom approach employs different methods to consider the full protein flexibility, such as the normal modes for the receptor analysis [ZS99] and the principal component analysis [Gar92]. Overall, these techniques support the hypothesis that rather using the original dimensional representation of the protein-ligand complex, a new lower-dimensional representation of this complex is generated by considering only the most significant modes of the protein's motion [TK03, ADK15]. The main advantages are low computational cost and high level of flexibility considered in the receptor-ligand complex. However, collective degrees of freedom approach have a well-known lack of accuracy in the results, as it attempts to account for moves identified during motion modes instead of native degrees of freedom of the protein [TK03].

The ensemble docking approach is the method used for docking a set of ligands into all MD conformations of the receptor [CKS+08]. An ensemble contains a collection of related but diverse crystal structures, obtained from experimental techniques, such as X-ray crystallography [BM05] and nuclear magnetic resonance spectroscopy [DC07]. Alternatively, it can be produced through MD simulations [ABG06, NBIM11], Monte Carlo simulations [RTA12] and normal mode analysis [SMP+10]. The employment of many receptor structures or snapshots derived from MD simulations is a well-establish method for representing the natural moves of atoms in molecular systems based on structural and physical aspects of biological macromolecules [CKS+08, NBIM11]. Ensemble docking is currently one of the best methods applied to achieve a broad range of all possible receptor conformations due to its level of exactness to recognize dynamic behaviors of proteins at different time-scales [ABG06]. Such precision is of fundamental importance to detect internal moves, conformational changes and even the correct folding of many proteins, which would be impossible to identify through experimental techniques [KM02, CKS+08]. However, the main drawback of this approach is the computational power required to deal with the size of the ensemble, which holds above 10^4 MD conformations in order to better explore the binding process [KOB+12].

Despite the pros and cons of each docking method accounting for protein flexibility, ensemble docking, particularly employing MD simulations, is considered the most appropriate and accessible method to produce a significant amount of protein conformations at reasonable cost [ABG06, NBIM11, KOB+12]. Korb et al. [KMC11] developed the Ensemble Performance Index (EPI) to evaluate the quality of the ensemble structures in eight different targets, concluding that some targets outperformed, and others improved the average of the rigid-protein docking results. Similarly, Nichols et al. [NBIM11] demonstrated great improvements of MD receptor conformations in virtual screening results compared with X-ray structures. Even though these studies show clear advantages, they highlight the need for developing ensemble selection protocols in order to select better MD structures. This may prevent performing a large number of poorly-docking conformations over a variety of ligands, saving considerable time in the RDD process.

2.4 Docking-Based Virtual Screening of MD Receptor Conformations

According to Alonso et al. [ABG06], an ensemble of receptor conformations increases the chances of finding a receptor in its right conformational state to accommodate a particular ligand. This study models the explicit flexibility of a receptor by using an ensemble of conformations or snapshots derived from its MD simulation. An ensemble of different protein structures is generated according to the measure of nuclear relation rate, which may vary from fastest ($<$ ns) until lowest internal moves (μs to ms), by using an MD software, such as AMBER 12 [CDCI+16] or GROMACS 4 [HKVDSL08].

As mentioned in section 2.3 , molecular docking simulations of MD receptor conformations is one of the most affordable and accurate methods to identify better conformational space of a protein-ligand complex. However, performing molecular docking simulations of the entire ensemble based on practical virtual screening of large libraries of small molecules, such as ZINC [ISM+12], PubChem [WXS+09] and DrugBank [KLJ+11], may become a computationally prohibitive task [ABM08, ADK15]. In these experiments, unlike traditional rigid-protein docking which considers only one protein structure [EB14, KBT+14], a molecular docking simulation is performed and analyzed for each MD receptor conformation against a set of different ligands [LPSM03]. In most practical cases, an ensemble of MD receptor structures can vary from hundreds to thousands or millions of conformations. Figure 2.2 illustrates a docking procedure of an ensemble of receptor structures generated from an MD trajectory, which is hereby referred to as Fully-Flexible Receptor (FFR) model [MWRNdS11].

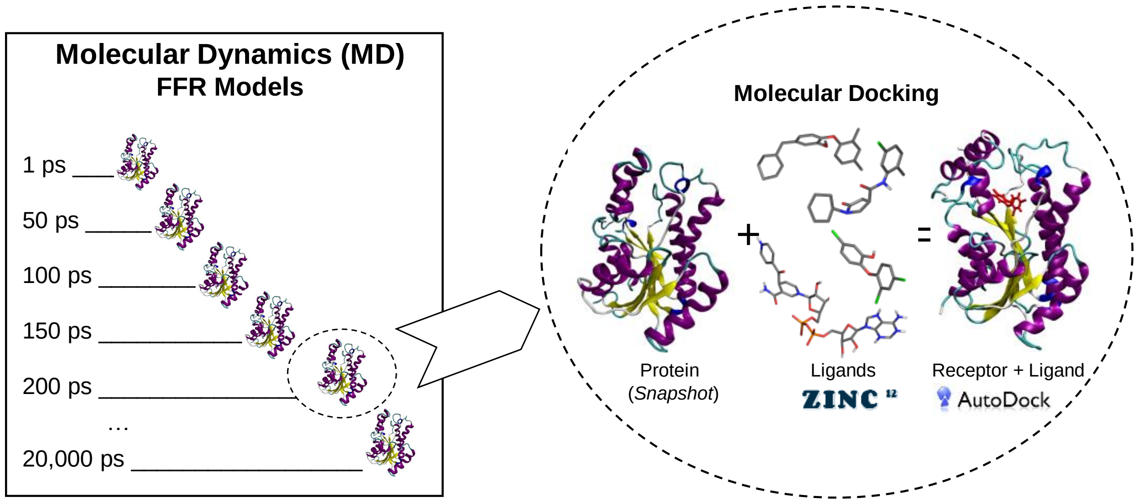


Figure 2.2 – Representation of an ensemble docking procedure by using an MD receptor conformation. The structures in the rectangle are the FFR model, representing the different conformational states of the protein generated from a 20 ns MD trajectory [Gar09]. The projection on the right hand illustrates the docking experiments performed for each snapshot against different ligands. Adapted from [QDPRNdS14].

To exemplify the complexity of the problem, let us estimate the time taken to perform a complete molecular docking simulation between the FFR model shown in Figure 2.2, which holds 20,000 snapshots and a small molecule, triclosan ligand (TCL from PDB ID: 2B35) [STB$^+$06] with 2 rotational bonds. If the time taken to execute 25 runs of LGA and 300,000 energy evaluations between one snapshots of the FFR model and the TCL ligand, in a i7 CPU with 12 GB RAM, is approximately 1 minute, it is expected that the overall time needed to sequentially execute the whole FFR model will be approximately 15 days. This time becomes even larger when the same model is used to screen libraries of small molecules. In such circumstances, it is paramount to seek for new and promising computational techniques able to enhance the docking performance and simplify the dimensionality of FFR models without losing the quality of the resulting models. Hence, it is expected to produce new Reduced Fully-Flexible Receptor (RFFR) models [DPFNdSR13] tailored for specific ligands by reducing or eliminating unnecessary biological information from the original FFR model and different ligands during docking experiments.

2.4.1 The FFR Model: InhA from *Mycobacterium tuberculosis*

The FFR model employed in this study was generated from the crystal structure of the InhA-NADH complex from *Mycobacterium tuberculosis* (PDB ID: 1ENY) [DQB$^+$95]. The structure contains the InhA enzyme with 269 amino acids residues, the NADH coenzyme, and 41 crystal water molecules. Figure 2.3 illustrates the 3D structure of the InhA enzyme backbone with NADH bound in the active site. The volume over the NADH, delimited by loops A, B, and the substrate-binding loop, is the substrate-binding cavity. The MD simulation was generated with the SANDER module from Amber9 suite of programs [CDCI$^+$16] using the ff99SB force field [HAO$^+$06] by Gargano [Gar09]. Data were saved at every 1 ps over the 20 ns simulation, yielding a total of 20,000 instantaneous receptor conformations. These 20,000 MD conformations constitutes the set of snapshots that form the FFR model of InhA, which is used to conduct all docking experiments performed in this thesis. Further details on the MD simulations preparation and execution can be found in [Gar09].

2.5 Scientific Workflows for Science

A workflow is a well-defined pattern or systematic organization of activities designed to perform certain transformations on data [Tal13]. According to Workflow Management Coalition [Hol95], it is the automation of a business process, in whole or in part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules. Even though this definition was originally created to

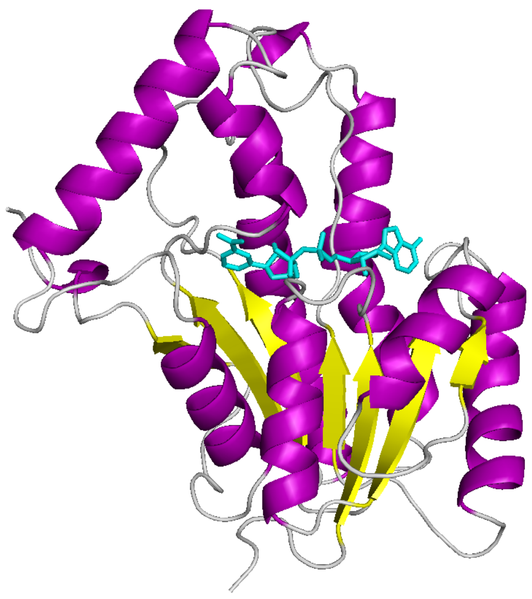


Figure 2.3 – 3D structure of the InhA-NADH complex (PDB ID: 1ENY). Ribbons representation of subunit C: α-helices in magenta, β-sheet in yellow and loops in gray. Stick representation of the NADH coenzyme (cyan) fitted in the protein binding site. Image generated with PyMol [DeL16].

define business workflows, it is also used to define scientific workflows. The term scientific workflow has been later assigned to scientists laboratories in order to manipulate large-scale and complex e-science research procedures. In this scenario, the repetitive cycle of moving data to worker nodes for analysis or simulation, launching the computations and managing the output results are automated by a Scientific Workflow Management Systems (SWfMS) [DGST09]. SWfMS are used to facilitate, expedite and streamline the management of high-throughput scientific experiments. Currently, some SWfMS are used intensively in the fields of biology, astronomy and computational engineering (Taverna [OAF$^+$04], Kepler [ABJ$^+$04], Pegasus [DSS$^+$05], e-Science Central [HWWC13], SciCumulus [DOOBM10] and others).

A scientific workflow is the assembly of complex sets of tasks or activities linked by a data flow with the intention of providing a service or generating a result [DGST09, Tal13]. Workflow tasks can be combined in a range of different scenarios which are defined taking into account the definition of control and data flow dependencies among tasks. To obtain an expected result, an execution sequence or orchestration of these tasks, which may contain one or more inputs, is defined to provide a workflow template. Moreover, a workflow template is capable of generating undetermined number of workflow instances to solve a number of problems. An important benefit of workflows is that, once defined, they can be stored and retrieved for modifications and re-executed in different scenarios by local computing resources or distributed environments [Tal13, LPVM15].

Aalst et al. [vDATHKB03] present a set of 20 workflow patterns. This study helps users to address the needs of a specific flow of execution control, such as loops, join, merge, parallel split, synchronization, multi-choice and other variations. Most of the scientific workflows are represented as Direct Acyclic Graph (DAG) (Figure 2.4(a)) and Direct Cyclic Graph (DCG) (Figure 2.4(b)), which can be performed as sequence, choice, or parallelism control patterns. The main difference is that the DCG-based workflow includes the iteration pattern control, also known as loop or cycle, and allows the workflow tasks to be repeated with a *whiledo* construct [YB05, LPVM15]. While DAG-based workflow is currently available in all SWfMSs, the DCG representation is supported by a smaller number of SWfMSs, such as Swift [ZHC+07], Kepler [ABJ+04], Triana [TSWH07] and Askalon [FPD+07].

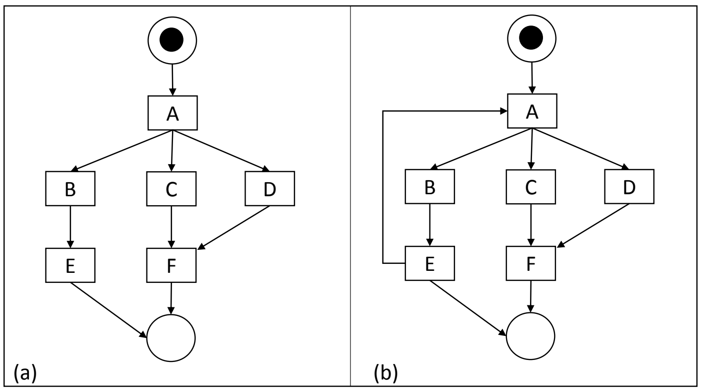


Figure 2.4 – Types of workflow structures. In (a) the DAG and in (b) the DCG, where the loop is represented by the edge from task E to A. Boxes represent tasks to be executed, while edges represent dependencies among tasks and arrows the data flow.

2.5.1 Phases of the Scientific Workflow Life Cycle

According to Deelman et al. [DGST09], the goal of SWfMSs is to provide a specialized programming environment to simplify the development effort required by scientists to orchestrate a computational science experiment. It is important to consider the four main phases of the workflow life cycle when developing scientific workflows using a SWfMS: composition, mapping, execution, and provenance [GSK+11, DGST09]. This life cycle involves the definition of the stage transitions for the workflow template from creation to completion. Although scientific workflows use the same classification areas initially purposed for business workflow, some modifications have been performed in the life cycle aspects to reflect the needs of scientist in developing simulations and experiments. The different life cycle concepts of conventional and scientific workflows were compared by Görlach et al. [GSK+11]. They state that scientific workflows centralize the life cycle counterpart only on the scientist, whereas business workflows are modeled by specialists that undertake the acknowledgment of specific life cycle parts. For instance, an IT specialist deploys the workflow, a client or employee enacts it and a business analyst analyses the outputs.

In another major study, Deelman et al. [DGST09] proposed a generalized taxonomy based on a feature set of various SWfMSs for supporting scientists and developers to make an appropriate decision about the suitability of scientific workflows. To generate the feature set, they also present a detailed assessment of the following four main phases of the scientific workflow life cycle [DGST09]:

1. Composition: in this phase an abstraction of the workflow is defined by the functionality of each task, their dependencies and data flow, taking into account the experiment requirements. The workflow specification is usually developed in either textual, and graphical workflow editing or mechanism-based semantic models.

2. Mapping: it is the transformation of the abstract composition to a concrete workflow or workflow instance, which consists of generating an executable workflow based on input data and concrete codes for the workflow activities. Moreover, the mapping by which a workflow or sub-workflow is invoked to be executed on an engine is also specified. This mapping is performed by the user, who directly selects the appropriate resources, and by the workflow system. In the latter case, users are able to specify local computing resources or a High Performance Computing (HPC) environment to execute the workflow instances. Besides, the resources are predefined by the user and the SWfMS usually chooses the location/resource where the workflow will be executed [HWWC13].

3. Execution: this phase is designed to define how the workflow will be executed, choosing the execution engine or enactment subsystem. SWfMS uses one or more workflow execution models to illustrate the different scenarios currently available. In some cases, only one system is responsible for scheduling, recovering, and reporting the set of programs submitted, such as DAGMan (Directed Acyclic Graph Manager) for Pegasus[DSS+05] and Condor [LLM88]. Alternatively, a set of services can be applied on a SWfMS, where each system has its own features, such as (a) REST API to build workflow instances directly on the cloud platform [HWWC13]; (b) models of computation from Kepler to execute workflow in different ways depending on which workflow director [ABJ+04]; and (c) using a service develop tool to support a number of different programming languages [HWWC13]. The tolerance fault is also specified in this phase with the intention of saving a state of execution and resuming after a failure.

4. Provenance: this phase defines how the workflow's operations will be analyzed by the domain specialist during or after the workflow execution. A variety of information is recorded at workflow runtime, including environments variables, process monitoring information, system environment information, tasks runtime, the hosts where tasks were executed, among others [LAB+09]. SWfMS also provides output data visualization, workflow evolution, and reproducibility. A more detailed review on provenance for scientific workflows can be found in [SPG05, DF08].

The process of running a scientific workflow generally takes long time to complete. As described in Section 2.4, a typical molecular docking simulation between 20,000 snapshots of the FFR model and a single ligand takes approximately 15 days, which may vary according to the ligand structure. Moreover, it becomes larger when a set of different ligands from libraries of small molecules is used. One of the most widely SWfMS execution models used today to speed up the scientific experiments is distributing the workflow executions on local parallel computers and HPC environments, such as Grids [YB05], computing clusters or Cloud computing [HWWC13]. In this process, one or more workflow instances are executed in parallel on an engine, which provides mechanisms to cope with workflow template, input data and instance adaptations at runtime. An SWfMS is able to invoke a number of workflow instances and orchestrate them on a set of engines that are located in HPC environments with the intention of distributing several concurrent activity executions and enhancing the computing time performance of scientific experiments [OBDO+14].

One of the most attractive HPC environment for scaling up scientific workflow instances is Cloud computing. Clouds are being widely used as a platform to traditional Grid and Cluster environments, since clouds provide a flexible on demand computing infrastructure for running large-scale scientific applications on parallel or distributed virtual machines [HMF+08]. Previous studies have been demonstrated the performance improvements by scaling a considerable number of worker cloud nodes to run scientific workflows of drug discovery experiments [CHWW13, OBDO+14]. Next section provides an explanation on cloud computing and its main platforms.

2.6 Cloud Computing

Cloud computing is a new and promising paradigm that has recently emerged to provide on-demand services over the Internet [BYV+09, ZCB10]. It is a well-established computing technology to make the resource provisioning ease and to provide scalability for performing large-scale experiments [ZCB10, KFJ14]. In a cloud computing environment, a number of services, such as applications, hardware and software, can be leased and released by users, which in turn pay for the computing resources allocated and used.

Currently, there is no consensus on one-size-fits-all cloud computing definition [VRMCL08]. Drawing on an extensive range of sources, the authors set out the different fields in which the cloud services are provisioned, taking into account how the facilities are delivered over the Internet by cloud providers, such as Amazon [AMA16], Microsoft Azure [AZU16], and Google [GOO16]. In this study, some well-known cloud computing definitions created to characterize different fields are outlined below.

According to NIST (National Institute of Standards and Technology) [MG11], cloud computing is a model for enabling on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Buyya et al. [BYV^{+}09] stated that a cloud is a type of parallel and distributed system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers.

A systematic literature review of cloud computing concepts was reported by Vaquero et al. [VRMCL08]. Although they compared the relationships and distinctions between grid and cloud approaches, their focus was on highlighting the major features of clouds that differ them from grids, based on a set of 22 definitions extracted from previous works. The authors concluded that virtualization and usability are the most important properties of clouds due to the ease of resource access. It means that resources can be configured as many times as necessary to adjust to a variable load (scale) in which Quality of Service (QoS) are ensured by the cloud providers through customized Service Level Agreement (SLA).

Once again, there is no consensus on the most suitable definition for cloud computing. However, some aspects converge to the same goal, such as cloud providers generally offer accessibility and support for scaling a large number of virtual machines (VMs) and a quick upload and download of resources to every location at anytime [AFG^{+}10]. Moreover, the aforementioned authors also highlight the following basic benefits:

- Reduced cost: the wide range of web-based services are large enough to provide heterogeneous resources on a pay as you go basis, which previously required tremendous hardware and software investments and professional skills [AFG^{+}10].
- Flexibility: the model allows users to dynamically scale hardware and software without concerns on limited resources, given that the storage capacity offered by cloud vendors is virtually endless.
- Easy access and level of abstraction: the processes of up and down load of resources happen in an entirely transparent manner, where technical configuration and deployment details are not viewed by the user.

The services provided by public cloud are sold as Utility Computing [ZCB10]. Utility computing is a computing business model in which the provider operates and manages its computing infrastructure and resources. Despite having other conceptions on grids and clusters, utility computing evolved its services to address public cloud infrastructure [BYV^{+}09]. Hence, public clouds are available on a pay as you go basis through a utility computing service, which, among other aspects, ensure an extensive capacity of storage and computing resources available on demand, fault tolerance control, image and data backups and the ability to pay for use of computational resources when you need them [AFG+10].

Different cloud computing aspects can be found in the literature, such as features [ZCB10, MG11], service and deployment models [MG11], architecture [ZCB10], technology [ZCB10], performance [BYV+09] and others. This section focuses on the essential features, the most commonly used web service models and the four well-known deployment models specified by the NIST [MG11]. Moreover, it presents the four layers of cloud computing architecture depicted by Zhang et al. [ZCB10], as well as their design principles.

2.6.1 Features

Essential features characterize the cloud computing model and distinguish the recent model from other computational paradigms (e.g. cluster and grids). NIST [MG11] points out the following five essential features for cloud computing:

1. On-demand self-service: a user is able to rent unilaterally computing resources needed, avoiding human interaction with the provider. This means that hardware and software resources can be dynamically assigned and reassigned according to the execution demand in a transparent manner.

2. Resource pooling: cloud vendors own a pool of different virtual and physical resources, such as storage, processing, memory, and network bandwidth, which are located in different sites. The pool is available to serve multiple consumers who specify the resource location at a high level of abstraction, normally country, region or data center, and the provider undertakes the knowledge over the exact location to assign the requested resource. The resources are visualized in a common pool, from which computer hardware are shared amongst the users and applications can be switched from one physical resource to another [KFJ14].

3. Rapid elasticity: the resource available can be released when no longer useful and often appear to be unlimited, thereby rewarding conservation to rapidly assign resources when, where and in any quantity needed.

4. Measured service: the resource usage can be monitored and controlled by metering capability, which is a detailed report for both provider and users to monitor the amount and value of used service based on a short-term charge (e.g., VMs are charged by hour and storage per day). Besides the metering capability, the SLA is also used to monitor resource usage. SLA provides information about the level of availability, usability, performance or other service attributes, such as penalties that are set when a level is broken to ensure the QoS offered by the provider. In this agreement service,

customers need to understand their responsibilities and those of the provider, before using a cloud service.

5. Broad network access: resources can be accessed over the network through a standard mechanism that enables thin[1] and thick[2] client platforms to be used in more devices (such as mobile phones, tablets, laptops and workstations). There are some software available by the provider to be locally installed and others are accessed using browsers, which provide the same work conditions and environment used in a local machine.

2.6.2 Service Models

According to Armbrust et al. [AFG+10], cloud computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services. Cloud computing consists of 20 different oriented services, which are classified into three major groups:

Software as a Service (SaaS). The purchases of services are made through a service provider, which controls and manages the underlying cloud infrastructure, including networking, services, operational systems, storage and other features inherent to the application. Applications are accessible over the Internet, typically via a web browser. The two main advantages of SaaS are low cost since the providers are responsible for deploying and executing the applications, as well as purchasing the software license. Examples of SaaS providers are Google Docs, Gmail and IBM SmartCloud [Sma16].

Platform as a Service (PaaS). PaaS provides the development environment for users to create their application by programming languages, libraries and other tools supported as a service. The platform may contain databases and middleware for the development tools of the application. Even though the services are managed and controlled by providers, users control the application deployed and hosted in the programming infrastructure. Microsoft Azure [AZU16] and Google App engine [GOO16] are examples of PaaS vendors.

Infrastructure as a Service (IaaS). IaaS makes accessible the provisioning of hardware, servers, network, storage and other resources to build on-demand environments. IaaS is also characterized by giving support to PaaS and SaaS, on which users can use the installed application or deploy their environment, such as operational system, database management system, services and so on. The virtualization software, which makes possible to run multiple operating systems and multiples applications on the same server at the same

[1]Thin client is a browser based application and most of the processing is done on the server side.

[2]Thick client stores local files and applications to perform most of the processing on the client side while it is still connected to the server.

time, is an integral part of this model. Some examples of this service model include Microsoft Azure [AZU16], Amazon web services [AMA16] and GoGrid [GoG16].

Although most of providers support all these service models, the users have to select only one due to particular characteristics of the on-demand services above described and illustrated in Figure 2.5. The consumer can decide which model contains the resources need to deploy and execute its applications taking into account the level of services provided by the suppliers. For instance, end users can execute applications on SaaS, developers can create their systems on PaaS, and IT professionals can build their application on IaaS platforms (Figure 2.5).

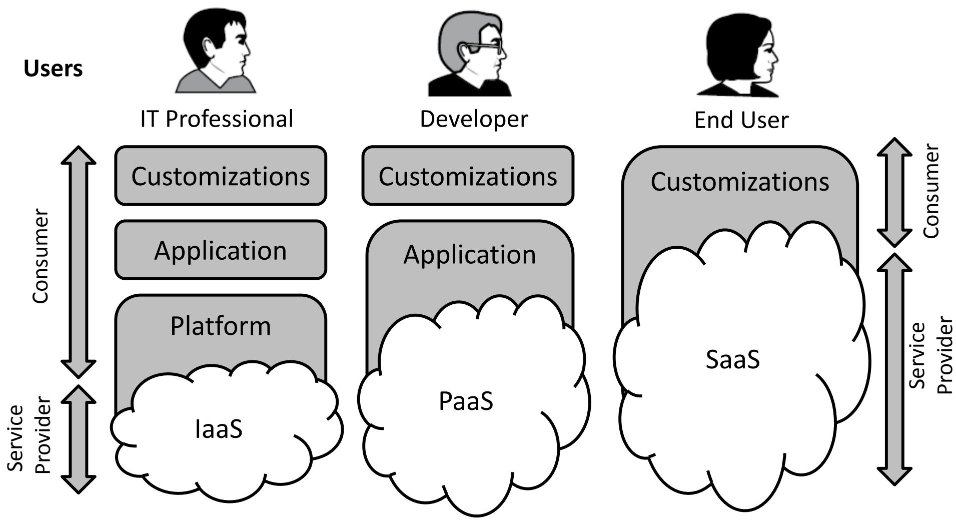


Figure 2.5 – Hierarchy of cloud service models. The level of services designated to users and providers are characterized by rectangles and clouds, respectively. The user profile changes as the level of supported service varies in the different models.

2.6.3 Deployment Models

Deployment models are used by SaaS, PaaS and Iaas providers, allowing to define the type of access and the availability of environments according to the level of service needed. This means that the access to resources, which are determined by consumers, can be controlled according to a business process and type of information. The deployment of cloud computing helps the users to select the model that is suitable to their requirements. It consists of the following four models [MG11]:

1. Private cloud: this model is provisioned for exclusive use by a private organization, and its services are not available to the general public [AFG$^+$10]. This model provides a high level of control, since the privacy and security rules can be dictated by a specific

client, which can also decide if the infrastructure will be hosted on-premises or at a third-party location. The main drawback of this model is that it requires high investment from the organization, which needs deploying, managing and maintaining its own infrastructure.

2. Community cloud: it is used by related communities (companies or users) that belong to a specific group with similar computing apprehensions (e.g., mission, security requirements, policy, and compliance considerations). The infrastructure can be hosted locally or remotely and it is managed by some company from the community, by a third party, or a combination of these.

3. Public cloud: the services of this model are leased and release for open use by the general public. Users are unable to access the physical location of the infrastructure and cloud providers provision services and infrastructure to different clients. Moreover, users have less control and visibility over services, since public clouds are shared and service providers are encouraged to provide a standardized offer to reduce costs. The main technical difference between public and private clouds is that cloud providers offer more levels of security that can be assigned to various services.

4. Hybrid cloud: this model is characterized by the composition of two or more distinct cloud infrastructures, where private, community and public clouds can be combined to unique entities. Its main advantage is the possibility of explicitly providing the portability of data and applications through a standard technology for combining different deployment models.

The deployment models may be run independently or as an integrated system, depending on the business policy assigned to the resources used and the cloud providers.

2.6.4 Architecture

The cloud computing architecture usually varies according to the application used. According to Zhang et al. [ZCB10], the cloud computing architecture is based on a layer model in order to solve the problem of planning a customized cloud architecture better for each application by combining similar services into an equivalent level of abstraction. The four different layers classified by [ZCB10] are hardware, infrastructure, platform and application. Figure 2.6 depicts these four layers adopted to strategically classify the different levels of service abstraction.

The hardware layer is responsible for manipulating the cloud physical resources, including servers, routers, switches, power and cooling systems. It consists of a large number of servers (also known as data centers) and their peripheral devices, organized in racks and

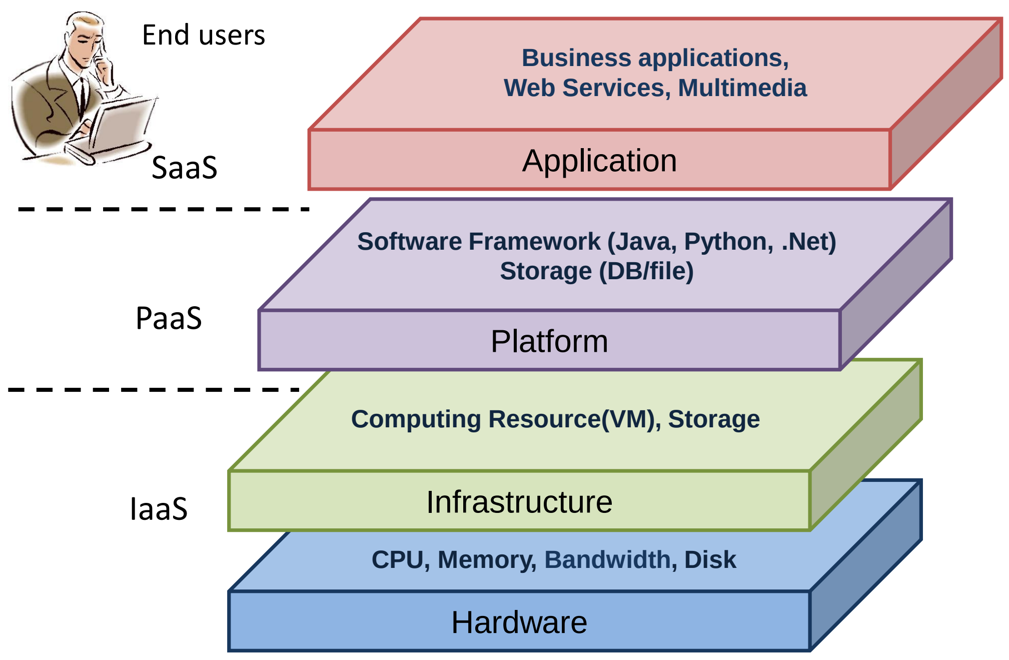

End users
SaaS
PaaS
IaaS
Business applications, Web Services, Multimedia
Application
Software Framework (Java, Python, .Net) Storage (DB/file)
Platform
Computing Resource(VM), Storage
Infrastructure
CPU, Memory, Bandwidth, Disk
Hardware

Figure 2.6 – Cloud layer architecture representation and its service models. Hardware and Infrastructure make up the lowest levels of the layers, which consist of data centers, clusters, desktops, and, other hardware resources. Adapted from [ZCB10].

interconnected through switches, routers or other resources. The typical issues addressed by this layer are fault tolerance, traffic management, power and cooler.

The infrastructure layer is also known as layer of virtualization. The storage and computing resources are virtually created in this layer by partitioning computer hardware using virtualization technologies, such as XenServer [XEN16], KVM [KVM16]. These technologies offer a number of attractive features for this component, including scalability, in which a number of computing power are dynamically assigned to obtain high levels of supercomputing performance [KFJ14].

The platform layer consists of operational systems and framework applications. It is responsible for minimizing the burden of deploying applications straightforwardly into the virtual machines. For example, Microsoft Azure [AZU16] and Google App Engine [GOO16] operate on this layer to provide API support to deploy storage, database, and other typical business web applications.

The application layer is at the highest level of the hierarchy and comprises the real cloud applications for final users. The main advantage of this layer is the low level of efforts by users since the scalability, availability, performance associated with QoS and other cloud features are undertaken by all the other layers.

One of the main advantages of this architectural modularity is the support to a wide range of application requirements, which in turn contribute to the QoS technical improvements and the reduction of the overload time designed for maintenance and management.

Another advantage of this architecture is the loose coupling whereby the computational resources from a layer can be added and replaced in a flexible and scalar manner without affecting other layers [ZCB10].

2.7 General Remarks

In this chapter, the essential concepts on RDD process were presented. A special emphasis was given to the molecular docking procedure of flexible receptor, which holds paramount importance for a better understanding of this thesis. As previously stated by Kuntz [Kun92], molecular docking simulation is the most important step in the RDD process, since it identifies, synthesizes and tests lead compounds in order to choose those that will be manufactured. A more accurate method to mimic the natural behavior of receptor and ligands in biological environments is to consider their explicit flexibility during the molecular docking experiments. In this context, Alonso et al. [ABG06], Amaro et al. [AL10] and Teodoro et al. [TK03] provide in-depth analyses of the constant structural changes that occur when protein and ligand flexibility are taken into account during the interaction process to find a favorable binding conformation. Alonso et al. [ABG06] indicates that protein flexibility increases the efficiency of molecular docking simulations compared with a rigid treatment of the protein, probably due to lower energy barriers when the ligand is allowed to explore a flexible binding site.

An ensemble of different receptor conformations generated from MD simulations is the approach adopted in this study. This approach was selected due to the lack of precision of existing docking methods in considering partially or totally the flexibility of proteins. Usually, scoring functions neglect the conformation changes of binding partners [TA08, BZ10]. Towards this end, a docking experiment is performed for each receptor conformation and ligand and its resulting poses are evaluated [ABM08, AL10]. However, as discussed in this section, the computational cost increases as the number of MD conformations from the FFR model raises [TK03, AL10].

There have been many efforts to reduce time spent in molecular docking simulations of FFR models by combining various computational techniques. For instance, the use of clustering algorithms to reduce ensembles of MD conformations into a manageable size while the critical structural information from the original MD trajectory is preserved [STTC07, LAB$^+$08, BQDPB15, DPQRNdS15, DPQR$^+$15]. Another example is the development of innovative solutions by using HPC environments to speed up docking experiments or by analyzing docking outputs to strategically select reduced ensembles of the promising MD conformations for specific ligands [MSR$^+$11, DPFNdSR13, QDPRNdS14]. Furthermore, Hübler et al. [HRFNdS15] proposed a data-flow pattern to identify the best MD conformations during the docking experiments by using a clustering of snapshots generated by

Machado [Mac11]. De Paris et al. [DPFNdSR13] and Quevedo et al. [QDPRNdS14] assess the performance gains achieved by incorporating this pattern in docking experiments of FFR models.

Chapter 5 presents the clustering methods commonly used for partitioning ensembles of MD conformations where a detailed study of these methods generates the clustering of snapshots used as input to the environment purposed in this study. The remaining computational techniques are reported throughout this Thesis since they comprise the advancements performed from the start to the end of this study, and therefore provides a deeper knowledge of the methodology applied to develop the scientific workflow that optimizes molecular docking simulations of FFR models.

SWfMS is the most widely-used system to handle biology experiments. It provides a variety of capabilities to manipulate data dependencies and perform activities (program or services invocations) into local computing resources or HPC environments [MDO^{+}15]. To address the computationally intensive and the long time taken for performing large-scale scientific experiments, the execution of scientific workflows is moving to a multisite environment [LPVM15]. In this scenario, SWfMSs are able to accommodate input/output data in different sites at runtime and send workflow invocations to be directly executed into resources distributed in different infrastructures (e.g. grid or cloud). Recently, Liu et al. [LPVM15] provide in-depth analysis of the data-intensive scientific workflow management in a multisite cloud and they concluded that a lot of improvements is necessary for terms of performance capability since the most familiar SWfMSs are suitable for single static computation and storage resources in grid environments. Despite the capability lacks presented by Liu et. al. [LPVM15], which address specific issues based on a set of 8 SWfMS, a number of studies has revealed the performance gains when large-scale scientific experiments are running using cloud-based workflow enactment systems [CHWW13, OBDO^{+}14].

For the reasons described above, the cloud-based environment proposed to optimize the docking-based virtual screening of FFR models is developed by using a SWfMS and a set of VMs from a cloud environment. Next section reports the several attempts performed by the research community regarding the challenges in extracting the most biologically relevant information when the dimensionality of FFR models is simplified or reduced.

3. CLUSTERING THE 20 NS InhA MOLECULAR DYNAMICS TRAJECTORY

Clustering algorithms applied to results of MD simulations is the most appropriate data mining technique to partition structural ensembles into groups of structures which share similar conformational features [HW79, STTC07, PCN11]. In this approach, every MD conformation is divided into several groups by using a measure of (dis)similarity. MD conformations within a group are, according to some criterion, similar to each other and dissimilar from the conformations of other groups [HW79].

A number of methods has been applied in many studies to cluster molecular dynamics trajectories. For instance, partitioning data sets into structurally homogeneous subsets for modeling [Li06, PCN11], picking representative chemical structures from individual clusters [LAB$^+$08, DLS$^+$05, CL09], and optimizing virtual screening results [DPFNdSR13, QDPRNdS14]. A more detailed analysis on clustering MD trajectories was done by Torda and van Gunsteren [TvG94] and Shao et al. [STTC07]. Torda and van Gunsteren [TvG94] created the distance measure D_{ab} for clustering an MD trajectory with 2,000 structures. By comparing hierarchical divisive algorithm and single linkage, they concluded that the divisive algorithm outperforms results when a trajectory configuration is evenly distributed across the conformational space. Shao et al. [STTC07] compared eleven different clustering algorithms to assess the performance and differences between such algorithms based on the pairwise RMSD distance. Shao and co-authors used the clustering metrics to find an adequate number of clusters in ensembles of structures taken from a "sieved clustering". In their approach, a subset of the data is clustered and the remaining data are added to existing clusters in order to handle very large data sets more efficiently.

Preliminary work on clustering MD trajectories for a selective docking protocol was undertaken by Machado et al. [Mac11]. In their study, the snapshots from a 3.1 ns InhA MD trajectory were clustered based on the number of interactions that each snapshot and ligand performs into the region of two different small molecules: the substrate analog (THT), and the THT together with the NADH coenzyme. According to Machado [Mac11], these small molecules were selected because THT substrate-binding position defines a cleft where InhA competitive inhibitors are expected to bind, and THT + NADH coenzyme binding region comprises the largest binding pocket of the InhA active site. The resulting clustering was originally created to be used as an input for the P-SaMI data pattern [Hüb10, DPFNdSR13]. However, Quevedo et al. [QDPRNdS14] stated that this clustering becomes very specific for ligands with a similar structure to the ones used to generate the clusters, since the number of interactions was taken from an exhaustive molecular docking simulation between the entire MD trajectory and a specific ligand.

To address this problem, we concentrate our efforts on identifying small and localized changes that are expected to have a major influence on the interactions between flexible receptors and different ligands. Instead of focusing on traditional pairwise distances between MD conformations, such as RMSD [STTC07, LAB$^+$08] and D_{ab} [TvG94], we proposed a method that groups similar behavior in the substrate-binding cavity of every MD conformation by extracting its structural properties.

This study presents two main contributions. First, it provides a detailed comparison of six clustering algorithms applied to three different data sets generated from the same MD trajectory. These six methods are: *k*-means, *k*-medoid, Complete linkage, UPGMA, WPGMA and Ward's. Each data set contains information from the 20 ns MD trajectory of the InhA-NADH complex, one regarding features from its substrate-binding cavity, while the other two are composed of well-known measures of similarity for clustering MD trajectories. Subsequently, this study identifies ensembles of representative MD conformations from the best clustering solutions. The optimal solution is selected based on dispersion measures of estimated FEB, which were extracted after executing docking experiments between the FFR model and 20 different ligands. We also investigated the variance of the RMSD values from the clustering generated to asses the level of similarity in the docking final poses from snapshots within the same group.

In this study, the best selected partition is used as input to the cloud-based environment. Thus, if a receptor conformation belongs to a cluster that interacts favorably with a specific ligand, one could assume that other conformations within the same cluster will behave similarly. Otherwise, the conformations belonging to this cluster are considered unpromising, and consequently may be discarded to reduce the number of docking experiments on the FFR model [DPQR$^+$15]. As a result, we expect to generate an RFFR model for each ligand by discarding groups of snapshots that have low or no binding affinity.

3.1 Clustering Algorithms Applied for Partitioning the MD Trajectory

A number of clustering algorithms have been used during this study with the intention of examining different ways to identify patterns among the snapshots that make up the FFR model based on their level of similarity. The InhA MD trajectory was clustered using algorithms implemented in R Programming Language [Tea12]. k-means [M$^+$67], k-medoids [KR90], and agglomerative hierarchical [KR90] methods and their variations were used to identify patterns among the snapshots that make up an FFR model based on their level of similarity. k-means and k-medoids belong to the set of partitioning clustering methods, which divide a set of data objects into non-overlapping subsets with spherical shape such that each data object is in exactly one subset [TSK13, HKP11].

k-means is a well-known clustering algorithm that locally optimizes the average squared distance of points from their nearest cluster center (centroid). It randomly generates k centroids and refines them throughout several iterations, where it computes the distance of every point to the k centroids in order to determine the cluster memberships [M+67]. To create clusters more compact and separated, the k-means algorithm minimizes the sum of squared errors between all objects p of a given cluster C_h and its centroid c_h for all clusters k according to the following equation:

$$Means = \sum_{h=1}^{k} \sum_{p \epsilon C_h} dist(p, c_h)^2 \tag{3.1}$$

In contrast to k-means, whose centroid almost never correspond to an object, k-medoids uses PAM (Partitioning Around Medoids) algorithm for clustering data sets based on central objects. This algorithm generates a set of representative objects or medoids to determine whether a non-representative object is a good replacement for a current medoid [HKP11]. While k-means technique uses the sum of the squared error function to measure the within-cluster variation, k-medoids algorithms apply an absolute error criterion. In this method, the objects (n) are grouped into k clusters by minimizing the sum of the dissimilarities between each object and its corresponding representative. Then, the sum of the absolute error for all objects p in the data set is defined as:

$$Medoids = \sum_{h=1}^{k} \sum_{p \epsilon C_h} dist(p, o_h) \tag{3.2}$$

where o_h is the representative object of C_h. As the k-medoids method calculates the distance between objects to obtain the mean values of the cluster, it is less sensitive to outliers than k-means. The complexity of k-medoids is squared to the number of instances, since a distance matrix is used to calculate the representative object. Compared to k-means algorithm, its main drawback is the high computational cost. While PAM takes $O(k(n-k)^2)$ to compute each iteration, k-means runs in $O(nkh)$, where the number of clusters (k) and the number of iterations (h) are usually less than the number of objects (n). In such circumstances, k-means is relatively more scalable and efficient in processing large data sets. k-medoids also uses CLARA (Clustering LARge Applications) algorithm to address large data sets [KR90]. CLARA generates multiple random samples from the entire data set and uses the PAM algorithm to compute k-medoids on these samples. Therefore, the complexity of computing CLARA is $O(ks^2 + k(n-k))$, where s is the size of the sample. In this method, a representative sample contains medoids that approximate the medoids of the original data set. Partitioning clustering methods use k random cluster centers or seeds to start partitioning the data set. In this study, we performed multiple runs with different initial centers and representative objects in order to have consistency between k-means and k-medoids runs.

Unlike partitioning clustering, hierarchical clustering methods aim to group data into levels such as in a dendrogram or “tree” of clusters [HKP11]. It has two basic approaches known as agglomerative and divisive. The agglomerative hierarchical clustering, which uses the bottom-up strategy, starts with each object as an individual cluster and interactively merges the closest pair of clusters until all the objects are in a single cluster or the maximum number of clusters is reached. The divisive hierarchical clustering, which uses the top-down strategy, starts with all objects in the same cluster and splits a cluster into smaller clusters in each iteration until each object becomes a singleton cluster or a termination condition holds [HKP11, TSK13]. In this study, we use only the agglomerative algorithms since the divisive method does not handle efficiently large data sets due to its computational costs. The limiting factor of the divisive method is that there are $2^{n-1} - 1$ possible ways to partition a set of n objects into two subsets.

To measure the proximity between two points in two different clusters, agglomerative algorithms widely use the methods known as single linkage, complete linkage, median, centroid, group average and Ward’s. All these methods were applied in this study for clustering data sets using the AGNES (AGlomerative NESting) method [KR90]. However, analyzing the resulting partitions, we noted that single linkage, median and centroid methods showed singleton clusters, high sensibility to outliers and, in some cases, a cluster with more than 50.00% of all objects into a single large cluster. For this reason, we decided to analyze only the partitions from Complete, Unweighted Pair Group Method using Arithmetic averages (UPGMA), Weighted Pair Group Method using Arithmetic averages (WPGMA) and Ward’s.

The complete linkage version of hierarchical clustering tends to minimize the increase in diameter of the clusters at each iteration by determining the proximity of two clusters (C_i, C_j) as the maximum distance based on the following equation:

$$Complete(C_i, C_j) = \max_{x \epsilon C_i, y \epsilon C_j} dist\{|x - y|\} \tag{3.3}$$

where $|x - y|$ is the distance between two objects or points x and y.

In UPGMA and WPGMA, which are group average agglomerative methods, the distance between two clusters is defined as the average pairwise proximity among all pairs of points or objects in different clusters [KR90]. The difference between these methods is the weight given to the points in different clusters to measure the pairwise proximity. While UPGMA takes into account the number of points in each cluster making a linkage between groups, WPGMA treats all clusters equally making a linkage within groups, defined as:

$$UPGMA(C_i, C_j) = \frac{1}{n_i n_j} \sum_{x \epsilon C_i} \sum_{y \epsilon C_j} dist|x - y| \tag{3.4}$$

$$WPGMA(C_i, C_j) = \frac{1}{2} \sum_{x \epsilon C_i} \sum_{y \epsilon C_j} dist|x - y| \tag{3.5}$$

where n_i and n_j are the number of objects from cluster C_i and C_j, respectively, and $|x - y|$ is the distance between two objects or points x and y.

The Ward's method, also called the minimum variance method, aims to merge pairs of clusters with minimum variance. As the k-means algorithm, this method calculates the squared error to merge two clusters. The “merging cost”, which is evaluated when two clusters are merged, is defined as:

$$Ward = \sum_{h=1}^{k} \sum_{x_i \epsilon C_h} \sum_{j=1}^{p} dist(x_{ij} - \overline{x}_{hj})^2 \tag{3.6}$$

with the cluster mean $\overline{x}_{hj} = \frac{1}{n_h} \sum_{x_i \in C_h} x_{ij}$, where x_{ij} denotes the value for the i^{th} individual in the j-cluster, k is the total number of clusters at each stage, and n_j is the number of individuals in the j^{th} cluster.

Clustering of MD conformations is especially useful for molecular docking simulations since it provides clusters of similar receptor structures. The main contribution of this study is on investigating the partitions generated by different clustering methods and identifying the one which provides the optimal solution. An optimal solution is a clustering that contains high similarity among MD conformations placed in the same group (cohesion), and high dissimilarity from the conformations of other groups (scattered) [HW79]. The best clustering of snapshots solution will be used as an input to the docking-based virtual screening method purposed in this study.

3.2 Data Sets for Clustering the MD Trajectory

To investigate the best measure of similarity, the quality of partitions was compared by using our new set of cavity attributes and two traditional data sets used for clustering MD trajectories. The specification from the MD trajectory used to generate the data sets is described in Section 2.4.1. Hence, we generated the following data sets:

1. Protein RMSD. The pairwise RMSD distance between the first and every MD structure, considering all structure residues as applied by [ZS04, LZ06, STTC07].

2. Cavity RMSD. This data set contains the RMSD distance between the first and every MD's structure, considering the residues that enclose the substrate-binding cavity of the InhA-NADH complex and the C16 substrate analog (PDB ID: 1BVR) enzyme [RVS+99]. Application examples of this measure of similarity are in [LAB+08, LWW+08].

3. Cavity Attributes. The data set proposed in this study, built by using a set of features extracted from the substrate-binding cavity of each conformation generated by the MD trajectory. A detailed explanation on how each attribute was obtained is given in this section.

It is expect that the partitions generated will be able to group MD conformations with similar features within their binding site by using the Cavity Attributes data set. This approach allows to cover small and local protein movements, in order to further improve the selection of suitable ligand-protein interactions. The structural features extracted from each MD conformation are:

1. the volume of the substrate cavity (in Å^3);

2. the number of heavy atoms of the 1BVR structure [RVS+99] that are present in the substrate-binding cavity; and

3. the pairwise RMSD distance relative from the first to the current snapshot (in Å).

Pairwise RMSD distances were evaluated by using the differences among backbone atoms from the first structure against each MD conformation, using the following equation:

$$RMSD_{r_t,r_{ref}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}|r_{t,i} - r_{ref,i}|^2} \tag{3.7}$$

where $r_{t,i}$ and $r_{ref,i}$ are the positions of equivalent atoms in the conformation at time $t(r_t)$ along the MD simulations and the reference structure (r_{ref}), respectively. The RMSD was calculated using the *ptraj* module from AmberTools14 [CDCI+16].

The 1BVR enzyme was chosen to compute the features that composes the data set for clustering the MD trajectory because it contains the THT ligand [RVS+99], the substrate analog in which its region encloses the InhA substrate-binding cavity. CASTp [BNL03] is the web tool used to identifies and calculates the two firsts structural features extracted from the InhA substrate-binding cavity. It is used to identify the solvent accessible surface area and the volume of the substrate cavity. A heuristic function was created to identify the substrate cavity on an ensemble of conformations generated by the MD simulation. This function is based on the number of heavy atoms present in the substrate-binding cavity of the 1BVR structure. The substrate cavity and the largest number of atoms were identified by considering the residues that enclose the substrate analog to the 1BVR crystallographic structure [RVS+99]. Thus, we computed the volume and the number of heavy atoms for each MD conformation considering the number of heavy atoms of the 1BVR structure that are present in the substrate-binding cavity of each MD conformation. Figure 3.1 illustrates the 1BVR substrate cavity and the residues that enclose it.

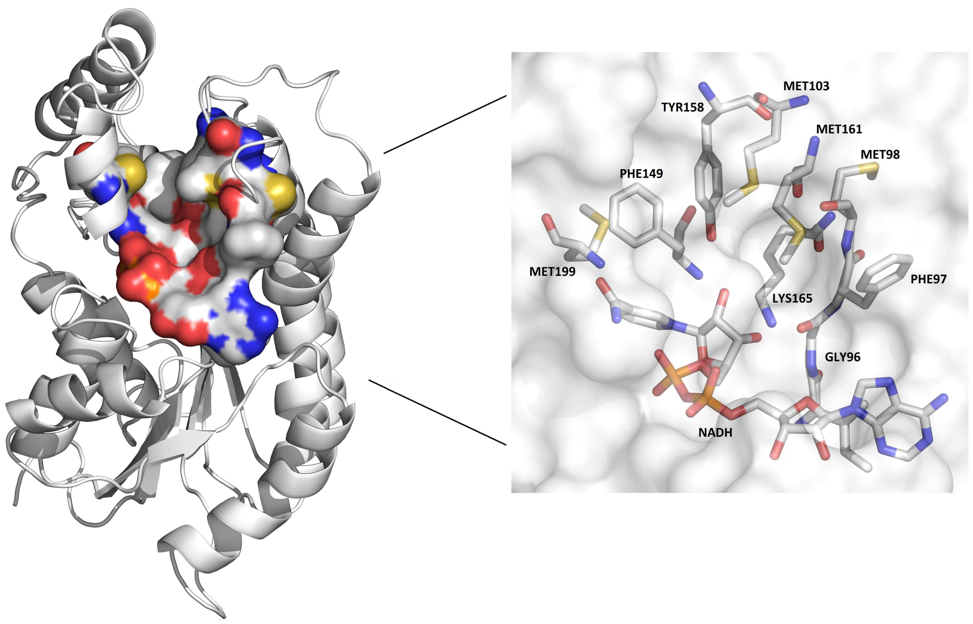


Figure 3.1 – Existing residues in the InhA crystal structure substrate-binding cavity (PDB ID: 1BVR). The 1BVR substrate-binding cavity is represented by molecular surface on the left side. The projection displays all residues from the binding pocket in stick representation. Residues and the substrate-binding cavity are colored by atom types [DPQRNdS15].

The volume of the substrate-binding cavity was chosen as one of attributes from Cavity Attributes data set since it varies considerably along the MD simulation. This is evidenced by analyzing the substrate-binding cavity volumes generated by CASTp, which ranged from 45.4 Å^3 to 2,852.9 Å^3 for the entire 20 ns MD simulation trajectory.

While volume encloses the substrate-binding cavity based on the solvent accessible surface area, the heavy atoms recognize the substrate-binding cavity dimension to fit a ligand based on a weight system. The weight system gives different scores for each residue considering geometric properties. Specifically, an MD conformation substrate-binding cavity placed on the NADH coenzyme will receive more weight than other residues, since it shapes the base of the target cavity and, therefore, increases the chances of finding a receptor in the right conformational state to accommodate a particular ligand [ABG06]. Thus, we decided to assign weight 1 for atoms whose residues determine the substrate analog in the crystal structure, and 3 for atoms whose residues surround the NADH nicotinamide ring. Table 3.1

exemplifies the features extracted from the substrate-binding cavity of each MD conformation, where a score is given for each residue. The maximum number of heavy atoms that a residue can have is indicated in the table header.

Table 3.1 – An excerpt of Cavity Attributes data set specification.

RMSD (Å)	Volume ($Å^3$)	GLY96 (4 atoms)	PHE97 (4 atoms)	MET98 (8 atoms)	MET103 (8 atoms)	PHE149 (11 atoms)	TYR158 (12 atoms)	MET161 (8 atoms)	LYS165 (9 atoms)	MET199 (8 atoms)	NADH (9 atoms)
0.39	779.80	2	4	2	2	1	2	2	0	4	6
0.39	821.30	2	4	3	2	3	3	2	0	4	6
...	...	...	...	...	...	...	...	...	...	...	...
1.34	1385.20	3	7	4	2	3	8	3	6	3	5
1.44	1249.50	3	7	4	2	2	5	4	4	1	6

Attributes are shown in Table 3.1 in their original format. To improve accuracy of clustering methods, we normalized these attributes using the interval [0,1] and generated a CSV format file. In addition, we removed the first 500 MD conformations from the FFR model, due to significant variations in the temperature that occurs during the equilibrium state of MD simulations. In this state, To achieve stability of simulations conditions, a number of properties are monitored, such as structure, pressure, energy and temperature. After the initial state, MD properties may be investigated by keeping the system in a steady non-equilibrium state. Figure 3.2 depicts the irregular RMSD changes in the equilibrium state from the binding cavity of the 20 ns InhA MD trajectory. The RMSD variation of our MD trajectory is stabilized between 1.0 Å and 1.8 Å, reaching a plateau around 1.4 ± 0.1 Å.

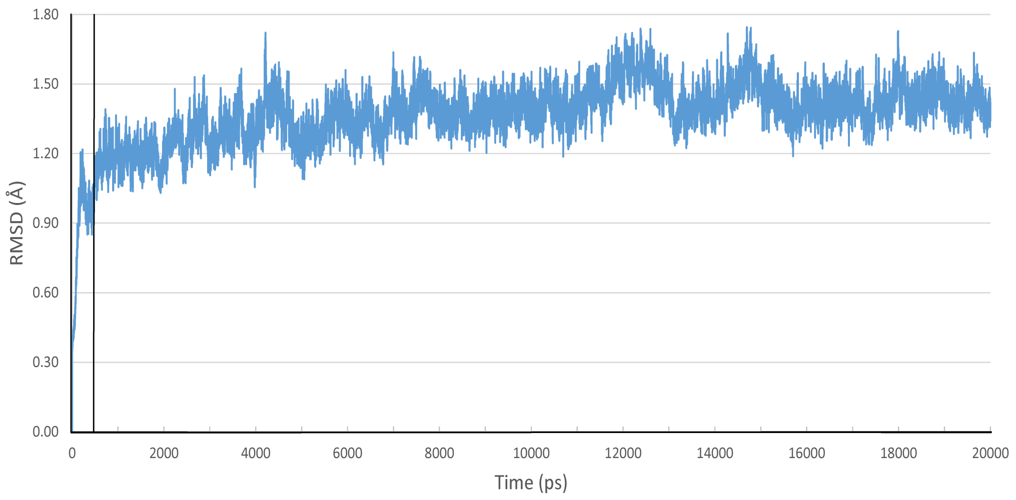


Figure 3.2 – Binding cavity pairwise RMSD distance analyses from the 20ns InhA MD trajectory. The black vertical line encloses the non-equilibrate state of the MD trajectory at 500 ps. The average RMSD variation is 1.37 ±0.14 Å.

3.3 Approach for Validating Data Partitions

The performance gain among clustering solutions generated for each algorithm and the three different data sets were evaluated by comparing the docking results obtained by performing ensemble docking experiments between the FFR model and 20 different ligands. The quality of partitions were assessed computing the first, second and third quartile values of medoids for each clustering solution. Quartile values have the power to indicate central tendency and dispersion of the data points inside a data set, being also resistant to outliers. As we seek to group MD conformations with high similarity in their substrate-binding cavities, the binding mode conformations of different drug candidates to the enzyme under study were investigated. In this regard, 20 compounds experimentally tested (Figure 3.3) were used in docking experiments against the FFR model to predict their lowest energy bound conformation (i.e. pose). This set of ligands was extracted from 20 InhA crystal structures available on Protein Data Bank (PDB) [BWF+00].

The set of ligands used in this study was chosen because the overall topology RMSD of their complexes is significantly different. According to Pauli et al. [PdSR+13], the binding cavity volumes of the 36 InhA crystal structures available on PDB ranges from 1,597.3 Å^3 to 3,046.7 Å^3 for the whole cavity (NADH + substrate-binding cavity). This range represents the different sizes and structures of the ligands that are bound in such structures, being a good sample to dock in the MD trajectory, since its binding cavity volume ranges from 419.4 Å^3 to 2,032.8 Å^3 (InhA-NAD complex).

To obtain the best clustering solution, the dispersion of resulting partitions was evaluated based on predicted FEB values extracted from the ensemble docking experiments. Dispersion corresponds to the first, second and third quartile values, which were calculated to compare the level of convergence between the resulting partitions and the MD's full trajectory. This assessment consists of three steps:

1. Generate a set of representative objects for every partition by selecting the medoid of each cluster;

2. Compute the sum of each quartile of all ligands for every partition based on predicted FEB values; and;

3. Compute the Sum of the Quartile Differences (SQD) based on the results from step 2.

SQD is the metric proposed in this study to select the optimal solution among the partitions generated from the six different clustering methods. This metric intends to find an ensemble in which the FEB dispersion between medoids and the MD's full trajectory is similar based on the following equations:

$$SQD = \sum_{j=1}^{q} |(\overline{x}_j - \overline{y}_j)| \tag{3.8}$$

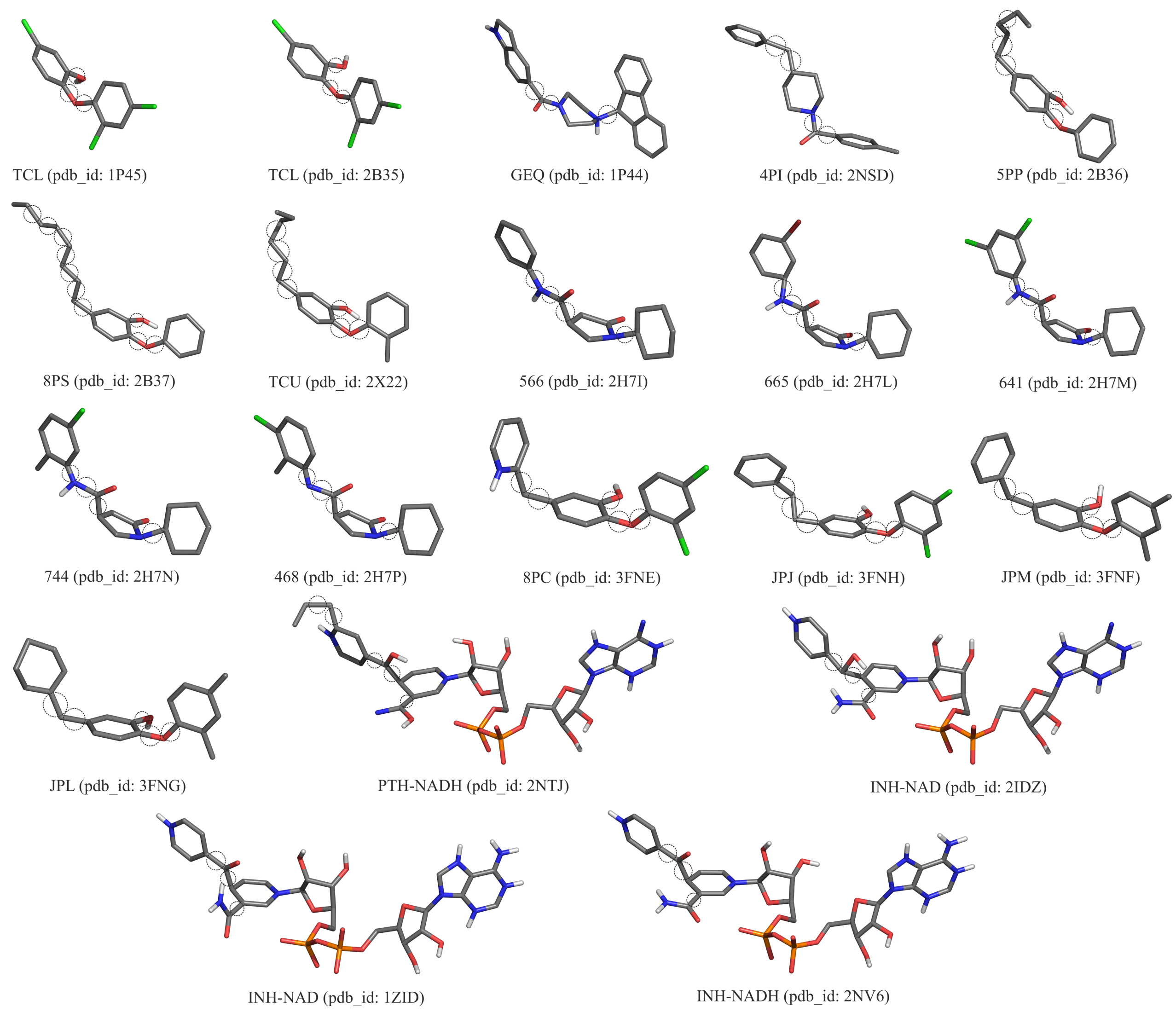


Figure 3.3 – 3D structure representation of the 20 ligands used to perform the ensemble docking experiments. The ligands are identified by PDB ID and coloured by atom type (carbon and hydrogen: light grey; nitrogen: blue; oxygen: red; chloro: green; phosphorus: orange; sulphur: yellow). The dashed circle depicts the rotatable bounds selected by AutoDock Tools 1.5.6. Ligand abbreviations: **TCL**: triclosan; **GEQ**: 5-[4-(9H-fluoren-9-YL)piperain-1-yl]carbonyl-1H-indole; **4PI**: N-4-methylbenzoyl-4-benzylpiperidine; **5PP**: 5-pentyl-2-phenoxyphenol; **8PS**: 5-octyl-2-phenoxyphenol; **TCU**: 5-hexyl-2- (2-methylphenoxy)phenol; **566**: (3S)-1-cyclohexyl-5-oxo-N-phenylpyrrolidine-3-carboxamide; **665**: (3S)-N-(3-bromophenyl)-1-cyclohexyl-5-oxopyrrolidine-3-carboxamide; **641**: (3S)-1-cyclohexyl-N-(3,5-dichlorophenyl)-5-oxopyrrolidine-3-carboxamide; **744**: (3S)-N-(5-chloro-2-methylphenyl)-1-cyclohexyl-5-oxopyrrolidine-3-carboxamide; **468**: (3S)-N-(3-chloro-2-methylphenyl)-1-cyclohexyl-5-oxopyrrolidine-3-carboxamide; **8PC**: 2-(2,4-dichlorophenoxy)-5-(pyridin-2-ylmethyl)phenol; **JPJ**: 2-(2,4-dichlorophenoxy)-5-(2-phenylethyl)phenol; **JPM**: 5-benzyl-2-(2,4-dichlorophenoxy)phenol; **JPL**: 5-(cyclohexylmehyl)-2-(2,4-dichlorophenoxy)phenol; **PTH-NAD** Prothionamide + NADH coenzyme; **INH-NAD** Isoniazid + NADH coenzyme [DPQRNdS15].

$$\overline{x}_j = \frac{1}{n_j} \sum_{x_i \epsilon Q_j} x_{ij} \tag{3.9}$$

$$\overline{y}_j = \frac{1}{n_j} \sum_{x_i \epsilon Q_j} y_{ij} \tag{3.10}$$

where q identifies the first, second and third quartiles, x_{ij} and y_{ij} denote the FEB value for the i^{th} ligand in the j^{th} quartile for the ensemble of medoids and the MD's full trajectory, respectively. The lower the SQD value, the higher the similarity of dispersion to the MD's full trajectory.

According to Jain and Dubes [JD88], there is no perfect clustering algorithm that assures the best solutions for all data sets. The exploratory analysis and understanding on data sets are important decisions for selection of the strategy (such as number of clusters, prototype and clustering method) to be adopted [JD88]. In this study, we analyzed partitions from three different data sets and six clustering methods. Even though we are using predicted FEB values to select the optimal solution, we believe that our methodology may provide an effective strategy for improving clustering MD trajectory approaches, and consequently, optimizing the total elapsed time taken to perform ensemble docking experiments. Further, we expect that the ensemble of medoids will also be able to label a fairly good level of distinct binding modes from the FFR model for accommodating different ligands. This is more likely with clustering solutions generated from the Cavity Attributes data set.

3.4 Comparative Analyses in the Partitioning Solutions

Our hypothesis is that the methodology used for clustering the MD trajectory can effectively group conformations with high similarity in their binding cavity. Specifically, we seek to reduce the computational time of using a very large MD trajectory, i.e. more than thousands of conformations, to perform ensemble docking in thousands or millions of ligands. One way to address this issue is to discard groups of MD conformations that are inadequate to bind productively with a given ligand during docking experiments. Thus, we concentrate our efforts on using clustering methods and evaluating their results to validate the working hypothesis. Our main contribution is on investigating different clustering methods to find the best strategy for clustering MD trajectories. Towards this end, six different methods were first employed, and subsequently, every clustering solution was analyzed based on the distribution of FEB values between ensembles of representative conformations (medoids) and the MD's full trajectory.

Clustering was performed on the 20 ns InhA MD trajectory using three different data sets as a measure of similarity and a series of independent runs varying the cluster numbers from 10 to 200. We started the clustering analyses from 10 clusters since low *k* values showed poor level of scatter and, consequently, it would be unable to reflect all possible movements from an MD trajectory of 20,000 conformations. The initial maximum cluster variation was 1,000. However, the total time to run molecular docking simulations increases as the quantity of clusters rises, since a sample of snapshots should be processed for each cluster for quality evaluation. Furthermore, good levels of accuracy were evidenced for different clustering methods when the number of clusters was equal or less than 200.

To select the optimal clustering solution, the dispersion among partitions generated from 10 to 200 clusters was analyzed by assessing the SQD values (equation 3.8). Figures 3.4 and 3.5 show SQD variations for the three data sets as a function of the cluster count for each clustering method. Comparing the two graphs, it can be seen that SQD values of partitioning-based clustering algorithms (Figure 3.4) constantly oscillated throughout the number of clusters for the three data sets. This is due to the fact that partitioning methods have difficulty on detecting natural clusters, i.e. when clusters have non-spherical shapes or widely different sizes or densities [HKP11]. Usually, the measures of similarity used for clustering MD trajectories have low dispersion. For instance, the RMSD normalized values varied from 0.41 to 0.95 for cavity, and from 0.58 to 0.99 for protein, having a maximum amplitude of 0.54.

Unlike partitional-based clustering methods, hierarchical algorithms outperformed the partitioning of all data sets. As can be seen in the graphs from Figure (3.5), SQD values varied constantly in the beginning, but after a certain number of clusters, they reached a peak and remained steady. Best hierarchical solutions were selected after SQD values reached more stable behaviors. In this analysis, Cavity Attributes data set showed best SQD values for every hierarchical method.

Regarding hierarchical clustering performance, Figure 3.5 shows that the lowest SQD values were reached by the Ward's method for all data sets. This method also outperformed the affinity with the MD's full trajectory for the Protein RMSD data set, which in turn showed the worst solutions on other hierarchical methods. Jain and Dubes [JD88] identifies Ward's as the better method for clustering functional data, particularly those with periodic tendencies, but the clusters tend to be spherical due to the equally distance computed for all cluster object. This is because the sum of squares criterion tends to merge small clusters given the same amount of splits. Regarding *k*-means algorithm, Ward's method also was able to reach a similarity dispersion, but it was unable to reach the similar central tendency undertaken for the entire ensemble of MD conformations.

The hierarchical clustering solutions reflected the main advantages from hierarchical agglomerative methods, i.e. these methods are more versatile and embed flexibility regarding seeking a proper level of granularity. Comparing the performance from clustering methods, Figure 3.5 shows that UPGMA, WPGMA and Complete are good methods for clustering data sets with target cavity features, while Ward's methods can be considered a good solution for every type of data set. However, the ability of Ward's of grouping objects that are as homogeneous as possible ended in partitions with central tendencies. Further,

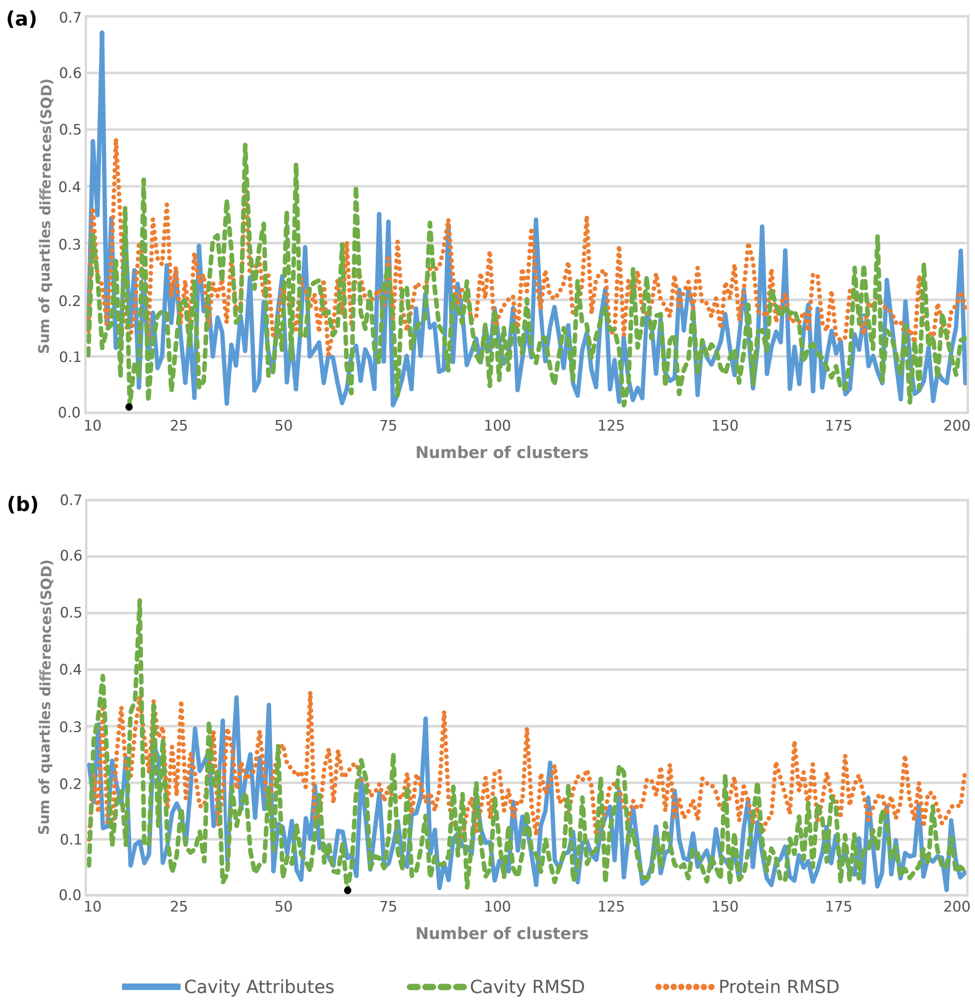


Figure 3.4 – Comparative performance of partitional clustering methods for Protein RMSD, Cavity RMSD and Cavity Attributes data set. SQD variations as a function of number of clusters for, k-means and k-medoid methods in graphs (a) and (b), respectively. Black points identify the optimal partitioning solution for each method [DPQRNdS15].

the high cohesion in the clusters generated from UPGMA and WPGMA methods were unable to reach low SQD values and number of clusters. Complete method looks for maximum distance to merge a new object in a cluster, and therefore, it becomes more susceptible to noise and outliers.

We also analyzed the partitioning solutions by using docking final poses, i.e. the RMSD values which were extracted from docking experiments performed between the 20 ns InhA MD trajectory and 20 ligands experimentally tested. In this regard, the mean variance for each clustering solution and the cumulative mean variance by ligand were computed to investigate the average changes in the RMSD values over the clustering produced for each method, and for each data set [1]. Figures 3.6 and 3.7 depicts the variation in the RMSD values for the clustering methods used. The line graphs compares the mean of accumulated RMSD variances in three different data sets namely, Cavity Attributes, Cavity RMSD and Protein RMSD, within 190 clustering solutions.

[1]Cluster with only one snapshot received weight one.

Comparing the RMSD values from partitional clustering methods, Figure 3.6 shows the minimal level of variance in clusters generated from Cavity Attributes data set for both, *k*-means and *k*-medoid algorithms. Further, it can be seen a considerable difference in the means of RMSD accumulated variance between the Cavity Attributes data set and the other two data sets, particularly for k-means algorithm where the mean of RMSD accumulated variance decreases as the number of clusters increases. These findings suggest that Cavity Attributes data set is the best similarity function when partitional clustering methods are used to group snapshots with high level of similarity in their docking final poses.

As can be seen from the Figure 3.7, hierarchical methods presented means of the RMSD accumulated variance higher than clustering solutions from k-means algorithms for all data sets. The lowest RMSD variances are reached by Complete and Ward's methods for Cavity Attributes data set. Even though UPGMA and WPGMA methods reported the lowest RMSD variances for Protein RMSD data set, it can be seen from graphs in Figure 3.7 that Complete and Ward's methods were able to reach the lowest means of RMSD accumulated variance for Cavity Attributes data set. In summary, these results show that Cavity Attributes data set is also the best similarity function for clustering MD trajectory when Complete and Ward's hierarchical methods are used.

It is somewhat surprising that the mean of the RMSD accumulated variance from Cavity RMSD and Protein RMSD data sets remained relatively unchanged in all clustering solutions produced by the methods used. In contrast, Cavity Attributes data set reported smaller RMSD variances and notable differences with the other two data sets for all clustering methods used. This means that the data set proposed in this study has the ability to detect small and localized docking pose changes, which, otherwise, are not possible when Protein RMSD and Cavity RMSD data sets are used.

The complexity of clustering algorithms is strongly related to the number n of data objects and the number k of clusters [WKQ+08]. From all experiments performed in this study, CLARA (*k*-medoid algorithm for large data sets) was the algorithm that required the longest execution time. The time noticeably increased due to the size the of sample, which proportionally grew as the number of clusters rose. This is due to the own nature of *k*-medoid which is more robust to noise and outliers. The complexity to compute and select a new medoid from representative objects by PAM algorithm is $O(k(n-k)^2)$. Algorithms from hierarchical agglomerative methods are expensive in their computational and storage requirements [TSK13]. Agglomerative methods compute the proximity matrix that needs $O(n^2)$ time to store and keep track of the clusters. The total time required for these algorithms is $O(n^2 logn)$ where $logn$ is the additional complexity of keeping data in a sorted list. In contrast to the hierarchical algorithms, which have the quadratic asymptotic running time with respect to the number of objects, *k*-means produces a number of partitions for every k in a linear time complexity for any aspect of the problem size [WKQ+08]. The complexity of *k*-

means algorithm is $O(nkh)$, where the number of clusters (k) and the number of interactions (h) are usually less than the number of objects (n).

3.5 Selecting the Optimal Clustering Solution

To identify the optimal clustering solution, we investigated the dispersion within the clusters based on FEB values and the mean of the RMSD accumulated variance from 20 known ligands of the InhA enzyme. These values were investigated since the existing clustering validity criteria, such as Davies-Bouldin (DB) [DB79], gap statistic [TWH01] and Dunn's [HBV02] indexes, showed to be ineffective as the number of clusters rose for every data set. The optimal clustering solution used as input for the method proposed in this study was selected by examining two different metrics: (i) the level of similarity between the MD's full trajectory and the optimal solution of each clustering method, and; (ii) the level of similarity in the snapshots within the clusters based on docking poses. For that, we first calculated the variance, average and standard deviation for the MD trajectory and the optimal set of medoids for the six clustering methods. After identifying the optimal clustering solution based on the SQD values, we analyzed the RMSD variance to identify if the selected solution has clusters composed of snapshots with some similarity in their docking final poses. Table 3.2 summarizes all clustering solutions with the lowest SQD values of each clustering method along with the corresponding number of clusters.

Table 3.2 – Statistical assessments from the optimal solution for each clustering method.

Clustering Method	Data Set	*k* cluster	SQD	Average	Std. Dev.	Variance
k-means	Protein RMSD	19	0.01	-6.61	±0.70	0.54
k-medoid	Protein RMSD	66	0.01	-6.63	±0.70	0.55
UPGMA	Cavity Attribute	133	0.04	-6.58	±0.72	0.56
WPGMA	Cavity Attribute	84	0.03	-6.90	±0.73	0.58
Complete	Cavity Attribute	48	0.01	-6.59	±0.69	0.51
Ward's	Cavity Attribute	95	0.01	-6.60	±0.68	0.51
-	MD trajectory	20,000	0.00	-6.58	±0.72	0.57

As can be seen in Table 3.2, UPGMA and WPGMA methods showed the highest SQD values, i.e. low similarity of dispersion between the clustering solutions and the MD's full trajectory. This low level of similarity can also be evidenced in the graphs (a) and (b) from Figure 3.7 where UPGMA and WPGMA methods presents the highest mean of the RMSD accumulated variance in all data sets and number of clusters tested. One reason for this low accuracy was the existence of many singletons, i.e. clusters with a single object. For cluster analysis, singleton clusters are likely to be outliers and their medoids may not be representative, which in turn means a higher SQD. According to Han [HKP11], one way to solve this problem is to discover outliers and eliminate them beforehand. However, there are some clustering applications for which outliers should not be removed, as in the clustering solution employed in this study. The removal of objects will affect the quality of RFFR models since a single snapshot that shows poor interactions with a specific ligand may produce favorable interactions with another set of different ligands.

It can be seen from the data in Table 3.2 that Complete, Ward's, *k*-means and *k*-medoid methods presented the lowest SQD values. Because of these same SQL values, we decided to compare the FEB averages between medoids from these methods and all MD conformations, aiming at clustering solutions with FEB average close to the MD trajectory. Therefore, Complete and Ward's are methods that presented FEB average values closer to the MD trajectory, i.e. -6.59 for Complete and -6.60 for Ward's against -6.58 from the MD trajectory. As the difference between average and standard deviation from these methods were minimal, we decided to select the experiment with the lowest number of clusters. We also believe that due to the farthest neighbor method, the clustering solutions were composed by medoids belonging to compact clusters of approximately equal diameters. Hence, the partitioning with 48 clusters from Complete method was the optimal solution selected for the MD trajectory in study.

Besides identifying clustering solutions with high binding affinity in the snapshots within the groups, we also analyzed the level of similarity in docking final poses based on the RMSD accumulated variance assessments. These results are in accord with the SQD values indicating that the optimal clustering solutions selected for both Complete and Ward's methods also contains low RMSD variances in the clustering generated. It can be seem from graphs (c) and (d) in Figure 3.7 that the lowest RMSD variances were reached when the number of cluster is higher than 40 and 95, for Complete and Ward's method, respectively. Although Ward's method reached means of RMSD accumulated variances lower than Complete method in the clustering generated for the Cavity Attributes data set (0.50 for Ward's against 0.51 for Complete), we decided to select the optimal clustering solution based on SQD values. This is due to the method proposed in this study, which identifies groups of (un)promising snapshots by assessing the binding affinity (FEB values) at docking runtime.

3.6 Chapter Remarks

Several studies explore the relative accuracy of various clustering algorithms in extracting the right number of clusters from generated data [KR90, JD88]. According to Hartigan et al. [HW79], it is impracticable to point out the best clustering method since different approaches are right for different purposes. Chen and Lonardi [CL09] state that the most popular method for clustering MD conformations is agglomerative hierarchical, since its linkage method is able to use the attributes for describing chemical structures. More specifically, linkage is the only method able to calculate the dissimilarities between two clusters of chemical structures using Euclidean distance. Alternatively, Shao et al. [STTC07] found that UPGMA, k-means, and SOM outperformed COBWEB, Bayesian, and other hierarchical clustering methods by using the pairwise RMSD distance as a measure of similarity. Although our analyses also show hierarchical agglomerative methods as the best choices for all data sets, *k*-means and *k*-medoids algorithms demonstrated to be the least advisable choice for all data sets. Each study has particularities regarding data generation and identification of the best clustering algorithm. Indeed, an appropriate solution depends on a given analysis or application scenario, so data collection, data representation, and cluster interpretation are crucial for selecting a suitable strategy [JD88, WKQ+08].

The purpose of this investigation was to assess the quality in clustering solutions presented by six different clustering methods and three measures of similarities (Protein RMSD, Cavity RMSD and Cavity Attributes) for partitioning MD trajectories. Docking experiments on the FFR model were performed for 20 different ligands with the intention of validating the proposed methodology. In addition to investigating RMSD-based clustering, we also provided a novel measure of similarity. It is based on the following feature from the substrate-binding cavity: pairwise RMSD, volume and number of heavy atoms. We demonstrated that the use of binding cavity properties for clustering MD trajectory was able to generate clusters of snapshots similar to the MD's full trajectory (Figures 3.4 and 3.5), as well as grouping snapshots with high affinity in their docking final poses (Figures 3.6 and 3.7). It is noteworthy that this novel approach for clustering MD trajectories may be used for any structure. The major limitation is the high dependency on the prior knowledge on the target cavity for the protein under study.

The novel set of features used in this study revealed to be a promising solution to identify relevant conformational changes that occur into the substrate-binding cavity along an MD simulation trajectory. We expect that the clustering selected will also contribute to reduce the high computational cost required for performing docking-based virtual screening experiments. The high level of binding cavity similarity within a cluster is a decisive factor for ensuring the accuracy of resulting RFFR models. Next section presents the method proposed to evaluate and rank the different interaction modes between receptor conformations and ligand at docking runtime.

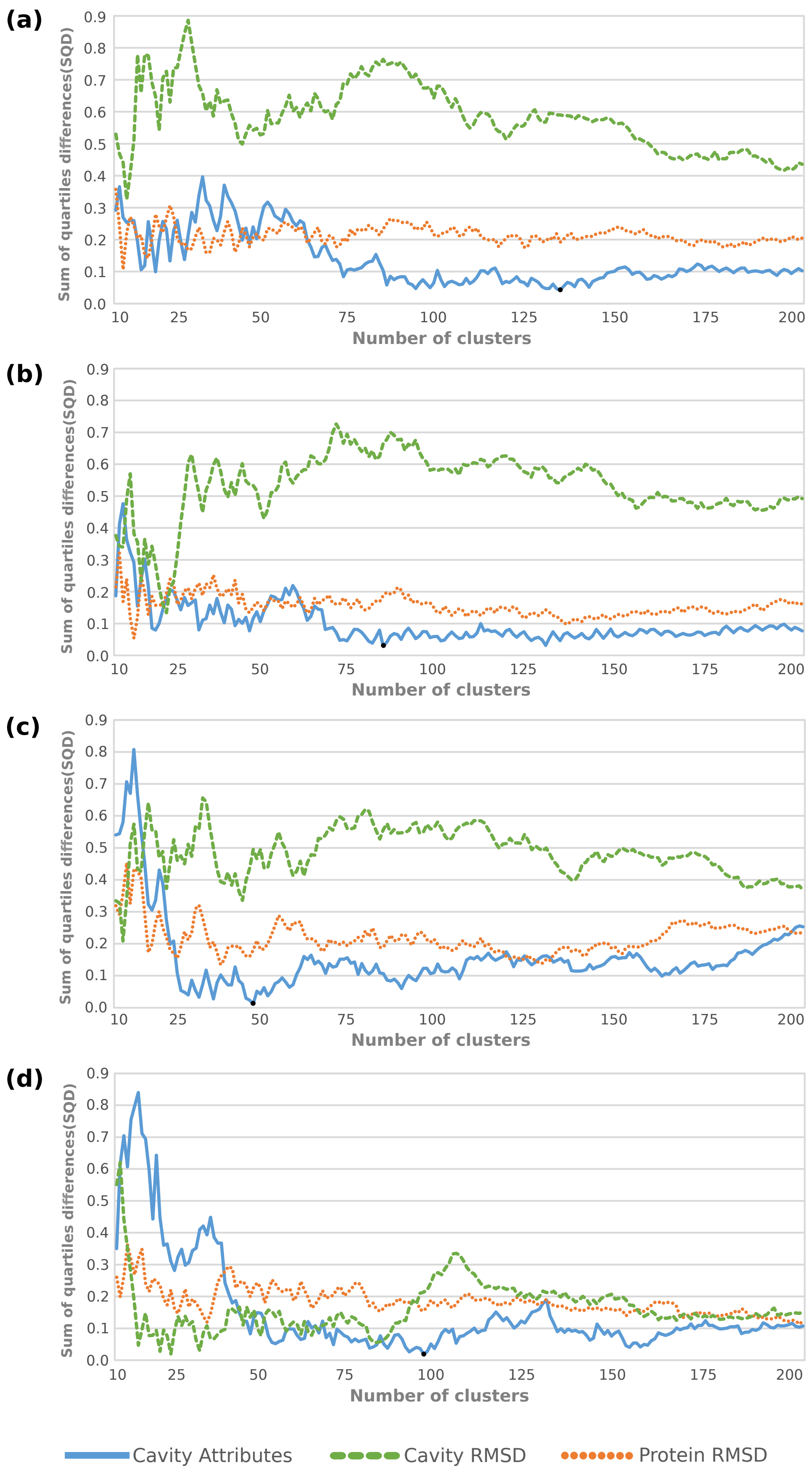


Figure 3.5 – Comparative performance of hierarchical clustering methods for Protein RMSD, Cavity RMSD and Cavity Attributes data sets. SQD variations as a function of number of clusters for, UPGMA, WPGMA, Complete and Ward's methods in graphs (a), (b), (c) and (d), respectively. Black points identify the optimal hierarchical solution for each method [DPQRNdS15].

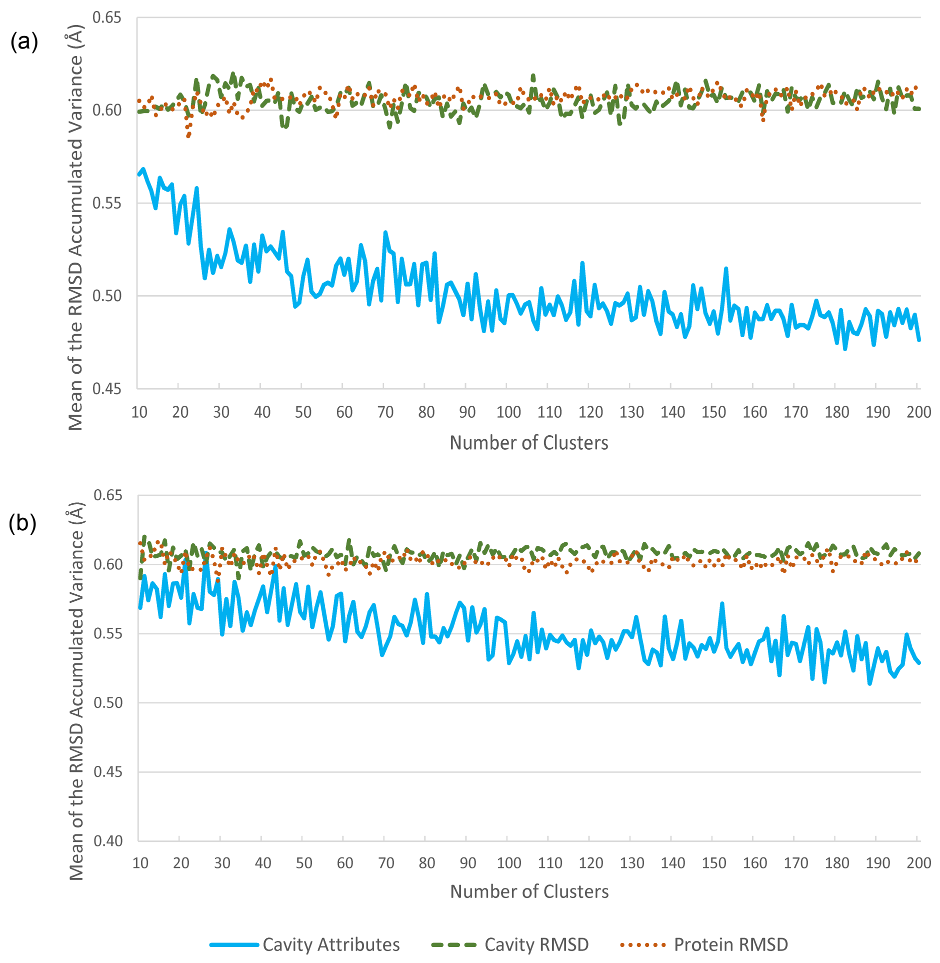


Figure 3.6 – Mean variance of the RMSD values in the partitional clustering methodos for Protein RMSD, Cavity RMSD and Cavity Attributes data sets. The mean of the RMSD accumulated variance as a function of number of clusters for k-means and k-medoid methods are in graphs (a) and (b), respectively. Each data set is represented by a color, where blue is the Cavity Attributes, green is the Cavity RMSD, and orange is the Protein RMSD.

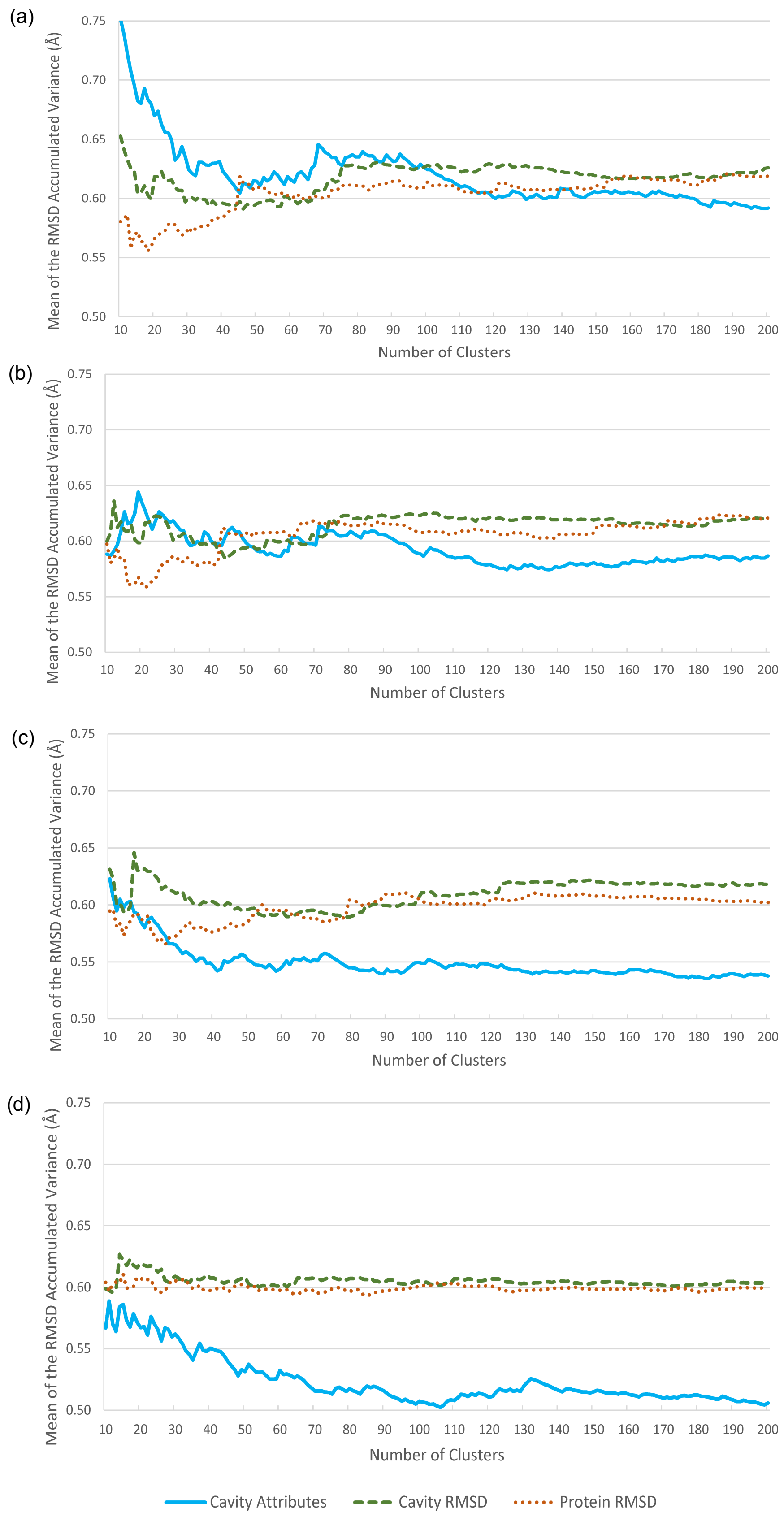


Figure 3.7 – Mean variance of the RMSD values in the hierarchical clustering methods for Protein RMSD, Cavity RMSD and Cavity Attributes data sets. The mean of the RMSD accumulated variance as a function of number of clusters for, UPGMA, WPGMA, Complete and Ward's methods, are in graphs (a), (b), (c) and (d), respectively.

4. THE METHOD TO OPTIMIZE DOCKING-BASED VIRTUAL SCREENING IN FFR MODELS

This chapter presents the method developed to reduce the overall time spent in molecular docking simulations of FFR models by discarding groups of unpromising snapshots, using the clustering of snapshots described in the previous chapter. Amaro et al. [AL10] assert that applying new computational techniques to reduce the entire MD ensemble without losing critical structural information is a promising alternative for making the practice of performing virtual screening experiments on MD trajectories computationally tractable. Similarly, Antunes et al. [ADK15] suggest that to predict an ensemble of top-ranked alternative binding modes instead of only one best structure is a promise strategic to reflect a more realistic exploration of the binding event in the flexibility of the protein-ligand complex. In this regard, we propose a novel method to reduce the dimensionality of FFR models by discarding snapshots belonging to groups that present poor quality in the docking results for a particular ligand. The quality of snapshot groups is determined by evaluating the FEB values from the snapshots processed. A new RFFR model is generated for each ligand by the end of each experiment.

The chapter starts by describing the Pattern Self-adaptive Multiple Instances (P-SaMI), whose is the basis to the method proposed. Following, we present the proposed method and its procedure to perform selective ensemble docking experiments. This chapter ends with a set of empirical tests to support the prediction of the top-ranked alternative binding modes accounting for FFR models.

4.1 Self-adaptive Multiple Instance (P-SaMI) Data Pattern for Scientific Workflows

P-SaMI is the data pattern formalism for scientific workflows developed by Hübler [Hüb10, HRFNdS15]. It identifies (un)promising conformations in different groups of snapshot and determines whether a specific group can be discarded during a docking experiment. The main objective of P-SaMI is to eliminate the exhaustive execution of molecular docking simulations of FFR models, keeping a fairly good level of accuracy in the resulting models (the RFFR models). P-SaMI procedures split clusters of snapshots into batches before starting the experiment and, while receptor conformations are progressively submitted to docking executions in a computing resource, the batches of (un)promising snapshots are identified. Figure 4.1 illustrates this model.

As shown in Figure 4.1, P-SaMI investigates FFR-ligand docking results and decides dynamically whether to take action based on a quality criterion. The quality criterion defines a priority and a status per batch. These parameters are assigned after the per-

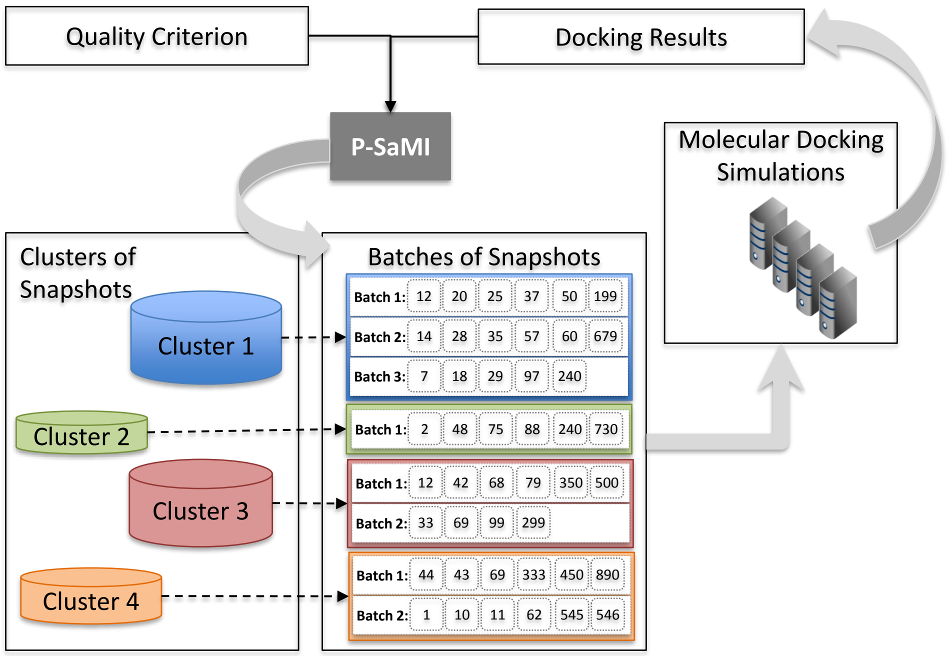

Quality Criterion
Docking Results
P-SaMI
Molecular Docking Simulations
Clusters of Snapshots
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Batches of Snapshots
Batch 1: 12 20 25 37 50 199
Batch 2: 14 28 35 57 60 679
Batch 3: 7 18 29 97 240
Batch 1: 2 48 75 88 240 730
Batch 1: 12 42 68 79 350 500
Batch 2: 33 69 99 299
Batch 1: 44 43 69 333 450 890
Batch 2: 1 10 11 62 545 546

Figure 4.1 – Model of P-SaMI data pattern operation. The clustering of snapshots is split into batches before starting the docking experiments on an HPC environment. The docking results (FEB values) and the quality criterion dictate the priority and status of the batches. Adapted from [DPFNdSR13].

centage of snapshots have been docked, from which the best FEB values are extracted. A set of parameters should be configured before staring the experiments to ensure the proper P-SaMI operation. The four P-SaMI parameters are:

- Sample size: the percentage of snapshots from a group to be added in a batch.
- Start analyses: the minimal percentage of snapshots required by P-SaMI to start the analysis of the batches.
- FEB maximum value: the worst FEB value of a ligand to its flexible receptor.
- FEB minimum value: the best FEB value of a ligand to its flexible receptor.

The last two parameters are determined by the domain expert, who has the knowledge about receptor-ligand interactions needed to provide such information. P-SaMI uses the midpoint average between the worst and best FEB values to assess the quality of a batch and assign a priority. An active batch varies its priorities on a scale of 1 to 3 (1 meaning low-priority and 3 high-priority), whereas priority 0 indicates a discarded batch.

Priority indicates how promising the snapshots are belonging to a batch and dictates whether a snapshot should be sent into a job queue or not. For instance, if batch B_{11} contains snapshots with bad interaction with the ligand L_1, it receives low priority, fewer slots in the job queue, and, eventually, B_{11} can be discarded before the end of the whole execution. Conversely, if a substantial amount of conformations from B_{21} presents favorable interaction with that same ligand L_1, it is assumed that B_{21} contains promising snapshots, being credited with a high priority on the job queue.

Quevedo et al. [QDPRNdS14] introduce the first experimental application of P-SaMI in a larger FFR model (a 20 ns MD trajectory). They proposed to perform a representative selection of docking poses using clustering methods to significantly reduce the high number of MD conformations in the resulting RFFR models. The set of representative poses was generated by clustering RMSD values, which were extracted from the crystallographic reference poses of the ligands, using *k*-means algorithm. The major benefit of this selection of representative poses is to reduce or eliminate the demanding task of domain experts to manually select promising snapshots. However, due to the measure of similarity used for clustering the docking poses, this representative selection has two noticeable limitations. First, it should be applied only when the crystallographic structure is known in advance, since the RMSD values were extracted based on a referential ligand position. Second, RMSD docking results may be imprecise for FFR models because conformational changes from a given snapshot may be different from another. The reference ligand is generally placed in the first conformation of the FFR model. The constant conformational changes between 500 ps and 20,000 ps can be evidenced in Figure 3.2.

In addition to selecting a significant quantity of false positives, P-SaMI also demands previous knowledge on the receptor-ligand complex. This means that pre-defined parameters are used to prioritize groups of promising snapshots. According to Hübler [HRFNdS15], the average between the best and worst FEB is the value used to identify groups of promising snapshots. Best and worst FEB values are usually determined by the expert domain. However, the accurate designation of these values is not a trivial task. The two limitations are: (i) the domain expert does not know the expected fitting of all set of ligands; (ii) the use of a single MD conformation to discover potential ligand poses, which is the most usual technique, is unable to represent all alternative flexible parts of the FFR model.

Cozzini et al. [CKS$^+$08] state that it is not known in advance which conformations the target flexible structure will adopt in response to the binding of a particular ligand. To obtain a more realistic worst and best FEB values for two ligands, Quevedo et al. [QDPRNdS14] performed docking experiments into a sample of 1% from the 20,000 snapshots of the FFR model. This is a suitable technique to reveal the representative behavior that a ligand may assume in the entire FFR model. A clear disadvantage of this manually pre-processing step is the required time, which increases considerably as the number of ligands rises. Such scenario occurs when one intends to perform virtual screening in large libraries of small molecules, where for each ligand a total of 200 docking experiments for the FFR model under study should be executed and analyzed. Furthermore, this quantity of docking experiments may be even greater if the FFR model dimensionality increases.

To address the above limitations, we proposed a method capable of:

1. predicting the behavior of FFR models and ligands automatically, and;
2. determining a criterion to assess the quality of groups based on a sample of docking results.

4.2 Input Parameters

The input parameters for the method proposed are based on the P-SaMI data pattern model. A set of specifications is determined to prepare groups of snapshots and control the docking experiments. The original P-SaMI model operation remained unchanged, but some parameters were modified with the intention of improving quality and reducing the time consumed by ensemble docking experiments. The set of P-SaMI specifications consists of eight input parameters, five for preparing snapshots and three for evaluating and handling docking experiments. These parameters are detailed below.

1. Clustering of snapshots. The input file with all MD receptor conformations grouped according to some similarity metric. This is the optimal clustering solution presented in Chapter 3.
2. Analyses by cluster or batch. Determines whether the quality of docking results will be evaluated by cluster or by batches. The latter splits clusters of snapshots into subgroups and enables the next three parameters.
3. Percentage of snapshots in a batch sample. The portion of snapshots taken from a cluster to create a new batch, considering the next two parameters.
4. Minimum quantity of snapshots per batch. It is the minimum number of snapshots required to create a batch. If a batch has fewer snapshots than the amount defined, the snapshots are added to the previous ones.
5. Maximum quantity of snapshots per batch. It is the maximum number of snapshots that a batch may contain. If a batch is overloading, the extra snapshots are returned to the original cluster and rearranged to another batch. This parameter aims at generating a well-balanced distribution of snapshots among batches to enhance the level of accuracy in docking analyses.

Analysis per batch or cluster allows specifying the evaluation method for a group of similar snapshots. In this study, only analyses per batch are employed since Hübler [Hüb10] identified better quality results when small samples of snapshots from equal groups are gradually processed by P-SaMI data pattern. Thus, the three parameters used to assess the quality of batches during the docking experiments are:

1. Percentage threshold to start analyses ($startAnal$). In the original P-SaMI data pattern model, analyses start when the percentage of processed snapshots from a batch reaches this parameter. Alternatively, in this study, analyses start when the percentage of processed snapshots from all batches reach this parameter. This is due to the new strategic function, which defines priorities based on samples of docking results from all batches.

2. Percentage threshold to discard a batch ($discB$). Determines when a batch with low priority is unable to increase the priority, and therefore, the remaining snapshots may be discarded. Hence, if low quality docking results are reached after processing the defined percentage of snapshots, the batch is considered unpromising and the docking stops.

3. Percentage threshold to discard a cluster ($discC$). Determines when a cluster with discarded batches is unable to reach better docking results, and therefore, the remaining batches may be also discarded.

The purpose of the last item is to identify unpromising snapshots by analyzing the quality among batches within a cluster. Specifically, when a cluster *c1* reaches a percentage of batches with low quality docking results (i.e. percentage to discarded batches), cluster *c1* is considered unpromising and its batches stop running docking experiments. This discarding is applied for all batches with low priority (i.e. less than 3). If an unpromising cluster contains batches with high priority, these batches are preserved and further docking results will be used to analyze their quality.

4.3 Approach Developed to Perform Selective Ensemble Docking Experiments

To reduce systematically the ensemble docking experiments and outperform the accuracy in the RFFR models produced by the method proposed, improvements were performed in the P-SaMI data pattern. This approach attempts to find the best receptor conformations for a ligand without prior information about the interactions between the FFR model and a specific ligand. Favorable binding modes are discovered and ranked during the docking experiments, based on predicted FEB values extracted from snapshots already processed. The approach developed to perform selective ensemble docking experiments is divided into preprocessing and processing stages. Procedures of both stages are described in this section. An overview of the schematic process is given in the flowchart shown in Figure 4.2.

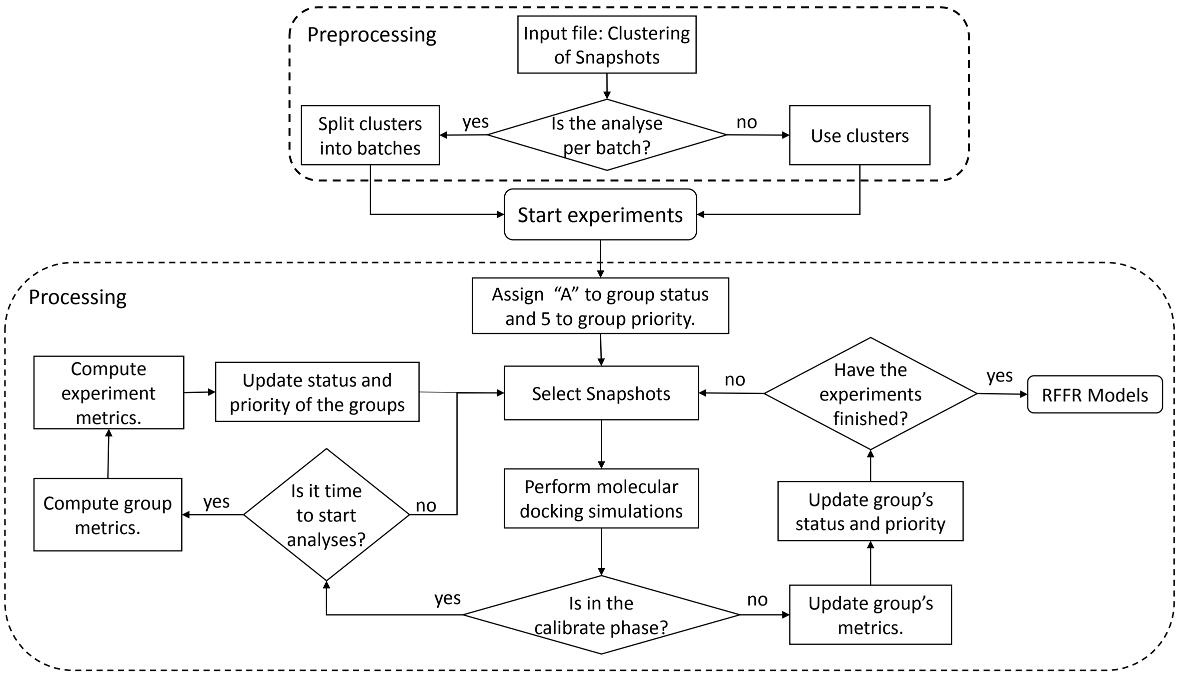

Preprocessing
Input file: Clustering of Snapshots
Is the analyse per batch?
yes
no
Split clusters into batches
Use clusters
Start experiments
Processing
Assign "A" to group status and 5 to group priority.
Select Snapshots
Have the experiments finished?
yes
no
RFFR Models
Compute experiment metrics.
Update status and priority of the groups
Compute group metrics.
Is it time to start analyses?
yes
no
Perform molecular docking simulations
Update group's status and priority
Is in the calibrate phase?
yes
no
Update group's metrics.

Figure 4.2 – Strategic method for performing selective ensemble docking experiments. A group can be a cluster or batch of snapshots. The processing stage comprises the calibration and the batch analyses phases. Calibration phase is the process of quantitatively defining interactions between a sample of MD conformations and a ligand. Batch analyses phase is the process of qualitatively identifying the best protein-ligand complex predictions.

An experiment is created when a clustering of snapshots and a ligand are submitted as input for the docking experiments. Each batch of snapshots contains a status and priority, used for handling which snapshots will perform molecular docking simulations. Thus, the priority indicates how promising a group of snapshots is on a scale from 0 to 5 (5 being the most promising), whereas status denotes one of the following five possibilities:

1. Active (A): the batch contains pending snapshots, which will be docked according to its priority.

2. Calibrate phase (C): the batch is waiting for other batches to reach the defined percentage to start the analyses. If a batch has this status, its priority is 0 and pending snapshots are blocked until all batches from the same experiment reach the percentage of docked snapshots.

3. Priority changed (P): the batch increased or decreased its priority and it contains snapshots waiting to be processed.

4. Discarded (D): the batch was discarded and the pending snapshots will no longer be processed. This status indicates a batch with unpromising snapshots.

5. Finalized (F): the batch processed all snapshots because it is more likely to have promising snapshots.

In this approach, when a docking experiment is submitted to be executed, all batches receive status "A" and priority 5. The highest priority was used in this stage to accelerate the conclusion of the experiment calibration. Status "C" and priority 0 are assigned to the batches that reach the percentage threshold to start analysis during the calibration phase, but they are waiting for all batches achieve the same percentage of processed snapshots. After all batches reach the percentage threshold to start analysis in a experiment, the batch's status changes to "A" and a priority is assigned for each batch based on FEB results of the processed snapshots. The selective ensemble docking operation starts by selecting batches of snapshots with status equal to "A" and uses the priority to dictate the order in which the batches are processed. The higher the priority, the sooner the processing and the larger portion of snapshots selected. An experiment ends when all batches hold status equal to "D" or "F". Unless in the calibrate phase, which uses the maximum priority to reach the percentage of processed snapshots to start analyses faster, the promising batches are those with the higher priority.

A set of metrics are computed while the snapshots are executed, which are: percentage of processed snapshots (ps_i), sampling average ($\overline{x}_i$), estimated average ($\overline{ex}_i$), sampling lower quartile (lq_i), sampling 13th percentile (p_i), and sampling minimum value (min_i). During the calibrate phase only the processed snapshots and sampling average are computed per batch. The sampling average is given by

$$\overline{x}_i = \frac{1}{n_i} \sum_{x_k \epsilon B_i} x_k \tag{4.1}$$

where i denotes the i-batch, x_k is the best estimated FEB value achieved from a docking experiment between a snapshot and a ligand, and n_i indicates the number of processed snapshots from batch B_i. In the calibrate phase the percentage of processed snapshots per batch is verified to update metrics and start docking analyses. When all batches of snapshots from an experiment reach the percentage to start analyses, the batch metrics are first computed, followed by the experiment metrics (Figure 4.2). The set of experiment metrics, which are computed when the percentage to start analyses is reached for all batches once, is the criteria used to categorize the quality of batches.

The estimated average is defined by Hübler et al. [HRFNdS15] as

$$\overline{ex}_i = \frac{1}{n_i}(\sum_{x_k \epsilon B_i} x_k + (0.4985 \times r_i \times (2\overline{x}_i - s_i))) \tag{4.2}$$

where

$$s_i = \sqrt{\frac{1}{n_i - 1}(\sum_{x_k \epsilon B_i} (x_k - \overline{x}_i)^2} \tag{4.3}$$

where n is the amount of snapshots in batch i, r is the set of remaining snapshots to be processed from batch B_i, x_k is the best FEB value for each snapshots from batch B_i, and $\overline{x}_i$ is the sampling average (Equation 5.1).

The sampling lower quartile, sampling 13th percentile, and sampling minimum value were computed with the aim of predicting more accurately binding affinity in protein-ligand complexes. With this metrics, it is expected to outperform the quality in the RFFR models produced not only by considering the FEB values average, but also by identifying the snapshots that account for at least 25.00% more negative FEB values of a batch.

The sampling average, the sampling lower quartile, the sampling 13th percentile, and the sampling minimum are structured into four vectors to compute the experiment metrics, where each vector's position contains the batch identification. Figure 4.3 depicts in a vector the five metrics computed for each experiment based on the four metrics computed ($\overline{x}_i$, lq_i, p_i, min_i) for the batches. The five metrics of an experiment (Exp) are formally represented as follows

$$Exp(M_{\overline{x}}, LQ_{\overline{x}}, LQ_{lq}, LP_p, LQ_{min}) = \sum_{i=1}^{N}(\overline{x}_i, lq_i, p_i, min_i) \tag{4.4}$$

where i denotes the i-batch, $M_{\overline{x}}$ is the median computed from the sampling average of the experiment batches, $LQ_{\overline{x}}$ is the lower quartile computed from the sampling average of the experiment batches, the LQ_{lq} is the lower quartile computed from the sampling lower quartile of the experiment batches, the LP_p is the lower quartile computed from the sampling 13th percentile of the experiment batches, and LQ_{min} is the lower quartile computed from the sampling minimum value of the experiment batches. The set of metrics, status and priority of each batch are updated while the snapshots are executed.

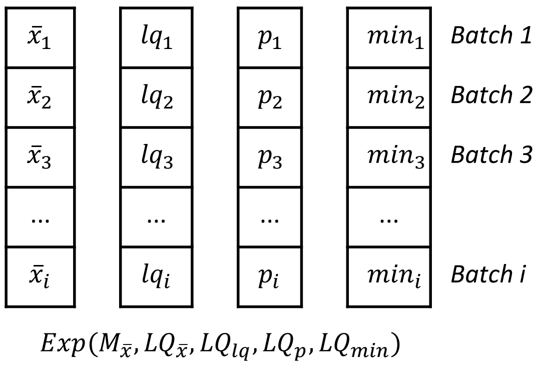


Figure 4.3 – Representation of the experiment and batch metrics computation. A vector representation is generated for each experiment after reaching the percentage to start analyses. Experiment metrics, which are defined after calibrate phase, are the parameters for selecting and sorting the docking experiments.

Algorithms 4.1 and 4.2 are the two function versions implemented to determine the priority in different batches of snapshots according to the level of quality produced at docking runtime. Both functions were developed on the basis of batches that contain the lowest FEB values, accounting simultaneously for favorable protein-ligand interactions, high priority, as well as larger number of processed snapshots. To enhance the quality of RFFR models and identify a larger number of unpromising groups of snapshots, we developed two function versions: batch and cluster analysis functions. The first (Algorithm 4.1) investigates the internal batch quality based on the following assumption: if a batch is unable to reach a specific level of priority after processing the percentage threshold to discard a batch ($discB$), then it is discarded or its priority is penalized by a decline of 2 levels, depending on the outcome of comparative statistical tests on accuracy. Alternatively, cluster analysis function (Algorithm 4.2) investigates both batch and cluster quality, based on the following assumption:

- if a cluster c reaches the percentage threshold to discard a cluster ($discC$), and;
- the batches from c has priority less than 3, then the remaining batches from c are discarded.

We noticed that the cluster analysis function without the percentage threshold to discard a batch ($discB$) processed a significant quantity of unpromising snapshots (i.e. batches with priority 1 or 2) during the empirical tests. For this reason, the ($discB$) parameter was inserted in the Algorithm 4.2 to operate in the same manner as Algorithm 4.1. Thus, batches that are unable to achieve high levels of priority after processing a certain percentage of snapshots are discarded.

We opted for creating a second function version to compare the different levels of analyses, i.e. batch and cluster. By evaluating the quality of clusters, we expect not only to find similarity among snapshots from the same batch, but also among snapshots from different batches belonging to the same cluster. If this approach does not succeed, we can assume: (i) the clustering quality is poor, or (ii) the clustering may be suitable for some ligand and unsuitable for others. Next section presents empirical experiments to compare the two distinct functions and propose an appropriate parametrization to run each function based on a set of docking results between the InhA FFR model and 16 known ligands.

4.4 Evaluating Empirical Experiments to Select a Suitable Parametrization for the Proposed Method

In this section, we describe the empirical evaluation performed to optimize docking-based virtual screening in FFR models. These experiments were divided in two steps. First, we generated batches by picking snapshots sequentially and randomly to assess the level of similarity between the cluster and its batches. In the sequential selection the snapshots were picked in the ascending order, i.e. as they were generated by the MD simulation, whereas in the random selection they were selected randomly. After finding an affordable selection to generate the batches of snapshots, we performed empirical experiments by using batch and cluster analysis functions (Algorithms 4.1 and 4.2) with six parameters in order to choose a more suitable parametrization. All empirical experiments were conducted based on exhaustive docking experiments between the FFR model and a set of compounds experimentally tested from the InhA structure.

The docking results used to evaluate these experiments were taken from the dataset employed to validate the data partitions presented in Chapter 3. A total of 16 ligands with 20,000 docking results each were used to execute the empirical experiments. The ligands complexed with the NADH coenzyme (PTH-NAD and INH-NAD from 2NTJ, 2IDZ, 1ZID and 2NV6 enzymes) were not used in this set of experiments because the proposed methods aims to perform docking experiments of the FFR model and ligands from database of small molecules. The FFR model was generated from an MD simulation of the InhA-NADH enzyme complex and the presence of the NADH coenzyme in the InhA substrate-binding cavity is essential to find a ligand capable of inhibiting the mycobacteria growth [QDS+96]. Thus, the predicted FEB values between the FFR model and the set of ligands from Figure 3.3 were also extracted in order to assess: (i) the level of similarity from both batches and their clusters; and (ii) the level of quality in the RFFR models by using the *settingPriority1* and *settingPriority2* functions with six parameters each.

The level of homogeneity between a cluster and its batches was measured when the snapshots were picked from clusters in a sequential and random manner. To evaluate this level of similarity, the percentage of batch sampling, the minimum and maximum quantity of snapshots to split clusters into batches used for both sequential and random snapshots selection, were 20, 50 and 150, respectively. Although these parameters can be modified according to the user purposes, we suggest to create batches with similar quantity of snapshots, mainly due to (i) the uneven distribution of snapshots into clusters, whose total number of snapshots ranges from 11 to 3,640; and (ii) the statistical analyses performed among batches, which demands small number of objects inside a group to indicate a more realistic dispersion over data. Based on these parameters, a total of 247 batches were generated, and the similarity of each batch and its corresponding cluster was compared based on FEB values obtained from exhaustive docking experiments of the FFR model with a set of 16 ligands. The FEB average difference between clusters and their respective batches from the sequential and random selection were computed as follows:

$$AvgDiff(C_i, B_j) = \sum_{\overline{x}_c \epsilon C_i} \sum_{\overline{x}_b \epsilon B_j} \mid \overline{x}_c - \overline{x}_b \mid \qquad (4.5)$$

$$\overline{x}_c = \frac{1}{n_i n_j} \sum_{i=1}^{l} \sum_{x_k \epsilon C_{i,j}} x_k \tag{4.6}$$

$$\overline{x}_b = \frac{1}{n_i n_j} \sum_{i=1}^{l} \sum_{x_k \epsilon B_{i,j}} x_k \tag{4.7}$$

where l is the total of ligands, x_k is the FEB value from the i^{th} ligand and the snapshot in the j^{th} cluster for $\overline{x}_c$ and the FEB value from the i^{th} ligand and the snapshot in the j^{th} batch for $\overline{x}_b$. Similar equation was used to measure the standard deviation difference ($StdDiff(C_i, B_j)$). Figure 4.4 shows the significant difference between the sequential and random distribution of clusters into batches. Random selection clearly showed batches more homogeneous and similar to their clusters, while sequential selection was more unsteady. These findings suggest that batches composed of snapshots taken in a sequential manner present a substantial difference in the binding modes among these batches and their original clusters. In contrast, batches created by a random selection are able to obtain better levels of homogeneity in the snapshots that form the same batch, as well as high similarity of binding modes among batches from the same cluster. The average of sequential and random selections were 0.14±0.12 kcal/mol and 0.03±0.03 kcal/mol for $AvgDiff$, and 0.07±0.05 kcal/mol and 0.02±0.02 kcal/mol for $StdDiff$.

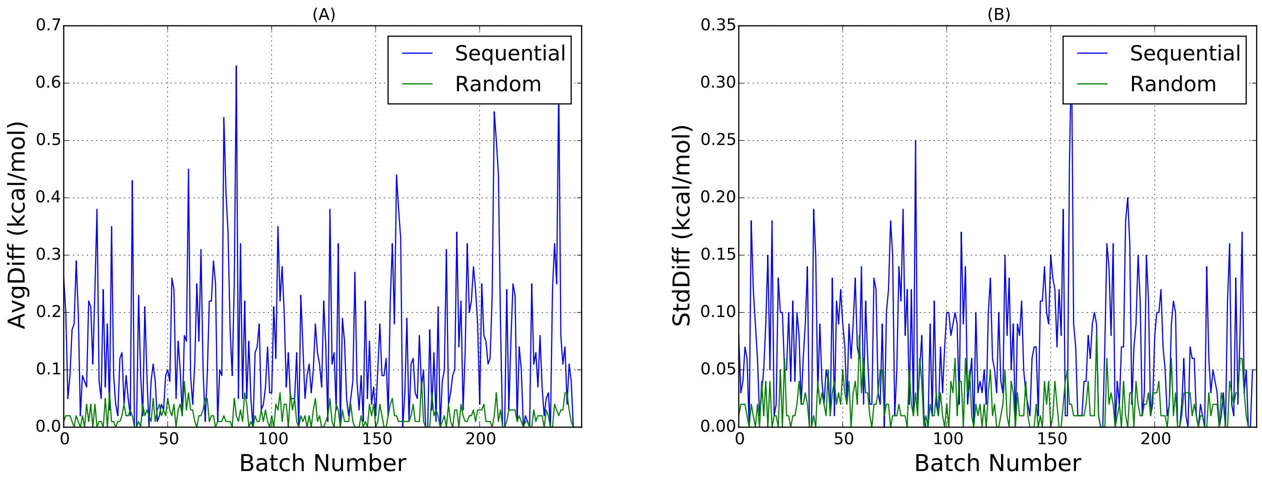


Figure 4.4 – Comparative level of homogeneity between batches and their clusters for sequential and random snapshots selection. Average differences of the FEB average and standard deviation as a function of the number of batches are shown in graphs (A) and (B), respectively.

Both sequential and random selections were applied to pick snapshots during the preparation and execution of experiments. Sequential selection was the strategy used by the batch analysis function and random selection was the strategy used by the cluster analysis function. Since random selection showed batches with better level of similarity with their clusters, we expect that it is also an appropriate strategy for finding promising snapshots during docking runtime.

The suitable parametrization for the proposed method was chosen considering the percentage of best snapshots interactions selected for each set of assigned values. Based on exhaustive docking ensemble results, the snapshots which are in the top 10, 20, 30 and 100 best FEB values were ranked for each ligand, aiming at comparing the accuracy from the RFFR models produced. In this assessment, we evaluated the percentage means obtained by the processed snapshots and four top best rankings, when a different set of values were assigned to the method's parameters. A total of 30 different combinations of values were assigned to the method, half of which for the batch analysis function and half for the cluster analyses function. Parameters and their values were combined for each function as follows:

- Batch analysis function (sequential selection): 10, 15 and 20 for percentage to start analyses, and 30, 40, 50, 60 and 70 for percentage to discard a batch parameter.
- Cluster analysis function (random selection): 10, 15 and 20 for percentage to start analyses, and 10, 20, 30, 40, 50 and 60 for percentage to discard a cluster parameter. The percentage to discard a batch, which is used for batches with low priorities (1 and 2) is 50.00% for all experiments.

As snapshots from the cluster analysis function were randomly selected, a total of 20 executions were performed for each different values combination. The purpose of running the same method's parametrization several times is to ensure the generalization of results when different snapshots are chosen by chance. It means that we expect to avoid biased results by reaching equal performance level in every cluster analysis configuration. Appendix B displays a complete investigation from the results obtained by executing the cluster analysis function. In this investigation, the average percentages of the 20 executions were computed for processed snapshots and the top 10, 20, 30 and 100 best snapshots per ligand. A similar investigation is presented to the batch analysis function in Appendix A. Both appendixes show tables of performance assessments for each ligand. However, unlike cluster analysis function, snapshots from batch analysis function were selected sequentially, and therefore, a unique execution was performed for each method's parametrization.

The results obtained from empirical experiments of the method to optimize docking experiment in FFR models are summarized in Figures 4.5 and 4.6. These figures display the performance percentage means of 16 ligands for each method's parametrization based on percentage values from Appendixes A and B. It is clear that there were better similarities among four performance percentages from cluster analysis than batch analysis regardless of method's parameterizations. Figure 4.5 indicates that percentage of processed snapshots increased over different percentages to discard batches, compared with cluster analysis function (Figure 4.6), with percentages of processed snapshots ranging from 8.00 and 12.00% for cluster analysis function, and 6.00 and 8.00% for batch analysis function.

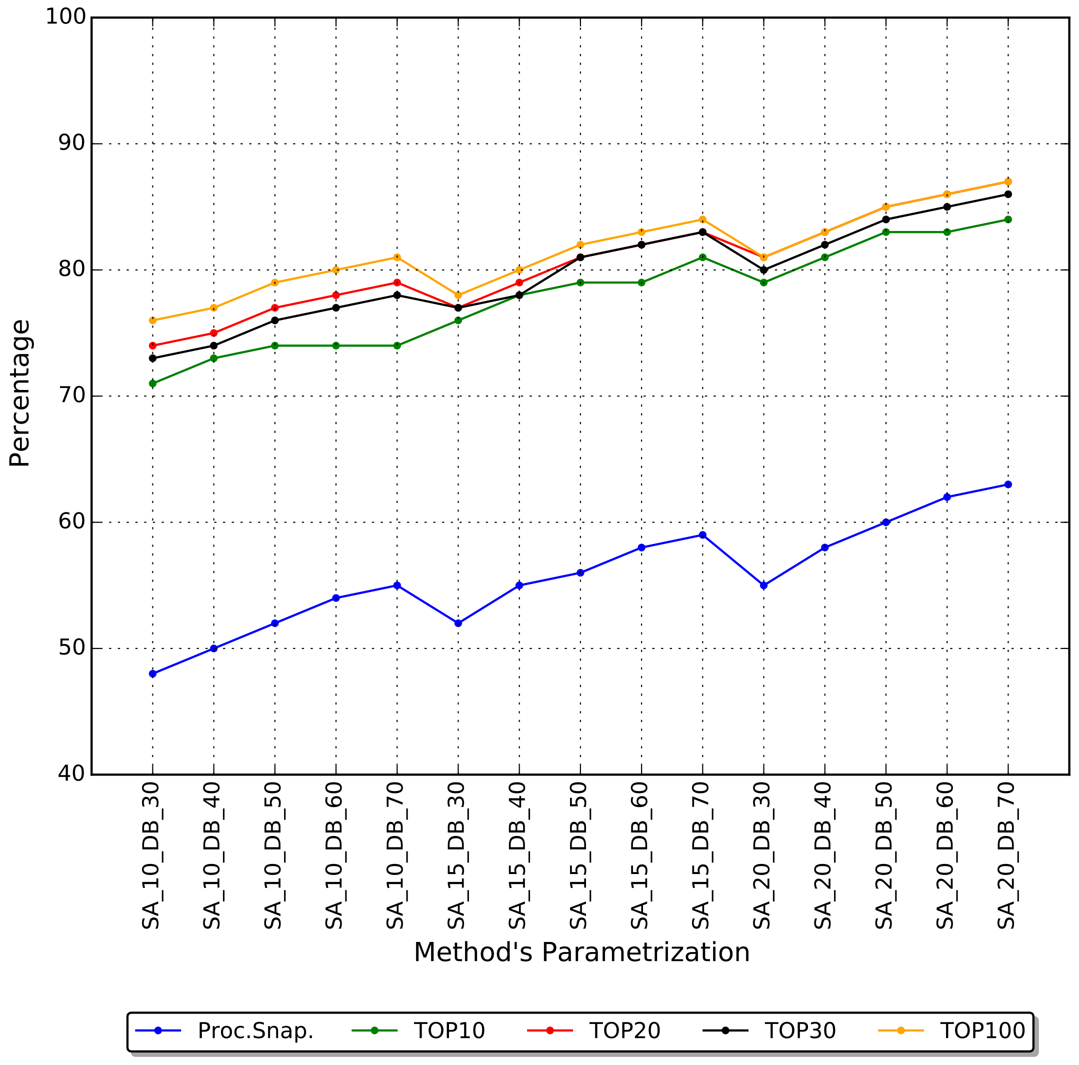


Figure 4.5 – Performance analyses of the empirical experiments applying different parameterizations for the batch analysis function. TOP10, TOP20, TOP30 and TOP100 identify the best 10, 20, 30 and 100 snapshots and ligand interactions. Each method's parametrization is encoded as SA_XX_DB_YY, where XX is the percentage of processed snapshots to start the analyses (SA), and YY is the percentage of processed snapshots to discard a batch (DB).

Figure 4.5 demonstrates that TOP100 selected high percentages of best snapshots in all cases. Conversely, TOP10 selected the worst percentages and obtained the biggest differences from the TOP100 percentages. TOP20 and TOP30 showed better similarity over different method's parametrization. The results obtained from batch analysis method revealed that sequential and centralized selection of snapshots were unable to represent the real binding behavior among the snapshots inside a batch. This is evidenced by comparing the percentage of top best snapshots selected for each method's parametrization in both functions, where batch analysis function produced biased results. Highest accuracies were presented by cluster analysis function, even when the same percentage of processed snapshots was achieved.

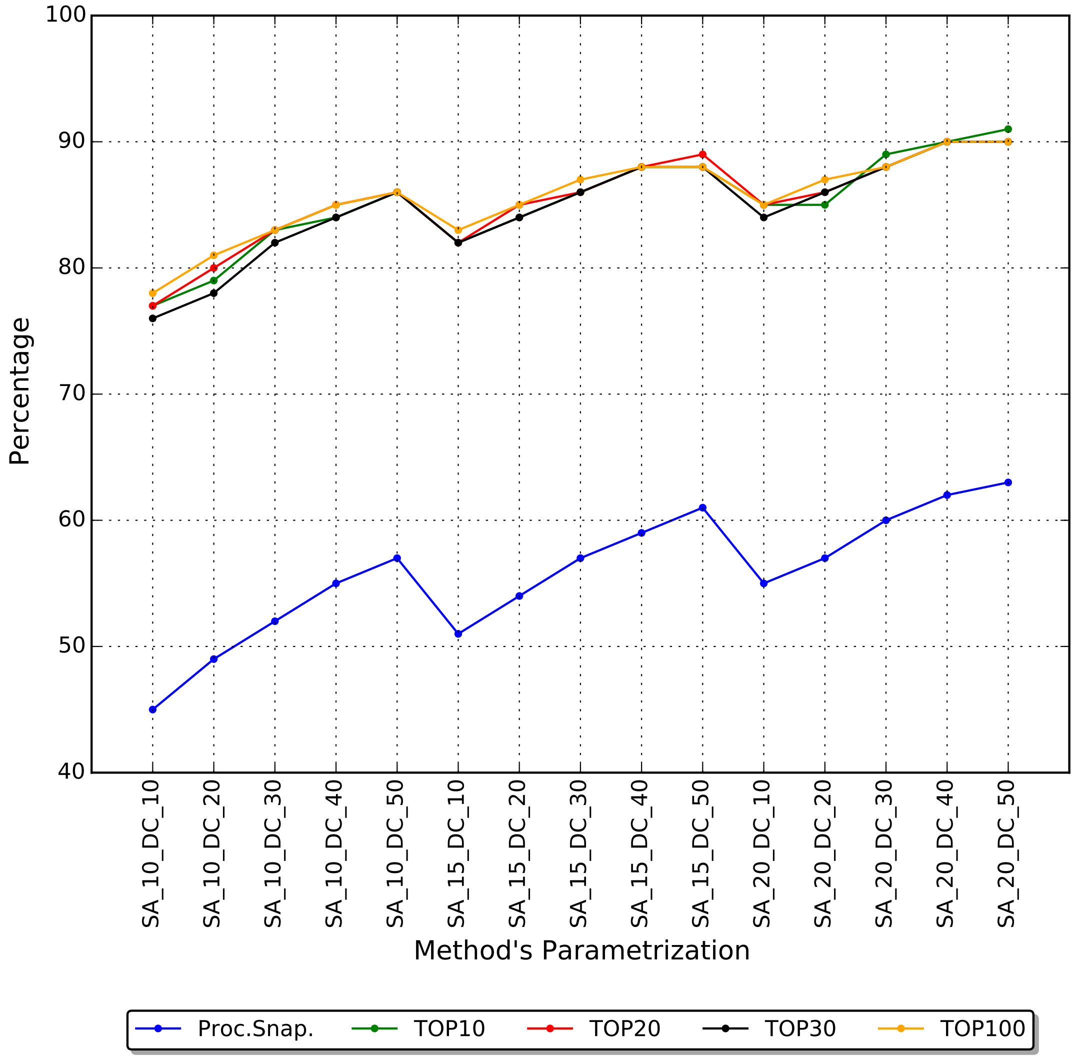


Figure 4.6 – Performance analyses of the empirical experiments applying different parameterizations for the cluster analysis function. Each method's parametrization is encoded as SA_XX_DC_YY, where XX is the percentage of processed snapshots to start the analyses (SA), and YY is the percentage of processed snapshots to discard a cluster (DC).

Since the performance obtained by the empirical experiments was heavily dependent on the function and parametrization chosen, we expect to achieve similar result quality when different ligands are used to perform docking-base virtual screening experiments in the cloud-based workflow environment. To corroborate this expectancy, a parametrization for each function was chosen based on the comparative performance showed in Figures 4.5 and 4.6. As the batch analysis function presented biased results and we were not expecting to obtain high levels of accuracy, we opted for selecting a parametrization that halved the quantity of snapshot of the FFR model. Thus, the percentages for batch analysis function were: 10.00% to start the analyses and 40.00% to discard a batch. By executing this function in the cloud-based workflow environment under these conditions, it is expected to process around 50.00% of snapshots from the FFR model and reach an accuracy of 74.00%.

Contrary to the batch analysis function, results from cluster analysis function showed higher accuracy levels in most of the parametrization tested. To select an option, we considered parametrization that processed the lowest percentage of processed snapshots, and achieved good levels of quality. Hence, based on the empirical experiments, the satisfactory parametrization adopted for cluster analysis function was: to start analysis after 15.00% of snapshots have been processed, and to discard a cluster when 20% of its batches have been discarded. At the end of the experiments, we expect to process around 54.00% of snapshots from the FFR model and reach accuracy of 85.00%.

It is worth mentioning that the configurations chosen for each method are not the only option to set the cloud-based workflow environment purposed in this study. Empirical experiments, which were based on docking results between the FFR model and 16 ligands, were performed to investigate the proposed functions, and identify a suitable configuration. The parameters may be changed depending on different factors, such as time available for executing, accuracy expected, number of parallel executions, budget and other purposes.

4.5 General Remarks

According to Korb et al. [KOB+12], the development of ensemble selection protocols able to identify optimally performing ensembles for different scoring functions and targets will be one of the major challenges in the future. This study contributed to this challenging area of research by proposing a method capable of discarding groups of unpromising snapshots for specific ligands using a clustered FFR model as input data. To validate this new protocol for ensemble selection, empirical experiments were performed by investigating results from exhaustive docking simulations between the FFR model and 16 compounds experimentally tested from the InhA structure. Empirical results revealed that the proposed method is able to achieve high levels of accuracy. This was evidenced by assessing the percentage of processed snapshots, and the best snapshots interactions (up to 100) selected for each ligand when the two analysis functions were executed with different parameters (Appendixes A and B). After validating the proposed method, a parametrization option was suggested for each function based on the performance obtained from the 16 ligands (Figures 4.5 and 4.6). These parameters are used to execute this new ensemble selection method in the cloud-based workflow environment developed in this study.

The method proposed in this chapter outperforms the original P-SaMI data pattern in the following aspects:

- Accuracy: The P-SaMI validation was performed by using FEB values obtained from interactions between a FFR model of InhA with only 3,100 snapshots and two ligands, where one of them is NADH, the coenzyme complexed with the InhA enzyme.

Conversely, our empirical tests used FEB values obtained from interactions between the FFR model of InhA with 20,000 snapshots and a set of 16 ligands experimentally tested.

- Self-contained: Whereas P-SaMI requires that the expert domain provides parameters to identify promising snapshots, our method is able to select promising docking results and reducing the number of docking experiments, without having any previous information on the protein-ligand interactions. Input parameters required by the proposed method are particularly related to the AutoDock software and operating controls in the batch/group of snapshots.

The method presented in this chapter aims at identifying promising snapshots based on the binding affinity produced by screening a large database of organic molecules for putative ligands that fit into the FFR model binding site. Towards this end, the FEB value obtained from docking result was the measure selected for evaluating quality interactions between MD conformations and different ligands. Another measure option would be the RMSD value, which provides the docking poses quality from InhA's known ligands. However, in this study, we intend to exploit opportunities for the discovery of potential new ligands in large libraries of compounds, which in turn contains unknown ligands of the InhA enzyme.

To verify whether the method proposed is able to select snapshots with the best docking final poses, an analysis on the level of accuracy in RMSD values is presented in Chapter 6, which describes the experimental results achieved by performing e-FReDock on cloud platforms. Next chapter provides details on e-FReDock, the cloud-based scientific workflow developed to optimize molecular docking simulations of FFR models and multiple ligands based on the method introduced in this chapter.

```
Let $ps$ be the percentage of processed snapshots from batch $i$.
Let $discB$ be the percentage of processed snapshots to discard a batch in an experiment.
**function** settingPriority1 $(\bar{x}_i, \bar{ex}_i, lq_i, p_i, min_i, ps_i)$
**if** $\bar{x}_i \leq LQ_{\bar{x}}$ **and** $\bar{ex}_i \leq LQ_{\bar{x}}$ **then**
   **if** $lq_i \leq LQ_{lq}$ **and** $p_i \leq LQ_p$ **and** $min_i \leq LQ_{min}$ **then**
      $priority \leftarrow 5$
   **else if** $ps_i \geq discB$ **then**
      $priority \leftarrow 3$
   **else**
      $priority \leftarrow 4$
   **end if**
**else if** $\bar{x}_i > LQ_{\bar{x}}$ **and** $\bar{ex}_i \leq LQ_{\bar{x}}$ **then**
   **if** $lq_i \leq LQ_{lq}$ **and** $p_i \leq LQ_p$ **and** $min_i \leq LQ_{min}$ **then**
      $priority \leftarrow 4$
   **else if** $ps_i \geq discB$ **then**
      $priority \leftarrow 2$
   **else**
      $priority \leftarrow 3$
   **end if**
**else if** $\bar{x}_i > LQ_{\bar{x}}$ **and** $\bar{ex}_i > LQ_{\bar{x}}$ **and** $\bar{x}_i \leq M_{\bar{x}}$ **and** $\bar{ex}_i \leq M_{\bar{x}}$ **then**
   **if** $lq_i \leq LQ_{lq}$ **and** $p_i \leq LQ_p$ **and** $min_i \leq LQ_{min}$ **then**
      $priority \leftarrow 3$
   **else if** $ps_i \geq discB$ **then**
      $priority \leftarrow 1$
   **else**
      $priority \leftarrow 2$
   **end if**
**else if** $\bar{x}_i > M_{\bar{x}}$ **and** $\bar{ex}_i \leq M_{\bar{x}}$ **then**
   **if** $ps_i \geq discB$ **then**
      $priority \leftarrow 0$
   **else if** $lq_i \leq LQ_{lq}$ **or** $p_i \leq LQ_p$ **or** $min_i \leq LQ_{min}$ **then**
      $priority \leftarrow 2$
   **else**
      $priority \leftarrow 1$
   **end if**
**else if** $\bar{x}_i > M_{\bar{x}}$ **and** $\bar{ex}_i > M_{\bar{x}}$ **then**
   **if** $ps_i \geq discB$ **then**
      $priority \leftarrow 0$
   **else if** $lq_i \leq LQ_{lq}$ **or** $p_i \leq LQ_p$ **or** $min_i \leq LQ_{min}$ **then**
      $priority \leftarrow 1$
   **else**
      $priority \leftarrow 0$
   **end if**
**end if**
**return** $priority$
```

Algorithm 4.1 – Batch analysis function for setting priority to the batches of snapshots. The six function parameters represent the following values from batch i: sampling average ($\overline{x_i}$), estimated average ($\overline{ex_i}$), sampling lower quartile (lq_i), sampling 13th percentile (p_i), sampling minimum value (min_i), and percentage of processed snapshots (ps_i)

```
Let $ps$ be the percentage of processed snapshots from batch $i$.
Let $discB$ be the percentage of processed snapshots to discard a batch in an experiment.
function settingPriority2 $(\bar{x}_i, \bar{ex}_i, lq_i, p_i, min_i, ps_i)$
if $\bar{x}_i \leq LQ_{\bar{x}}$ and $\bar{ex}_i \leq LQ_{\bar{x}}$ then
    if $lq_i \leq LQ_{lq}$ and $p_i \leq LQ_p$ and $min_i \leq LQ_{min}$ then
        $priority \leftarrow 5$
    else
        $priority \leftarrow 4$
    end if
else if $\bar{x}_i > LQ_{\bar{x}}$ and $\bar{ex}_i \leq LQ_{\bar{x}}$ then
    if $lq_i \leq LQ_{lq}$ and $p_i \leq LQ_p$ and $min_i \leq LQ_{min}$ then
        $priority \leftarrow 4$
    else
        $priority \leftarrow 3$
    end if
else if $\bar{x}_i > LQ_{\bar{x}}$ and $\bar{ex}_i > LQ_{\bar{x}}$ and $\bar{x}_i \leq M_{\bar{x}}$ and $\bar{ex}_i \leq M_{\bar{x}}$ then
    if $lq_i \leq LQ_{lq}$ and $p_i \leq LQ_p$ and $min_i \leq LQ_{min}$ then
        $priority \leftarrow 3$
    else
        $priority \leftarrow 2$
    end if
else if $\bar{x}_i > M_{\bar{x}}$ and $\bar{ex}_i \leq M_{\bar{x}}$ then
    if $lq_i \leq LQ_{lq}$ or $p_i \leq LQ_p$ or $min_i \leq LQ_{min}$ then
        $priority \leftarrow 2$
    else if $ps_i \geq discB$ then
        $priority \leftarrow 0$
    else
        $priority \leftarrow 1$
    end if
else if $\bar{x}_i > M_{\bar{x}}$ and $\bar{ex}_i > M_{\bar{x}}$ then
    if $ps_i \geq discB$ then
        $priority \leftarrow 0$
    else if $lq_i \leq LQ_{lq}$ or $p_i \leq LQ_p$ or $min_i \leq LQ_{min}$ then
        $priority \leftarrow 1$
    else
        $priority \leftarrow 0$
    end if
end if
return $priority$
```

Algorithm 4.2 – Cluster analysis function for setting priority to the batches of snapshots. The six function parameters represent the following values from batch i: sampling average ($\overline{x}_i$), estimated average ($\overline{ex}_i$), sampling lower quartile (lq_i), sampling 13th percentile (p_i), sampling minimum value (min_i), and percentage of processed snapshots (ps_i)

5. e-FReDock: A CLOUD-BASED SCIENTIFIC WORKFLOW TO OPTIMIZE INTENSIVE MOLECULAR DOCKING SIMULATIONS OF FFR MODELS

One of the major challenges in performing docking experiments of FFR models is the computational demand to screen large databases of small compounds and extract potential binders. According to Amaro et al. [AL10], performing virtual screening experiments in the full set of structures is computationally intractable and likely unnecessary. In the same way, Bier et al. [BZ10] states that both the steady increase of computer power and the implementation of advanced sampling strategies will broaden the applicability of MD simulation methods for pre-selection of relevant conformation transitions. For this reason, we propose a method that selects groups of promising snapshots during the ensemble docking runtime in a cloud-based scientific workflow. The most promising receptor-ligand bound conformations are identified in groups with high level of similarity in their substrate-binding cavities, and consequently, it is expected to find a binding pattern in each of the clusters of snapshots.

This chapter introduces e-FReDock, the cloud-based scientific workflow to optimize intensive molecular docking simulations of FFR models based on the method presented in Chapter 4. This scientific workflow was developed in e-Science Central (e-SC) [HWWC13], the workflow enactment system created by the scalable computing research group based at Digital Institute, Newcastle University, UK. We created a set of blocks in e-SC to design and run the e-FReDock scientific workflow, which can be hosted in public and private clouds. The e-SC platform allows to write blocks in a variety of languages and connect in different database management systems, such as MongoDB [Cho13]. MongoDB is the NoSQL database used to store and manage the large scale of data produced during e-FReDock execution. The e-sc platform also allows to use Azure Tables

A preliminary conceptual architecture of e-FReDock was published earlier as a PhD Consortium on the 2015 IEEE 7th International Conference on the Cloud Computing Technology and Science [DPRNdS15]. The scientific workflow approach introduced in the paper is different from the final architecture presented in this section as it was an ongoing work and some improvements have been made to meet the primary objective of this study. This chapter presents the final e-FReDock conceptual specification developed in e-SC, the blocks created to design the workflows and the data model designed in MongoDB. We also give a brief introduction to the services provide by e-SC workflow enactment system and the concepts of MongoDB database.

5.1 Workflow Enactment System: e-Science Central

The e-SC platform is a cloud-based web workflow enactment system for e-Science projects [HWWC13]. This system, created in 2008, can be deployed on both private and public clouds. The cloud-based platform provided by e-SC includes essential services to support scientists and developers, who can design scientific workflows using available services or building new applications. Figure 5.1 illustrates the set of virtualized services from e-SC on a cloud platform. Each platform service is described below [HWWC13].

- Data storage: all e-SC files are versioned and placed in the cloud storage. These files are services, workflows and user data.
- Processing: a set of VMs are attached to the e-SC server to execute workflow invocations. These VMs can be any worker node in a HPC environment (cloud computing or physical environments) and are also called e-SC engines as they provide an execution environment that can simultaneously support code in different languages.
- Security: workflows, data and services are controlled by the users, which grant different access for specific users or groups. Access control lists are used to store in a database the actions performed by users, which can be read, written, deleted or added.
- Analysis service: e-SC allows users to upload and deploy services into the platform. Services can be written in a variety of languages including Java, JavaScript, R [Tea12] and Octave [Eat16]. A binary library is also provided by e-SC to manipulate other files formats.
- Workflow enactment: to provide scalable performance, the workflow services are stored on the file system of a VM and their instances are sent as a temporary invocation to be deployed on each of a pool of VMs. The file system can be placed in the e-SC server VM or another VM on the cloud.
- Provenance: all system events such as workflow operation, data control, and other interactions are stored in a database. This information can be retrieved by users to reproduce their experiments or extract particular data. Moreover, e-SC allows users to analyze the life cycle of every stage of data and process within the system thought the Open Provenance Model (OPM) [MCF$^+$11], a data model that provides the necessary tools to follow the history of an object.

The e-SC core set of services includes components to design workflow by selecting blocks. A typical workflow in e-SC is composed by blocks (or services) which are connected by arrows to orchestrate the execution flows based on the DAG representation. Each block has any number of inputs and outputs, but only one connection is accepted on any given block input, whereas any number of connections can be accepted on any given block output. The data representation options of input and output files are: table structured format (CSV file), serialized Java object, and a list of files in any format.

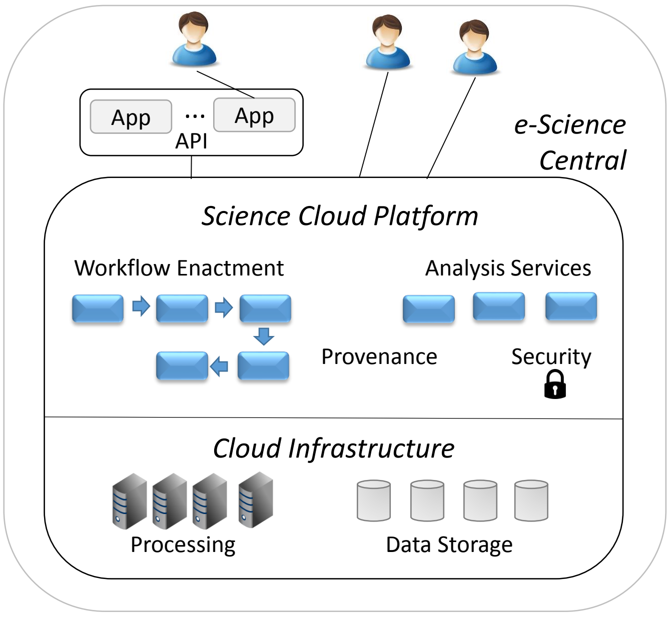


Figure 5.1 – e-Science Central Cloud Platform Architecture. The Application Program Interface (API) allows users to build their external application and execute workflow invocation directly on the cloud platform. Provenance and analysis are collected during the workflow execution and stored into the data storage cloud infrastructure. Adapted from [HWWC13].

The invocation of a workflow enables a sequence of blocks that run when all input ports have input data buffered and ready to use [CHWW13]. A block is a thin layer of an API that allows users to customize their own algorithms, as well as to access input data and properties to generate result invocations. A number of e-SC services is portable and transparent to end users. Code and workflows can be prototyped on a laptop and then executed at scale by transferring it to a hosted cloud environment [HWWC13].

Data, workflows and code are stored within logical folders. Each e-SC user has its own home folders containing the user's designed workflows, blocks developed, and files uploaded or created into the platform. All these operations can be performed by users through a web interface, or using external tools on which blocks are written and deployed into the e-SC platform. User and e-SC data can be stored on the physical disk of the system running e-SC server, or within a data storage service, which allows to store large amounts of data as a persistent storage [1] in public cloud platforms.

Another service provided by e-SC is the Java REST API, which allows developers to build applications directly on the cloud platform. The external API service has the

[1]Persistent storage or non-volatile storage is any data storage device that retains data after power to that device is shut off.

advantage of developing specific functionalities by accessing and manipulating data files and workflows stored in the e-SC file system, which would not be possible to design on the e-SC platform. A further advantage of this external e-SC service is the possibility of making decisions on the data before invoking a workflow engine. This is particularly useful in enabling run-time assessments to identify consistent groups of promising snapshots in molecular docking simulations of FFR models.

The other two main advantages that motivated the use of e-SC to develop the cloud-based environment proposed are its web interface and its application for drug discovery procedure [HWWL10, WLC^{+}11, HWWC13, CHWW13]. Recently, Cala et al. [CHWW13] designed a scalable system based on e-SC for reducing the time taken to generate QSAR models in the Azure cloud platform. By assessing the scalability of prediction of chemical activity experiments, they demonstrated efficiency of 90.00% with 200 work nodes on Azure Cloud. One critical problem of using the e-SC is the lack of documentation for many services, which can be a drawback for new users.

5.2 MongoDB NoSQL Database

MongoDB [Cho13] is the open source NoSQL database used to store the information from molecular docking experiments performed on e-FReDock. It is a non-relational database designed to support bulk processing of great amounts of data in distributed horizontal scaling, e.g. large number of VMs on the cloud [GGM12]. NoSQL systems are typically used to deal with scalability issues, in which relational databases are inefficient [GGM12]. MongoDB supports an easy-to-use protocol for storing large files based on a document-oriented database. The document-oriented approach is a more flexible model to represent a single record, where a row from an SQL table can be replaced by a complex hierarchical structure.

According to Chodorow [Cho13], the five main concepts that distinguish MongoDB from SQL databases are document structure, unique key, collection, multiple independent databases and JavaScript shell. As mentioned above, documents are analogous to rows in a relational database. MongoDB supports many data types. The most commonly used are: integer, double, boolean, string, date, timestamp, JavaScript code, binary data, array, and embedded document or sub-documents. Embedded document is a vector field composed of a set of sub-documents with equal or different field names. A document also contains an object *id* field, which stores a unique identification for each document. The object *id* consists of 12 bytes generated in the following sequence: four bytes for the timestamp (in seconds), three bytes for the machine number, two bytes for the process id, and an incremental number in the three last bytes. A collection (equivalent to tables in relational databases), contains a group of documents. Unlike relational databases, a collection is a dynamic schema as within a single collection, each document can have a number of different fields. MongoDB also groups collections into database. The main difference is that multiple independent databases are stored in separate files on the same disk, allowing many applications and users to store data in the same MongoDB server. The interaction with MongoDB databases may be through management tools (e.g. RoboMongo [Sch16]), http interfaces or a JavaScript Shell. The last is a essential tool, since it is provided along with MongoDB installation as a full-featured JavaScript interpreter, capable of managing databases by running arbitrary JavaScript programs.

MongoDB NoSQL database was chosen for this study due to the need for high scalability and availability when large amounts of data are processed in distributed cloud-based environments. Recent studies reveal the performance gains of MongoDB when compared to SQL databases such as PostgreSQL [JYBC15] and Microsoft SQL Server [ARMG15]. Another major feature of MongoDB is its wide use for the management of big data for a variety of distinct areas [GGM12]. For instance, The New York Times uses MongoDB for backing up all submitted photos in its form-building application, eBay Inc. uses MongoDB for its internal cloud manager and LinkedIn uses MongoDB for storing a huge variety of resources for its internal learning platform.

5.3 e-FReDock Conceptual Specification

As mentioned in the Introduction, the foremost objective of this study is to propose a new method to assist in performing practical virtual screening on FFR models through three strategies: (i) speeding up ensemble docking experiments; (ii) reducing systematically the resultant FFR models (RFFR models); and (iii) preserving the most biologically relevant information in the RFFR models produced. To achieve this goal, we developed the scientific workflow e-FReDock. e-FReDock is able to: (i) scale docking experiments out onto cloud resources; (ii) store docking data required to run experiments in a NoSQL database; and (iii) manage data and services required to evaluate and discard groups of snapshots at docking runtime, based on the method proposed in Chapter 4. It also incorporates e-SC services, which allow to reduce costs in cloud platforms, executes workflows in a loop as a DCG representation, and access the scientific workflow functionalities entirely through a web browser.

The conceptual architecture of e-FReDock scientific workflow is presented in Figure 5.2. It contains two sub-workflows: Create Experiment, which creates new docking experiments of FFR models; and Selective Ensemble Docking, which performs the molecular docking simulations and analyzes the quality of docking results. MongoDB database and e-SC server are usually hosted in the same machine; however, it is possible to deploy them into different machines or in storage data services from cloud platforms, such as Azure blob storage [AZU16]. Create Experiment workflow, Selective Ensemble Docking workflow, and e-SC Share Library are designed by users and stored in the e-SC server. The workflow enactment is executed on one of a number of virtual machines attached to the main e-SC server. These attached virtual machines represent the Workflow Enactment Nodes (Figure 5.2). A workflow enactment node is composed of: an e-SC Engine, a Java Runtime, and a Library Directory structure. The e-SC Engine component contains the e-SC code necessary to install workflow blocks, execute workflow invocations, and communicate with the e-SC server. Java development toolkit and an execution directory structure are required by engines to run workflow invocations.

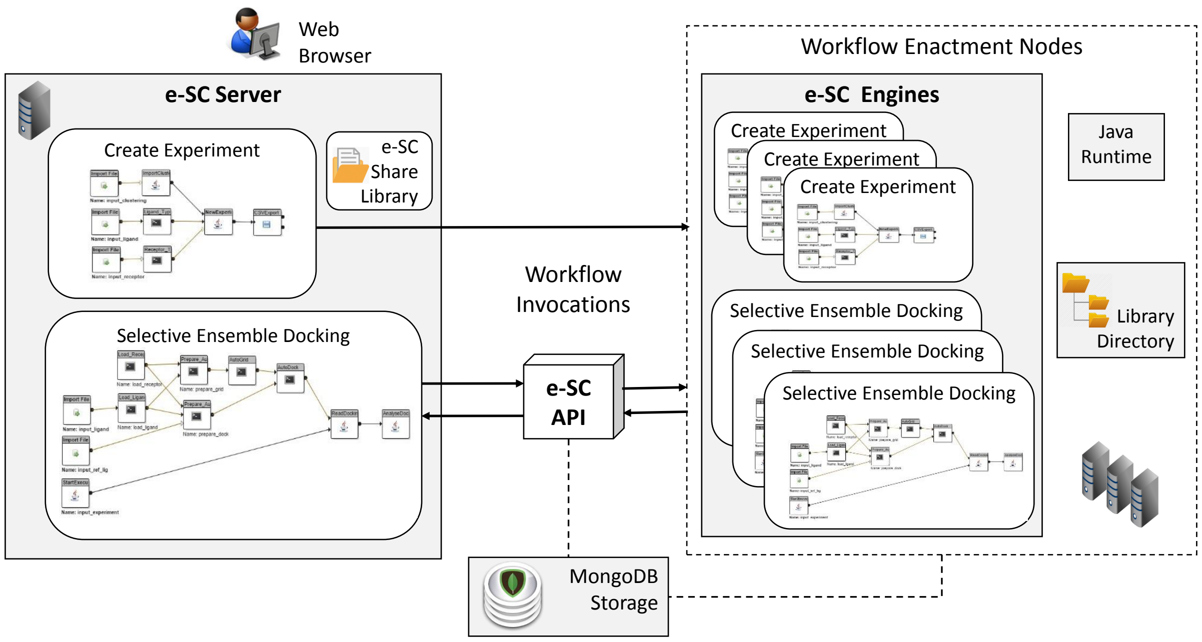


Figure 5.2 – Conceptual architecture of e-FReDock scientific workflow. A workflow instance starts on the e-SC server and its invocation is sent to be executed on one of the enactment nodes. The right box on the right represents the pool of virtual machines attached to the e-SC server from which workflows are executed. Data and control flows are monitored by e-SC, which is also responsible for scaling VMs onto cloud platforms.

There are a number of events performed by the e-SC Server from start to finish of a workflow invocation. The system initiates by placing the workflow invocation onto a queue with its parameters and settings required. When an idle node is found, the e-SC engine installed on this node removes the execution request from the queue and starts to execute the blocks. During workflow execution, the e-SC engine automatically deploys workflow blocks into the library on the enactment node. The four main steps performed by an e-SC Engine for each block within the workflow invocation are:

1. To verify if the Library Directory deployed on the enactment node contains the block code and its dependencies, including software, packages and files.

2. To download block code or dependencies from the e-SC Server and install them within the Library Directory, in case they are missing. The snapshots and ligands input files to execute the molecular docking simulations, which are in the e-SC Share Library, are also transferred from the e-SC Server to the Library Directory on the enactment node.

3. To initiate the block code execution, in the case of block code and dependencies are not missing.

4. To execute the main and post processing routine of the block.

It is worth mentioning that e-SC performs the deployment of blocks on the enactment node only when the workflow invocation is executed for the first time. One advantage of transferring all block files from e-SC server to the e-SC engine at once is that it avoids the high-throughput data transfer produced at runtime, thereby minimizing delays or failures caused by network connection issues, and data transfer costs charged by public cloud platforms. The e-SC Share Library was used to store all snapshots from the FFR model and ligands used to perform docking experiments. Thus, a total of 3.60 GB files were transferred to deploy the blocks within the Library Directory on the enactment nodes.

The e-SC engine creates a file directory for each workflow invocation in order to store input and output files that are used during the execution. By default, temporary workflow files are deleted from the enactment node immediately after ending the workflow executions; unless one or more target folders are indicated to save required files. This is especially useful for performing docking-based virtual screening experiments, since a simple molecular docking simulation executed on AutoDock4.2 [MHL$^+$09] generates approximately 3.15 MB files, varying depending on the ligand size. For instance, an exhaustive docking simulations between the FFR model and a small ligand would produce approximately 60 GB of input and output files, the majority of which are temporary files. This clearly indicates the importance of having a database to store essential information from docking results, and then delete all files produced by the Selective Ensemble Docking sub-workflow during each docking experiment.

Before presenting the database model and explaining all activities of the sub-workflows that constitute the e-FReDock conceptual architecture (Figure 5.2), the basic steps of AutoDock4.2 [MHL$^+$09] are presented, in order to better understand the next section of this chapter. This software is used to perform the molecular docking simulations into the e-FReDock scientific workflow.

5.3.1 AutoDock4.2 Software

AutoDock, a suite of automated docking tools, has been widely used to perform virtual screening of a huge database of potential ligands against a variety of receptors. It has been successfully applied to design new inhibitors or bioactive compounds and, consequently, to improve the RDD efforts. A selection of the most commonly used protein-ligand docking programs is presented by Sousa et al. [SRC+13]. In their review of analyzing the protein-receptor docking evolution in the period of 2001-2011, Sousa et al. ranked 53 docking software by number of citations and revealed that AutoDock [MHL+09] is the most popular docking software used in the academic community. Due to wide acceptance along with good accuracy and easy access by academic members, AutoDock is the software used for performing molecular docking simulations in this study [MHL+09].

AutoDock4.2 software contains two executables AutoGrid and AutoDock, which are responsible for preparing and executing docking experiments. The free energy of binding (FEB) is estimated by an empirical force-field-based scoring function that incorporates a set of atom types and charges, and by a stochastic algorithm that generates random conformations of receptor and ligand [MHL+09]. The four main steps to perform molecular docking simulations between a target receptor and a ligand with AutoDock4.2 are summarized below.

1. Preparation of ligand and receptor files. In this step, from a coordinate file, generally in PDB format, it is possible to insert the polar hydrogen atoms, partial charges and atom types [MHL+09]. Furthermore, the torsion degree can be selected to limit the flexibility of the ligand. After all configured, a PDBQT file is created for both receptor and ligand.

2. Configuration grid and docking files. After creating ligand and receptor files, the grid and docking files are prepared. Each file contains parameters that are specified to execute the third and fourth steps. A grid parameter file (GPF extension) is needed to run *AutoGrid*, specifying the grid point spacing, the grid center, and the name of output files written during the grid calculation [MHL+09]. One of the docking inputs, the DPF file, contains the parameters to run the search algorithm. The algorithms used by *Autodock4* are Lamarckian Genetic Algorithm (LGA) [MGH+98], Genetic Algorithm [MGH+98] and Simulated Annealing [GMO96].

3. Autogrid execution. For each atom type present in the ligand being docked (i.e. carbon, oxygen, nitrogen and hydrogen), the energy of interaction of this single atom with the protein is assigned at each grid point [MGH+98]. This step is essential because, besides the fact that the results of this pre-calculation will be the Autodock input parameters, it speeds up the docking computations. The grid output files consist of a log file with grid calculation, information about the coordinates and specifications to create a grid box. The quantity of grid map files depends on the number of atom types in each small molecule.

4. Autodock execution. Finally, AutoDock is executed by one of the search methods. The docked conformations and FEB results are independently generated by the search algorithm employed to perform a number of receptor and ligand interactions. There

are several parameters to improve the docking performance. However, different values can be selected depending on the search approach chosen. Besides specification of docking calculations, grid maps (step 3) and ligand files (step 1) are also specified in the input file (DPF extension). At the end of the execution, an output file (DLG extension) is produced with final docked coordinates, final binding energy values, such as RMSD and FEB, and other values derived from each evaluation.

LGA is the search algorithm employed in this study do execute the docking experiments. The main limitation of this algorithm is related to time constraints. LGA is the most efficient search for docking procedure, but depending on the input parameters, such as ligand structure, number of runs and number of energy evaluations, it can become computationally expensive. For instance, the total time spent for 10 runs of LGA and 300,000 energy evaluations between one snapshot of the FFR model and the TCL ligand, in a i7 CPU with 3.40 GHz and 12 GB RAM, is 40 seconds. This time doubles when the number of LGA runs and the energy evaluations are 25 and 150,000, respectively. On a practical level, performing millions of possible protein-ligand interactions using only one physical CPU is an impracticable task. One of the most intuitive ways to accelerate the execution time of high-throughput docking experiments is to enhance the computational performance [KFJ14]. Cloud or grid computing or GPUs, for example, are recent technologies that provide powerful capabilities to facilitate and streamline molecular docking-based virtual screening practices.

5.3.2 MongoDB Storage Component

The MongoDB Storage component is the database created to store and manipulate data generated during the e-FReDock workflow execution. These data consist of input and output information from AutoDock4.2 software [MHL+09], and all specifications needed to execute the method described in Chapter 4. A total of 8 collections were created to conduct the docking-based virtual screening experiments on e-FReDock scientific workflow. The E-R based conceptual diagram from the MongoDB database [Cho13] is presented in Figure 5.3, and a description on its collection is detailed in next subsections.

The only information needed to setup the high-throughput workflow invocations of the Selective Ensemble Docking sub-workflow are stored into the *Config* collection. Attributes from *Config* collection are used by the e-SC API component, which controls the quantity of workflow invocations and the quantity of selected snapshots per batch. Both, *quantProc* and *aveExpBatch* attributes are used by the e-SC API component to avoid that a large number of snapshots become processed even after their batches have been discarded. Section 5.3.5 provides more details on this e-SC API operation.

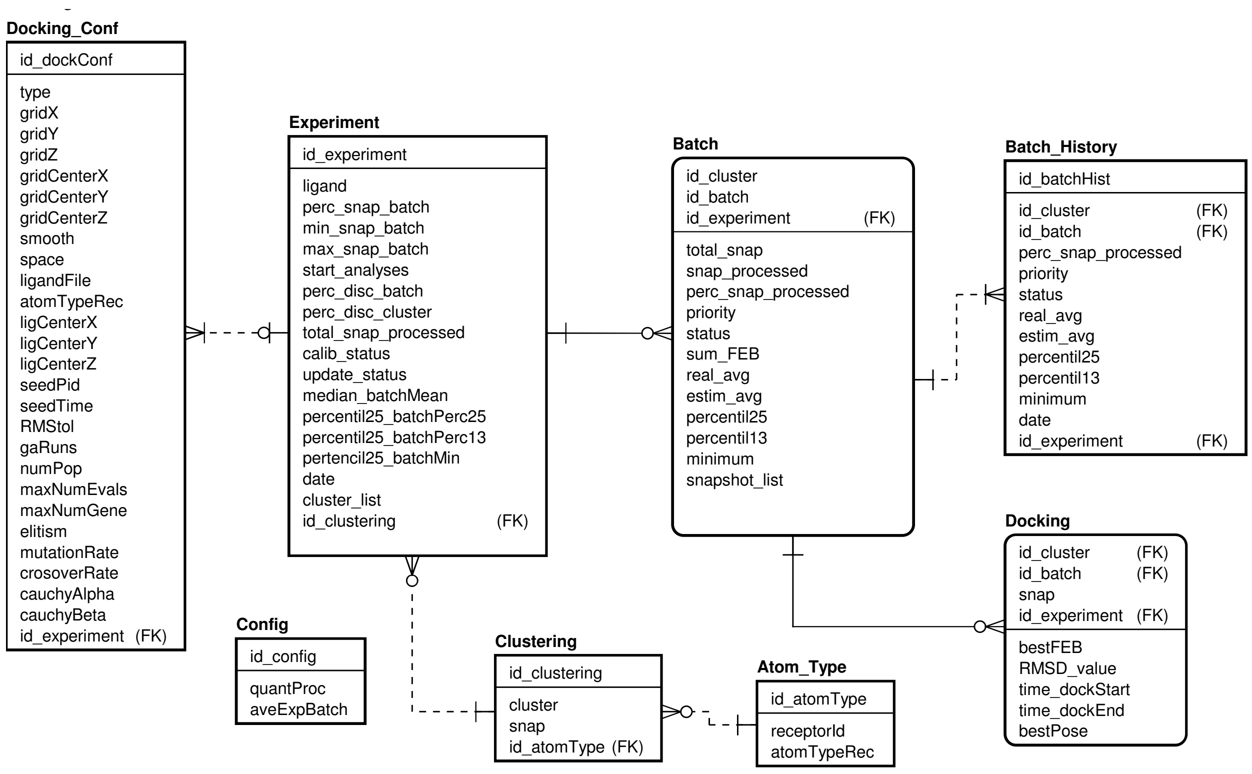


Figure 5.3 – Database model designed for e-FReDock scientific workflow. The collections were created to store data docking and control the selective ensemble docking experiments. Diagram produced with Astah Professional Software Development [Hir16].

5.3.2.1 Clustering Collection

The *Clustering* collection stores information regarding snapshots from the FFR model and their clusters. It consists of a attribute to identify each cluster of snapshots (*id_clustering*) and two attributes to save cluster and snapshot numbers. The *id_atomtype* attribute is the foreign key from *Atom_Type* Collection, which stores the receptor atom types. This structure allows storing more than one clustering for the same FFR model and clustering of new FFR models. *Clustering* is one of the most important collections for e-FReDock as, besides providing the ensemble of receptor conformations that will be used to the docking experiments, it also contains initial specifications to conduct the selective ensemble docking experiments.

5.3.2.2 Experiment Collection

As mentioned in Chapter 4, an experiment is composed by a clustering of snapshots and a ligand. *Experiment* collection stores every ensemble docking experiments submitted to the e-FReDock workflow, embodying input parameters needed to prepare batches of snapshots and analyze their quality. Most attributes from this collection are used to initialize and control experiments. For instance, *perc_snap_batch*, *min_snap_batch*, *max_snap_batch*, *start_analyses*, *perc_disc_batch* and *perc_disc_cluster* attributes are updated only when the experiment is created and used to control the batches execution. Each experiment also stores the date and time of experiment generation, and information related to the clustering of snapshots. The *cluster_list* attribute is an embedded document that contains the quantity of snapshots within each cluster.

Remaining attributes are updated at experiment runtime. The quantity of snapshots already processed is computed while the experiment is active by the *total_snap_processed* attribute, whereas experiment indexes and status are updated only when the analyses start to be performed. According to the method specification, experiment indexes should be computed after the end of calibration phase. To control the calibration phase during an experiment, e-FReDock uses *calib_status* and *update_status* attributes, where *calib_status* indicates an experiment waiting for all batches to reach the percentage to start analyses, and *update_status* signalizes when batches and experiment indexes should be computed to initialize the batch quality analyses.

5.3.2.3 Docking Parameters: Atom_Type and Docking_Conf Collections

All information required to create AutoGrid and AutoDock input files are stored in the *Docking_Conf* collection. To store all AutoDock4.2 inputs in one collection, the *type* attribute indicates "G" when a document has AutoGrid input parameters and, "D" when a document has LGA input parameters. Thus, each experiment is related to two documents of the *Docking_Conf* collection. A ligand file is taken from the Library Directory on the e-SC engine when it matches to the *ligandFile*, which stores the file name in a string attribute.

5.3.2.4 Batch and Batch_History Collections

Once the experiment is created, *Batch* and *Batch History* collections save the batches of snapshots generated by splitting the clustering of snapshots considering experiment input parameters. Batches are identified by sequential numbers starting from 1 to the identifier of each cluster. The quantity of snapshots placed in a batch is stored into *total_snap* attribute, whereas the list of snapshots is saved into *snapshot_list* attribute. This attribute is an embedded document with the following parameters:

- *id_snap*: the snapshot integer identification.
- *text_snap*: the snapshot string identification used to recognize the snapshot file into the Library Directory.
- *status*: denotes whether the snapshot has not been processed (0), has selected to be processed or is been processed by an e-SC engine (1), or has been processed by an e-SC engine (2).

Attributes from *Batch* collection were created according to the method described in Chapter 4, where priority and status retained the equal name, and batch indexes were referred to the following attribute names: *perc_snap_processed* for ps_i, *real_avg* for $\overline{x}_i$, *estim_avg* for $\overline{ex}_i$, *percentil25* for lq_i, *percentil13* for p_i, and *minimum* for min_i. To trace the changes on batch values during or after e-FReDock experiments, every update makes to these attributes creates a record into the *Batch_History* collection. Thus, during or after the executions all batch progress, including the data and time the attributes were updated, can be followed and retrieved by querying *Batch_History* collection.

5.3.2.5 Docking Collection

Crucial docking results are extracted from the AutoDock output file and stored in *Docking* collection. Such data include: (i) the best predicted FEB value, which is saved into the *bestFEB* attribute and used to update the batch indexes; and (ii) the 3D coordinates from the best ligand pose, which is saved into the *bestPose* attribute. The latter allows users to retrieve a final docking pose and plot it on a molecular graphical visualization tool. The *RMSD_value* attribute stores the distance between the ideal ligand position and its final docking pose when a known ligand position is used as reference. To retrieve the time taken for each docking experiment, *time_dockStart* and *time_dockEnd* attributes record when the Selective Ensemble Docking sub-workflow starts and ends its execution.

5.3.3 Create Experiment Sub-Workflow

All experiments submitted to run on e-FReDock are created in the Create Experiment sub-workflow. This sub-workflow is responsible for storing and organizing all data into the database to execute the Selective Ensemble Docking sub-workflow. All collections, except for *Config* and *Docking*, are updated when a new experiment is added to e-FReDock. Figure 5.4 displays the set of blocks that were used to design the Create Experiment sub-workflow. *Import File* and *CSVExport* are services from e-SC platform. All other blocks were developed for the purposes of this study.

The *Import File* block downloads files in any format from the e-SC file system to the workflow (Figure 5.4-A). This block was added in the Create Experiment sub-workflow to download the clustering of snapshots, the ligand, and the receptor files. If a ligand or receptor file is taken from the e-SC Share Library, the *Import File* block should be disconnected to the *Ligand_Type* or *Receptor_Type* block as shown Figure 5.4-B. In this case, the physical file name should be set in the parameters to the block corresponding to the file.

The *CSVExport* block generates a CSV file format from the data sets received in its input port. The e-SC system allows to provide the name and the folder to place the file into the e-SC file system. If no target folder is assigned, e-SC stores the file within the workflow invocation folder.

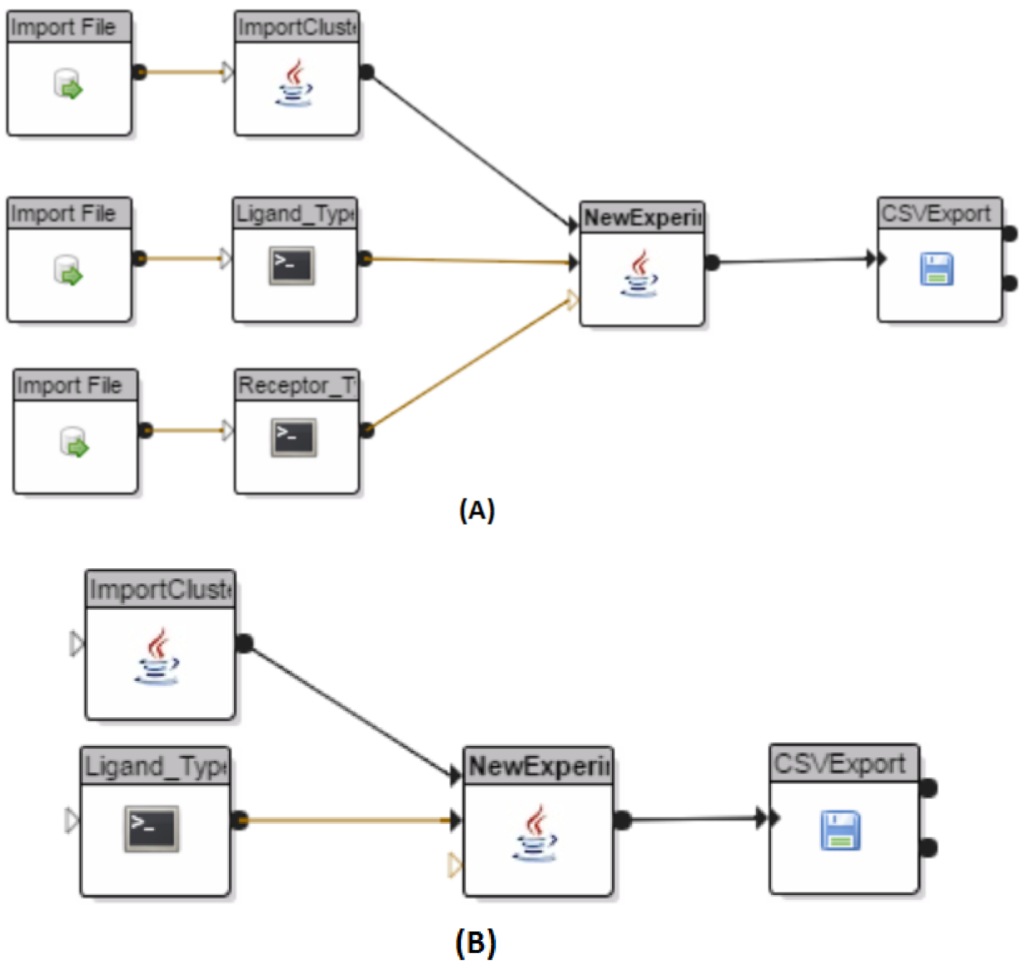


Figure 5.4 – Create Experiment sub-workflows designed on the e-SC platform. White arrows attached to the blocks indicate optional input data, while black arrows are mandatory to start the block execution. Both sub-workflows insert new ensemble docking experiments to e-FReDock. The main difference is that sub-workflow (A) adds a new receptor and clustering of snapshots to the database, while sub-workflow (B) searches for an existing receptor and clustering of snapshots into the database. The images within blocks indicate the programming language used to write the blocks, where Java is represented by its logo, C is identified by a black terminal box. Remaining images indicates blocks from e-SC system and they are used to manipulate input/output files.

Receptor_Type and *Ligand_Type* blocks were developed to extract atom types from receptor and ligand PDBQT files. The *Receptor_Type* block is linked to the *NewExperiment* block whether a new receptor is used in the experiment. Figure (Figure 5.4-B) shows the sub-workflow design to create an experiment when the receptor atom types are taken from the database through *ImportClustering* block. Besides extract atom types from the ligand file, the *Ligand_Type* block also obtains the number of rotatable bonds. Each block creates a text file and places it in the output port. Receptor and ligand PDBQT files can be uploaded from the e-SC file system (*Import File* block) or from the Library Directory (Figure 5.2).

The input file with all snapshots and related clusters is read and saved into the database by using *ImportClustering* block. This block inserts a new record to the *Clustering* collection when the *Import File* block is linked to it input port; otherwise, an existing clustering is selected to be used in the experiment. The textitImportClustering output is a file containing the clustering identification, which should be provided by the user as a block parameter.

When all input ports have data ready to use, the *NewExperiment* block is invoked to create a new experiment. This block creates a new record into *Experiment*, *Batch*, *Batch_History* and *DockingConf* collections, according to the input parameters provided by the user (Figures 5.5 and 5.6). *Clustering* and *AtomType* collections are updated only when sub-workflow from Figure5.4-A is executed. All these data stored in the database are used to execute the Selective Ensemble Docking sub-workflow invocation.

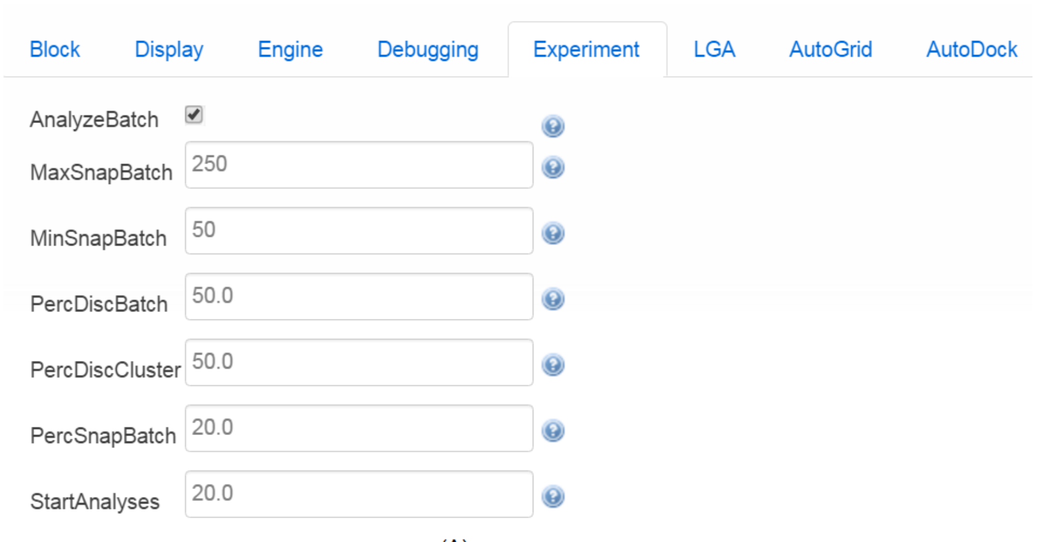
5.3.4 Selective Ensemble Docking Sub-Workflow

The Selective Ensemble Docking sub-workflow, designed to scale onto cloud VMs, contains a set of blocks for performing molecular docking simulations based on Autodock4.2 features, and measuring the quality of batches of snapshots. Even though this sub-workflow can be executed through the e-SC interface, we created an external program by using the e-SC API. The e-SC API is described in more detail in the next section. Basically, it contains essential functionalities to monitor workflows and send required information to every workflow's blocks. Figure 5.7 shows the Selective Ensemble Docking sub-workflow designed to run and select the best docking results of FFR models. A total of nine blocks was developed to execute the selective ensemble docking experiments. The activities executed for each block are detailed above.

- *Load_Receptor* block: It downloads the PDBQT snapshot file from the Library Directory to the workflow invocation folder and generates a TXT file with the atom types from the receptor. The receptor physical file name and atom types are sent by the e-SC API as a parameter.

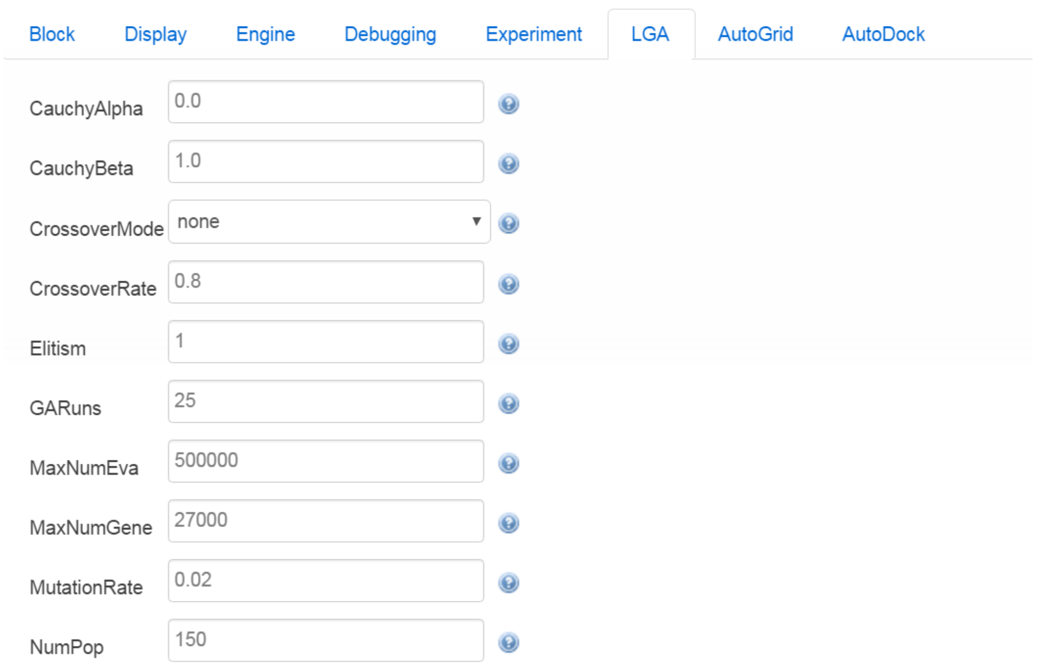
- *Load_Ligand* block: It downloads the PDBQT ligand file from the Library Directory to the workflow invocation folder, and generates a text file with the atom types and number of rotatable bonds from the ligand. The ligand physical file name, atom types and number of rotatable bonds are sent by the e-SC API as a parameters.

- *StartExecution* block: It generates a CSV file containing the experiment, cluster, batch and snapshot identification, and the date and time the experiment starts. All identification data are sent by the e-SC API as a parameter.



AnalyzeBatch	✓
MaxSnapBatch	250
MinSnapBatch	50
PercDiscBatch	50.0
PercDiscCluster	50.0
PercSnapBatch	20.0
StartAnalyses	20.0

(A)



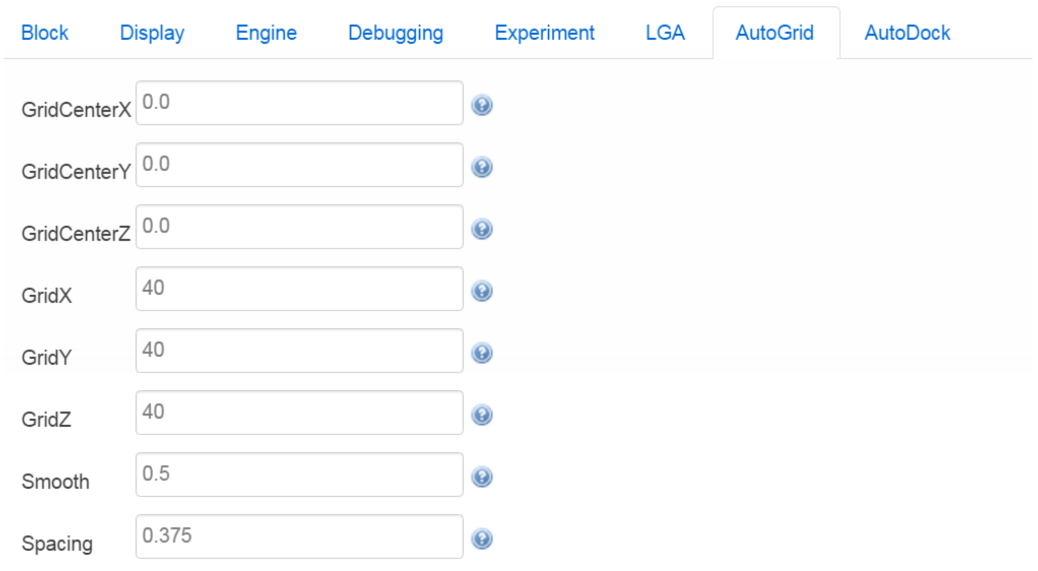
CauchyAlpha	0.0
CauchyBeta	1.0
CrossoverMode	none
CrossoverRate	0.8
Elitism	1
GARuns	25
MaxNumEva	500000
MaxNumGene	27000
MutationRate	0.02
NumPop	150

(B)

Figure 5.5 – Experiment and LGA input parameters from *NewExperiment* block used to create new experiments on e-FReDock. Parameters used to controls and analyze batches of snapshots before and during the experiments, and to configure AutoDock4.2 based on the LGA are in (A) and (B) interfaces, respectively.

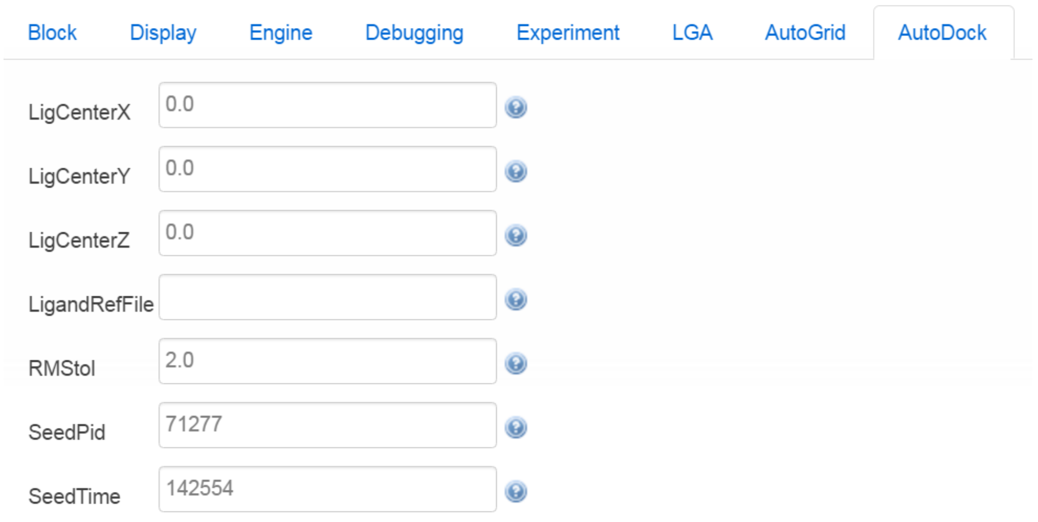
- *Prepare_AutoGrid* block: It receives the receptor and ligand files through the input port, and generates a GPF file with the input parameters assigned to the AutoGrid. Autogrid parameters are taken from the database and sent by the e-SC API as a parameter. This is step 2 from AutoDock4.2 software (Section 5.3.1).

- *Prepare_AutoDock* block: It receives the receptor and ligand files through the input port, and generates a DPF file with the input parameters assigned to LGA and AutoDock. This block has an optional input port to load the ligand reference file.



GridCenterX	0.0
GridCenterY	0.0
GridCenterZ	0.0
GridX	40
GridY	40
GridZ	40
Smooth	0.5
Spacing	0.375

(A)



LigCenterX	0.0
LigCenterY	0.0
LigCenterZ	0.0
LigandRefFile	
RMStol	2.0
SeedPid	71277
SeedTime	142554

(B)

Figure 5.6 – AutoGrid and AutoDock input parameters from *NewExperiment* block used to create new experiments on e-FReDock. Parameters used to generate AutoGrid and AutoDock files are in (A) and (B) interfaces, respectively.

Autodock input parameters are taken from the database and sent by the e-SC API as a parameter. This is step 2 from AutoDock4.2 software (Section 5.3.1).

- *AutoGrid* block: It receives a GPF file through the input port, and execute AutoGrid program from AutoDock4.2 toolkit. This is step 3 from AutoDock4.2 software (see section 5.3.1).

- *AutoDock* block: It receives a DPF file and all *AutoGrid* output files through the input port, and execute *AutoDock* from AutoDock4.2 toolkit. This is step 4 from AutoDock4.2 software (Section 5.3.1).

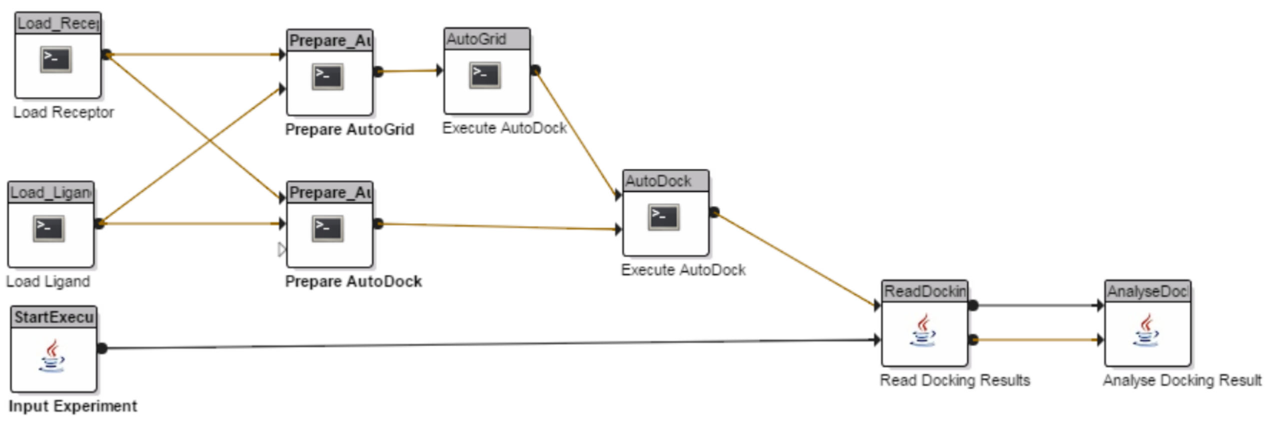


Figure 5.7 – Selective Ensemble Docking sub-workflow designed on the e-SC platform. The images within blocks indicate the programming language used to write their activities, where Java is represented by its logo, and C is identified by a black terminal box.

- *ReadDocking* block: It receives the AutoDock output DLG file, and extracts the best predicted FEB value along with its 3D coordinates pose and, when applicable, its RMSD value. This information are saved in two different files: (i) a CSV file with the data received from *StartExection* block, the best docking values (FEB and RMSD); and (ii) a TXT containing the 3D coordinates from the best ligand-receptor pose.

- *AnalyseDockExperiment* block: It receives the files from *StartExection* block and connects to the database to insert docking results. This block also updates priority and status of batches by using the functions created to determine the priority in different batches of snapshots (Algorithms 4.1 and 4.2). For each docking experiment, the quantity of processed snapshots (*total_snap_processed* field) is updated in *Experiment* collection, and all attributes from *Batch* collection are also updated. *Batch_History* and *Docking* collections generate a new record, where *Batch_History* replicates the attributes updated in *Batch* collection, and Docking stores the information from the input files.

As described earlier, e-SC system removes all data at the workflow execution sites when they are no longer needed. To store data into the e-SC file system, workflows need a file export service, such as the *CSVExport* block from Create Experiment sub-workflow; otherwise, all data used and generated by the workflow's blocks are considered temporary, and deleted after the end of its execution. We decided to store necessary docking data into MongoDB database and eliminate all files generated during the Selective Ensemble Docking sub-workflow execution for two critical reasons: disk space and data transfer overhead.

In this study, the selective Ensemble Docking sub-workflow invocation was performed from the e-SC API. However, it also can be executed through e-SC interface, or by designing new workflows, such as if one wants to perform a single molecular docking simulation. By using e-SC services and e-FReDock blocks, users can modify or design new docking workflows according to their needs, enacting them from the e-SC interface or via API.

5.3.5 The e-SC API Component for Handling Selective Ensemble Docking Sub-Workflow Invocations

The e-SC API is a set of programming instructions and standards that allows developers to deal with e-SC services and integrate them with existing systems by including a JAR file into any Java project. It has a set of e-SC components to execute workflow instances on cloud resource, and manage data files by accessing the e-SC file system. We decided to use this component because e-SC enactment system is a DAG-based workflow. It means that e-SC system does not allow to repeat the execution of the workflow blocks, which is a essential feature to correctly run the method proposed in this study. Hence, the e-SC API Java client was included in our e-FReDock project to monitor the Selective Ensemble Docking sub-workflow invocations in a queue of docking tasks. It also selects which snapshots can be executed based on the processing stage presented in the workflow scheme from Figure 4.2. In this schema, the two aspects from the experiment that need to be controlled by the e-SC API are:

1. Calibration status: snapshots from all batches are selected to be executed until the percentage to start analyses is reached (*start_analyses* attribute from *Experiment* collection).

2. Execution status: snapshots from all batches are selected to be executed according to their status and priority, i.e., snapshots belonging to batches with high priority and status equal to "A" (active) or "P" (Priority changed) are executed earlier.

Algorithm 5.1 depicts how the Selective Ensemble Docking sub-workflow iterations were controlled via e-SC API. Most procedures are based on statuses from snapshots, batches and experiments. These statuses are updated in the database by *AnalyseDock-Experiment* block. Status from batches and snapshots are attributes from *Batch* collection, while $updateStatus$ is an attribute from *Experiment* collection. This experiment status is changed only when the percentage to start analyses is reached by all batches, in which case batches and experiment indexes are computed, and batches status are changed from "C" (calibrate) to "A" (active). Each Selective Ensemble Docking sub-workflow invocation is added into a queue of docking tasks, whose size is stored in the $quantProc$ attribute from *Config* collection, according to the number of processors from VMs. The queue of docking tasks contains a list of invocation identifiers, which are taken after the API runs a workflow. In addition to the identifier, status and folder are also attributes from the invocation object, which is returned to the API and represents a record of the workflow run within the e-SC system.

```
 1: Let $Q$ be the queue of docking tasks and $i$ its first position;
 2: Let $W$ be the workflow invocation within $Q$;
 3: Let $S$ be the status from a snapshot;
 4: Let $B$ be the status from a batch;
 5: while $B$ is equal to "C", "A", or "P" do
 6:    Create one $Q$ with $quantProc$ positions;
 7:    while $Q$ is not empty do
 8:       Remove $W[i]$ from $Q$;
 9:       if $W[i]$ status is equal to $Finished$ or $ExecutionError$ then
10:          if $W[i]$ status is equal to $Finished$ then
11:             $S \leftarrow 2$;
12:             Delete folder from $W[i]$;
13:          end if
14:          Select a snapshot from a batch;
15:          Make a new workflow invocation;
16:          Add the new workflow invocation into $Q$;
17:          if $W[i]$ status is equal to $ExecutionError$ then
18:             $S \leftarrow 0$;
19:          end if
20:          if $updateStatus$ from experiment is equal to true then
21:             Compute batches and experiment indexes;
22:          end if
23:       else
24:          Add $W[i]$ to $Q$;
25:       end if
26:    end while
27:    Retrieve $Q$;
28: end while
```

Algorithm 5.1 – Algorithm designed to control workflow invocations via e-SC API.

The progress of each workflow invocation is monitored by calling the object invocation. In this procedure, the workflow engine reports progress back to the main e-SC server and classifies the workflow as *Running*, *Queued Finished* or *ExecutionError*. Algorithm 5.1 uses the e-SC API to check periodically if a workflow execution is *Finished* or *ExecutionError* to send a new invocation. Further, the experiment is set to rerun when the workflow execution fails (*ExecutionError*) due to some error within one of the workflow blocks, and the invocation folder is removed when the workflow completes. This latter avoids system delays and failures caused by many of folders created for each invocation.

Regardless of the selection procedure, whenever a workflow run ends or fails, a snapshot is selected from the set of active batches considering its priorities. This procedure, performed when line 14 of Algorithm 5.1 is executed, also deals with the problem of false-positives risk by selecting many or even all snapshots from the same batches at once.

Towards this end, we restricted the quantity of snapshots being processed by batch with the following conditions:

- If $CalibrationPhase = true$ and $SnapProcessing > 0$, then go to the next batch;
- Else if $CalibrationPhase = false$ and $BatchPriority \geq 4$ and $SnapProcessing > MaxSnapProcessing$, then go to the next batch;
- Else if $CalibrationPhase = false$ and $BatchPriority < 4$ and $SnapProcessing > (MaxSnapProcessing/2)$, then go to the next batch;

The *qtdProcessing* attribute from *Batch* collection is used and referred above as $SnapProcessing$ to count the number of snapshots being processed. To control the quantity of docking experiments will be executed in parallel the $MaxSnapProcessing$ parameter stores the maximum quantity of snapshots. This parameter is taken from the *maxDockBatch* attribute in *Config* collection and may be set by the user before starting the experiment. In the calibration phase, only one parallel docking execution is allowed per batch, in order to prevent that the percentage of processed snapshots exceeds the percentage to start the batch analyses. In the execution phase the maximum quantity of parallel executions is dictated by the priority. By giving more parallel docking executions for the highest priority batches, we expect to execute less unpromising snapshots.

5.4 General Remarks

The main challenge faced by incorporating an FFR model in docking experiments is the high computational cost of performing and analyzing each receptor conformation and ligand interactions. This chapter introduced e-FReDock, the cloud-based scientific workflow designed to optimize molecular docking simulations of FFR models based on the method proposed in Chapter 4. The e-SC platform [HWWC13] was the SWfMS chosen to deal with the high-throughput docking-based virtual screening experiments, where a set of blocks were developed to prepare and perform a new selective ensemble docking protocol. The set of workflow components and the MongoDB database scheme presented in this chapter were developed to address the specifications described by the proposed method in Chapter4. The four essential components presented in this chapter that make up the e-FReDock are:

1. The Create Experiment sub-workflow. A set of blocks were developed to prepare the experiments that are submitted to the e-FReDok workflow, where a clustered FFR model, a PDBQT receptor file, and a PDBQT ligand file are the input data required to configure a new experiment.

2. The Selective Ensemble Docking sub-workflow. A set of blocks were developed to execute docking experiments, analyze docking results, and discard batches of snapshots based on the best binding free energies obtained from the snapshots already processed.

3. A database for the e-FreDock workflow. MongoDB was the NoSQL database chosen to store input/output data and control data execution. It stores all information generates from create experiment and selective ensemble docking sub-workflows.

4. The e-SC API Component. It is one of the most important component of the e-FReDock conceptual architecture. This component contains every procedure required to scale the selective ensemble docking sub-workflow out onto VMs, monitor the Selective Ensemble Docking sub-workflow invocations, and selects snapshots that are likely to represent the most promising conformations between an FFR model and a specific ligand.

Next chapter presents the results obtained by executing e-FReDock on private and public cloud platforms. It is provided a detailed investigation on the performance gains obtained by reducing the dimensionality of FFR models at docking runtime and executing parallel e-FReDock workflow invocation onto a set of VMs. Further, the accuracy of the resulting RFFR models is also evaluated.

6. EXPERIMENTAL RESULTS OF e-FReDock ON CLOUD PLATFORMS

This chapter presents the set of results of this thesis. It begins with an overview of the cloud platforms used to perform the experiments, focusing on the resources employed to execute e-FReDock. After, a description is given on the performance evaluation performed to select the most cost-effective cloud instance setting for executing the docking-based virtual screening experiments. We divide the experimental results into two distinct scenarios: (i) execution of e-FReDock for batch docking analyses; and (ii) execution of e-FReDock for cluster docking analyses. In each scenario, we analyze the differences in performance by comparing the accuracy reached from the RFFR models produced with the best docking results obtained when all snapshots of the FFR model and its crystal structure interact with a set of different ligands. In addition, we analyze the performance gains obtained by using e-FReDock to optimize molecular docking experiments of FFR models.

6.1 Cloud Computing Platforms

Clouds are currently the most cost effective HPC alternative to considerably improve virtual screening performance. According to D'Agostino et al. [DCQ+13] cloud computing is able to provide clear advantages for small-to-medium laboratories working in the field of biotechnology, which typically do not have the possibility to invest a sufficient amount of time and money in creating and maintaining on-site hardware infrastructure. Despite having intrinsic security and data transfer limitations, cloud computing has been demonstrated a more viable option in scientific environments, especially for processing short peaks on demand [KHB+13, Sul14]. The cloud platforms selected for performing the docking-based virtual screening were: Microsoft Azure public cloud [AZU16] and Cloud Innovation Centre (CIC) private cloud [Hor16]. A brief overview of these platforms and services used in this study are presented below.

6.1.1 Microsoft Azure Cloud

Microsoft Azure is a SaaS solution for computing in public clouds. It provides a comprehensive collection of propriety development tools and protocols to support developers in developing, hosting and controlling their application [BYV+09]. Currently, Microsoft Azure platform offers a variety of services which are classified according to their functionalities, such as computing, database, storage, networking, developer tools and others [AZU16].

Services from different components can work conjointly or separately, providing more flexibility in the applications developed on the cloud. This study will focus particularly on compute and blob storage, which are the on demand services assigned to deploy the cloud environment proposed.

Azure VMs or instances provide IaaS to virtually deploy applications on any language and Operating System (OS). These services are billed on a per-minute basis when the machines are being used. The basic VM configuration includes the following specifications: virtual hard disk, size and region [AZU16]. A virtual hard disk contains the OS and data. The user can select a virtual hard disk from Azure gallery, with a variety of OS and applications, or create its own virtual hard disks and insert it on the gallery. Azure provides a set of hardware, also called size options, to deploy a VM. The size is classified based on purposes of processing, where a variety of CPU, RAM and disk size are offered among different categories. Region is the location where the new VM will be hosted, e.g. Central US, North Europe, Asia, Brazil, among others. The price is usually dictated by the size and OS of the VM, but in some cases it is also affected by the region.

A blob, the acronym of Binary Large Object, is one of the data management services that stores unstructured object data. This service is useful for storing large amounts of data at a low cost or to provide a persistent storage by building an operational file system on an Azure VM. There are two types of blobs: block, and page [AZU16]. Block blobs can contain up to 195 GB of data distributed in 50,000 blocks of 4 MB each. They are suitable for storing text and binary files, such as documents and media files. Page blobs can store up to 1 TB and are commonly used as a data disk or operational system in the VMs due to their efficiency in frequent read/write operations. Both block and pages are in a container, which provides a grouping of a set of blobs. The number of blobs that a container can store and the number of containers that an user account can contain are unlimited. Figure 6.1 illustrates the Azure blob storage structure.

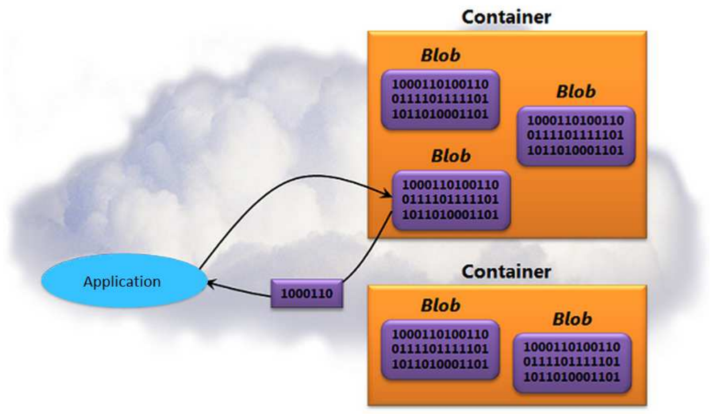


Figure 6.1 – Azure blob storage. Each container consists of one or more pages or block blobs for storing a collection of binary information [AZU16].

The Azure storage service provides a geo-replication system to avoid hardware faults and the loss of critical data [AZU16]. In this process, each blob is transversally replicated in four accessible secondary computers from the same data center region. This step is performed to ensure access to a secondary computer in case the primary fails. The system only allows specification of computers from other regions if they are within a distance of 500 kilometers.

The Azure cloud platform was chosen for this study for two main reasons. First, the performance observed when e-SC server is deployed on Azure instances for scaling workflow invocations among 200 worker instances [CHWW13]. Second, the deployment of different cloud environments for performing docking-based virtual screening on the Azure cloud platform [KBT$^+$14, NMT$^+$15].

6.1.2 CIC Private Cloud

The second cloud platform used to execute the docking-based virtual screening experiments was CIC, the acronym of Cloud Innovation Center. This private cloud is located at Newcastle University and built by the School of Computing Science to support cloud research, staff and students' mass-scale visualization requirements and third party partners. CIC private cloud infrastructure is a large visualization platform, consisting of 27 nodes with 20 cores each, resulting in a total of 540 cores and 7,424 GB total RAM memory. The storage area network uses a 10 Gb Ethernet Storage LAN and 4 nodes with 12 cores, 64 GB RAM and 37 TB storage per node. Furthermore, 3 nodes with 12 cores, 64 GB RAM and 1.4 TB storage each are used for management purposes. Horizon Dashboard [Hor16] is the web based user interface for OpenStack Nova services.

The CIC Private Cloud is available for students and staff members from Newcastle University since January 2016. Its access was granted by the project coordinators for the sole purpose of running the experiments of this research.

6.2 e-FReDock Performance Analyses on Cloud Virtual Machines

The primary goal of this set of experiments is to assess the e-FReDock performance on different Azure VMs located in the North Europe data center, and VMs from the Private Cloud located at Newcastle University. We executed docking experiments without discarding snapshots (percentage to start analysis equal to 100%) in order to create a more scalable system. One VM on each cloud platform was turned into an e-SC server as we executed e-FReDock separately. Azure e-SC server was hosted in a Standard D2 VM instance

(Intel Xeon 2.4 GHz, 7 GB RAM), while CIC e-SC server was hosted in a Large flavor (Intel Xeon 3 GHz, 8 GB RAM). MongoDB database was hosted in each one of these servers.

6.2.1 Azure Virtual Machines Performance

In an effort to better understand which choices to make regarding price and performance of a commercial cloud system used in the e-FReDock scientific workflow, we tested several instances in the Microsoft Azure cloud. A total of 100 Selective Ensemble Docking sub-workflow invocations with identical ligand and docking parameters were used to estimate the overall time, cost and efficiency to complete the docking experiments in different Dv2-series Ubuntu 14.04 instances. This was performed to define a baseline cost, which could be extended to multi-VM situations. Further, assuming AutoDock program can linearly scale when using multiple cores, the goal was also to investigate whether AutoDock could take advantage of larger memory allocations.

In these experiments, the LGA and its parameters were used to execute the molecular docking simulations between the first 100 snapshots from the InhA FFR model [Gar09] and the TCL ligand from PDB ID 2B35 [STB$^+$06] with 2 rotatable bonds. Twenty-five LGA independent runs were executed with a maximum of 500,000 energy evaluations. The other LGA parameters were kept at default values. Table 6.1 lists the different VMs instances we tested along with their corresponding features and costs. The Dv2-series instances were used to scale the Selective Ensemble Docking sub-workflow, since they are based on the 2.4 GHz Intel Xeon E5-2673 v3 processor with Intel Turbo Boost Technology 2.0 that can go up to 3.2 GHz. According to Azure website [AZU16], Dv2-series instances carry more powerful CPUs which are on average about 35.00% faster than D-series instances for the same memory and disk configuration.

Table 6.1 – Types of Azure Dv2-series instances used to assess e-FReDock performance.

Instance Name	**Cores**	**RAM (GB)**	**Disk Size (GB)**	**Price (US$)**[a]
D2 v2	2	7	100	0.14
D3 v2	4	14	200	0.28
D4 v2	8	28	400	0.55
D5 v2	16	56	800	1.11
D11 v2	2	14	100	0.18
D12 v2	4	28	200	0.37
D13 v2	8	56	400	0.74
D14 v2	16	112	800	1.48

[a] Pricing information from the Azure website as of January 15, 2016 [AZU16].

Figures 6.2 and 6.3 present the observed time and cost to complete the docking experiments by the number of threads used in each Azure instances. Interestingly, the similar times for instances with equal number of cores suggest that the amount of RAM does not greatly affect the docking simulations completion time, regardless of the number of threads. The number of cores had the largest impact, with the time to completion reduced by half as the number of cores exponentially increased. On the contrary, the cost to completion increased approximately 25.00% between instances with equal number of cores and different amount of RAM. As the pricing per instance is proportional to the amount of RAM, it is reasonable to use the price/core/hr to derive cost estimates for docking experiments. Thus, the cost to perform an exhaustive docking experiment between the 20,000 snapshots from the FFR model and the TCL ligand on Azure resources in the conditions above described is US$ 34.00 for the D12 v2 instance. Without changing the docking parameters, the D3 v2 instance, which has equal number of cores but half amount of RAM, will perform this experiment at the same rate resulting in a cost of approximately US$ 26.00. This analysis clearly indicates that D11 v2, D12 v2, D13 v2, and D14 v2 instances do not outperform the other instances with equal number of cores, in spite of higher RAM amounts.

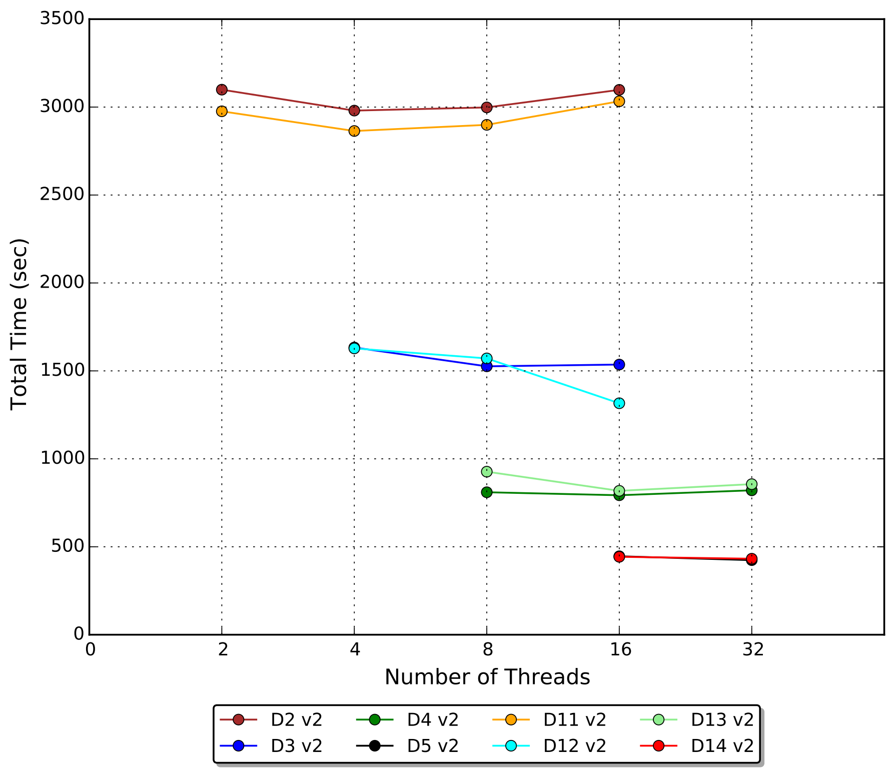


Figure 6.2 – Comparing the overall processing time of Dv2-2 Azure instances with different number of threads.

We also observed that running e-FReDock on small VMs (2 or 4 cores) is slightly more efficient than running the same workload on 16 core VMs. One possible explanation for this performance loss is the logical isolation, which is a common practice of malicious users to maximize utilization rate of physical sever [HCAL14]. This behavior can be seen in Figure 6.4, where the efficiency of e-FReDock shows lowest percentage when the number of cores is 16. Figure 6.4 also shows that the highest percentages of efficiency were reached by D2 v2, following by D11 v2, D12 v2, D3 v2, and D4 v2 instances. However, the cost

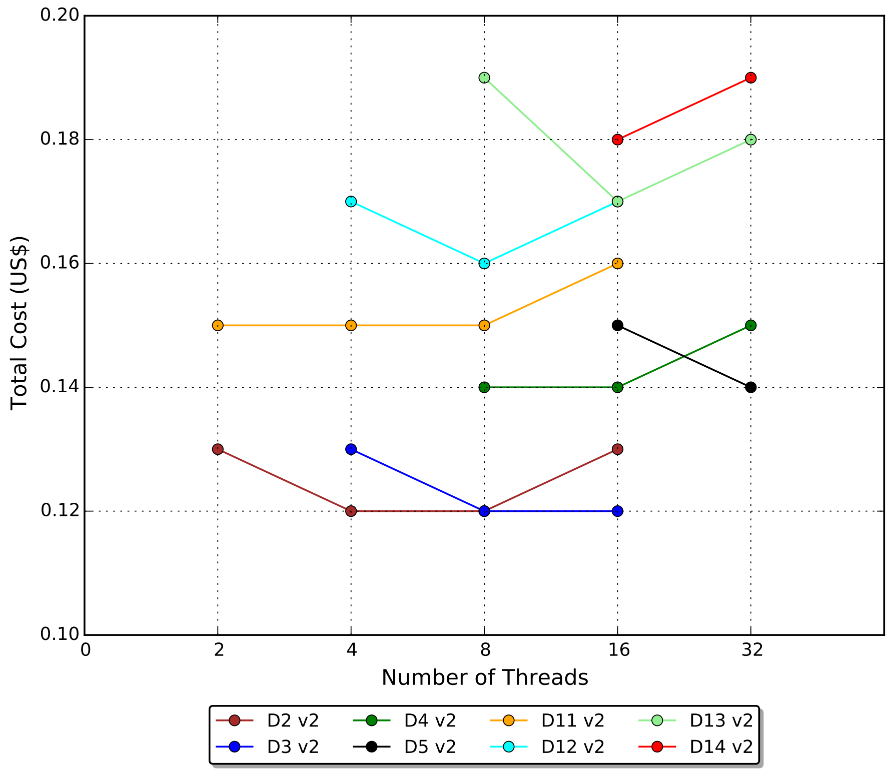


Figure 6.3 – Comparing the overall processing cost of Dv2-2 Azure instances with different number of threads. Price per machine/hour based on Azure website applied to North Europe region in January 15, 2016 [AZU16].

analyses above described revealed that D11 v2 and D12 v2 were unable to outperform VMs with equivalent cores. Thus, we decided to execute the e-FReDock workflow in the most efficient and less costly instances, i.e. D2 v2 and D3 v2 instances.

Cala et al. [CHWW13] demonstrated the high scalability achieved by e-SC system on the cloud when 200 worker nodes were used for generating Quantitative Structure-Activity Relationships (QSAR) models. We therefore would like to know if e-FReDock based on e-SC API is also able to achieve performance gains for the docking-base virtual screening in the Azure cloud platform. To answer this question, we executed 500 Selective Ensemble Docking sub-workflow invocations, instead of 100, and evaluated the speedup levels when the number of LGA runs was 10 and 25. Figure 6.5 presents the behavior of performance gains according to the number of D2 v2 VMs attached to e-SC server. The D2 v2 instance using 4 threads (i.e. 4 concurrent workflow invocations at the same VM) was chose to perform this experiment, since it showed the best efficiency when compared with other instances from Dv2-series (Figure 6.4).

Comparing the speedup of e-FReDock with 10 and 25 LGA runs, it is clear that e-SC introduces an overhead to manage the distribution of activities when the number of the Selective Ensemble Docking sub-workflows is configured for 10 LGA independent runs, and the number of cores increases from 8 to 16 (Figure 6.5). Consequently, some VMs may remain idle. This happens because the workflow executions in the VMs performed faster

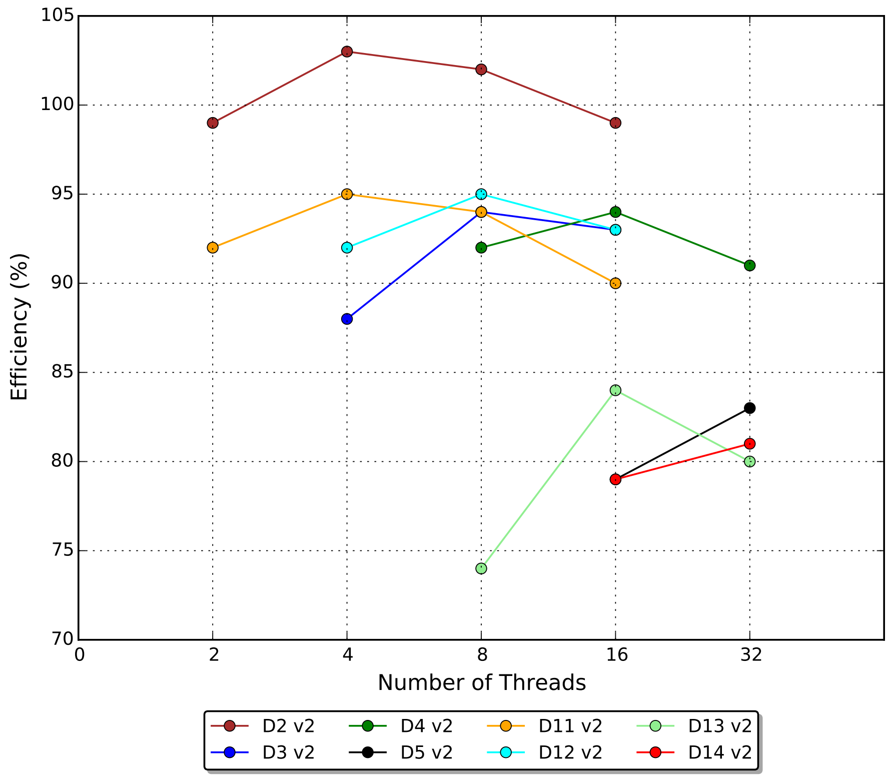


Figure 6.4 – Comparing the efficiency of Dv2-2 Azure instances with different number of threads.

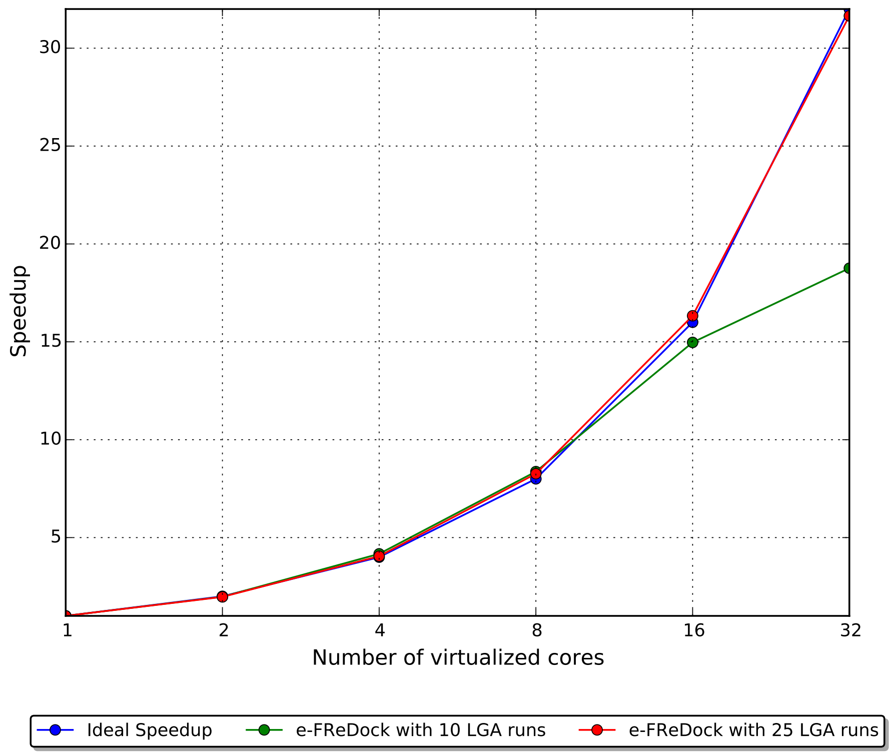


Figure 6.5 – Scalabity of e-SC for e-FReDock on Azure D2 v2 virtual machines. The time to complete each e-FReDock experiment is approximately 40 seconds for 10 LGA runs and 100 seconds for 25 LGA runs.

than the e-SC API workflow calls. We found that e-SC has a native weighted processing cost associated with workflow invocation procedure, which includes ensuring the communication among the set of VMs, storing data for provenance and performance, and other system controls [HWWC13]. For this reason, when we increase the amount of invocations and the number of available VMs, the algorithm designed to control the workflow invocation in e-SC API (Algorithm 5.1) tends to require more time to process the workflow invocation. However, this overhead can be solved when the total time to execute each Selective Ensemble Docking sub-workflow is approximately 100 seconds, which was achieve when the number of LGA runs in a docking simulation was 25.

As described in Section 5.3.1, the docking total time execution depends on the ligand structure, maximum number of evaluations, and the number of LGA runs. The lower these values, the faster the docking simulation completion time. TCL was chosen to evaluate the performance analyses in the cloud VMs because it is the smallest molecule used to screen the FFR model in this study. In addition, all docking experiments were performed with a maximum of 500,000 energy evaluations and 20 LGA runs. Thus, it is expected that the time to complete each Selective Ensemble Docking sub-workflow execution will be more than 100 seconds, avoiding the possibility of overhead, observed in Figure 6.5.

6.2.2 CIC Private Cloud Virtual Machines Performance

CIC private cloud has a small set of flavors with limited hard disk (Table 6.2). Disk size was the determining factor to select the VM flavors since the Ubuntu 14.04.3 LTS installation, which is required for e-SC server and engines, takes 7.5 GB from the total disk size. For this reason, Large was the flavor chosen to evaluate the performance gains and execute e-FReDock in the CIC cloud platform.

Table 6.2 – Types of virtual machine flavors from CIC private cloud.

Flavor Type	**Cores**	**RAM**	**Disk Size (GB)**
Tiny	1	512 MB	1
Small	1	2 GB	8
Medium	2	4 GB	8
Large	4	8 GB	16

When comparing the processing rates at each cloud resource, it was found that the efficiency from Large flavor is better than Dv2 series with 4 cores, i.e 97.00% against 88.00% for D3 v2 and 92.00% for D12 v2. Table 6.3 presents the performance evaluation of the Large private VM to execute 100 docking experiments employing the same parameters as those used for assessing the Azure efficiency. The similar times for the Large VM suggest that the amount of threads does not affect the total docking execution time. Thus, four threads on a four cores VM seems to be the best option, since it prevent *thrashing* in operating system, a common problem caused when the system spends more time changing the active threads than running the contents of the threads themselves.

Table 6.3 – Comparative performance of Large CIC cloud VMs according to the number of threads.

Threads	**Total Time (Sec)**	**Efficiency (%)**
1	5846	100
4	1499	97
8	1496	98
16	1500	97

6.3 The Set of Ligands Used to Screen against the FFR model

The first database used for extraction of ligands was the PDB [BWF+00]. This database has currently a total of 70 available *Mycobacterium tuberculosis* InhA crystallographic structures. Among them, 13 structures were chosen and their ligands were extracted to perform the experiments on e-FReDock. These ligands were selected for two main reasons. First, only the ligands without the coenzyme as part of their structures were used since the NADH coenzyme was considered as part of the protein receptor in all snapshots of the FFR model. Second, this set of ligands has being investigated in Chapters 4 and 3, as well as in other studies performed by our research group [PdSR+13, DPQRNdS15, DPQR+15, BQDPB15].

ZINC is the second database used to build the set of ligands [ISM+12]. It is a public access database of commercially-available compounds for virtual screening. ZINC15, the current ZINC version, contains 120 million purchasable "drug-like" compounds, of which approximately 25.00% are commercially available [SI15]. This database was chosen since it is a free and well-known platform for research tool development. It is clear that performing virtual screening of 30 million ZINC compounds, specially using FFR models, is an impracticable task. For this reason, the ranked list of drug candidates of the FFR model generated by Quevedo [Que16] was used to select the ligands from ZINC database to perform the docking-based virtual screening in e-FReDock. Quevedo [Que16] investigated the same FFR model used in this study and developed a heuristic function to rank databases of ligands by scoring the most promising InhA drug candidates based on the assessment of physiochemical properties between the 3D ligand structures and the substrate-binding cavity from the MD receptor conformations. The result of this study is a list of 957 ligands, which in turn were sorted by the minimum predicted FEB values obtained from performing docking experiments into a set of 25 representative structures of the FFR model [Que16]. The first 89 compounds from this list of ranked compounds were selected to conduct the analysis described in this section.

6.4 e-FReDock Setup

The e-FReDock setup consists of installing and configuring e-SC system and MongoDB into the server machine, and attaching the VMs to the e-SC server. An Azure blob storage with 30 GB was used to deploy the e-SC server, and a hard disk with 40 GB was attached to the e-SC server on the CIC private cloud. After preparing the e-SC environment, each experiment executed into e-FReDock is created by using Create Experiment sub-workflow.

As described in Section 5.3.3, the input parameters and files for the method proposed, grid and docking are defined in this sub-workflow. For all experiments, the maximum number of energy was set to 500,000 and the number of runs was set to 20. The grid box was centered in the middle coordinates of each ligand with a dimension of 48 X 48 X 44 used for ZINC's compounds, and customized dimensions were configured from PDB's ligands. All ligands were treated as flexible during the docking experiments. A PDBQT file for each snapshot from the FFR model was created before starting the experiments and placed into the e-SC Share Library. We set the atom types used by AutoDock4.2, added the Kollman charges and merged all receptor snapshots from the FFR model with the non-polar hydrogens.

Input parameters related to the method were configured based on the empirical evaluation described in Chapter 4. We selected a parametrization for each of the two function proposed: batch analysis and cluster analysis. Both functions were first developed into the *AnalyseDockExperiment* block from Selective Ensemble Docking sub-workflow, and then executed to compare their accuracy in practical experiments. Table 6.4 specifies the parameters used to run the following scenarios:

- Scenario I: Execution of e-FReDock using the batch analysis function (Algorithm 4.1).
- Scenario II: Execution of e-FReDock using the cluster analysis function (Algorithm 4.2).

Table 6.4 – Set of settings employed to execute e-FReDock on Scenario I and II.

	PerSnapBatch	MinSnapBatch	MaxSnapBatch	StartAnalyses	PercDiscCluster	PercDiscBatch
Scenario I	20.00%	50	150	10.00%	-	40.00%
Scenario II	20.00%	50	150	15.00%	20.00%	50.00%

The virtual machines from Azure and CIC cloud platforms, which executed the workflow invocations, were selected based on the performance analyses presented in Section 6.2. We attached 10 D2 v2 Azure VMs into the e-SC server, where each VM was set to run 4 parallel workflow invocations (4 threads) for Scenario I, and 10 D3 v2 Azure VMs into the e-SC server with 8 parallel workflow invocations (8 threads) per VM for Scenario II. These VMs were chosen as they obtained the lowest cost among Dv2-series VMs by performing 100 docking simulations (Figure 6.3).

Another contributing factor behind the above choices was the efficiency rate, which reached 102% when D2 v2 was executed with four threads and 94.00% when D3 v2 was executed with 8 threads. We also observed that D3 v2 instance with 8 parallel workflow executions was able to achieve the same e-FReDock efficiency obtained by instances with higher number of cores (D4 and D12 v2 instances), which in turn presented higher costs to complete the docking experiments.

We limited the number of VMs to 10 considering two main aspects: cost and scalability. Before attaching the poll of VMs to the e-SC server, we investigated the best cost/performance ratio to choose a number of VMs that could fit our initial budget. Based on the docking parameters used in the performance analyses (Section 6.2), we estimated that D2v2 would take 16 days and D3 v2 would take 8 days to execute approximately 450,000 molecular docking simulations. Further, we also predicted the amount of experiments that could be performed based on the empirical tests described in Chapter 4. Considering that the percentage expectancy to reduce the number of snapshots processed for Scenario I was 50.00% and for Scenario II was 46.00%, it was estimated a total of 46 and 42 experiments for Scenario I and Scenario II, respectively. Thus, we believe that this quantity of experiments would be reasonable to validate the method proposed in this study.

The second aspect to consider in order to limit the number of VMs was scalability. Even though Figure 6.5 indicates that e-SC for e-FReDock on Azure D2 v2 VMs scales linearly with the addition of resources when longer molecular docking simulations are executed, the overall time to complete every compound screen is unpredictable. Furthermore, due to time constraints, the scalability of D3 v2 instance was not measured. Thus, it is uncertain whether the same performance achieved by D2 v2 instances will be obtained by D3 v2 instance when more than 10 VMs are attached to the e-SC server. Based on this hypothesis, we opted to use the same number of Large VMs in the CIC private cloud.

6.5 Evaluating e-FReDock Results

The assessment performed on e-FReDock results was divided in two distinct sets of analyses. The first set compares the predicted FEB and RMSD values from all snapshots that make up the FFR model and those selected by e-FReDock for different ligands. It aims to evaluate the binding affinity quality and the accuracy in final docking poses from the resulting RFFR models. The second set provides an analysis on the advantages of performing docking experiments in ensemble of MD conformations using e-FReDock. In this analysis, the minimal FEB values reached by RFFR models produced by different ligands were compared with those obtained by the FFR model in its rigid structure (PDB ID: 1ENY) [DQB+95].

6.5.1 Comparing the Accuracy of e-FReDock Results based on the entire FFR Model

The method proposed in this study aims at reducing the dimensionality of FFR models during the ensemble docking experiments without losing the most biological relevant information. To validate the accuracy of e-FReDock results, the method proposed is compared with an ensemble docking at random, composed by a reduced set of snapshots selected by chance to dock with a specific ligand. Thus, the following hypotheses are addressed:

- Null Hypothesis: the method does not result in gains.
- Alternative Hypothesis: the method results in gains.

To reject the null hypothesis, the accuracy of e-FReDock results should be higher than the selective ensemble docking at random, considering the same percentage of processed snapshots.

Aiming to compare the estimated accuracy between empirical and practical experiments, we submitted all 20,000 conformations that make up the FFR model and 17 distinct ligands to execute exhaustive molecular docking simulations. These simulations were performed on e-FReDock, which also allows to disable the quality analyses during docking runtime by processing 100% of the snapshots from the FFR model. The accuracy of results obtained from e-FReDock was evaluated by selecting the snapshots which are in the top 10, 20, 30, 100 and 200 best FEB values from the exhaustive ensemble docking results of each ligand. These assessments are shown in Table 6.5, where 9 known inhibitors of the InhA enzyme and 8 chemical compounds from ZINC database were used to compare the e-FReDock accuracy from Scenario I and Scenario II. Ligands from PDB were chosen based on the empirical results showed in Appendix B. The first 7 best experiments (4PI, GEQ, JPJ, 566, 468, 5PP, 665 and 8PC ligands) in the method's parametrization chosen to run cluster analysis function, and the well-known sub-micromolar inhibitor of the InhA enzyme (TCL ligand) were the criteria used to select the PDB's ligands. The set of compounds from ZINC database was chosen based on the 8 best predicted FEB values obtained from docking experiments of 25 representative structures from the FFR model performed by [Que16].

Table 6.5 shows that Scenario II outperforms Scenario I regarding the larger number of promising snapshots selected per ligand. One reason for this performance gains, is the random selection of snapshots applied to split cluster into batches in Scenario II. Another relevant aspect that also contributes to the quality improvements in Scenario II is the percentage of processed snapshots, which was 3.49% higher when compared with the average percentage of processed snapshots from all 17 ligands. However, we cannot support this information based on the average percentage of processed snapshots since e-FReDock assessments vary according to the ligand. For instance, 4PI and 91870997 ligands presented

Table 6.5 – Accuracy assessments in the e-FReDock scientific workflow for 17 different ligands.

		Scenario I - Batch Analysis Function						Scenario II - Cluster Analysis Function					
PDB ID	**Ligand**	**Proc. Snap. (%)**	**TOP10 (%)**	**TOP20 (%)**	**TOP30 (%)**	**TOP100 (%)**	**TOP200 (%)**	**Proc. Snap. (%)**	**TOP10 (%)**	**TOP20 (%)**	**TOP30 (%)**	**TOP100 (%)**	**TOP200 (%)**
2NSD	4PI	53.61	100.00	95.00	86.00	89.00	86.00	48.22	100.00	95.00	96.00	98.00	97.00
1P44	GEQ	48.92	100.00	95.00	93.00	86.00	86.00	49.71	90.00	85.00	83.00	83.00	84.00
3FNH	JPJ	47.89	100.00	100.00	100.00	96.00	93.00	50.15	100.00	100.00	96.00	94.00	95.00
2H7I	566	52.33	80.00	90.00	83.00	79.00	78.00	49.75	90.00	95.00	96.00	92.00	90.00
2H7P	468	49.84	90.00	95.00	83.00	82.00	82.00	53.10	100.00	100.00	96.00	89.00	90.00
2H7L	665	50.62	100.00	100.00	100.00	98.00	98.00	53.64	100.00	100.00	96.00	96.00	96.00
3FNE	8PC	52.46	90.00	90.00	90.00	93.00	90.00	52.04	100.00	95.00	93.00	92.00	92.00
2B35	TCL	50.67	80.00	55.00	60.00	75.00	80.00	52.42	80.00	80.00	80.00	87.00	87.00
2B36	5PP	50.97	60.00	70.00	66.00	67.00	73.00	48.51	80.00	75.00	83.00	73.00	77.00
-	91870997	47.23	90.00	90.00	93.00	90.00	86.00	44.88	80.00	80.00	76.00	78.00	77.00
-	35361468	43.79	100.00	90.00	86.00	89.00	86.00	55.73	90.00	90.00	86.00	85.00	82.00
-	12047789	42.64	100.00	100.00	100.00	92.00	88.00	53.75	100.00	90.00	86.00	83.00	81.00
-	56919632	46.33	100.00	100.00	93.00	85.00	84.00	49.81	100.00	95.00	90.00	85.00	86.00
-	63479951	35.30	80.00	75.00	73.00	72.00	74.00	45.69	100.00	100.00	96.00	86.00	84.00
-	4073149	45.45	60.00	65.00	70.00	66.00	64.00	56.14	90.00	80.00	76.00	80.00	79.00
-	39532319	47.03	90.00	90.00	90.00	88.00	85.00	51.44	90.00	90.00	93.00	94.00	93.00
-	34378053	45.03	90.00	90.00	93.00	87.00	84.00	54.65	100.00	100.00	96.00	92.00	92.00
Average	-	**47.67**	**88.82**	**87.65**	**86.59**	**84.65**	**83.82**	**51.16**	**93.53**	**91.18**	**89.29**	**87.47**	**86.76**

less percentage of processed snapshots for Scenario II, while their accuracy were kept and even enhanced when compared with the top best assessments in Scenario I.

Regarding the empirical experiments expectancy, both scenarios outperform accuracy and increase the number of discarded snapshots in practical experiments. While the average percentage of processed snapshots decreased approximately 2.50% in both methods, the average percentage of the top best selected snapshots ranges from 83.82% to 87.65%, and from 93.53% to 86.76% for Scenario I and Scenario II, respectively. These ranges in empirical experiments was from 73.00% to 77.00%, and from 84.00% to 85.00% for Scenario I and Scenario II, respectively. It is not surprising that e-FReDock results outperform empirical experiments, since most of ligands used to assess the accuracy are structurally different, especially those from ZINC database. Empirical experiments were used as a baseline accuracy, which may be extended to other method's parametrization reported in Appendixes A and B along with the findings presented in Table 6.5.

To validate the accuracy of the resulting RFFR models, we also assessed the RMSD values from the snapshots selected by e-FReDock for six known ligands of the InhA enzyme. Figure 6.6 compares the variation in the RMSD values between the FFR model and the produced RFFR models in Scenario I and Scenario II. As can be seen in the graph from Figure 6.6, boxplots from Scenario I and II report central tendencies and dispersions lower than the FFR model. It can therefore assumed that e-FReDock was able to select snapshots with the best docking final poses for each ligand, even though the method proposed to optimize docking-based virtual screening in FFR models is based only on FEB values.

Comparing boxplots from Scenario I and II, Figure 6.6 shows that the difference in central tendencies and quartiles is minimal. For instance, 2H7L_665 and 2B35_TCL ligands present their central tendencies lower in Scenario I but their lower quartiles and minimum values are equals. Furthermore, except for 1P44_GEQ ligand, Scenario II has central tendencies and minimum values lower or equal to Scenario I for all tested ligands. These results provide further support for the alternative hypothesis, since the proposed method is able to select a considerable number of snapshots that contains the best binding affinity as well as high quality in their docking final poses.

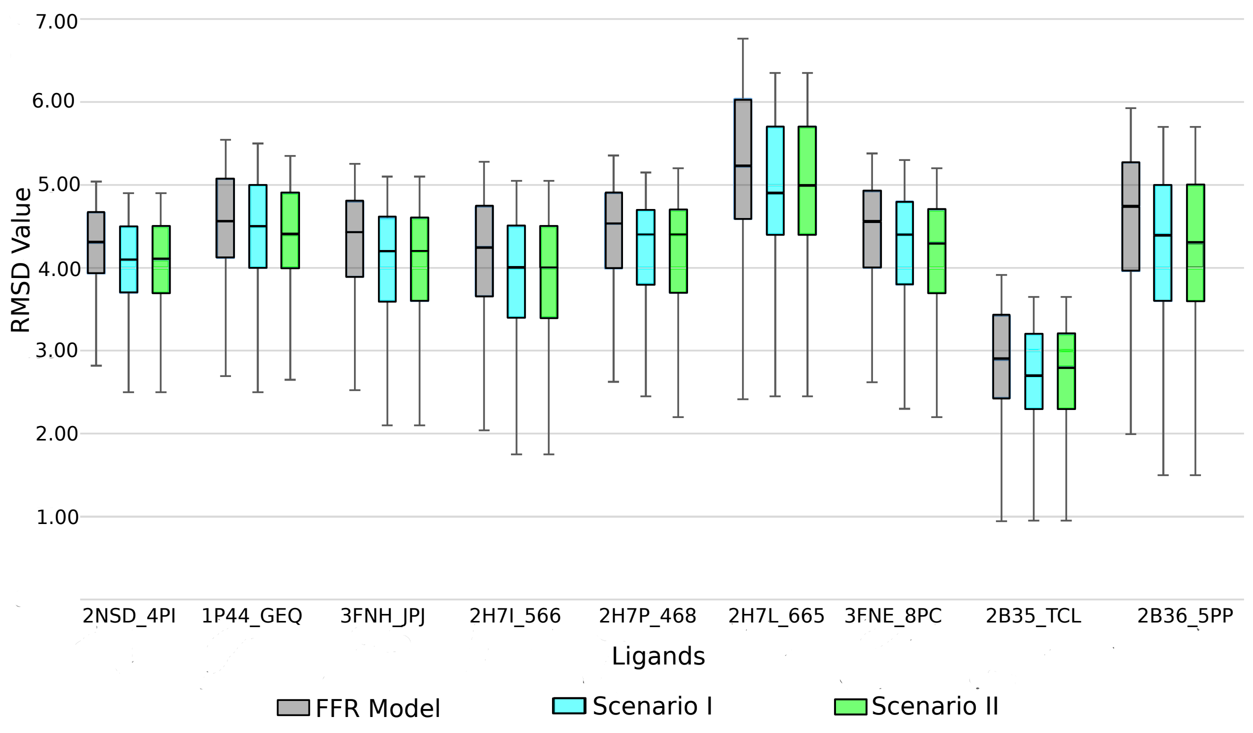


Figure 6.6 – Comparison between the RMSD values obtained by the FFR model and the resulting RFFR models for six InhA's known ligands. Boxplots represent the trends in RMSD values changes per ligand for Scenario I and II. Values range from the first quartile to the third with the median RMSD values denoted by the black line across the central box region. Each set of docking results is represented by different colors, where blue and green are RMSD values from Scenario I and Scenario II, respectively, and gray is the RMSD values from the FFR model.

6.5.2 Comparing the Accuracy of e-FReDock Results based on 1ENY Crystal Structure

The second set of analyses aims at assessing the gains/losses obtained by performing molecular docking simulations in an ensemble of dynamic structures, based on the RFFR models produced by e-FReDock. For this purpose, we compare the minimal FEB values achieved from the interactions between the tested ligands and the crystal structure that originates the FFR model, considering the following hypotheses:

- Null Hypothesis: the 1ENY crystal structure reached predicted FEB values better than the RFFR model.

- Alternative Hypothesis: the RFFR model reached predicted FEB values better than the 1ENY crystal structure.

If the null hypothesis is rejected, then the selective method to strategically reduce the number of MD conformations processed during the molecular docking simulations was able to achieve benefits, which would be unreachable by the rigid version of the FFR model.

A total of 139 experiments were performed in e-SC, 101 for Scenario I and 38 for Scenario II. The snapshots that reached the best predicted FEB values, along with their clusters, batches and ligand identifiers are shown in Appendix C. To validate the hypotheses above described, we compared each best FEB value achieved by the interactions between crystal structure and ligands based on the evaluation method proposed by Quevedo [Que16]. Quevedo's method consists of classifying each e-FReDock experiment as follows [Que16]:

1. RFFR model winner: If the best predicted FEB value reached by the RFFR model outperforms the best predicted FEB value from the 1ENY structure in more than 1 kcal/mol.
2. Tie for RFFR model: If the best predicted FEB value reached by the RFFR model outperforms the best predicted FEB value from the 1ENY structure with a difference equal to or less than 1 kcal/mol.
3. Tie for 1ENY structure: If the best predicted FEB value reached by the 1ENY structure outperforms the best predicted FEB value from the RFFR model with a difference equal to or less than 1 kcal/mol.
4. 1ENY structure winner: If the best predicted FEB value reached by the 1ENY structure outperforms the best predicted FEB value from the RFFR model in more than 1 kcal/mol.

The rate threshold of 1 kcal/mol used to identify the winners is based on the set of docking experiments performed by Morris et al. [MHL$^+$09] to validate accuracy on Autodock4.2. They concluded that AutoDock4.2 was able to satisfactorily predict the binding affinities for about 80.00% of docking results when the final pose and the FEB value vary up to 2.0 Å and 1 kcal/mol from the crystal structure. Table 6.6 summarizes the results obtained by executing e-FReDock in both scenarios according to the four categories used in this set of analysis. Surprisingly, the resulting RFFR models were able to outperform the crystal structure version of the FFR model for all ligands used in both scenarios. Most of ligands that showed differences with the crystal structure were extracted from other InhA crystal structures available at the PDB. We believe this is due to the fact that InhA crystal structures from ligands used are unable to be reproduced by any MD conformations of the FFR model.

Table 6.6 – Docking experiment analyses from the RFFR models produced by e-FReDock and the 1ENY crystal structure with 14 PDB's ligands and 89 ZINC's compounds for Scenario I and II.

	Database	Total Ligands	Winner 1ENY	Winner RFFR Model	Tie for 1ENY Structure	Tie for RFFR Model
Scenario I	PDB	12	0.00%	25.00%	25.00%	50.00%
	ZINC	89	0.00%	87.64%	0.00%	12.36%
Scenario II	PDB	14	0.00%	21.43%	14.28%	64.29%
	ZINC	24	0.00%	87.50%	0.00%	12.50%

Conversely, Table 6.6 indicates that compounds from ZINC database show greater binding affinities for the FFR model used in this study, even when they are in a case of tie.

In addition to comparing with the 1ENY crystal structure, the four categories used to classify the e-FReDock results were also employed to compare the quality of the RFFR models produced in this study and the representative set of snapshots used by Quevedo [Que16]. Quevedo [Que16] generated a set of 25 representative snapshots from the FFR model to be docked with 957 compounds from ZINC database. We found that the results produced by e-FReDock were able to outperform the crystal structure better when compared to the set of representative structures generated by Quevedo. Table 6.6 indicates that the percentage of RFFR model winner and tie are 87.64% and 12.36%, while the set of representative snapshots from Quevedo for the same categories and compounds tested were 32.00% and 54.00%, respectively (see [Que16]). Further, the 1ENY crystal structure outperformed the set of representative snapshots in 3.00% of the 89 compounds. By analyzing the results obtained by Quevedo, we conclude that our solution is capable of reducing the dimensionality of the FFR model, maintaining a high level of quality in the RFFR models produced for distinct ligands.

6.6 General Remarks

This section analyzed the results obtained by executing e-FReDock with the functions developed to reduce the size of FFR models without losing relevant biological information. We also investigated the performance gains on Azure Dv2-series and CIC Large VMs by using a small set of snapshots from the FFR model. The best pool of cloud VMs settings was selected to run the docking-based virtual screening experiments in e-FReDock.

To assess the accuracy of the results obtained from the experiments performed in this study, we created two different hypotheses based on analyzing the quality of RFFR models produced. Section 6.5.1 showed that the average reduction in the FFR model size by executing e-FReDock with 17 different ligands was, on average, 51.16%, while the selection in the best 10, 20, 30, 100 and 200 receptor-ligand interactions was, on average, in the range of 84.65% and 93.53%, where the highest percentages were reached by TOP 10 best docking results. These results reject the null-hypothesis, defined as *the method does not result in gains*, since a random selection of 9,976 snapshots (equivalent to the 51.16%) from the 20 ns InhA MD trajectory would statistically select roughly 50.00% of the best 10, 20, 30, 100 and 200 receptor-ligand interactions from the FFR model. However, Table 6.5 shows that the lowest percentage of the top docking selections by analyzing each experiment individually was 73.00% for the best 100 interactions between the FFR model and 2B36_5PP ligand.

Besides investigating the binding affinity achieved by the ligand tested, we also extracted the RMSD values from the 1ENY's known ligands to asses the final pose quality. Based on this analysis, we concluded that, in addition to select the best binding affinity from the FFR model under study, e-FReDock was also able to select the best docking poses through the FEB-based selective method proposed in this study.

Section 6.5.2 described the analyses performed to validate the second hypothesis from e-FReDock results. It verified if the RFFR models produced by e-FReDock were able to outperform the crystal structure from the FFR model, as well as evaluated the gains achieved by including flexibility of the 1ENY structure (target receptor) in the docking procedures, even with only 50.00% of processed snapshots. A total of 102 distinct ligands were executed by e-FReDock in Scenario I and II. Most of these experiments rejected the null hypothesis, since the best predicted FEB values achieved from the RFFR models outperformed the best predicted FEB values achieved from the 1ENY structure. Precisely, 97.00% of the ligands tested indicated the predicted FEB values from the RFFR models better than 1ENY crystal structure. We noticed that a high percentage of ligands from PDB presented differences in the FEB values for the RFFR model and 1ENY structure. This is not surprising, because even though 75.00% of them outperformed 1ENY structure, these ligands were not selected from the heuristic function that ranked the most promising InhA drug candidates [Que16], and they probably establish unfavorable interactions with the FFR model under study.

The e-FReDock scientific workflow generated a total of 1,253,887 Selective Ensemble Docking sub-workflow invocations, among which 440,492 were executed on Azure cloud platform. In total 292.61 hours were taken for D2 instance (e-SC server), 1,154.74 hours for 10 D2 v2, and 1,360.08 hours for 10 D3 v2 instances (e-SC engines). Even though D3 v2 instances are more powerful and spent more hours than D2 v2, the amount of workflow invocations executed by D3 v2 was only 35,695 more than D2 v2 instances. Table 6.7 depicts the Azure costs regarding the computation and data storage. According to the performance evaluation on Dv2-series Azure instances showed in Section 6.2.1, we estimated that the cost for executing the same quantity of workflow invocations in D2 v2 and D3 v2 instances would be similar, but D2 v2 instances would take twice the time of D3 v2 instances. The first reason for this lack of performance in D3 v2 instances was the two VMs failures faced during the experiments, which executed workflow invocation errors for approximately

8 hours in one VM and 3 hours in the other. The second reason is related to the fact that different ligands were submitted to Scenario I and Scenario II, with fastest docking performed on D2 v2 (Scenario I).

Table 6.7 – Cost specification spent to run e-FReDock on Azure cloud platform

Cost Description	**Price (US$)**
e-FReDock Deployment	10.06
e-FReDock Performance Tests	22.16
e-FReDock for Scenario I	360.06
e-FReDock for Scenario II	438.75
Blob Storage	80.51
File Transfer	6.48
Total	**918.02**

7. RELATED WORKS

A number of different methods in the literature propose to reduce the overall time taken for performing docking-based virtual screening. Most of these methods perform docking experiments with flexible ligands, but rigid receptor. As described in Chapter 2, considering the flexibility of receptor in docking simulations is still a challenging task. In order to describe the current works associated with the contributions presented in this study, we classify them into three topics: .

1. High performance environments to optimize docking-based virtual screening;
2. Approaches to reduce the ensemble of MD conformations for docking simulations;
3. Methods to optimize Molecular Docking Simulations of FFR Models.

7.1 High Performance Environments to Optimize Molecular Docking Simulations

Several computational approaches have focused on reducing the elapsed time taken to perform virtual screening of small molecules against rigid receptors using High Performance Computing (HPC) environments, such as computing clusters, grids [FK03], and clouds [CSSB11, ZWL13, KBT$^+$14, EB14, OBDO$^+$14, NMT$^+$15]. Most of these methods treat the receptor as rigid bodies to scale up the simulations based on the volume of compounds to be docked. For instance, Collignon et al. [CSSB11] and Zhang et al. [ZWL13] enhanced the performance of high-throughput virtual screening executing simultaneous docking experiments on a large number of processors from a Linux cluster using AudoDock4.2 [MHL$^+$09]. Unlike these studies, e-FReDock was developed to execute simultaneously AutoDock4.2 in worker nodes from clusters, grids or cloud VMs.

A variety of docking environments deployed on cloud platforms to enhance the performance of docking simulation of rigid receptors is available. Kiss et al. [KBT$^+$14] performed virtual screening practices by using VMs from Azure cloud platform [AZU16]. They compared the scalability of docking experiments using 5, 10 and 20 Azure VMs with a grid structure and analyzed the performance gains achieved in each environment. In a recent work, Nguyen et al. [NMT$^+$15] proposed a multi-site cloud environment for molecular virtual screening. They combined small allocations of VMs in multiple locations (Azure WestUS, University of Florida and University of California) connected through a virtual networking system, and compared the differences in performance with the individual cloud sites by running low accuracy screening (docking completion time of approximately 10 seconds/compound), using DOCK program [MLP$^+$06]. Ocaña et al. [OBDO$^+$14] also analyzed the performance gains obtained by scaling large scale receptor-ligand docking experiments out on cloud VMs in a different scenario. They developed a docking-based virtual screening workflow and compared a total of 10,000 receptor-ligand interactions using AutoDock4 and AutoDock Vina [MHL$^+$09]. Even though all these studies provided advances to facilitate large scale docking experiments, their environments were developed to analyze rigid receptors. Our study is focused on the development of an environment that can be deployed on different HPC environments, in order to optimize docking-based virtual screening of FFR models.

7.2 Approaches to Reduce the Ensemble of MD Conformations

To make feasible the incorporation of protein flexibility in docking experiments, some studies attempted on reducing the ensemble of MD conformations to a manageable size generating a small set of representative MD conformations from partitions of MD trajectories [LZ06, HZ07, STTC07, ZCZ$^+$08, LAB$^+$08, LZ12, FTW12]. Such partitions are generated from clustering methods, which use measure of similarity to group MD conformations. RMSD values obtained by pairwise or matrix error distances is the most traditional and popular measure of similarity used by clustering methods. For instance, Lyman and Zuckerman [LZ06] generated a set of reference structures by enforcing a cutoff radius in RMSD for cluster assignment from biomolecular simulation trajectories of metenkephalin, a pentapeptide neurotransmitter. Shao et al. [STTC07] make use of several clustering algorithms and two validity metrics to find the best clustering partition based on the pairwise RMSD values of small samples from an MD trajectory. Even though the meaningful trajectories cover very different portions of the conformation space, Shao et al. [STTC07] limited the structural metrics by using only a portion of the data, and then the remaining data were added to existing clusters. However, the RMSD may not be the most appropriate measure of similarity to cluster conformations for docking simulations, since it is influenced by changes that occur on the whole protein. In contrast to previous studies, which use pairwise RMSD vales as measure of similarity, we concentrate our efforts on identifying small and localized changes that occur in the cavity of FFR models. Features from the substrate-binding cavity can play a role in providing potential ligands to FFR models, and may identify all possible interactions of the particular ligand with the alternative receptor conformations.

Another approaches to reduce the ensemble of MD conformations for docking simulations are presented by Flick et al. [FTW12] and Leis and Zacharias [LZ12]. While Flick et al. [FTW12] confine the protein conformational changes to the relevant flexible sections of the backbone, Leis and Zacharias [LZ12] represent a flexible receptor by a series of potential grids, each corresponding to one discrete receptor. Indeed, limiting flexibility regions makes the problem much more tractable [TA08]. However, depending on the system, methods that treat only partial protein flexibility may not be sufficient for modeling realistic molecular interactions between ligands and receptors. A different approach to reduce the number of receptor conformations is presented by Huang and Zou [HZ07]. They developed a method to automatically select an optimal structure that contains the best fit with the ligand from an ensemble of experimental structures based on a SIMPLEX local optimization procedure. Although this method provided satisfactory results, the authors applied an unusual evaluation methodology, making it a target of criticism [TA08].

7.3 Methods to Reduce the Molecular Docking Simulations of FFR Models

FReDoWS [MSR$^+$11] and wFReDoW [DPFNdSR13] are applications developed in our research group with the intention of optimizing molecular docking simulations of FFR models by eliminating poor receptor snapshot and ligand interactions. FReDoWS [MSR$^+$11] is a scientific workflow that accelerates molecular docking experiments based on docking results obtained from an exhaustive execution of an FFR-ligand docking simulation. The main drawbacks of FReDoWS are the time taken to perform a full ensemble docking experiment for each new ligand, and the serial execution of AutoDock, by which one docking experiment is processed on one CPU for a given snapshot.

P-SaMI [Hüb10] is also a data pattern for scientific workflows, whose purpose is to identify cluster of promising snapshots at docking runtime from a clustering of snapshots. It is the preliminary version of the method proposed in this study (Chapter 4). Hübler [Hüb10] created and validate the P-SaMI hough empirical tests in a small FFR model with only two ligands. The deployment of P-SaMI was presented by De Paris et al. [DPFNdSR13] and Quevedo et al. [QDPRNdS14]. De Paris et al. [DPFNdSR13] developed wFReDoW, the cloud-based environment to handle molecular docking simulations of FFR models based on P-SaMI specifications. The main objective of this thesis is similar to wFReDoW approach: generating RFFR models by discarding clusters of unpromising snapshots at docking runtime, preserving the quality of the original FFR models. Even though De Paris et al. [DPFNdSR13] and Quevedo et al. [QDPRNdS14] presented promising RFFR models, wFReDoW has the following drawbacks:

- It does not perform docking-based virtual screening since only one experiment, which constitutes docking simulations between an FFR model and a ligand, is allowed to be executed at a time.
- It is not a scalable solution since it uses the MPI cluster model to execute parallel docking experiments.
- It requires expert domain skills to provide information about interactions between the FFR model and a specific ligand.

- It does not provide a friendly user interface. Input files and system are manually prepared by the user, being thus prone to failures.

The e-FReDock workflow developed in this study is able to address wFReDoW limitations above described. Besides creating a novel environment to outperform wFReDoW, this study also addresses P-SaMI's disadvantages by developing an effective method to perform selective ensemble docking experiments for large number of ligands on cloud platforms.

8. FINAL CONSIDERATIONS

With the advances in X-ray crystallography and nuclear magnetic resonance spectroscopy, the number of protein structures available in biological databases has been increasing. This encourages the pharmaceutical industry that expects to benefit from the wealth of data on new targets, and hence produce newer and more efficient drugs [GAG11]. Molecular docking simulations is an integral part of current drug design efforts. The quantity of docking data has grown due to the amount of putative ligands and several parameters used for the purpose of truly mimic biological systems in natural environments, especially those considering the explicitly plasticity and flexibility of receptors and ligands. The major challenge behind these experiments is to achieve an optimal compromise between satisfactory accuracy and computational effort [BZ10].

This study aimed to make progress on reducing the computational cost involved in using FFR models to perform practical virtual screening in database of small molecules. For this purpose, we created e-FReDock, a cloud-based scientific worklfow to optimize intensive molecular docking simulations of FFR models, based on the two questions described in the Introduction of this thesis. The first question was solved by developing a scientific workflow, and attaching a set of cloud VMs to it in order to balance the computational efficiency and prediction accuracy in docking-based virtual screening of FFR models. The answer to the second question is the free-parameter method described in Chapter 4. This method was developed to eliminate poor interactions between snapshots of the FFR model and small molecules to generate RFFR models that best complements the shape of specific ligands. e-FReDock, which is based on this method, was developed in the e-SC workflow enactment system [HWWC13] and deployed on two cloud platforms: Microsoft Azure public cloud [AZU16] and Cloud Innovation Centre (CIC) private cloud [Hor16]. The results are very promising.

This thesis started by describing the theoretical principles for the understanding problem at study. Chapter 2 provided explanations on the molecular docking procedure, the existing approaches for considering the flexibility of receptors, highlighting those used in this study, and the model used to perform all experiments. This chapter also presented a brief overview of the main features of scientific workflow, pointing out the four phases of its life cycle. It ended giving a better comprehension of the features, services and architecture from cloud computing, used to deploy e-FReDock and run the selective ensemble docking experiments.

Chapter 3 reported the six clustering methods applied to group snapshots of the FFR model with similar features in their substrate-binding cavities. To assess the performance gains of this novel measure of similarity, traditional clustering-based RMSD approaches were also submitted to the two partitioning and four hierarchical clustering algorithms. The optimal clustering solution was selected by evaluating binding conformations of drug candidates to the InhA enzyme, performed by cross-docking experiments between the FFR model and 20 different ligands experimentally tested. Furthermore, they also revealed that the binding cavity features outperform other two traditional RMSD measures of similarity in the clustering algorithms applied in this study.

Chapter 4 specified the method developed to discard groups of unpromising snapshots at docking runtime based on P-SaMI [HRFNdS15]. The input for this method is a ligand, the optimal clustering solution from Chapter 3, and a set of method and docking parameters. The clustering of snapshots is split into small and balanced batches to improve the analyses of protein-ligand interactions. The set of method parameters contains information regarding the clusters/batches controls. Unlike P-SaMI data pattern, no specific information about interactions between the protein and ligands or previous docking experiments is required in order to benefit from the proposed method. Our method is able to identify ensemble of snapshots with proper conformational states to accommodate a particular ligand without previous information from a domain expert. To prioritize putative hits and select the better protein-ligand interactions at docking runtime, two distinct analysis functions were proposed: batch and cluster analysis functions. The batch analysis function discards a batch if its processed snapshots present unsatisfactory FEB values. Alternatively, the cluster analysis function discards a batch if the processed batches belonging to the same cluster present poor quality in their snapshots. Such functions were validated by performing a set of empirical experiments, from which a set of parameters was chosen to run experiments in the e-FReDock workflow.

Chapter 5 introduced e-FReDock, the scientific workflow built in the e-SC enactment system [HWWC13] that reduces the computational time involved in using FFR models to perform practical virtual screening of ligands. It is composed of two sub-workflows: (i) the Create Experiment, which prepares and insert new docking experiments of FFR models with different ligands; and (ii) the selective ensemble docking, which executes docking experiments, analyses docking results, and discards snapshots based on the best binding free energies for the ligand and snapshots already processed. This chapter also presented the e-SC API component used to monitor the selective ensemble docking sub-workflow invocations based on batches's priority and the number of VMs linked to the e-SC system. A strategic function was created in the e-SC API to give more virtual processors to batches with high priority (higher than 3), and less to batches with low priority (less or equal to 3).

The experimental results from e-FReDock conceptual architecture were presented on Chapter 6. First, a sample of snapshots from the FFR model in study was executed on e-FReDock in Azure and CIC private cloud platforms. The performance results showed that running e-FReDock on small VMs (2 or 4 cores) is more efficient than performing the same workload on 16 core VMs. In addition, it was found that the amount of RAM does not affect the docking simulation completion time. Based on this findings, a set of VMs were attached to the e-SC server from each cloud platform in order to scale up selective ensemble docking workflow invocations. The foremost contribution presented in Chapter 6 is the high accuracy obtained from e-FReDock results when the dimensionality of the model is reduced by 50.00%. To evaluate the quality of e-FReDock results, two sets of analyses were presented. The first analysis showed that e-FReDock was able to preserve between 85.00 and 90.00% of the quality of the FFR models, while the dimensionality of the model reduced on average 49.68%. The second analysis reported that the predicted FEB values achieved by the resulting RFFR models outperformed the predicted FEB values obtained from the rigid structure of the FFR model in 97.00% of the ligands tested.

Chapter 7 showed the state of art associated with this study, focusing on the key contributions this thesis compared to the current works of scientific community.

8.1 Limitations

The generalizability of e-FReDock is subject to certain limitations. For instance, if a new FFR model is introduced to execute the method proposed, a novel clustering of snapshots should be investigated and generated. The clustered FFR model used as input to the e-FReDock was generated by partitioning features from the substrate-binding cavity of each MD conformation using the complete linkage method. Chapter 3 showed that this solution was selected by comparing the level of similarity between the medoids from each partition and the MD's full trajectory when two traditional RMSD similarity function, and the cavity attributes were used as input data to six different clustering methods. This study revealed the high level of accuracy achieved by the partitioning solutions when the features from substrate-biding cavity of each MD conformation was used as input to the hierarchical clustering methods (see Figure 3.5). This means that the set of substrate-binding cavity features is a promising measure of similarity for MD trajectories, and it can be extended to other protein/receptors. The only constraint is that the binding pocket from the FFR models should be known in advance.

The quality of the clustering solution is crucial for the proper operation of the method proposed in Chapter 4. To obtain high quality in the RFFR models produced by e-FReDock, the clustering methods and the input dataset should group snapshots that are as homogeneous as possible. Chapter 6 showed the gains obtained by using e-FReDock with the optimal clustering solution selected in Chapter 3. However, to obtain similar or greater level of quality to those achieved in the RFFR models generated in this study, the clustering of snapshots should present high quality.

Finally, the overhead imposed when fast docking experiments are performed using more than 8 VMs identified in Chapter 6, Section 6.2.1, is also a current limitation of the e-FReDock. A way to improve e-FReDock performance can be the creation of threads of execution to control the queue of tasks rather than controlling the queue of tasks sequentially.

8.2 Future Works

In addition to the promising results showed in this study, more research is needed to further reduce the ensemble of snapshots from the RFFR models in a manageable size. An interesting possibility is to perform a careful analysis on docking results to select a small set of representative candidate poses of the resulting RFFR models, for example, by using clustering methods to partition snapshots into groups of dissimilar features. This would avoid the grunt work of manually selecting the potential ligand poses by the domain expert.

A natural progression of this study is to work on the limitations described in the previous section. The empirical tests described in Chapter 4, Section 4.4, are used to validate the method proposed and identify a suitable set of parametrization to execute the selective ensemble docking experiments. Results and analyses presented in this study are very useful to perform new experiments with the same FFR model. However, further empirical tests are needed in order to determine a suitable set of parameters when new FFR models are introduced to e-FReDock. Thus, further studies on methods to automatically create batches and discard them during docking runtime, instead of depending on a set of parameters, would add knowledge to the field.

Another direction for future research would be to examine FFR InhA model interactions with a larger number of compounds, particularly those already ranked as drug candidates to the FFR model by Quevedo [Que16]. This can assist in discovering new potential lead compounds to the InhA enzyme, as well as support the method proposed in this study.

8.3 Publications

Throughout the research carried out during my PhD, I have published conference and journal papers. Some of them are directly related to this thesis, whereas others are only indirectly-related to main topic presented.

The following papers refer to the approach I developed and co-developed to the area of clustering methods for partitioning MD trajectories:

- De Paris, R.; Quevedo, C.V; Ruiz, D.D.; and Norberto de Souza, O. An Effective Approach for Clustering InhA Molecular Dynamics Trajectory Using Substrate-Binding Cavity Features. *PloS One*, 10(7), 1-25, 2015.

- De Paris, R.; Quevedo, C.V.; Ruiz, D.D.; Norberto de Souza, O.; and Barros, R.C. Clustering Molecular Dynamics Trajectories for Optimizing Docking Experiments. *Computational Intelligence and Neuroscience*, 2015.
- Barros, R.C.; Quevedo, C. V.; De Paris, R.; and Basgalupp, M. P. Clustering Molecular Dynamics Trajectories with a Univariate Estimation of Distribution Algorithm. *IEEE Congress on Evolutionary Computation (CEC)*, 2058-2065, 2015.

The following paper refers to the preliminary conceptual architecture of e-FReDock:

- De Paris, R.; Ruiz, D. D.; and Norberto de Souza, O. A Cloud-Based Workflow Approach for Optimizing Molecular Docking Simulations of Fully-Flexible Receptor Models and Multiple Ligands. *IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)*, 495-498, 2015.

The following papers refer to ensemble docking experiments I performed for two different clustered FFR models of InhA using wFReDoW:

- Quevedo, C.V.; De Paris, R.; Ruiz, D.D; and Norberto de Souza, O. A Strategic Solution to Optimize Molecular Docking Simulations Using Fully-Flexible Receptor Models. *Expert Systems with Applications*, 41, 7608-7620, 2014.
- De Paris, R.; Frantz, F.A.; Norberto de Souza, O.; and Ruiz, D.D. wFReDoW: A Cloud-Based Web Environment to Handle Molecular Docking Simulations of a Fully-Flexible Receptor Model. *BioMed research international*, 2013.

The following abstract refers to works performed from our research group (Labio and Gpin) during 10 years:

- Cunha, H.; De Paris, R.; Quevedo, C.V.; Ruiz, D.D.; and Norberto de Souza, O. Recent Advances in Molecular Docking Experiments of Fully-Flexible Receptor Models. *In: Poster papers of the ISCB-Latin America together with X-Meeting, BSB and SoiBio*, 2014.

REFERENCES

[ABG06] Alonso, H.; Bliznyuk, A. A.; Gready, J. E. “Combining docking and molecular dynamic simulations in drug design”, *Medicinal Research Reviews*, vol. 26–5, 2006, pp. 531–568.

[ABJ$^+$04] Altintas, I.; Berkley, C.; Jaeger, E.; Jones, M.; Ludascher, B.; Mock, S. “Kepler: an extensible system for design and execution of scientific workflows”. In: International Conference on Scientific and Statistical Database Management, 2004, pp. 423–424.

[ABM08] Amaro, R. E.; Baron, R.; McCammon, J. A. “An improved relaxed complex scheme for receptor flexibility in computer-aided drug design”, *Journal of Computer-Aided Molecular Design*, vol. 22–9, 2008, pp. 693–705.

[ACBI09] Armen, R. S.; Chen, J.; Brooks III, C. L. “An evaluation of explicit receptor flexibility in molecular docking using molecular dynamics and torsion angle molecular dynamics”, *Journal of Chemical Theory and Computation*, vol. 5–10, 2009, pp. 2909–2923.

[ADK15] Antunes, D. A.; Devaurs, D.; Kavraki, L. E. “Understanding the challenges of protein flexibility in drug design”, *Expert Opinion on Drug Discovery*, vol. 10–12, 2015, pp. 1301–1313.

[AFG$^+$10] Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A. D.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I.; Zaharia, M. “A view of cloud computing”, *Communications of the ACM*, vol. 53–4, 2010, pp. 50–58.

[AL10] Amaro, R. E.; Li, W. W. “Emerging methods for ensemble-based virtual screening”, *Current Topics in Medicinal Chemistry*, vol. 10–1, 2010, pp. 2–13.

[AMA16] AMAZON. “Amazon web services”. Source: http://aws.amazon.com/, Accessed at Jun 2016.

[ARMG15] Aboutorabi, S. H.; Rezapour, M.; Moradi, M.; Ghadiri, N. “Performance evaluation of SQL and MongoDB databases for big e-commerce data”. In: International Symposium on Computer Science and Software Engineering, 2015, pp. 1–7.

[AZU16] AZURE. “Microsoft Azure: The cloud for modern business”. Source: http://azure.microsoft.com/en-us/, Accessed at Jun 2016.

[BM05] Barril, X.; Morley, S. D. “Unveiling the full potential of flexible receptor docking using multiple crystallographic structures”, *Journal of Medicinal Chemistry*, vol. 48–13, 2005, pp. 4432–4443.

[BNL03] Binkowski, T. A.; Naghibzadeh, S.; Liang, J. “CASTp: computed atlas of surface topography of proteins”, *Nucleic Acids Research*, vol. 31–13, 2003, pp. 3352–3355.

[BQDPB15] Barros, R. C.; Quevedo, C. V.; De Paris, R.; Basgalupp, M. P. “Clustering molecular dynamics trajectories with a univariate estimation of distribution algorithm”. In: IEEE Congress on Evolutionary Computation, 2015, pp. 2058–2065.

[BRM15] Buonfiglio, R.; Recanatini, M.; Masetti, M. “Protein flexibility in drug discovery: From theory to computation”, *ChemMedChem - Chemistry Enabling Drug Discovery*, vol. 10–7, 2015, pp. 1141–1148.

[BWF$^+$00] Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T.; Weissig, H.; Shindyalov, I. N.; Bourne, P. E. “The protein data bank”, *Nucleic Acids Research*, vol. 28–1, 2000, pp. 235–242.

[BYV$^+$09] Buyya, R.; Yeo, C. S.; Venugopal, S.; Broberg, J.; Brandic, I. “Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility”, *Future Generation Computer Systems*, vol. 25–6, 2009, pp. 599–616.

[BZ10] Beier, C.; Zacharias, M. “Tackling the challenges posed by target flexibility in drug design”, *Expert Opinion on Drug Discovery*, vol. 5–4, 2010, pp. 347–359.

[CDCI$^+$16] Case, D.; Darden, T.; Cheatham III, T.; Simmerling, C.; Wang, J.; Duke, R.; Luo, R.; Walker, R.; Zhang, W.; Merz, K.; et al.. “AMBER 12”. Source: http://ambermd.org/, Accessed at Jul 2016.

[Cho13] Chodorow, K. “MongoDB: the definitive guide”. Sebastopol, CA: O’Reilly Media, 2013, 409p.

[CHWW13] Cała, J.; Hiden, H.; Woodman, S.; Watson, P. “Cloud computing for fast prediction of chemical activity”, *Future Generation Computer Systems*, vol. 29–7, 2013, pp. 1860–1869.

[CKS$^+$08] Cozzini, P.; Kellogg, G. E.; Spyrakis, F.; Abraham, D. J.; Costantino, G.; Emerson, A.; Fanelli, F.; Gohlke, H.; Kuhn, L. A.; Morris, G. M.; Orozco, M.; Pertinhez, T. A.; Rizzi, M.; Sotriffer, C. A. “Target flexibility: An emerging consideration in drug discovery and design”, *Journal of Medicinal Chemistry*, vol. 51–20, 2008, pp. 6237–6255.

[CL09] Chen, J. Y.; Lonardi, S. “Biological data mining”. New York: CRC Press, 2009, 733p.

[CSSB11] Collignon, B.; Schulz, R.; Smith, J. C.; Baudry, J. “Task-parallel message passing interface implementation of autodock4 for docking of very large databases of compounds using high-performance super-computers”, *Journal of Computational Chemistry*, vol. 32–6, 2011, pp. 1202–1209.

[DB79] Davies, D. L.; Bouldin, D. W. “A cluster separation measure”, *IEEE Transactions on Pattern Analysis Machine Intelligence*, vol. PAMI-1–2, 1979, pp. 224–227.

[DC07] Damm, K. L.; Carlson, H. A. “Exploring experimental sources of multiple protein conformations in structure-based drug design”, *Journal of the American Chemical Society*, vol. 129–26, 2007, pp. 8225–8235.

[DCQ$^+$13] D’Agostino, D.; Clematis, A.; Quarati, A.; Cesini, D.; Chiappori, F.; Milanesi, L.; Merelli, I. “Cloud infrastructures for in silico drug discovery: Economic and practical aspects”, *BioMed Research International*, vol. 2013, 2013, pp. 1–19.

[DeL16] DeLano, W. L. “PyMol: The molecular graphics system”. Source: https://www.pymol.org/, Accessed at May 2016.

[DF08] Davidson, S. B.; Freire, J. “Provenance and scientific workflows: challenges and opportunities”. In: ACM SIGMOD International Conference on Management of Data, 2008, pp. 1345–1350.

[DGST09] Deelman, E.; Gannon, D.; Shields, M.; Taylor, I. “Workflows and e-science: An overview of workflow system features and capabilities”, *Future Generation Computer Systems*, vol. 25–5, 2009, pp. 528–540.

[DLS$^+$05] Deng, J.; Lee, K. W.; Sanchez, T.; Cui, M.; Neamati, N.; Briggs, J. M. “Dynamic receptor-based pharmacophore model development and its application in designing novel hiv-1 integrase inhibitors”, *Journal of Medicinal Chemistry*, vol. 48–5, 2005, pp. 1496–1505.

[DOOBM10] De Oliveira, D.; Ogasawara, E.; Baião, F.; Mattoso, M. “Scicumulus: A lightweight cloud middleware to explore many task computing paradigm in scientific workflows”. In: IEEE International Conference on Cloud Computing, 2010, pp. 378–385.

[DPFNdSR13] De Paris, R.; Frantz, F. A.; Norberto de Souza, O.; Ruiz, D. D. “wFReDoW: A cloud-based web environment to handle molecular docking simulations of a fully flexible receptor model”, *BioMed Research International*, vol. 2013, 2013, pp. 1–12.

[DPQR$^+$15] De Paris, R.; Quevedo, C. V.; Ruiz, D. D.; Norberto de Souza, O.; Barros, R. C. “Clustering molecular dynamics trajectories for optimizing docking experiments”, *Computational Intelligence and Neuroscience*, vol. 2015, 2015, pp. 1–10.

[DPQRNdS15] De Paris, R.; Quevedo, C. V.; Ruiz, D. D.; Noberto de Souza, O. “An effective approach for clustering InhA molecular dynamics trajectory using substrate-binding cavity features”, *PloS one*, vol. 10–7, 2015, pp. 1–25.

[DPRNdS15] De Paris, R.; Ruiz, D. A.; Norberto de Souza, O. “A cloud-based workflow approach for optimizing molecular docking simulations of fully-flexible receptor models and multiple ligands”. In: IEEE International Conference on Cloud Computing Technology and Science, 2015, pp. 495–498.

[DQB$^+$95] Dessen, A.; Quemard, A.; Blanchard, J. S.; Jacobs Jr, W. R.; Sacchettini, J. C. “Crystal structure and function of the isoniazid target of *Mycobacterium tuberculosis*”, *Science*, vol. 267–5204, 1995, pp. 1638–1641.

[DSS$^+$05] Deelman, E.; Singh, G.; Su, M.-H.; Blythe, J.; Gil, Y.; Kesselman, C.; Mehta, G.; Vahi, K.; Berriman, G. B.; Good, J.; et al.. “Pegasus: A framework for mapping complex scientific workflows onto distributed systems”, *Scientific Programming*, vol. 13–3, 2005, pp. 219–237.

[Eat16] Eaton, J. W. “GNU Octave”. Source: http://www.gnu.org/software/octave/, Accessed at Jun 2016.

[EB14] Ellingson, S. R.; Baudry, J. “High-throughput virtual molecular docking with AutoDockCloud”, *Concurrency and Computation: Practice and Experience*, vol. 26–1, 2014, pp. 907–916.

[EMBS14] Ellingson, S. R.; Miao, Y.; Baudry, J.; Smith, J. C. “Multi-conformer ensemble docking to difficult protein targets”, *The Journal of Physical Chemistry B*, vol. 119–3, 2014, pp. 1026–1034.

[FK03] Foster, I.; Kesselman, C. “The Grid 2: Blueprint for a new computing infrastructure”. San Francisco, CA: Morgan Kaufmann, 2003, 748p.

[FLSM14] Feixas, F.; Lindert, S.; Sinko, W.; McCammon, J. A. “Exploring the role of receptor flexibility in structure-based drug discovery”, *Biophysical Chemistry*, vol. 186, 2014, pp. 31–45.

[FPD$^+$07] Fahringer, T.; Prodan, R.; Duan, R.; Hofer, J.; Nadeem, F.; Nerieri, F.; Podlipnig, S.; Qin, J.; Siddiqui, M.; Truong, H.-L.; et al.. "Askalon: A development and grid computing environment for scientific workflows". In: *Workflows for e-Science: Scientific Workflow for Grids*, Lodon: Springer, 2007, pp. 450–471.

[FRTA02] Fernández-Recio, J.; Totrov, M.; Abagyan, R. "Soft protein–protein docking in internal coordinates", *Protein Science*, vol. 11–2, 2002, pp. 280–291.

[FTW12] Flick, J.; Tristram, F.; Wenzel, W. "Modeling loop backbone flexibility in receptor-ligand docking simulations", *Journal of Computational Chemistry*, vol. 33–31, 2012, pp. 2504–2515.

[GAG11] Garg, V.; Arora, S.; Gupta, C. "Cloud computing approaches to accelerate drug discovery value chain", *Combinatorial Chemistry & High Throughput Screening*, vol. 14–10, 2011, pp. 861–871.

[Gar92] García, A. E. "Large-amplitude nonlinear motions in proteins", *Physical Review Letters*, vol. 68–17, 1992, pp. 2696.

[Gar09] Gargano, F. "Efeito da temperatura na enzima 2-trans-enoil-ACP(CoA) redutase (EC 1.3.1.9) de *Mycobacterium tuberculosis* em complexo com o NADH: um estudo por simulação por dinâmica molecular", Ph.D. Thesis, Programa de Pós-Graduação em Biologia Celular e Molecular, PUCRS, Porto Alegre, RS, Brasil, 2009, 129p.

[GdMD14] Guedes, I. A.; de Magalhães, C. S.; Dardenne, L. E. "Receptor–ligand molecular docking", *Biophysical Reviews*, vol. 6–1, 2014, pp. 75–87.

[GGM12] Gupta, R.; Gupta, H.; Mohania, M. "Cloud computing and big data analytics: what is new from databases perspective?" In: International Conference on Big Data Analytics, 2012, pp. 42–61.

[GMO96] Goodsell, D. S.; Morris, G. M.; Olson, A. J. "Automated docking of flexible ligands: applications of autodock", *Journal of Molecular Recognition*, vol. 9–1, 1996, pp. 1–5.

[GoG16] GoGrid. "GoGrid Cloud Plataform". Source: http://www.gogrid.com, Accessed at Jun 2016.

[GOO16] GOOGLE. "Google Cloud Platform". Source: https://cloud.google.com/, Accessed at Jun 2016.

[GSK$^+$11] Görlach, K.; Sonntag, M.; Karastoyanova, D.; Leymann, F.; Reiter, M. "Conventional workflow technology for scientific simulation". In: *Guide to* *e-Science: Next Generation Scientific Research and Discovery*, London: Springer, 2011, pp. 323–352.

[HAO$^+$06] Hornak, V.; Abel, R.; Okur, A.; Strockbine, B.; Roitberg, A.; Simmerling, C. “Comparison of multiple amber force fields and development of improved protein backbone parameters”, *Proteins: Structure, Function, and Bioinformatics*, vol. 65–3, 2006, pp. 712–725.

[HBV02] Halkidi, M.; Batistakis, Y.; Vazirgiannis, M. “Clustering validity checking methods: part II”, *ACM Sigmod Record*, vol. 31–3, 2002, pp. 19–27.

[HCAL14] Han, Y.; Chan, J.; Alpcan, T.; Leckie, C. “Virtual machine allocation policies against co-resident attacks in cloud computing”. In: IEEE International Conference on Communications, 2014, pp. 786–792.

[Hir16] Hiranabe, K. “Astah: Software Design Tools for Agile Teams”. Source: http://astah.net/, Accessed at Jun 2016.

[HKP11] Han, J.; Kamber, M.; Pei, J. “Data Mining Concepts and Techniques”. Waltham, MA: Morgan Kaufmann, 2011, 744p.

[HKVDSL08] Hess, B.; Kutzner, C.; Van Der Spoel, D.; Lindahl, E. “GROMACS 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation”, *Journal of Chemical Theory and Computation*, vol. 4–3, 2008, pp. 435–447.

[HMF$^+$08] Hoffa, C.; Mehta, G.; Freeman, T.; Deelman, E.; Keahey, K.; Berriman, B.; Good, J. “On the use of cloud computing for scientific workflows”. In: IEEE Fourth International Conference on eScience, 2008, pp. 640–645.

[Hol95] Hollingsworth, D. “Workflow management coalition: The workflow reference model”, Technical Report, Workflow Management Coalition, Hampshire, UK, 1995, 55p.

[Hor16] Horizon. “Horizon: The openstack dashboard project”. Source: http://docs.openstack.org/developer/horizon/, Accessed at Jun 2016.

[HRFNdS15] Hübler, P.; Ruiz, D.; Ferreira, J. E.; Norberto de Souza, O. “P-SaMI: a data-flow pattern to perform massively-parallel molecular docking experiments using a fully-flexible receptor model”. In: Annual ACM Symposium on Applied Computing, 2015, pp. 54–57.

[HSN$^+$12] Hartkoorn, R. C.; Sala, C.; Neres, J.; Pojer, F.; Magnet, S.; Mukherjee, R.; Uplekar, S.; Boy-Röttger, S.; Altmann, K.-H.; Cole, S. T. “Towards a new tuberculosis drug: pyridomycin–nature's isoniazid", *EMBO Molecular Medicine*, vol. 4–10, 2012, pp. 1032–1042.

[Hüb10] Hübler, P. "P-SaMI: Self-adpting multiple insntaces - a data pattern to scientific workflows (*in portuguese:* P-MIA: Padrão múltiplas instâncias autoadaptáveis - um padrão de dados para workflows científicos)", Ph.D. Thesis, Programa de Pós-Graduação em Ciência da Computação, PUCRS, Porto Alegre, RS, Brasil, 2010, 179p.

[HW79] Hartigan, J. A.; Wong, M. A. "A k-means clustering algorithm", *Applied Statistics*, vol. 28, 1979, pp. 100–108.

[HWWC13] Hiden, H.; Woodman, S.; Watson, P.; Cala, J. "Developing cloud applications using the e-science central platform", *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 371–1983, 2013, pp. 1–12.

[HWWL10] Hiden, H.; Watson, P.; Woodman, S.; Leahy, D. "e-science central: Cloud-based e-science and its application to chemical property modelling", Technical Report, School of Computing Science Newcastle University, Newcastle upon Tyne, UK, 2010, 11p.

[HZ07] Huang, S.-Y.; Zou, X. "Ensemble docking of multiple protein structures: considering protein structural variations in molecular docking", *Proteins: Structure, Function, and Bioinformatics*, vol. 66–2, 2007, pp. 399–421.

[HZ10] Huang, S.-Y.; Zou, X. "Advances and challenges in protein-ligand docking", *International Journal of Molecular Sciences*, vol. 11–8, 2010, pp. 3016–3034.

[ISM$^+$12] Irwin, J. J.; Sterling, T.; Mysinger, M. M.; Bolstad, E. S.; Coleman, R. G. "ZINC: a free tool to discover chemistry for biology", *Journal of Chemical Information and Modeling*, vol. 52–7, 2012, pp. 1757–1768.

[Jai07] Jain, A. N. "Surflex-Dock 2.1: Robust performance from ligand energetic modeling, ring flexibility, and knowledge-based search", *Journal of Computer-Aided Molecular Design*, vol. 21–5, 2007, pp. 281–306.

[JD88] Jain, A. K.; Dubes, R. C. "Algorithms for clustering data". New Jersey: Prentice-Hall, 1988, 334p.

[JWG$^+$97] Jones, G.; Willett, P.; Glen, R. C.; Leach, A. R.; Taylor, R. "Development and validation of a genetic algorithm for flexible docking", *Journal of Molecular Biology*, vol. 267–3, 1997, pp. 727–748.

[JYBC15] Jung, M.-G.; Youn, S.-A.; Bae, J.; Choi, Y.-L. “A study on data input and output performance comparison of mongodb and postgresql in the big data environment”. In: International Conference on Database Theory and Application, 2015, pp. 14–17.

[Kap08] Kapetanovic, I. “Computer-aided drug discovery and development (CADDD): In silico-chemico-biological approach”, *Chemico-Biological Interactions*, vol. 171–2, 2008, pp. 165–176.

[KBT$^+$14] Kiss, T.; Borsody, P.; Terstyanszky, G.; Winter, S.; Greenwell, P.; McEldowney, S.; Heindl, H. “Large-scale virtual screening experiments on Windows Azure-based cloud resources”, *Concurrency and Computation: Practice and Experience*, vol. 26–10, 2014, pp. 1760–1770.

[KDFB04] Kitchen, D. B.; Decornez, H.; Furr, J. R.; Bajorath, J. “Docking and scoring in virtual screening for drug discovery: methods and applications”, *Nature Reviews Drug Discovery*, vol. 3–11, 2004, pp. 935–949.

[KFJ14] Korb, O.; Finn, P. W.; Jones, G. “The cloud and other new computational methods to improve molecular modelling”, *Expert Opinion on Drug Discovery*, vol. 9–10, 2014, pp. 1121–1131.

[KHB$^+$13] Knaus, J.; Hieke, S.; Binder, H.; Schwarzer, G.; et al.. “Costs of cloud computing for a biometry department”, *Methods of Information in Medicine*, vol. 52–1, 2013, pp. 72–79.

[KLJ$^+$11] Knox, C.; Law, V.; Jewison, T.; Liu, P.; Ly, S.; Frolkis, A.; Pon, A.; Banco, K.; Mak, C.; Neveu, V.; Djoumbou, Y.; Eisner, R.; Guo, A. C.; Wishart, D. S. “DrugBank 3.0: a comprehensive resource for ‘omics’ research on drugs”, *Nucleic Acids Research*, vol. 39–suppl 1, 2011, pp. D1035–D1041.

[KM02] Karplus, M.; McCammon, J. A. “Molecular dynamics simulations of biomolecules”, *Nature Structural & Molecular Biology*, vol. 9–9, 2002, pp. 646–652.

[KMA$^+$03] Kuo, M. R.; Morbidoni, H. R.; Alland, D.; Sneddon, S. F.; Gourlie, B. B.; Staveski, M. M.; Leonard, M.; Gregory, J. S.; Janjigian, A. D.; Yee, C.; Musser, J. M.; Kreiswirth, B.; Iwamoto, H.; Perozzo, R.; Jacobs, W. R.; Sacchettini, J. C.; Fidock, D. A. “Targeting tuberculosis and malaria through inhibition of enoyl reductase compound activity and structural data”, *Journal of Biological Chemistry*, vol. 278–23, 2003, pp. 20851–20859.

[KMC11] Korb, O.; McCabe, P.; Cole, J. “The ensemble performance index: an improved measure for assessing ensemble pose prediction performance”, *Journal of Chemical Information and Modeling*, vol. 51–11, 2011, pp. 2915–2919.

[KOB^{+}12] Korb, O.; Olsson, T. S.; Bowden, S. J.; Hall, R. J.; Verdonk, M. L.; Liebeschuetz, J. W.; Cole, J. C. “Potential and limitations of ensemble docking”, *Journal of Chemical Information and Modeling*, vol. 52–5, 2012, pp. 1262–1274.

[KR90] Kaufman, L.; Rousseeuw, P. J. “Finding Groups in Data: An Introduction to Cluster Analysis”. New York: Willey, 1990, 342p.

[Kun92] Kuntz, I. D. “Structure-Based Strategies for Drug Design and Discovery”, *Science*, vol. 257–5073, 1992, pp. 1078–1082.

[KVM16] KVM. “Kernel based virtual machine”. Source: http://www.linux-kvm.org/, Accessed at Jun 2016.

[LAB^{+}08] Landon, M. R.; Amaro, R. E.; Baron, R.; Ngan, C. H.; Ozonoff, D.; Andrew McCammon, J.; Vajda, S. “Novel druggable hot spots in avian influenza neuraminidase H5N1 revealed by computational solvent mapping of a reduced and representative receptor ensemble”, *Chemical Biology & Drug Design*, vol. 71–2, 2008, pp. 106–116.

[LAB^{+}09] Ludäscher, B.; Altintas, I.; Bowers, S.; Cummings, J.; Critchlow, T.; Deelman, E.; Roure, D. D.; Freire, J.; Goble, C.; Jones, M.; et al.. “Scientific process automation and workflow management”, *Scientific Data Management: Challenges, Existing Technology, and Deployment, Computational Science Series*, vol. 230, 2009, pp. 476–508.

[LBM^{+}09] Lang, P. T.; Brozell, S. R.; Mukherjee, S.; Pettersen, E. F.; Meng, E. C.; Thomas, V.; Rizzo, R. C.; Case, D. A.; James, T. L.; Kuntz, I. D. “DOCK 6: Combining techniques to model RNA–small molecule complexes”, *RNA*, vol. 15–6, 2009, pp. 1219–1230.

[Lea94] Leach, A. R. “Ligand docking to proteins with discrete side-chain flexibility”, *Journal of Molecular Biology*, vol. 235–1, 1994, pp. 345–356.

[Li06] Li, Y. “Bayesian model based clustering analysis: application to a molecular dynamics trajectory of the HIV-1 integrase catalytic core”, *Journal of Chemical Information and Modeling*, vol. 46–4, 2006, pp. 1742–1750.

[LLM88] Litzkow, M. J.; Livny, M.; Mutka, M. W. “Condor-a hunter of idle workstations”. In: International Conference on Distributed Computing Systems, 1988, pp. 104–111.

[LPSM03] Lin, J.-H.; Perryman, A. L.; Schames, J. R.; McCammon, J. A. “The relaxed complex method: Accommodating receptor flexibility for drug design with an improved scoring scheme”, *Biopolymers*, vol. 68–1, 2003, pp. 47–62.

[LPVM15] Liu, J.; Pacitti, E.; Valduriez, P.; Mattoso, M. “A survey of data-intensive scientific workflow management”, *Journal of Grid Computing*, vol. 13–4, 2015, pp. 457–493.

[LWW$^+$08] Liu, Z.-P.; Wu, L.-Y.; Wang, Y.; Zhang, X.-S.; Chen, L. “Protein cavity clustering based on community structure of pocket similarity network”, *International Journal of Bioinformatics Research and Applications*, vol. 4–4, 2008, pp. 445–460.

[LZ06] Lyman, E.; Zuckerman, D. M. “Ensemble-based convergence analysis of biomolecular trajectories”, *Biophysical Journal*, vol. 91–1, 2006, pp. 164–172.

[LZ12] Leis, S.; Zacharias, M. “Reflexin: a flexible receptor protein-ligand docking scheme evaluated on hiv-1 protease”, *PloS one*, vol. 7–10, 2012, pp. 1–13.

[M$^+$67] MacQueen, J.; et al.. “Some methods for classification and analysis of multivariate observations”. In: Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.

[Mac11] Machado, K. d. S. “Seleção eficiente de conformações de receptor flexível em simulações de docagem molecular”, Ph.D. Thesis, Programa de Pós-Graduação em Ciência da Computação, PUCRS, Porto Alegre, RS, Brasil, 2011, 180p.

[MCF$^+$11] Moreau, L.; Clifford, B.; Freire, J.; Futrelle, J.; Gil, Y.; Groth, P.; Kwasnikowska, N.; Miles, S.; Missier, P.; Myers, J.; et al.. “The open provenance model core specification (v1. 1)”, *Future Generation Computer Systems*, vol. 27–6, 2011, pp. 743–756.

[MDO$^+$15] Mattoso, M.; Dias, J.; Ocaña, K. A.; Ogasawara, E.; Costa, F.; Horta, F.; Silva, V.; de Oliveira, D. “Dynamic steering of hpc scientific workflows: A survey”, *Future Generation Computer Systems*, vol. 46, 2015, pp. 100–113.

[MG11] Mell, P.; Grance, T. “The NIST definition of cloud computing (draft)”, *NIST Special Publication*, vol. 800–145, 2011, pp. 1–7.

[MGH$^+$98] Morris, G. M.; Goodsell, D. S.; Halliday, R. S.; Huey, R.; Hart, W. E.; Belew, R. K.; Olson, A. J.; et al.. “Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function”, *Journal of Computational Chemistry*, vol. 19–14, 1998, pp. 1639–1662.

[MHL+09] Morris, G. M.; Huey, R.; Lindstrom, W.; Sanner, M. F.; Belew, R. K.; Goodsell, D. S.; Olson, A. J. “AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility”, *Journal of Computational Chemistry*, vol. 30–16, 2009, pp. 2785–2791.

[MLP+06] Moustakas, D. T.; Lang, P. T.; Pegg, S.; Pettersen, E.; Kuntz, I. D.; Brooijmans, N.; Rizzo, R. C. “Development and validation of a modular, extensible docking program: Dock 5”, *Journal of Computer-Aided Molecular Design*, vol. 20–10-11, 2006, pp. 601–619.

[MMM09] Mandal, S.; Moudgil, M.; Mandal, S. K. “Rational drug design”, *European Journal of Pharmacology*, vol. 625–1, 2009, pp. 90–100.

[MSR+11] Machado, K. S.; Schroeder, E. K.; Ruiz, D. D.; Cohen, E. M.; Norberto de Souza, O. “FReDoWS: a method to automate molecular docking simulations with explicit receptor flexibility and snapshots selection”, *BMC Genomics*, vol. 12–Suppl 4, 2011, pp. 2–13.

[MWRNdS11] Machado, K. S.; Winck, A. T.; Ruiz, D. D.; Norberto de Souza, O. “Mining flexible-receptor molecular docking data”, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1–6, 2011, pp. 532–541.

[NBIM11] Nichols, S. E.; Baron, R.; Ivetac, A.; McCammon, J. A. “Predictive power of molecular dynamics receptor structures in virtual screening”, *Journal of Chemical Information and Modeling*, vol. 51–6, 2011, pp. 1439–1446.

[NMT+15] Nguyen, A.; Matsunaga, A.; Tsugawa, M.; Ichikawa, K.; Haga, J. H.; et al.. “Deployment of a multi-site cloud environment for molecular virtual screenings”. In: International Conference on e-Science, 2015, pp. 145–154.

[OAF+04] Oinn, T.; Addis, M.; Ferris, J.; Marvin, D.; Senger, M.; Greenwood, M.; Carver, T.; Glover, K.; Pocock, M. R.; Wipat, A.; et al.. “Taverna: a tool for the composition and enactment of bioinformatics workflows”, *Bioinformatics*, vol. 20–17, 2004, pp. 3045–3054.

[OBDO+14] Ocaña, K.; Benza, S.; De Oliveira, D.; Dias, J.; Mattoso, M. “Exploring large scale receptor-ligand pairs in molecular docking workflows in HPC clouds”. In: IEEE International Parallel & Distributed Processing Symposium Workshops, 2014, pp. 536–545.

[Org15] Organization, W. H. “Global tuberculosis report 2015”, Technical Report, World Health Organization, Geneva, SUI, 2015, 204p.

[PCN11] Phillips, J. L.; Colvin, M. E.; Newsam, S. “Validating clustering of molecular dynamics simulations using polymer models”, *BMC Bioinformatics*, vol. 12–1, 2011, pp. 445–468.

[PdSR$^+$13] Pauli, I.; dos Santos, R. N.; Rostirolla, D. C.; Martinelli, L. K.; Ducati, R. G.; Timmers, L. F.; Basso, L. A.; Santos, D. S.; Guido, R. V.; Andricopulo, A. D.; et al.. “Discovery of new inhibitors of *Mycobacterium tuberculosis* InhA enzyme using virtual screening and a 3d-pharmacophore-based approach”, *Journal of Chemical Information and Modeling*, vol. 53–9, 2013, pp. 2390–2401.

[PMD$^+$10] Paul, S. M.; Mytelka, D. S.; Dunwiddie, C. T.; Persinger, C. C.; Munos, B. H.; Lindborg, S. R.; Schacht, A. L. “How to improve r&d productivity: the pharmaceutical industry’s grand challenge”, *Nature Reviews Drug discovery*, vol. 9–3, 2010, pp. 203–214.

[QDPRNdS14] Quevedo, C. V.; De Paris, R.; Ruiz, D. D.; Norberto de Souza, O. “A strategic solution to optimize molecular docking simulations using fully-flexible receptor models”, *Expert Systems With Applications*, vol. 41–16, 2014, pp. 7608–7620.

[QDS$^+$96] Quemard, A.; Dessen, A.; Sugantino, M.; Jacobs, W. R.; Sacchettini, J. C.; Blanchard, J. S. “Binding of catalase-peroxidase-activated isoniazid to wild-type and mutant *Mycobacterium tuberculosis* enoyl-acp reductases”, *Journal of the American Chemical Society*, vol. 118–6, 1996, pp. 1561–1562.

[Que16] Quevedo, C. V. “Triagem virtual em banco de dados de ligantes considerando propriedades físico-químicas de um modelo de receptor completamente flexível”, Ph.D. Thesis, Programa de Pós-Graduação em Ciência da Computação, PUCRS, Porto Alegre, RS, Brasil, 2016, 143p.

[RTA12] Rueda, M.; Totrov, M.; Abagyan, R. “Alibero: evolving a team of complementary pocket conformations rather than a single leader”, *Journal of Chemical Information and Modeling*, vol. 52–10, 2012, pp. 2705–2714.

[RvDBR12] Ruddigkeit, L.; van Deursen, R.; Blum, L. C.; Reymond, J.-L. “Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17”, *Journal of Chemical Information and Modeling*, vol. 52–11, 2012, pp. 2864–2875.

[RVS$^+$99] Rozwarski, D. A.; Vilchèze, C.; Sugantino, M.; Bittman, R.; Sacchettini, J. C. “Crystal structure of the *Mycobacterium tuberculosis* enoyl-ACP reductase, InhA, in complex with NAD+ and a C16 fatty acyl substrate”, *Journal of Biological Chemistry*, vol. 274–22, 1999, pp. 15582–15589.

[SBBW12] Scannell, J. W.; Blanckley, A.; Boldon, H.; Warrington, B. “Diagnosing the decline in pharmaceutical r&d efficiency”, *Nature Reviews Drug Discovery*, vol. 11–3, 2012, pp. 191–200.

[Sch16] Schetnikovich, D. “Robomongo: Native management MongoDB tool”. Source: https://robomongo.org/, Accessed at Jun 2016.

[SI15] Sterling, T.; Irwin, J. J. “Zinc 15-ligand discovery for everyone”, *Journal of Chemistry Information and Modeling*, vol. 55–11, 2015, pp. 2324–2337.

[Sma16] SmartCloud, I. “IBM smartcloud for social business”. Source: https://www.ibm.com/cloud-computing/social/us/en/, Accessed at Jun 2016.

[SMP$^+$10] Sperandio, O.; Mouawad, L.; Pinto, E.; Villoutreix, B. O.; Perahia, D.; Miteva, M. A. “How to choose relevant multiple receptor conformations for virtual screening: a test case of cdk2 and normal mode analysis”, *European Biophysics Journal*, vol. 39–9, 2010, pp. 1365–1372.

[SPG05] Simmhan, Y. L.; Plale, B.; Gannon, D. “A survey of data provenance in e-science”, *ACM Sigmod Record*, vol. 34–3, 2005, pp. 31–36.

[SRC$^+$13] Sousa, S.; Ribeiro, A.; Coimbra, J.; Neves, R.; Martins, S.; Moorthy, N.; Fernandes, P.; Ramos, M. “Protein-ligand docking in the new millennium–a retrospective of 10 years in the field”, *Current Medicinal Chemistry*, vol. 20–18, 2013, pp. 2296–2314.

[STB$^+$06] Sullivan, T. J.; Truglio, J. J.; Boyne, M. E.; Novichenok, P.; Zhang, X.; Stratton, C. F.; Li, H.-J.; Kaur, T.; Amin, A.; Johnson, F.; et al.. “High affinity InhA inhibitors with activity against drug-resistant strains of *Mycobacterium tuberculosis*”, *ACS Chemical Biology*, vol. 1–1, 2006, pp. 43–53.

[STTC07] Shao, J.; Tanner, S. W.; Thompson, N.; Cheatham, T. E. “Clustering molecular dynamics trajectories: 1.Characterizing the performance of different clustering algorithms”, *Journal of Chemical Theory and Computation*, vol. 3–6, 2007, pp. 2312–2334.

[Sul14] Sultan, N. “Making use of cloud computing for healthcare provision: Opportunities and challenges”, *International Journal of Information Management*, vol. 34–2, 2014, pp. 177–184.

[TA08] Totrov, M.; Abagyan, R. “Flexible ligand docking to multiple receptor conformations: a practical alternative”, *Current Opinion in Structural Biology*, vol. 18–2, 2008, pp. 178–184.

[Tal13] Talia, D. “Workflow systems for science: Concepts and tools”, *ISRN Software Engineering*, vol. 2013, 2013, pp. 1–15.

[Tea12] Team, R. C. “R: A language and environment for statistical computing”, Technical Report, R Foundation for Statistical Computing, Vienna, Austria, 2012, 1706p.

[TIM^{+}93] Telenti, A.; Imboden, P.; Marchesi, F.; Matter, L.; Schopfer, K.; Bodmer, T.; Lowrie, D.; Colston, M.; Cole, S. “Detection of rifampicin-resistance mutations in *Mycobacterium tuberculosis*”, *The Lancet*, vol. 341–8846, 1993, pp. 647–651.

[TK03] Teodoro, M. L.; Kavraki, L. E. “Conformational flexibility models for the receptor in structure based drug design”, *Current Pharmaceutical Design*, vol. 9–20, 2003, pp. 1635–1648.

[TSK13] Tan, P.-N.; Steinbach, M.; Kumar, V. “Introduction to Data Mining”. Boston: Addison-Wesley, 2013, 742p.

[TSP^{+}14] Tian, S.; Sun, H.; Pan, P.; Li, D.; Zhen, X.; Li, Y.; Hou, T. “Assessing an ensemble docking-based virtual screening strategy for kinase targets by considering protein flexibility”, *Journal of Chemical Information and Modeling*, vol. 54–10, 2014, pp. 2664–2679.

[TSWH07] Taylor, I.; Shields, M.; Wang, I.; Harrison, A. “The triana workflow environment: Architecture and applications”. In: *Workflows for e-Science: Scientific Workflow for Grids*, London: Springer, 2007, pp. 320–339.

[TvG94] Torda, A. E.; van Gunsteren, W. F. “Algorithms for clustering molecular dynamics configurations”, *Journal of Computational Chemistry*, vol. 15–12, 1994, pp. 1331–1340.

[TWH01] Tibshirani, R.; Walther, G.; Hastie, T. “Estimating the number of clusters in a data set via the gap statistic”, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63–2, 2001, pp. 411–423.

[vDATHKB03] van Der Aalst, W. M.; Ter Hofstede, A. H.; Kiepuszewski, B.; Barros, A. P. “Workflow patterns”, *Distributed and Parallel Databases*, vol. 14–1, 2003, pp. 5–51.

[VRMCL08] Vaquero, L. M.; Rodero-Merino, L.; Caceres, J.; Lindner, M. “A break in the clouds: towards a cloud definition”, *ACM SIGCOMM Computer Communication Review*, vol. 39–1, 2008, pp. 50–55.

[VWA$^+$06] Vilchèze, C.; Wang, F.; Arai, M.; Hazbón, M. H.; Colangeli, R.; Kremer, L.; Weisbrod, T. R.; Alland, D.; Sacchettini, J. C.; Jacobs, W. R. “Transfer of a point mutation in *Mycobacterium tuberculosis* InhA resolves the target of isoniazid”, *Nature Medicine*, vol. 12–9, 2006, pp. 1027–1029.

[WKQ$^+$08] Wu, X.; Kumar, V.; Quinlan, J. R.; Ghosh, J.; Yang, Q.; Motoda, H.; McLachlan, G. J.; Ng, A.; Liu, B.; Philip, S. Y.; et al.. “Top 10 algorithms in data mining”, *Knowledge and Information Systems*, vol. 14–1, 2008, pp. 1–37.

[WLC$^+$11] Watson, P.; Leahy, D.; Cala, J.; Sykora, V.; Hiden, H.; Woodman, S.; Taylor, M.; Searson, D. “Cloud computing for chemical activity prediction”, Technical Report No. CS-TR-1242, Newcastle University, Newcastle upon Tyne, UK, 2011, 10p.

[Won08] Wong, C. F. “Flexible ligand–flexible protein docking in protein kinase systems”, *Biochimica et Biophysica Acta (BBA)-Proteins and Proteomics*, vol. 1784–1, 2008, pp. 244–251.

[WWF$^+$04] Wei, B. Q.; Weaver, L. H.; Ferrari, A. M.; Matthews, B. W.; Shoichet, B. K. “Testing a flexible-receptor docking algorithm in a model binding site”, *Journal of Molecular Biology*, vol. 337–5, 2004, pp. 1161–1182.

[WXS$^+$09] Wang, Y.; Xiao, J.; Suzek, T. O.; Zhang, J.; Wang, J.; Bryant, S. H. “PubChem: a public information system for analyzing bioactivities of small molecules”, *Nucleic Acids Research*, vol. 37–suppl 2, 2009, pp. W623–W633.

[XEN16] XENSERVER. “Xenserver: Open source virtualization”. Source: http://www.xenserver.org/, Accessed at Jun 2016.

[YB05] Yu, J.; Buyya, R. “A taxonomy of workflow management systems for grid computing”, *Journal of Grid Computing*, vol. 3–3-4, 2005, pp. 171–200.

[ZCB10] Zhang, Q.; Cheng, L.; Boutaba, R. “Cloud Computing: State-of-the-art and research challenges”, *Journal of Internet Services and Applications*, vol. 1–1, 2010, pp. 7–18.

[ZCZ$^+$08] Zhong, S.; Chen, X.; Zhu, X.; Dziegielewska, B.; Bachman, K. E.; Ellenberger, T.; Ballin, J. D.; Wilson, G. M.; Tomkinson, A. E.; MacKerell Jr, A. D. “Identification and validation of human dna ligase inhibitors using computer-aided drug design”, *Journal of Medicinal Chemistry*, vol. 51–15, 2008, pp. 4553–4562.

[ZHC$^+$07] Zhao, Y.; Hategan, M.; Clifford, B.; Foster, I.; Von Laszewski, G.; Nefedova, V.; Raicu, I.; Stef-Praun, T.; Wilde, M. “Swift: Fast, reliable, loosely coupled parallel computation”. In: IEEE Congress on Services, 2007, pp. 199–206.

[ZS99] Zacharias, M.; Sklenar, H. “Harmonic modes as variables to approximately account for receptor flexibility in ligand–receptor docking simulations: Application to dna minor groove ligand complex”, *Journal of Computational Chemistry*, vol. 20–3, 1999, pp. 287–300.

[ZS04] Zhang, Y.; Skolnick, J. “Spicker: A clustering approach to identify near-native protein folds”, *Journal of Computational Chemistry*, vol. 25–6, 2004, pp. 865–871.

[ZWL13] Zhang, X.; Wong, S. E.; Lightstone, F. C. “Message passing interface and multithreading hybrid for parallel molecular docking of large databases on petascale high performance computing machines”, *Journal of Computational Chemistry*, vol. 34–11, 2013, pp. 915–927.

APPENDIX A – RESULTS OF THE EMPIRICAL EXPERIMENTS USING DIFFERENT METHOD’S PARAMETRIZATION FOR THE BATCH ANALYSIS FUNCTION

Percentage of the best 10, 20, 30 and 100 ligand and snapshots interactions (TOP10, TOP20, TOP30 and TOP100), and processed snapshots obtained when the docking analyses start after 10% of snapshots have been docked.

	Ligand	TCL300	TCL400	4PI	665	8PC	8PS	468	566	641	5PP	744	GEQ	JPJ	JPL	JPM	TCU
	PDB ID	2B35	1P45	2NSD	2H7L	3FNE	2B37	2H7P	2H7I	2H7M	2B36	2H7N	1P44	3FNH	3FNG	3FNF	2X22
DB_30	TOP10	70.00	60.00	90.00	100.00	70.00	80.00	70.00	100.00	70.00	40.00	40.00	60.00	100.00	70.00	40.00	80.00
	TOP20	60.00	75.00	80.00	100.00	75.00	90.00	70.00	90.00	70.00	60.00	50.00	65.00	100.00	65.00	30.00	90.00
	TOP30	66.00	73.00	86.00	100.00	80.00	93.00	70.00	86.00	66.00	53.00	50.00	56.00	100.00	73.00	36.00	90.00
	TOP100	77.00	64.00	89.00	93.00	80.00	91.00	81.00	81.00	69.00	66.00	59.00	57.00	91.00	74.00	49.00	90.00
	Processed	47.00	48.00	48.00	45.00	51.00	48.00	46.00	50.00	47.00	50.00	47.00	44.00	48.00	50.00	48.00	54.00
DB_40	TOP10	70.00	60.00	90.00	100.00	70.00	80.00	80.00	100.00	80.00	40.00	40.00	60.00	100.00	80.00	40.00	80.00
	TOP20	60.00	75.00	80.00	100.00	75.00	90.00	75.00	90.00	75.00	60.00	50.00	65.00	100.00	70.00	35.00	90.00
	TOP30	66.00	73.00	86.00	100.00	80.00	93.00	73.00	86.00	70.00	56.00	50.00	56.00	100.00	76.00	40.00	93.00
	TOP100	77.00	65.00	89.00	93.00	80.00	91.00	84.00	81.00	74.00	68.00	59.00	58.00	91.00	77.00	51.00	92.00
	Processed	48.00	51.00	49.00	47.00	52.00	49.00	47.00	52.00	49.00	51.00	49.00	46.00	49.00	52.00	52.00	55.00
DB_50	TOP10	70.00	60.00	90.00	100.00	70.00	80.00	80.00	100.00	80.00	40.00	40.00	60.00	100.00	80.00	50.00	80.00
	TOP20	60.00	75.00	80.00	100.00	75.00	90.00	75.00	90.00	75.00	60.00	55.00	65.00	100.00	70.00	50.00	90.00
	TOP30	70.00	73.00	86.00	100.00	80.00	93.00	73.00	86.00	70.00	60.00	53.00	60.00	100.00	80.00	50.00	93.00
	TOP100	78.00	67.00	89.00	93.00	80.00	92.00	84.00	82.00	78.00	73.00	64.00	62.00	92.00	81.00	56.00	92.00
	Processed	51.00	53.00	51.00	49.00	54.00	51.00	49.00	54.00	52.00	53.00	52.00	50.00	51.00	54.00	55.00	57.00
DB_60	TOP10	70.00	60.00	90.00	100.00	70.00	80.00	80.00	100.00	80.00	40.00	40.00	60.00	100.00	80.00	50.00	80.00
	TOP20	70.00	80.00	80.00	100.00	75.00	90.00	75.00	95.00	75.00	60.00	55.00	65.00	100.00	70.00	50.00	90.00
	TOP30	76.00	80.00	86.00	100.00	80.00	93.00	76.00	93.00	70.00	60.00	53.00	60.00	100.00	80.00	50.00	93.00
	TOP100	82.00	69.00	89.00	93.00	80.00	92.00	85.00	87.00	79.00	73.00	65.00	63.00	92.00	81.00	58.00	92.00
	Processed	52.00	54.00	53.00	50.00	56.00	52.00	51.00	56.00	54.00	54.00	53.00	52.00	53.00	56.00	56.00	58.00
DB_70	TOP10	70.00	70.00	90.00	100.00	70.00	80.00	80.00	100.00	80.00	40.00	40.00	60.00	100.00	80.00	50.00	80.00
	TOP20	70.00	85.00	80.00	100.00	75.00	90.00	75.00	95.00	75.00	60.00	55.00	65.00	100.00	75.00	50.00	90.00
	TOP30	76.00	83.00	86.00	100.00	80.00	93.00	76.00	93.00	73.00	63.00	53.00	60.00	100.00	83.00	50.00	93.00
	TOP100	82.00	73.00	89.00	93.00	81.00	93.00	85.00	87.00	80.00	75.00	65.00	64.00	92.00	83.00	59.00	92.00
	Processed	53.00	55.00	54.00	52.00	57.00	53.00	52.00	58.00	56.00	56.00	55.00	54.00	54.00	57.00	58.00	59.00

DB = Percentage threshold to discard a batch.

Percentage of the best 10, 20, 30 and 100 ligand and snapshots interactions (TOP10, TOP20, TOP30 and TOP100), and processed snapshots obtained when the docking analyses start after 15% of snapshots have been docked.

	Ligand	TCL300	TCL400	4PI	665	8PC	8PS	468	566	641	5PP	744	GEQ	JPJ	JPL	JPM	TCU
	PDB ID	2B35	1P45	2NSD	2H7L	3FNE	2B37	2H7P	2H7I	2H7M	2B36	2H7N	1P44	3FNH	3FNG	3FNF	2X22
DB_30	TOP10	70.00	60.00	90.00	100.00	80.00	80.00	70.00	100.00	80.00	60.00	60.00	70.00	100.00	70.00	40.00	80.00
	TOP20	60.00	75.00	80.00	100.00	80.00	90.00	70.00	95.00	80.00	70.00	65.00	75.00	100.00	65.00	30.00	90.00
	TOP30	66.00	73.00	86.00	100.00	83.00	93.00	70.00	90.00	76.00	63.00	63.00	70.00	100.00	73.00	36.00	90.00
	TOP100	77.00	64.00	89.00	94.00	84.00	91.00	82.00	85.00	72.00	71.00	67.00	66.00	94.00	74.00	49.00	90.00
	Processed	51.00	51.00	52.00	51.00	53.00	52.00	49.00	53.00	50.00	54.00	50.00	51.00	52.00	54.00	51.00	56.00
DB_40	TOP10	70.00	60.00	90.00	100.00	80.00	80.00	80.00	100.00	90.00	60.00	60.00	70.00	100.00	80.00	40.00	80.00
	TOP20	60.00	75.00	80.00	100.00	80.00	90.00	75.00	95.00	85.00	70.00	65.00	80.00	100.00	70.00	35.00	90.00
	TOP30	70.00	73.00	86.00	100.00	83.00	93.00	73.00	90.00	80.00	70.00	63.00	73.00	100.00	76.00	40.00	93.00
	TOP100	78.00	66.00	89.00	94.00	84.00	91.00	84.00	85.00	77.00	74.00	68.00	69.00	95.00	79.00	53.00	92.00
	Processed	53.00	54.00	54.00	54.00	55.00	53.00	51.00	55.00	53.00	56.00	53.00	55.00	56.00	58.00	55.00	58.00
DB_50	TOP10	70.00	60.00	90.00	100.00	80.00	80.00	80.00	100.00	90.00	60.00	60.00	80.00	100.00	90.00	50.00	80.00
	TOP20	70.00	75.00	80.00	100.00	80.00	90.00	75.00	95.00	85.00	70.00	70.00	85.00	100.00	75.00	50.00	90.00
	TOP30	76.00	73.00	86.00	100.00	83.00	93.00	73.00	90.00	80.00	70.00	66.00	76.00	100.00	83.00	50.00	93.00
	TOP100	82.00	67.00	89.00	94.00	84.00	92.00	84.00	86.00	81.00	78.00	73.00	72.00	95.00	84.00	57.00	92.00
	Processed	55.00	56.00	56.00	55.00	56.00	54.00	53.00	57.00	56.00	58.00	55.00	58.00	58.00	60.00	57.00	59.00
DB_60	TOP10	70.00	60.00	90.00	100.00	80.00	80.00	80.00	100.00	90.00	60.00	60.00	80.00	100.00	90.00	50.00	80.00
	TOP20	70.00	80.00	80.00	100.00	80.00	90.00	75.00	100.00	85.00	70.00	70.00	85.00	100.00	75.00	55.00	90.00
	TOP30	76.00	80.00	86.00	100.00	83.00	93.00	76.00	96.00	80.00	70.00	66.00	76.00	100.00	83.00	53.00	93.00
	TOP100	82.00	69.00	89.00	94.00	84.00	92.00	85.00	91.00	81.00	78.00	73.00	73.00	95.00	84.00	60.00	92.00
	Processed	56.00	57.00	57.00	56.00	58.00	55.00	54.00	59.00	57.00	59.00	57.00	61.00	59.00	61.00	58.00	61.00
DB_70	TOP10	70.00	70.00	90.00	100.00	80.00	80.00	80.00	100.00	90.00	70.00	60.00	80.00	100.00	90.00	50.00	80.00
	TOP20	70.00	85.00	80.00	100.00	80.00	90.00	75.00	100.00	85.00	75.00	70.00	85.00	100.00	80.00	55.00	90.00
	TOP30	76.00	83.00	86.00	100.00	83.00	93.00	76.00	96.00	83.00	76.00	66.00	76.00	100.00	86.00	53.00	93.00
	TOP100	82.00	74.00	89.00	96.00	85.00	95.00	85.00	91.00	82.00	82.00	74.00	76.00	95.00	86.00	61.00	93.00
	Processed	57.00	59.00	58.00	58.00	59.00	57.00	55.00	61.00	59.00	61.00	59.00	63.00	60.00	63.00	60.00	62.00

DB = Percentage threshold to discard a batch.

Percentage of the best 10, 20, 30 and 100 ligand and snapshots interactions (TOP10, TOP20, TOP30 and TOP100), and processed snapshots obtained when the docking analyses start after 20% of snapshots have been docked.

	Ligand	TCL300	TCL400	4PI	665	8PC	8PS	468	566	641	5PP	744	GEQ	JPJ	JPL	JPM	TCU
	PDB ID	2B35	1P45	2NSD	2H7L	3FNE	2B37	2H7P	2H7I	2H7M	2B36	2H7N	1P44	3FNH	3FNG	3FNF	2X22
DB_30	TOP10	80.00	70.00	100.00	100.00	80.00	80.00	80.00	100.00	80.00	70.00	60.00	70.00	100.00	70.00	50.00	80.00
	TOP20	70.00	80.00	90.00	100.00	80.00	90.00	85.00	90.00	80.00	75.00	70.00	85.00	100.00	65.00	35.00	90.00
	TOP30	76.00	76.00	93.00	100.00	83.00	93.00	83.00	86.00	80.00	70.00	66.00	80.00	100.00	73.00	43.00	90.00
	TOP100	82.00	69.00	93.00	94.00	85.00	91.00	87.00	83.00	75.00	74.00	70.00	71.00	97.00	77.00	54.00	89.00
	Processed	54.00	57.00	56.00	55.00	55.00	55.00	52.00	53.00	53.00	56.00	54.00	54.00	57.00	58.00	57.00	58.00
DB_40	TOP10	80.00	70.00	100.00	100.00	80.00	80.00	90.00	100.00	90.00	70.00	60.00	70.00	100.00	80.00	50.00	80.00
	TOP20	70.00	80.00	90.00	100.00	80.00	90.00	90.00	95.00	85.00	75.00	70.00	85.00	100.00	70.00	40.00	90.00
	TOP30	80.00	76.00	93.00	100.00	83.00	93.00	86.00	90.00	83.00	73.00	70.00	80.00	100.00	76.00	46.00	93.00
	TOP100	83.00	70.00	93.00	94.00	85.00	92.00	89.00	85.00	83.00	76.00	71.00	74.00	97.00	82.00	60.00	91.00
	Processed	56.00	60.00	58.00	58.00	58.00	56.00	54.00	56.00	57.00	58.00	56.00	58.00	59.00	61.00	60.00	59.00
DB_50	TOP10	80.00	70.00	100.00	100.00	80.00	80.00	90.00	100.00	90.00	70.00	60.00	80.00	100.00	80.00	70.00	80.00
	TOP20	70.00	80.00	90.00	100.00	80.00	90.00	90.00	95.00	85.00	75.00	75.00	90.00	100.00	70.00	65.00	90.00
	TOP30	80.00	76.00	93.00	100.00	83.00	93.00	86.00	90.00	83.00	73.00	73.00	83.00	100.00	80.00	66.00	93.00
	TOP100	83.00	71.00	93.00	94.00	85.00	92.00	89.00	88.00	85.00	80.00	76.00	79.00	97.00	86.00	69.00	91.00
	Processed	57.00	61.00	59.00	60.00	59.00	59.00	55.00	59.00	60.00	60.00	59.00	61.00	61.00	62.00	62.00	61.00
DB_60	TOP10	80.00	70.00	100.00	100.00	80.00	80.00	90.00	100.00	90.00	70.00	60.00	80.00	100.00	80.00	70.00	80.00
	TOP20	80.00	85.00	90.00	100.00	80.00	90.00	90.00	95.00	85.00	75.00	75.00	90.00	100.00	70.00	70.00	90.00
	TOP30	86.00	86.00	93.00	100.00	83.00	93.00	90.00	90.00	83.00	73.00	73.00	83.00	100.00	80.00	70.00	93.00
	TOP100	87.00	74.00	93.00	96.00	85.00	92.00	90.00	88.00	85.00	80.00	77.00	80.00	97.00	86.00	74.00	92.00
	Processed	59.00	62.00	61.00	62.00	61.00	60.00	57.00	61.00	62.00	61.00	61.00	64.00	63.00	64.00	64.00	62.00
DB_70	TOP10	80.00	80.00	100.00	100.00	80.00	80.00	90.00	100.00	90.00	70.00	60.00	80.00	100.00	80.00	70.00	80.00
	TOP20	80.00	90.00	90.00	100.00	80.00	90.00	90.00	95.00	85.00	75.00	75.00	90.00	100.00	75.00	70.00	90.00
	TOP30	86.00	90.00	93.00	100.00	83.00	93.00	90.00	90.00	86.00	76.00	73.00	83.00	100.00	83.00	70.00	93.00
	TOP100	87.00	76.00	93.00	96.00	86.00	95.00	90.00	88.00	86.00	83.00	77.00	82.00	97.00	88.00	74.00	93.00
	Processed	60.00	63.00	61.00	63.00	62.00	61.00	58.00	62.00	64.00	62.00	62.00	65.00	64.00	65.00	66.00	63.00

DB = Percentage threshold to discard a batch.

APPENDIX B – RESULTS OF THE EMPIRICAL EXPERIMENTS USING DIFFERENT METHOD’S PARAMETRIZATION FOR THE CLUSTER ANALYSIS FUNCTION

Percentage of the best 10, 20, 30 and 100 ligand and snapshots interactions (TOP10, TOP20, TOP30 and TOP100), and processed snapshots obtained when the docking analyses start after 10% of snapshots have been docked.

	Ligand	TCL300	TCL400	4PI	665	8PC	8PS	468	566	641	5PP	744	GEQ	JPJ	JPL	JPM	TCU
	PDB ID	2B35	1P45	2NSD	2H7L	3FNE	2B37	2H7P	2H7I	2H7M	2B36	2H7N	1P44	3FNH	3FNG	3FNF	2X22
DC_10	TOP10	68.50	61.50	96.00	99.00	72.50	55.50	77.50	94.00	70.00	78.50	76.50	87.00	94.50	71.00	44.00	78.00
	TOP20	75.25	68.00	95.25	94.75	78.50	62.00	75.00	91.50	66.00	72.50	77.25	90.50	90.50	64.50	39.00	81.25
	TOP30	76.05	69.45	93.20	96.00	79.40	69.70	74.70	90.00	67.25	68.90	75.15	88.95	88.65	69.25	46.30	85.65
	TOP100	76.50	70.10	95.05	93.45	78.55	75.40	84.60	85.55	67.25	71.85	79.05	81.15	86.65	71.95	53.65	81.20
	Processed	47.00	45.50	45.50	42.95	44.10	45.55	43.95	44.10	43.50	42.40	44.15	45.05	44.10	46.15	51.35	43.75
DC_20	TOP10	71.50	65.00	98.50	98.00	80.00	63.50	79.00	95.00	76.00	78.00	75.50	86.50	97.50	77.00	46.50	70.00
	TOP20	78.25	69.75	96.75	94.00	83.00	68.50	77.25	96.00	68.75	75.25	75.25	88.50	94.75	70.75	39.50	77.25
	TOP30	79.55	71.05	93.75	95.15	83.40	75.35	77.50	93.00	71.45	72.90	76.35	87.85	92.25	74.40	48.10	83.35
	TOP100	79.30	73.10	95.70	94.05	82.90	78.45	84.25	87.10	74.50	76.30	80.10	81.60	90.00	76.00	56.25	82.10
	Processed	49.50	47.85	47.75	45.30	47.80	48.10	46.65	48.30	50.05	44.95	46.90	50.00	47.50	53.55	53.90	49.40
DC_30	TOP10	74.50	76.50	96.50	99.50	86.00	76.00	83.50	97.00	82.50	78.00	81.50	88.00	98.00	75.50	55.00	75.50
	TOP20	81.75	82.50	97.25	95.50	86.25	79.00	80.75	97.50	73.00	77.00	80.25	91.25	95.25	67.50	47.00	81.00
	TOP30	82.30	81.35	95.00	96.45	85.55	84.70	81.90	95.25	76.35	74.65	80.25	90.65	92.85	71.90	54.35	85.30
	TOP100	82.90	78.80	95.70	95.30	85.45	84.25	87.80	90.65	76.75	78.70	83.05	83.45	90.80	75.60	60.60	85.55
	Processed	52.55	50.65	48.85	48.10	52.00	52.30	50.75	52.20	53.15	48.50	48.65	54.20	50.90	55.50	58.25	53.80
DC_40	TOP10	75.67	79.00	98.00	99.00	88.00	76.00	83.50	97.67	86.33	76.50	78.33	92.33	97.67	82.67	57.67	79.00
	TOP20	85.33	86.00	98.33	95.75	90.75	78.33	83.25	97.33	76.00	78.25	78.83	93.67	96.83	73.50	54.00	83.67
	TOP30	84.10	83.37	96.10	96.70	88.35	82.77	83.85	95.67	79.03	76.25	79.80	91.17	94.00	78.67	60.77	87.90
	TOP100	84.77	81.47	97.03	95.80	89.05	83.73	89.15	91.57	79.23	79.30	83.60	85.97	91.93	80.27	66.20	86.63
	Processed	54.27	53.57	50.60	51.05	55.05	53.97	53.30	55.40	57.23	52.70	52.13	60.43	53.23	58.93	60.93	57.00
DC_50	TOP10	79.50	81.00	98.00	100.00	87.50	77.00	89.50	96.00	82.50	77.00	82.00	91.50	99.00	83.50	67.00	82.00
	TOP20	87.75	87.00	96.50	96.00	91.50	80.50	87.50	97.50	77.75	79.50	81.25	94.50	95.75	74.00	64.25	87.75
	TOP30	86.60	83.20	94.75	96.65	89.70	84.65	87.30	95.25	79.70	77.70	80.00	92.10	93.10	79.10	67.60	90.20
	TOP100	85.05	82.00	96.25	96.75	89.80	84.65	90.20	91.50	79.90	81.70	83.95	85.70	91.90	81.30	72.00	88.55
	Processed	55.50	54.95	51.45	52.95	59.35	57.15	54.90	56.20	60.35	57.30	53.35	63.05	55.25	59.05	64.95	58.25

DC = Percentage threshold to discard a cluster.

Percentage of the best 10, 20, 30 and 100 ligand and snapshots interactions (TOP10, TOP20, TOP30 and TOP100), and processed snapshots obtained when the docking analyses start after 15% of snapshots have been docked.

	Ligand	TCL300	TCL400	4PI	665	8PC	8PS	468	566	641	5PP	744	GEQ	JPJ	JPL	JPM	TCU
	PDB ID	2B35	1P45	2NSD	2H7L	3FNE	2B37	2H7P	2H7I	2H7M	2B36	2H7N	1P44	3FNH	3FNG	3FNF	2X22
DC_10	TOP10	79.00	70.00	97.50	100.00	83.50	67.00	86.00	98.50	78.00	80.50	78.50	93.00	98.50	79.43	43.50	83.00
	TOP20	85.00	73.50	98.25	95.50	86.25	72.00	81.00	99.25	70.75	74.50	79.25	95.75	95.75	70.79	39.75	86.75
	TOP30	85.35	73.45	95.25	96.40	85.55	78.50	80.25	97.10	73.55	72.75	78.95	94.10	93.60	74.98	49.40	90.55
	TOP100	83.00	74.65	96.65	94.80	84.00	81.10	86.85	91.50	75.85	77.65	83.95	85.55	92.10	78.21	56.90	86.10
	Processed	53.15	50.50	50.65	48.85	50.00	50.50	49.25	50.95	51.05	47.30	50.05	51.05	49.80	54.49	56.65	50.95
DC_20	TOP10	76.00	72.50	100.00	100.00	90.00	73.50	89.00	98.50	86.00	78.00	80.00	93.00	100.00	80.50	54.00	78.00
	TOP20	85.25	76.75	98.50	96.25	92.50	76.25	84.50	98.75	77.25	78.25	79.50	95.50	98.00	72.25	50.75	84.50
	TOP30	84.80	75.15	95.25	97.00	91.00	82.40	84.10	96.65	79.55	76.05	79.65	95.30	95.30	76.20	58.00	88.80
	TOP100	84.55	76.45	96.85	95.90	88.85	83.35	88.95	91.30	80.05	79.95	83.95	87.45	93.00	78.50	63.75	86.00
	Processed	54.20	53.05	51.65	50.90	53.75	53.85	53.15	53.85	55.15	51.35	51.85	55.85	52.55	57.35	60.05	55.05
DC_30	TOP10	77.00	77.00	99.00	100.00	88.00	79.00	91.00	99.50	83.00	80.00	84.50	92.00	100.00	81.00	67.00	83.50
	TOP20	86.75	84.75	98.00	95.00	90.25	77.50	87.75	99.50	77.75	79.75	82.25	95.75	99.00	72.50	61.75	88.25
	TOP30	85.80	81.75	94.95	96.00	89.50	82.60	87.60	96.55	79.80	79.35	81.45	94.85	95.65	76.45	66.45	91.70
	TOP100	86.35	79.70	97.10	96.20	88.60	84.05	89.85	92.55	81.25	81.75	85.15	87.45	93.40	79.85	71.10	89.75
	Processed	56.55	55.55	53.15	53.50	57.25	56.85	55.70	56.50	58.85	55.85	53.80	61.05	54.80	60.55	64.45	58.75
DC_40	TOP10	78.50	82.50	100.00	100.00	89.00	82.00	91.00	98.00	89.00	82.50	84.00	95.50	99.00	89.00	70.50	85.00
	TOP20	86.75	88.75	99.00	96.25	92.75	83.75	89.00	99.00	82.00	82.75	83.25	96.25	98.00	80.00	66.00	88.50
	TOP30	86.65	87.55	97.00	96.80	91.45	87.55	89.05	97.65	82.90	80.00	82.30	95.80	94.15	83.60	69.65	91.15
	TOP100	86.85	85.80	97.85	97.25	91.20	86.10	91.85	93.65	82.55	82.85	86.05	89.70	93.10	84.45	73.15	89.90
	Processed	58.00	57.40	54.45	55.15	59.35	57.85	57.50	59.05	61.40	57.60	56.05	65.40	57.70	63.55	66.10	62.45
DC_50	TOP10	80.00	81.00	98.50	100.00	92.50	77.00	90.50	99.50	88.00	80.00	88.00	92.00	99.50	90.00	74.00	80.00
	TOP20	88.25	88.25	98.75	95.75	94.75	80.25	88.00	99.75	80.00	82.75	84.50	94.50	99.00	78.00	72.25	86.25
	TOP30	87.50	87.40	96.60	96.65	92.40	85.90	90.05	97.45	82.35	82.10	84.35	93.25	95.90	82.55	74.05	90.40
	TOP100	87.50	85.40	97.70	96.85	93.00	85.95	92.15	93.45	83.15	83.30	86.25	88.05	94.50	84.15	75.70	88.75
	Processed	59.15	58.55	55.45	57.60	62.55	59.50	59.75	59.65	63.70	60.75	57.70	67.20	59.40	63.85	67.70	61.95

DC = Percentage threshold to discard a cluster.

Percentage of the best 10, 20, 30 and 100 ligand and snapshots interactions (TOP10, TOP20, TOP30 and TOP100), and processed snapshots obtained when the docking analyses start after 20% of snapshots have been docked.

	Ligand	TCL300	TCL400	4PI	665	8PC	8PS	468	566	641	5PP	744	GEQ	JPJ	JPL	JPM	TCU
	PDB ID	2B35	1P45	2NSD	2H7L	3FNE	2B37	2H7P	2H7I	2H7M	2B36	2H7N	1P44	3FNH	3FNG	3FNF	2X22
DC_10	TOP10	76.50	81.50	100.00	100.00	91.50	69.00	87.50	98.50	84.50	78.50	79.50	94.50	99.00	82.50	58.50	84.00
	TOP20	84.00	82.75	99.00	96.00	93.00	73.00	82.75	99.00	75.00	76.50	80.50	96.25	96.50	71.50	53.75	87.50
	TOP30	83.45	81.05	97.00	96.80	91.50	79.70	82.25	96.20	77.95	75.05	80.35	95.55	94.20	76.90	60.90	90.70
	TOP100	84.25	80.15	97.70	95.35	89.55	82.75	87.95	91.70	78.55	79.10	85.80	86.85	93.00	80.20	65.00	86.30
	Processed	56.05	53.85	54.20	52.65	54.45	54.15	52.95	54.35	55.10	51.75	53.25	55.20	53.80	58.65	61.15	55.45
DC_20	TOP10	79.50	74.50	99.50	99.00	91.00	63.50	91.00	100.00	86.00	83.50	82.50	92.50	99.50	86.00	57.50	81.50
	TOP20	86.75	81.25	99.75	96.00	93.00	71.50	88.25	99.75	79.75	82.75	82.75	95.75	98.50	77.75	54.50	87.75
	TOP30	85.45	79.35	97.25	96.70	91.25	78.90	85.85	97.40	82.30	79.85	82.40	95.70	95.85	82.30	60.60	91.40
	TOP100	85.65	80.00	98.20	96.20	90.50	83.50	90.10	92.30	81.75	81.75	86.25	88.50	93.90	82.10	67.30	88.30
	Processed	57.60	56.55	54.85	54.20	56.80	56.20	56.05	56.85	59.25	55.20	55.15	59.80	55.25	60.35	63.05	58.50
DC_30	TOP10	82.50	83.00	99.50	100.00	92.00	76.00	92.00	100.00	88.00	84.50	87.00	95.00	99.50	80.50	72.00	86.50
	TOP20	89.00	89.25	99.00	96.50	93.50	79.25	89.00	99.50	81.75	83.25	83.00	97.25	99.50	73.00	71.75	90.50
	TOP30	87.55	86.60	97.30	97.20	92.15	84.35	88.70	97.10	84.30	79.65	82.75	96.45	96.10	77.10	73.75	92.75
	TOP100	88.25	84.90	98.30	96.95	91.20	85.90	92.50	93.45	84.45	82.75	86.50	88.30	94.30	80.20	75.90	90.30
	Processed	59.80	58.45	56.00	56.75	60.50	59.10	58.80	58.60	62.50	58.95	57.05	63.65	58.95	63.20	67.20	61.45
DC_40	TOP10	82.00	84.00	100.00	100.00	96.00	82.50	90.00	100.00	92.00	81.50	84.50	95.50	100.00	90.00	75.50	83.50
	TOP20	89.25	90.75	99.00	96.75	97.25	84.00	89.50	100.00	84.50	83.75	85.25	97.00	99.00	81.50	71.75	89.00
	TOP30	87.60	87.80	96.90	97.40	94.25	88.05	89.20	99.00	86.45	82.10	84.75	95.70	95.85	85.85	74.15	92.35
	TOP100	88.40	87.10	98.20	97.30	92.85	86.85	92.55	95.15	84.95	85.15	87.25	89.85	94.35	85.70	77.70	91.15
	Processed	60.65	60.00	57.30	59.20	62.65	60.80	61.15	61.05	63.90	62.20	58.30	66.60	59.85	65.25	68.85	64.75
DC_50	TOP10	81.50	87.00	100.00	100.00	95.00	85.50	89.50	99.00	92.00	83.50	89.00	96.50	99.50	90.50	78.50	86.00
	TOP20	90.25	91.50	99.50	95.75	96.25	86.75	88.75	99.50	84.00	84.75	85.25	97.75	99.25	80.00	71.75	91.25
	TOP30	89.80	88.55	96.95	96.60	93.75	89.70	90.20	98.25	85.55	83.40	84.75	97.40	96.30	84.40	75.70	93.50
	TOP100	89.95	86.40	98.20	97.30	93.05	88.70	93.70	94.75	85.20	86.15	88.10	90.65	95.65	85.50	77.50	91.30
	Processed	62.25	61.40	57.45	60.60	64.50	62.90	61.90	61.80	66.05	62.85	59.95	68.95	61.85	66.35	70.35	65.60

DC = Percentage threshold to discard a cluster.

APPENDIX C – E-FREDOCK RESULTS

Best FEB values from the tested ligands along
with their cluster and batch identifications for the batch and cluster analysis functions - Part I.

			ANALYSIS BATCH FUNCTION					ANALYSIS CLUSTER FUNCTION				
Ligand	**Database**	**1ENY FEB**	**Id Snap.**	**Id Batch**	**Id Cluster**	**Best FEB**	**Proc. Snap. (%)**	**Id Snap.**	**Id Batch**	**Id Cluster**	**Best FEB**	**Proc. Snap. (%)**
34378053	ZINC	-9.61	8625	9	10	-12.33	45.03	8625	2	10	-12.33	54.65
1P44_GEQ	PDB	-10.70	629	1	9	-12.21	48.92	629	5	9	-12.21	49.71
12047789	ZINC	-10.57	855	2	2	-12.13	42.64	855	6	2	-12.13	53.75
35361468	ZINC	-10.39	15889	1	36	-11.91	43.79	15889	1	36	-11.91	55.73
56919632	ZINC	-7.51	1043	2	3	-11.89	46.33	1043	2	3	-11.89	49.81
11871395	ZINC	-9.81	920	2	1	-11.79	43.82	920	4	1	-11.79	53.30
34378052	ZINC	-10.80	6525	6	10	-11.56	42.18	6525	2	10	-11.56	53.81
23360796	ZINC	-8.92	18107	2	46	-11.50	41.48	18107	1	46	-11.50	56.22
65298323	ZINC	-9.88	19497	2	39	-11.38	35.53	19497	2	39	-11.38	43.49
2H7I_566	PDB	-8.90	19888	5	11	-11.33	52.56	19888	2	11	-11.33	49.75
9251152	ZINC	-9.46	1319	3	1	-11.05	39.10	1319	2	1	-11.05	44.57
20027278	ZINC	-7.91	17810	6	13	-11.08	41.64	15870	1	32	-11.03	45.67
63479935	ZINC	-8.56	18465	3	36	-11.46	35.06	17216	4	32	-10.95	45.69
63479951	ZINC	-9.60	17216	9	32	-10.95	35.50	17216	2	32	-10.95	32.21
90914428	ZINC	-8.82	17760	6	45	-10.90	43.37	17760	8	45	-10.90	49.06
63349859	ZINC	-9.92	664	1	1	-11.49	44.25	19846	2	2	-10.84	47.24
53364786	ZINC	-8.32	19214	2	39	-11.00	41.11	18156	2	39	-10.83	32.95
1456628	ZINC	-8.26	1047	2	3	-10.78	45.21	1047	2	3	-10.78	51.84
3FNH_JPJ	PDB	-9.90	8483	8	10	-9.63	47.89	15817	1	41	-10.74	47.49
2NSD_4PI	PDB	-10.20	1390	2	3	-10.56	53.61	1390	2	3	-10.56	48.22
4073149	ZINC	-8.57	17778	2	46	-10.16	45.45	16649	2	41	-10.53	56.14
91870997	ZINC	-8.50	17618	2	46	-10.68	47.23	17779	3	36	-10.46	44.88
63503064	ZINC	-8.40	17613	2	41	-10.45	42.09	17613	1	41	-10.45	50.48
36676865	ZINC	-8.87	1038	2	3	-10.44	45.33	1038	2	3	-10.44	52.52
2347739	ZINC	-8.40	18445	8	13	-10.41	39.88	18445	4	13	-10.41	50.21
63738775	ZINC	-8.26	978	1	11	-10.52	45.40	979	2	3	-10.35	49.51
9197776	ZINC	-9.32	16715	6	32	-10.32	43.89	16715	7	32	-10.32	52.57
39532319	ZINC	-8.18	15837	2	32	-10.25	47.03	15837	11	32	-10.25	51.44
2H7M_641	PDB	-9.10	4257	3	10	-10.11	47.75	4257	8	10	-10.11	48.99
2H7L_665	PDB	-9.70	2539	2	4	-9.91	50.62	2539	3	4	-10.06	53.64
17243209	ZINC	-8.20	17935	6	18	-9.87	39.96	18715	10	32	-9.79	47.99
2H7P_468	PDB	-9.00	920	2	1	-9.62	49.84	15824	2	32	-9.75	53.10
11074320	ZINC	-7.97	1056	1	16	-10.38	40.34	16084	4	45	-9.74	50.15
3FNE_8PC	PDB	-9.60	1374	1	7	-9.71	52.46	1374	1	7	-9.71	52.04
2H7N_744	PDB	-9.00	3636	2	10	-9.52	50.65	3636	4	10	-9.52	49.66
2B36_5PP	PDB	-8.90	1377	4	1	-9.05	50.97	4499	16	10	-9.11	48.51
2B37_8PS	PDB	-8.80	546	1	1	-8.70	49.72	1299	2	1	-8.86	49.19
2B35_TCL300	PDB	-8.50	5584	1	26	-8.47	50.67	5584	1	26	-8.47	52.42
1P45_TCL400	PDB							5584	1	26	-8.45	52.52
2911927	ZINC	-9.50	18046	8	45	-10.53	37.04					
2924572	ZINC	-9.27	19207	2	45	-11.35	39.86					
4335232	ZINC	-7.97	15816	2	32	-10.31	44.86					
5200961	ZINC	-8.57	18720	3	43	-10.93	39.55					
6144048	ZINC	-9.53	1131	3	1	-10.80	46.06					
6648224	ZINC	-8.42	16117	2	45	-10.80	44.69					
8323837	ZINC	-8.19	1034	2	3	-11.10	43.54					
8971422	ZINC	-8.06	15870	2	32	-10.73	45.15					
9130690	ZINC	-8.54	1086	1	15	-11.07	45.93					
9130701	ZINC	-8.43	1043	2	3	-10.25	45.71					
9197790	ZINC	-9.38	16386	6	18	-10.36	42.19					
9197821	ZINC	-9.11	16386	6	18	-10.21	42.35					
9409766	ZINC	-8.51	17110	8	32	-10.05	40.57					
9522091	ZINC	-8.66	863	1	4	-10.63	45.56					
11783975	ZINC	-7.74	16392	4	32	-10.53	41.18					
14989185	ZINC	-8.47	3745	1	13	-10.19	46.13					
15038988	ZINC	-8.11	1319	3	1	-10.32	45.65					
19336692	ZINC	-7.73	4551	5	10	-9.70	40.93					
20285686	ZINC	-9.09	16692	6	32	-10.62	40.81					
20753806	ZINC	-9.47	18076	6	18	-10.12	38.13					
20836860	ZINC	-9.03	1034	2	3	-10.16	45.79					
24000894	ZINC	-8.50	1034	2	3	-10.28	42.77					
25286217	ZINC	-10.08	898	2	1	-11.47	47.10					
31165497	ZINC	-7.18	934	2	1	-10.23	44.90					
35727540	ZINC	-9.12	877	2	1	-11.27	42.32					
38570167	ZINC	-8.82	1036	2	3	-11.25	47.10					
39445440	ZINC	-7.85	4205	3	10	-9.92	45.57					

Best FEB values from the tested ligands along with their cluster and batch identifications for the batch and cluster analysis functions - Part II.

			ANALYSIS BATCH FUNCTION					ANALYSIS CLUSTER FUNCTION				
Ligand	Database	1ENY FEB	Id Snap.	Id Batch	Id Cluster	Best FEB	Proc. Snap. (%)	Id Snap.	Id Batch	Id Cluster	Best FEB	Proc. Snap. (%)
39923320	ZINC	-8.25	15840	2	32	-11.80	42.05					
40266184	ZINC	-8.67	979	1	3	-10.31	40.79					
41584148	ZINC	-8.72	3909	1	26	-9.97	42.38					
41584155	ZINC	-8.11	18031	6	13	-9.38	44.88					
41584161	ZINC	-7.99	18042	8	45	-9.93	41.86					
41584170	ZINC	-7.94	18042	8	45	-9.61	44.52					
41584175	ZINC	-8.11	3870	4	15	-9.48	44.47					
63234429	ZINC	-7.12	18532	2	43	-8.59	52.08					
63234437	ZINC	-7.00	6508	6	10	-8.22	51.94					
63362881	ZINC	-9.71	19023	3	14	-10.71	38.36					
63576270	ZINC	-7.00	18086	6	18	-8.39	53.77					
63772586	ZINC	-7.46	17778	2	46	-8.59	51.05					
63948238	ZINC	-7.00	920	2	1	-8.56	54.44					
64002358	ZINC	-9.48	5119	4	2	-10.48	41.69					
64019380	ZINC	-6.95	1034	2	3	-8.33	54.49					
64040264	ZINC	-8.27	8722	9	10	-9.67	44.41					
64040549	ZINC	-8.39	15550	1	32	-10.42	44.19					
64057877	ZINC	-8.12	1035	3	1	-10.75	45.66					
64074412	ZINC	-8.17	1035	3	1	-10.55	46.62					
64074451	ZINC	-8.51	934	2	1	-10.64	46.89					
64103060	ZINC	-8.19	1034	2	3	-10.70	46.41					
64604357	ZINC	-7.96	1061	2	3	-9.76	47.50					
64625806	ZINC	-8.67	19224	2	39	-9.58	44.73					
64626041	ZINC	-7.80	2903	2	5	-10.04	44.14					
64889693	ZINC	-9.31	2080	1	11	-11.15	43.07					
64889694	ZINC	-9.07	2080	1	11	-11.18	43.09					
65298175	ZINC	-9.59	19088	3	31	-10.46	39.48					
65298319	ZINC	-9.97	19009	6	30	-11.03	40.00					
67641394	ZINC	-9.11	19942	5	11	-10.08	44.98					
70656077	ZINC	-7.67	946	1	6	-11.84	49.46					
72047160	ZINC	-8.29	1043	2	3	-11.00	46.02					
89608939	ZINC	-10.41	886	1	6	-11.49	43.58					
90185596	ZINC	-9.10	7921	7	10	-10.18	45.59					
93933985	ZINC	-6.50	17803	3	41	-8.10	39.54					
94621875	ZINC	-7.46	18720	3	43	-8.91	52.22					
94621892	ZINC	-7.76	19857	7	14	-10.28	46.04					