universität
wien

DISSERTATION

Titel der Dissertation

„Development and Application of Distributed Computing Tools for Virtual Screening of Large Compound Libraries“

Verfasser

Yogesh D. Aher

angestrebter akademischer Grad

Doktor der Naturwissenschaften (Dr.rer.nat.)

Wien, 2012

Studienkennzahl lt. Studienblatt:	A 091 449
Dissertationsgebiet lt. Studienblatt:	Dr.-Studium der Naturwissenschaften Pharmazie
Betreuerin / Betreuer:	Univ.-Prof. Dr.Gerhard F. Ecker

Acknowledgements

“Sometimes words are hard to find… I’m looking for that perfect line… To let you know, you’re always on my mind….” (Bryan Adams)

It surely is an overwhelming task if not a daunting one because no matter how perfect a line may seem to me, it will never do justice in acknowledging all the individuals who have supported and guided me to reach this level. Although their interest for my well-being can never be repaid by mere words, I acknowledge their solicited (and especially unsolicited) advice, efforts and guidance.

To start with, I would like to thank the ‘Gentleman Upstairs’, the ALMIGHTY, for all the times when my wishes were fulfilled and for giving me the strength and spirit to carry on when they were not. In retrospect, I have realized that the apparently unpleasant episodes of my life have occurred for a reason and for my good. At the risk of sounding immodest, I can confidently say that I am a better and a more perceptive person, today in spite, or rather because of these disagreeable events.

I consider myself immensely fortunate and it will always be a matter pride and honor to be a student of University of Vienna.

I would like to express my deep sense of gratitude and immense respect to my esteemed teacher and revered supervisor, Prof. Gerhard F. Ecker, for providing me the opportunity to carry out my PhD in his prestigious ‘Pharmacoinformatics Research Group’. His continuous enthusiasm, words of inspiration, encouraging support and guidance during the project will always be cherished.

The grid development work was carried out in close collaboration with Prof. Wilfried Gansterer and Gerald Stampfel, faculty of Computer Sciences, University of Vienna. I am sincerely thankful to them for their valuable guidance and suggestions during the establishment of computational grid and preparation of the documentation.

I am very grateful to Prof. Michael Freissmuth and Christina Glöckel, Institute of Pharmacology, Medical University of Vienna for giving me the opportunity to work on a very exciting topic related to the role of SSRIs in lymphoma (section 3.2) and performing pharmacological studies on selected compounds.

I would like to thank all my colleagues and friends at Pharmacoinformatics Research Group. Starting with the senior members of the group, thank you Barbara Zdrazil, Dominik Kaiser, Rita Schwaha, Michael Demel, Khac-Minh Thai for your guidance and suggestions. Lars Richter (or shall I say – nature-man/Tarzan), thank you for your help in my initial days of Vienna and being such a nice neighbor in the lab. Daniela Digles, thank you for your support with respect to my writing, SOM and bureaucratic tasks which really helped me a lot to improve me scientifically and to lessen the language barriers. Ishrat ji, despite the critical political situation between our countries, we shared a special brother-sister relation, became the best friends in the lab and shared everything without hesitation. Your counseling, advices, motivations and critical suggestions will help me to become a better human-being. Nathan Poongavanam, thank you for being such a nice, talented friend and helping me understand classification problems alongwith showing new aspect to look towards life. René Weissensteiner, thank you for introducing me to Austrians traditions and ways to look at different stages of human life starting from birth, sacrament, marriage, house-warming till demise of loved ones. Your enthusiasm for great scientific work, playing musical instruments and photography will always be remembered. Andrea Schiesaro, thank you for the Italian touch in the lab and nice talks we shared. My best wishes to you and Oyuna for expecting new angel soon in your home. Andy Jurik, thank you for a wonderful company and whole-hearted cheers whenever we see each other alongwith so many rides in your car with your perfect driving skills. Freya Klepsch and Marta Pinto, don't worry, our FC Barcelona will win champions league as well as La Liga next year. I would like to thank other colleagues of the group- Amir Seddik, Daria Tsareva, Rainer Dangl, Kathy, Vicky, Sabine, Maria, Kamil, Daniela, Anika and Christoph for their cheerful company and help.

I would especially like to thank Virag Sharma for his constant encouragement, advices, support, much needed talks, recalling past wonderful memories from NIPER and helping me during the project and my hard times.

I would like to extend my thanks to Veronika and Doblhoff-Dier family for introducing me to Austrian culture, food, festivities and sports. Learning Austrian national sports, skiing for the whole week in Kitzbühel with the help of her, Katharina and Stefan was one of the best periods of my stay in Austria. I would like to thank all Austrian and international friends who made my stay in Vienna enjoyable and sociable. Sarolta Gall, Emanuel Lohninger, Johannes Stefan, Sophie Schoditsch, Verena Hofecker, Miroslav Miho, Vierka, Dusan, Ansku, Ines, Elena, Raphie, Tamara, etc. thanks for all the good times. Miro, I had really wonderful times of my life when I visited you every summer in Banska Bystrica, Slovakia. Visiting families of Sarolta (Miercurea Ciuc, Romania), Johannes (Klagenfurt, Austria) Sophie (Oberwart, Austria), Verena (Vienna) and Raphie (in Freilassing, Germany) only reinforced me to realize that all mothers are same, caring their children.

Of course, jovial company of all my Indian friends in Vienna – Dhairyasheel Desai, Sudarshan Patil, Sanjay Boddul, Sachin Thorat, Rashmi Jois, Pradip Hiwale, Jayaraj Manepalli, Ravi & Smita Gaware, Abhishek Singh, Ajinkya & Sunetra Sase, Harshad Bidwai, Aparna Date, Shivanand & Shrilaxmi Hegde, Vaibhav Gawali, Priyanka Saxena, Prakash Chandak, Vineet Mahajan and Aniket Nikam made my stay and work easier than it would ever have been.

All my deepest sense of gratitude goes to my loving grandparents, parents, brother and whole Aher family for their unconditional love, moral support and believing in me throughout my studies. Sorry Apeksha, I couldn't be there for your wedding. Hope you forgive your brother.

I would like to end this infinite gratitude, which cannot possibly cover all the names, with short Sanskrit stanza from Bhagawada Gita (chapter 2.47)

कर्मण्येवाधिकारस्ते मा फलेषु कदाचन ।
मा कर्मफलहेतुर्भूर्मा ते सङ्गोऽस्त्वकर्मणि ॥

English transliteration:

Karmaňye Vădhikăraste, Mă phaleShu kadăchana,
Mă Karma Phala Hetur Bhurmătey SaNgostvaAkarmani

English translation:

You have a right to perform your prescribed duty, but you are not entitled to the fruits of action
Never consider yourself the cause of the results of your activities, and never be attached to not doing your duty
(In simple terms it means: Keep on performing your duties without expecting for any reward in return, leading a selfless life – this is what it is all about.)

Contents

Acknowledgements .... **i**

Contents .... **iii**

1. Introduction **1**

1.1 Drug Discovery and *in silico* Techniques 3

1.1.1 Target identification 4

1.1.2 Lead Discovery and Lead Optimization 5

1.1.3 Key Aspects and Challenges of *in silico* Drug Discovery 6

1.2 Grid Technologies for *in silico* Drug Discovery 8

1.2.1 Grids - Resource Sharing, Information Flow and Speed Acceleration 9

1.2.2 Basic Characteristics of Grids 11

1.2.3 Grid Applications in Drug Discovery Projects 15

1.2.3.1 Cycle Scavenging Screensaver projects 15

1.2.3.2 Drug Discovery on Computational Grids 17

1.2.3.3 Enterprise Grid Computing 20

1.3 Aim of the Thesis 21

2. Grid Development and UVieCo – University of Vienna Condor Pool **22**

2.1 Introduction 24

2.1.1 Grid Components 27

2.1.2 Types of Grids 29

2.2 UVieCo and High Throughput Computing using Condor 31

2.2.1 Condor 32

2.2.2 UVieCo Architecture 38

2.2.3 Firewall Transversal 40

2.2.4 Security 43

2.2.5 Virtualization and CoLinux Package 46

2.2.6 Applications and Performance Efficiency Measurement 47

2.2.7 Conclusions and Future Tasks 53

3. NCI-60 Data Mining and Grid Applications 54
3.1 NCI-60 Data Mining 55
3.1.1 Gene expression profiles and drug activity patterns 57
3.1.2 Expression profiling of ABC transporters in NCI-60 58
3.1.3 Correlation of mRNA expression of ABC transporters and drug sensitivity 59
3.1.4 Calculation of Pearson correlation coefficients 60
3.2 Shape Similarity Approach 63
3.2.1 Identification of P-gp substrates 65
3.2.2 Identification of SSRI-like compounds using shape similarity approach 72
3.3 Imbalanced Data Classification Problem 86
3.3.1 Prediction of P-glycoprotein substrates and non-substrates from highly imbalanced dataset using cost-sensitive machine learning methods 89
Bibliography 108
Appendix 121
UVieCo Administration 121
UVieCo Implementation 125
Perl Script to calculate Pearson correlation coefficients 139
Abstract 142
Zusammenfassung 144
Curriculum Vitae 146

CHAPTER 1

INTRODUCTION

1. Introduction

Despite the availability of numerous resources for identifying potential therapeutic compounds, the current drug discovery process is very tedious, expensive and time consuming. It takes about 12-15 years and more than US$800 million for a new molecule after being discovered to get approved from FDA and accessible to the public [1, 2]. The success rate in this process is very low, i.e. only a single New Molecular Entity (NME) out of >10,000 candidates passes the barriers of preclinical evaluations and clinical trials [3]. Late stage failures due to biopharmaceutical properties (oral bioavailability and formulation issues) and toxicity are also important to be considered. They contribute to 39% and 21% failures respectively along with 29% of failures occurring from lack of efficacy [4]. About 40% of the compounds are tested in humans after preclinical studies and out of these only 10% reach the market. This is a major challenge for the research-based pharmaceutical companies [5, 6].

Reduction in the late-stage attrition rate of NMEs can be achieved by specifying stringent and exhaustive quality assessment criteria at important and early stages of drug discovery process. A comprehensive assessment in terms of molecular characterization, chemical and functional behavior, integrity, synthetic feasibility, structure-activity relationships (SARs) as well as bio-physicochemical and pharmacokinetic properties is crucial in selection and prioritization of hits to be progressed into lead series, thus minimizing the research timeline in the discovery stage [7]. Technological developments in terms of genomics and proteomics, automated protein crystallization and structure determination, pathways and protein networks elucidation as well as high throughput screening and combinatorial chemistry gave rise to remarkable abundance of new potential therapeutic targets and drug candidates. This led in forcing pharmaceutical industries and R&D organizations to speed up the productivity and rapidly screening out irrelevant candidates with low probability of successful registration and commercialization [8].

The increasing role of computer sciences and information technologies in basic research and modern drug discovery process is evident since the past 3-4 decades. This maximizes productivity in various aspects of research, interpretation, management and collaboration within multidisciplinary R&D project teams to arrive at substantiated decisions with accelerated pace [9]. New *in silico* techniques are anticipated to improve the identification of new leads and prediction of activity, bioavailability, safety and efficacy as early as possible in the drug discovery and development process.

1.1 Drug Discovery and *in silico* Techniques

An important goal of the drug discovery process is to search for new drug molecules which can bind to a specific target known to be involved in causing a disease and change the target's function. In the traditional drug discovery process, identification of the suitable drug target is the first and foremost task. These targets are biomolecules which mainly include proteins such as receptors, transporters, enzymes and ion channels. Validation of such targets is necessary to exhibit a sufficient level of 'confidence' and to know their pharmacological relevance to the disease under investigation. This can be performed from very basic levels such as cellular, molecular levels to whole animal level. Once the target validation has performed, effective compounds such as inhibitors, modulators or antagonists for such target need to be identified. This process is called lead identification where the design and development of a suitable assay is done to monitor the effect on the target under study. High-throughput screening (HTS) plays a crucial role in this phase where large numbers of chemical compounds are exposed to the target. Compounds showing dose-dependent target modulation in terms of a certain degree of confidence are processed further as lead compounds. Subsequently, the experiments are performed on the animal models in the laboratories and the positive results are then optimized in terms of potency and selectivity. Physicochemical properties and their pharmacokinetic and safety features are also assessed before they become candidates for drug development [10]. Even though most of the processes depend on experimental tasks, i*n silico* approaches are playing important roles in every stage of this drug discovery pipeline (Figure 1.1).

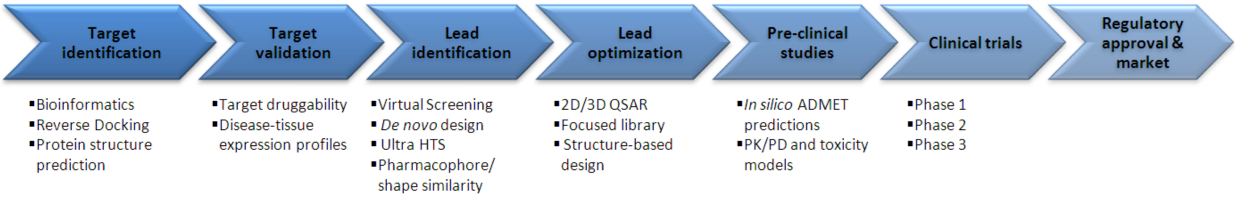


Figure 1.1: Role of *in silico* approaches in various phases of drug discovery & development pipeline.

In the following section, we will briefly highlight some of the challenging issues of data generation and advances of *in silico* techniques from the perspective of target identification and discovery of new leads.

1.1.1 Target identification

With the 'omics' revolution and advancements in the life sciences, especially in the field of biotechnology, a large amount of data is generated every day, starting from the whole-genome sequencing accompanied by microarray gene expression analysis and x-ray / mass spectroscopy of proteins and metabolites. Enormous data is being produced by an increasingly growing number of novel high-throughput and genome-wide bio-techniques with nanotechnology and lab-on-a-chip (LOC) applications providing icings on the cake. This renders the bioinformatics researchers with great opportunity to study whole genome rather than gene-by-gene analysis, guiding life sciences towards more data driven and rationalize approach [11]. As the target discovery is a primary and major aspect of *in silico* techniques' implications, it also helps to reveal all possible targets for therapeutic intervention. The haploid human genome contains about 23,000 protein-coding genes with ~8,000 targets of pharmacological interest and on the basis of ligand-binding studies, ~3,000 targets are predicted for small molecules opening broad perspective for target identification [12]. In another study, Drews and colleagues identified approximately 500 drug targets, but suggested there are likely as many as 5000- 10000 potential therapeutic targets [13, 14]. Although this unfolds a wide range for of biomolecules as drug targets, selection of the most relevant ones for a given disease is a big challenge.

Several sequence and structure-based approaches have been proposed for target identification. In the sequence-based method, functional information about the target and its positioning in the biological networks is provided to detect unique targets from the disease causing pathogens (e.g. bacteria or viruses) by comparing functional genomics of humans with corresponding genomics of pathogens [15]. In this way, it is possible to distinguish the targets in the pathogens as well as for endogenous diseases, subsequently resulting in the discovery of novel targets by genomics difference between normal and abnormal tissues. With the increasing number of 3D structures of proteins, structure-based approaches are becoming popular. These methods are useful to specify topographies of the binding complementarity in the protein for the given ligands and later these results can be validated by experimental procedures in the laboratories. The structure-based methods guide the optimization of new leads into drugs and now-a-days employed much earlier in the drug discovery process for accessing 'druggability' or tractability of targets [16].

1.1.2 Lead Discovery and Lead Optimization

Millions of chemical compounds consisting of natural products were synthesized and isolated in the past. Lipinski and Hopkins related this vast chemical space to the cosmological universe and small molecules are referred as stars. Identification of the biologically relevant chemical space, i.e. regions containing biologically active compounds from such universe is a challenging task [17]. With the availability of thousands of targets, it is not possible to screen these existing targets with all the available compounds. Thus, it is important to develop suitable assays to mine the existing huge chemical space for exploring a large number of new leads or hits. Although tremendous advances in HTS have been made which allow the screening of thousands of compounds every day, the cost and time would be greatly saved if the number of randomly tested compounds is reduced. Recent improvements in virtual screening have shown its importance in discovering lead compounds and the hit rate has enriched by about 100-1000-times over random screening [18]. However, virtual screening retrieves the hits from existing compounds or old drugs. For the discovery of new molecular entities, several *in silico* approaches have been proposed such as virtual/combinatorial compound library design, fragment-based screening, *de novo* drug design, etc. These methods aim at building new molecules from available building blocks/fragments rather than considering whole molecules. Target specific focused-libraries can also be generated using known information about three-dimensional complementarity (i.e. shape, size and physicochemical properties) of target-binding site to reduce number of compounds needed for synthesis and *in vitro* HTS screening [19]. Such approaches are very useful for the membrane proteins like receptors, transporters and ion channels where it is extremely difficult to obtain the x-ray crystal structures. In these cases, information about the known ligands for specified target can be used to derive e.g. *in silico* pharmacophore models which are then used for virtual screening to specify a subset of lead molecules. The computational demand in these cases scales according to the number of candidate molecules as well as the time required to perform these virtual screening experiments and to explore the conformational space of the novel compounds.

As discussed briefly in the earlier section, *in silico* screening techniques can be useful to detect complex structural and bioactivity relationships from the ligand-oriented information or the knowledge about protein structures. Ligand-based screening of new hits considers several methods such as techniques involving similarity principle, i.e. similar compounds have similar biological activity/profiles, in several forms such as classical similarity using descriptors or fingerprints, pharmacophore modeling, classification algorithms and self organizing maps. Structure-based approaches involve docking of large compound libraries into targeted protein structures and scoring and ranking of the poses/compounds, binding free energy calculations, molecular dynamic simulations and more accurate quantum and molecular mechanics methods. These methods suffer from high computational demands and manual intervention which put forward the need of high throughput computing environment for carrying out these tasks.

1.1.3 Key Aspects and Challenges of *in silico* Drug Discovery

Fundamental priorities of any pharmaceutical company are to reduce the research cost and time of the drug discovery process and to increase the reliability and confidence in discovered leads which will be processed through development stages. There are several challenges in the *in silico* drug discovery process which need to be addressed to accomplish these goals [20].

- Management of huge amount of versatile scientific data – The i*n silico* drug discovery process generates vast amount of data in terms of structures, sequences, compounds, models, databases and images. Interpretation and integration of such data is a challenging task which helps in enhancing knowledge discovery as well as simplification of complex workflow. This involves data manipulation, data format standardization, definition of dataflow in a distributed system, data storage and administration with respect to the infrastructure and software providers, database updates and registration of data and meta-data.
- Management of different softwares – Integration of a variety of software poses another challenge which helps in the construction of efficient and complex

workflows and aids in data mining and data management. Creation of web servers which are available via internet is another effective way to provide software in a distributed environment. Man power is another important issue to use software where several experts are needed to keep a record of updates and maintenance of software, managing workflows and pipelines to project new methodology, utilize remote services, analyze and exploit the outputs and come to a conclusion about which compounds to be proposed for further screening. The main advantage of scientific workflows is that they assist researchers and decision makers in organizing the complicated tasks in a flexible way and to pass over the data, information and knowledge to the organization.

- Deployment of intensive computing – Huge amount of computing resources and computing power is the main requirement of large bioinformatics calculations, e.g. docking hundreds of compounds in several targets retrieving thousands of poses or molecular dynamics simulations of even a small system require in the order of a few TFlops within one day. Several computational methods based on all-atom physics-based force fields comprising of implicit solvation and free-energy calculations also need large computational resources to represent accurate protein structure models. Generally, the computer centers' clusters and small university-wide servers are occupied with a large number of short tasks causing problems for such gigantic jobs [20].
- Collaborations between public and private partners – When life sciences are combined with computational approaches for *in silico* drug discovery, strong remote collaborations involving sharing of resources (data, information, knowledge, tools, software and workflows) as well as infrastructures (computers, servers, storage devices and networks) is required and need to be maintained. Such comprehensive interfaces not only help to integrate outputs from *in silico* drug discovery but also from experimental processes reducing development time of new methods.
- Security, firewalls and authentication - Important aspect of any academic organizations and pharmaceutical industries is to effectively protect critical data, information and intellectual properties to avoid mal-behavior. This includes – allowing access to the users from various institutions across firewall barriers and

their authentication, provision of flexible mechanisms and support to resource owners to manage user accounts, their privileges and privacy settings [20].

As discussed in this section, the *in silico* drug discovery process faces some vital challenges such as addressing huge amount of data, data and software integration, deployment of intensive computing, public-private remote collaboration, resources sharing and strong security and protection constraints. Thus, in house as well as remotely available tools, data, storage and computing infrastructure need to be shared and integrated in an influential and secured environment.

1.2 Grid Technologies for *in silico* Drug Discovery

Timely analysis and evaluation of incoming data in research and development is very important for the decision-making processes in this highly competitive world of industrial drug discovery. The degree of accuracy of these decision making processes is in turn dependent on the computational resources used to generate the precise hypotheses [21]. In the context of *in silico* screening and molecular modeling, the demand for computational power has increased drastically in recent years to perform rigorous calculations. Regardless of the present computing infrastructure, the emergence of various software, algorithms and programs in all scientific fields which require tremendous calculation time for their execution has demanded further enhancement. In particular, there is a need of investigating new innovative strategies for more efficient utilization of distributed computing resources by combining them in the form of clusters and computational grids with regards to the interdisciplinary research activities. Although availability of clusters and huge servers with thousands of processors eases the computational research, they are dedicated resources and their use is limited by their high costs, user complexity, permission issues, resource management, queueing mechanisms and preferences.

Dramatic advancements in computing hardware with improved networking technology have enabled the coordination of distributed resources within or outside the enterprise domain, providing access to computing facilities at much faster rate with affordable cost. This resulted in diverting the focus of the scientific community from the supercomputers to the personal desktop workstations and PCs. Despite their slow speed and limited performance, these small computers prove beneficial and circumvent the shortcomings of supercomputers when used in conjunction with many others. The environment of grid-enabled distributed computing in the institution increases the computational power to a great extent with individual ownership to each of the PC owner.

Grids render a new dimension to the emerging information technology paradigm providing and coordinating the resources such as various organizations, people, computing, storage and networking facilities as well as data, knowledge, software and workflows. In simple terms, a grid is the well organized combination of dispersed but networked computational resources and the intermediate middleware furnishing streamlined services to the users. Grid technology imparts an opportunity to combine different aspects of scientific disciplines such as life science research, pharmaceutical industries, healthcare systems, infrastructures and field work using the collaborative information technology environment. It provides a distinct possibility to collect and analyze the distributed information from a situation where all data are not localized at a single depository. The data can be stored and maintained at any place within a grid and still be accessed transparently by any authorized users. Similarly, the computational resources of a grid can also be shared and mobilized whenever needed on demand. The grids will be discussed further in details in chapter 2.

1.2.1 Grids - Resource Sharing, Information Flow and Speed Acceleration

The grids provide the multi-dimensional horizons to the *in silico* drug discovery in several forms. They present some unique and remarkable opportunities for sharing the resources and collaborations, expose interesting ideas and perspectives to deal with incoming information flows and supply the resources for speeding up the execution of time-consuming software [22] .

With respect to provision of the opportunities regarding resource sharing and collaborations, numerous distinct features imparted by grids need to be considered. Grids allow cross-organizational collaborative resource sharing among universities, academic research institutions and pharmaceutical industries, leading to increase in co-operation and communication between partners. This in turn helps in routine mobilization of information and resources or in an emergency situation. High performance computing and cycle-stealing is another benefit of grids where they exploit otherwise-wasted CPU power to maximize optimal usage of resources and take advantage of spare computation. Another interesting feature of grids includes networking and aggregating available hardware resources which assist in reduction of costs and manpower. Grids allow the use of large structural and chemical databases and time-consuming life-sciences software to take advantage of new information technology applications. Attainment of mutual benefits by collaborative exploration of distributed, large, diverse and complex information has been made possible with grids.

Handling the information flows is another important aspect of grids. In this scenario, grids open exciting perspectives which allows permission to handle relevant data and regular updates of the databases and publications. Grids also allow secured and transparent access to the storage and retrieval of large amount of archived data in self-organized and automated manner. They help in compliance of the pre-defined rules to analyze, organize and co-ordinate the data and information in an understandable way. Formation of relevant databases and workflows has been made possible to understand the task under study and keeping the records of the respective steps. Implementation of the infrastructures and services to aggregate the information from various firms and research centers assist in enhancing interpretation of the problem in an efficient fashion.

Execution of the time-consuming exercises is another major concern of the current *in silico* drug discovery processes and grids help to accelerate these tasks in efficient ways by allowing implementation of high performance computing to advanced areas of research for more rigorous and accurate calculations. They allow execution of stringent data analysis, mathematical modeling and *in silico* drug discovery tasks with the aid of large computing resources such as supercomputers, graphical processing units, servers and even desktop PCs. They assist in exchanging ideas, tasks, tools and workflows among the organizations and researchers. Creation of automated job scheduling and submission policies and distribution facilities to perform large compute-intensive tasks in effective way has become possible with grid technologies. With the successful application of grids, permission to the resources and services all time around the year (365x24) on heterogeneous platforms distributed across geographically diverse sites has become a reality. Thus, in total, grids are distinct implementations for information sharing and collection, taking proficient inputs and opinions from experts in networking, transfer and spreading of data and knowledge resources among the partners after certain time intervals or in an emergency, which helps in creation of a suitable environment for development of *in silico* drug discovery processes [22].

1.2.2 Basic Characteristics of Grids

Provision of different forms of resource sharing in terms of high-throughput (or distributed) computing for the life sciences applications has been one of the major goals of grid technologies in the past years. Although it mainly involves utilization of idle desktop workstation- or PC-cycles, exploiting various experimental equipments at the partner or collaborators' sites in a flexible, reliable and secure way offers significant benefits. For example, secure remote coupling of the essential databases and computers/servers with expensive and specialized unique devices such as high performance NMR spectroscope, x-ray-source beamline, etc. among the participating institutions assist in important data curation and analysis. This allows the scientists to access the data instantaneous across the enterprise boundaries rather than working locally [23].

Networking support in the forms of internet, corporate intranets or World Wide Web is considered to be a crucial aspect of grid computing environments which aids in maintaining heterogeneous sources together. Although internet is considered as the best option for this delivery system, its deployment on grids is strictly restricted by the connectivity and bandwidth sharing concerns along with the slow speed and inability to deal with large amount of data [24]. Another important issue to be considered about implementation of grids over internet is the cost of data transmission, where networking services which enable the fast communication at high capacity over less cost open promising opportunities for grid computing [25]. Several exciting techniques such as optical networking, optical burst switching (OBS) [26], Quality of Service enabled networks [27], wavelength division multiplexing (WDM) technology, micro-electro-mechanical systems (MEMS) technology, Transmission Control Protocol/Internet Protocol (TCP/IP), Novell Netware, fiber distributed data interface (FDDI) and various remote procedure call (RPC) systems are attractive propositions and hardware solutions help to reach global resources at affordable bandwidth for the better communication which are the requirements of the data-intensive grids [24, 28].

Regardless of the availability of high-performance networking capabilities and continuously decreasing costs of its maintenance, it has not really fostered inter- or intra-institutional collaborations. The virtual organizations (VOs) [29] formed by such sharing and co-ordination of resources within a group of people and/or organizations face more severe problems. Since internet is used as an underlying mechanism of communication for many VOs, the security concerns in terms of hazardous intrusion and sabotage need to be considered profoundly without hampering the accessibility to the resources. Drug discovery applications and related *in vivo*, *in vitro* and *in silico* data and information are strictly confidential and must also meet regulatory requirements, providing proper attention towards data protection, safety and reliability. The primary fundamentals of privacy and confidentiality have to be followed all the times. Authentication, identity and authorization levels of the users, firewall restrictions, different ownership and administrative complications with high-level security and access control policies are the important characteristics while constructing a grid. If these attributes are not addressed properly, the establishment of a grid environment becomes quite difficult. For such cases, various solutions have been put forward which include negotiation of the access to the computing facilities, configuration of the systems so that users can use them effectively without compromising the security of the resources. This becomes very much essential in parallel computations where multiple computational resources are spread across a wide range of administrative domains running thousands of processes simultaneously. The establishment of trustworthy relationships between these sites for execution of the applications is also hampered by the dynamic nature of the grids. Furthermore, the inter-domain security negotiations should work hand-in-hand with diverse intra-domain access control technologies rather than replacing them. Several constraints are laid down into the grid environment to avoid non-secure intrusion such as initial authentication to acquire, use, manipulate and release resources, protection of user credentials (passphrases, secured keys, etc.), local and inter-domain security policies, easy encryption and decryption codes, multiple security implementation technologies, secure group communication, etc [30].

In the early 1990s, before so called middlewares gained their popularity, different legacy applications such as distributed computing environment (DCE) [31] or Common Object Request Broker Architecture (CORBA) [32]have been put forward to address these networking and distributed computing issues. The distributed applications are exclusively constructed with a purpose of sharing the resources and their execution on multiple platforms. Distributed Computing Environment (DCE) software was developed by the Open Software Foundation (OSF, Cambridge) to bring together more than 330 worldwide members of hardware and software vendors, end users and research institutions. It mainly focused on the heterogeneity of the operating systems, their security policies along with networking and cooperation among the members. DCE helped the users to develop, deploy, and manage applications that efficiently use a network's scattered computing resources [33]. On the same stream, the Object Management Group (OMG) developed CORBA, which enables to merge a variety of software components written in different languages such as C, C++, Java, COBOL, python, etc., running on different computers to work with each other like a single application. It provides a flexible communication and acts as a catalyst for distributed heterogeneous object-oriented computing environments. The main characteristics of CORBA include its heterogeneity in terms of diverse programming and implementation on different computers and equipments; the ability to be used in a variety of environments, and its object-oriented software development techniques [34]. In both of these applications, resources are not allocated and processes are not created explicitly by programs, but they connect to established "services" that encapsulate hardware resources or provide defined computational services. When the different environments do not have more information about each others, the communication is achieved using remote procedure call or remote method invocation models via standardized protocols (based on TCP/IP) that are devised for portability and not for high performance. There are a variety of problems related to these technologies such as failure to satisfy the portability and performance requirements, difficulties in their sophisticated implementation, inability to share resources or harness idle computers, and failure to address flexibility, performance issues, reliability and trust.

Tremendous technical advances in the past few years helped to handle most of these concerns providing solutions to heterogeneous resource sharing, resource management, networking, security and speed of accession across administrative boundaries. Middlewares have emerged as the boosting technologies to solve these issues facilitating the access to the computational grids by coordination of heterogeneous computational and information technology resources across a network, allowing them to function in unison. In general terms, a middleware is the building block used to construct the grid. It can be generally conceived as the layer of software sandwiched between the operating system and applications, providing a common programming abstraction across a distributed system and a variety of services required by an application to function correctly [35]. It is specifically designed to mask the heterogeneity incurred in distributed systems, networks, hardwares, operating systems and programming languages as well.

The Globus Toolkit, a trademark of University of Chicago, is such a pioneer, open source middleware used for building grids [36]. It presents a software infrastructure that integrates diverse computing resources and users from different geographical locations to demonstrate them as a single entity. It provides basic functionalities and competencies required for building a computational grid. The toolkit contains various components for implementing elementary services required for grids such as authentication and communication, security, information services, data management, resource management and application development environments. It contributes in a variety of application-oriented tasks having a major collaborative role to develop wide-spread grid-enabled applications. The Globus toolkit helps in designing and constructing large-scale testbeds, both for research and production in various scientific as well as engineering fields. Several such projects, which depend on the Globus toolkit, include European DataGrid project [37], Teragrid [38], GriPhyN (Grid Physics Network) [39], DOE Science Grid [40], etc.

Several similar middlewares emerged after the success of Globus with different goals and specialties for effective utilization of unused computer power. These include Condor[41], Unicore [42], gLite [43], GEMSS [44], Legion [45], Nimrod [46] and many more to furnish object-based remote access systems for different studies. Companies such as Entropia [47], United Devices (now known as Univa UD) [48] or Oracle Corporation [49] also work on similar streams and provide softwares for different purposes such as harvesting idle cycles, cluster scheduling, load sharing facilities and portable batch systems within institutional PC grids. Data Grid technologies built on these and other technologies address issues related specifically to computations on data and its distributed management.

Grid systems such as EU DataGrid [50]or NSF TeraGrid [51]give an impression that the current grid technologies have transformed from the experimental phase to the point where production capabilities are possible. Still there are many areas where this large scale resource sharing has not really prospered up to the expectations which may be attributed to the absence of uniform applicable standards as well as heterogeneity and fragmentation of available grid services, tools and middlewares. Increasing use of service-oriented standards and commercial web service technologies influenced the development of grid-oriented services such as middlewares and tools. Although these middleware toolkits are available for cluster and grid computing, they are very complex to implement, as prior knowledge about their underlying mechanisms and computational resources management is of prime importance. Thus, mapping life sciences applications and tools used for *in silico* drug discovery processes on a grid infrastructure is a non-trivial exercise.

1.2.3 Grid Applications in Drug Discovery Projects

There are numerous efforts for supporting scientific applications by utilizing grid environments and distributed computational resources. Grids have become an integral part of the current drug discovery and virtual screening processes. They are widely used from pharmaceutical research to bioinformatics, chemoinformatics, genomics, proteomics and other life science applications. Grid technologies proved to be very promising in terms of reducing time and increase efficiency [23, 52-60]. Several CPU cycle scavenging projects were accomplished by several organizations where worldwide network resources were utilized via internet. Different institutions and enterprises have also realized the importance of distributed computing and implemented ae grid infrastructure within the organization to effectively harness wasted computing power. A brief overview of these projects along with their basis is presented in the following section.

1.2.3.1 Cycle Scavenging Screensaver projects

FightAIDS@Home (FAAH) [61], a part of World Community Grid, is the first biomedical distributed computing project run by Olson laboratories at The Scripps Research Institute in La Jolla, California. It uses AutoDock to perform virtual screening of novel inhibitors available from the NCI (National Cancer Institute, Bethesda, MD, USA) against wild-type HIV protease. FAAH was installed on about 450 000 clients worldwide using a small software package (downloaded from world community grid) which runs in the background of the computer and was capable of screening around 10 000 ligands per day [62].

Folding@home[63] is recognized as the most powerful distributed computing project in the world designed to execute compute-intensive simulations of protein folding and other molecular dynamics for targets linked to various diseases such as Alzheimer's, Mad Cow (BSE), Parkinson's, Huntington's, many cancers and cancer-related syndromes. The project recently accomplished simulated folding in the 1.5 millisecond range. Various molecular dynamics software packages are used in this project, which include – Gromacs, Amber, Protomol, Tinker and Desmond. The computing statistics are reviewed periodically and by august 2011, the peak speed of the project was 4 native PFLOPS (6.4 x86 PFLOPS) from 430,000 individual PCs and PlayStation 3s.

The cancer screensaver lifesaver project [64]is an initiative by the department of chemistry of the University of Oxford, which has accomplished screening of 3.5 billions of compounds in one year against twelve targets in a virtual screening experiment since its implementation in April 2001, by using more than 1.5 millions of computers from more than 200 countries. This has created a 65-teraflop machine that has provided more than 100,000 years of CPU time [54]. The THINK software has been used for matching pharmacophores and full conformational searches. It ranks the hits using the ChemScore function, thus providing estimation about the free energy of binding for the conformation of each bound molecule. Subsequently, after the success of the initial *in silico* drug discovery and design on a grid in the form of the cancer screensaver project, the same institute launched the Anthrax research project [65] and the smallpox protection project [66]. During the Anthrax protection project, a clustering algorithm based robust method was invented to locate ligand-binding sites on proteins and 3.57 billion small molecules were screened on more than one million PCs to retrieve about 300,000 hits as promising candidates for the treatment of cancer [67]. Similarly, in the Smallpox protection grid project, more than 2 million users in over 200 countries contributed their idle times to find drugs to combat the post-infection effects of smallpox virus. About 35 million compounds were screened against eight smallpox proteins to narrow down the number of drug-like molecules binding and ultimately inactivating smallpox proteins [66]. The virtual screening was performed using the LigandFit molecular docking software from Accelrys. The distributed computing aspect for all these projects was handled by United Devices (now Univa UD) [48], which distributes the work packages to individual PCs and collects and validates the returned results.

Docking@Home [68] is a small scale distributed computing project having more than 6000 volunteers worldwide with the aim to create new and improved medicines against major disease targets such as P38 alpha, Trypsin and HIV proteases. The protein-ligand docking protocol is based on the CHARMM package and is run on BOINC (Berkeley Open Infrastructure for Network Computing), a free, open-source middleware system for volunteer computing.

GPUGRID [69] is an innovative distributed supercomputing project designed for all-atom biomolecular simulations using NVIDIA's CUDA-compatible graphical processing units with BOINC as the basic software platform. Several molecular dynamics simulation experiments have been carried out successfully for targets such as triose phosphate isomerase (TPI) enzymes, the D2 dopamine receptor, and HIV protease, along with free binding of benzamidine to trypsin.

Rosetta@home [70] is another promising distributed computing project for predicting and designing 3-dimensional structures of proteins mainly of malaria, anthrax, HIV, Alzeimer's disease, Spanish Flu (H1N1) Influenza and others based on the BOINC (Berkeley Open Infrastructure for Network Computing) infrastructure. With the help of more than one million volunteer PCs, it processed 58 teraFLOPS on average by June, 2011.

In another distributed computing project called drug design and optimization lab (D2OL) [71] hosted by the Rothberg Institute, a platform was developed for the users from around the globe to dedicate their personal computers for computing the interactions of small molecule drug candidates with target molecules that likely play prominent roles in specific human disease pathways, including those involved in SARS, Avian Flu, Ebola and Smallpox and other potentially devastating infectious diseases. The parallel computing aspect was managed by Sengent Inc.'s CommunityOS framework [72]. This project was active for more than seven years (ended on April 15, 2009) and returned results for docking of over 150 million drug candidate into disease targets, utilizing over 3000 years worth of CPU time in the process.

1.2.3.2 Drug Discovery on Computational Grids

The Drug Discovery Grid (DDGrid) constructed by Zhang et al [73] demonstrates the use of Peer-to-Peer (P2P) environments and grid technologies based on BOINC middleware for execution of molecular docking based virtual drug screening services on geographically distributed resources. Docking applications (in the form of Dock v4.0.1 and gsDock v2.0, which is a generic algorithm optimized form of Dock), preprocessing software and toolkits (such as autogrid/autodock, combimark, combilib) and chemical databases (e.g. ZINC, Specs, National Cancer Institute database, China Natural Product Database and Traditional Chinese Medicine Database) were used for screening against the farnesoid X receptor (FXR) to finally discover 5 compounds with high binding affinities. This system was also used in drug screening for anti-SARS, anti-diabetic and anti-arthritis drug research projects.

Molecular docking was used as a virtual screening tool in the WISDOM (World-wide *In Silico* Docking On Malaria) project to identify novel hits and leads against malaria and other neglected diseases [74]. In this work, one million compounds obtained from the ChemBridge database were screened against 5 targets of the Plasmepsin family using FlexX and AutoDock. They reported that about 41 million dockings were performed within 6 weeks, which is equivalent to 80 CPU years [75]. The Wisdom-II project [76, 77] was subsequently launched after the success of the initial WISDOM project to continue the *in vitro* screening after VS experiments. Several new targets were explored such as Glutathione-S-transferase and tubulin along with two previously well known proteins: dihydrofolate reductase (DHFR) from *Plasmodium falciparum* and from *Plasmodium vivax.* 4.3 million compounds obtained from the ZINC database were docked against the crystal structures or homology models of the above mentioned targets. Technically, about 140 million docking runs (equivalent to 413 CPU years), representing an average throughput of almost 80,000 dockings per hour, were performed in 90 days. Both WISDOM projects used mainly the EGEE (Enabling Grid for E-sciencE) grid infrastructure [78]. Other than EGEE, several other grid infrastructures such as: AuverGrid [79], EELA [80], EUChinaGrid [81] and EUMedGrid [82] provided significant contributions to this project.

The BioinfoGRID project [83] promotes the bioinformatics grid applications, mainly in the fields of Genomics, Proteomics, Transcriptomics and Molecular Dynamics, reducing data calculation times by distributing the calculation on thousands of computers using the Grid infrastructure network created by the EGEE Project. Within the framework of this project, molecular dynamics simulations were performed on 5000, 5000 and 15,000 docked conformations of Plasmepsin, Glutathione-S-transferase and *P. falciparum* DHFR, respectively, retrieved from WISDOM. 25 days were taken to simulate 25,000 compounds which otherwise would have taken 347 days on a single CPU. The WISDOM project is a good representation of the advanced biomedical research and its implementation on computational Grids for generation, storage and sharing of huge scientific data located at different research institutions throughout the world for drug discovery against malaria and other neglected diseases.

BRIDGE (Bilateral Research and Industrial Development Enhancing and Integrating GRID Enabled Technologies)[84] is another WISDOM-type grid scenario based on the SIMDAT grid infrastructure for accessing the applicability on the data grids. It aims at performing malaria and birdflu docking experiments using various docking tools such as FlexX, AutoDOCK, DOCK, GasDock and AutoxX. Development of the interoperable grid-based virtual screening application using molecular docking for pharmaceutical R&D is a main focus of BRIDGE workflows.

The OpenMolGRID (Open Computing GRID for Molecular Science and Engineering) project [85] was developed to implement the grid approach for dealing with large-scale drug design, pharmacy, chemistry and engineering related tasks. The basic grid services and functionalities were based on the UNICORE infrastructure. The OpenMolGRID system provided several features such as 1. Creation of data warehouse technology, MOLDW for retrieving chemical compound data from public resources and storage and integration of structural information for other processes; 2. Use of different data mining techniques such as multi-linear regression (MLR), principle component analysis (PCA), partial least squares (PLS), and artificial neural networks (ANN) to develop QSPR/QSAR models and 3. Prediction of biological activities or ADMETox properties. Several programs such as MOLGEO, MOPAC and CODESSA PRO were integrated in this grid system which was used for analyzing millions of structures in short time and screening promising hits.

The SwissBioGrid (SBG) project [86] is a national Swiss Grid infrastructure, collaboration between academia and industrial organizations, developed for life sciences and computational biology research. ProtoGRID, a prototype of meta-scheduling system, was developed initially and then NorduGrid's ARC middleware solution [87] was chosen to build their production grid middleware in the later phases. Performance of the grid was measured by considering docking-based virtual screening experiments to identify potential candidates against dengue virus using Autodock 3.05.

1.2.3.3 Enterprise Grid Computing

High-throughput docking calculations were performed using the Dock 4.01 docking software package for the discovery of human casein kinase II (CK2) inhibitors at Novartis [88]. The corporate enterprise grid constructed was based on the grid computing architecture developed by United Devices (now Univa UD) [48]. Around 400,000 compounds were docked in the binding site of a homology model of the protein to retrieve 12 hits, one out of which was the most potent inhibitor for CK2 ever reported. A very efficient, hierarchical protocol comprising of rigid and semi-flexible docking along with molecular dynamics was presented by Ghemtio et al. [89] for high-throughput structure-based virtual screening utilizing the Grid5000 [90] cluster grid computing infrastructure, which is geographically distributed at nine sites in France, featuring more than 3,200 processors and 5,700 cores. A user-level middleware called a parameter sweep tool (APST) was used for the virtual screening manager for the grid computing (VSM-G) platform to build a multi-cluster environment. The ZINC database comprising of 13 million purchasable compounds was used as the starting point for virtual screening against the 3 β-isoforms of liver X receptors, employing the Omega software for conformation generation, SHEF and GOLD docking programs for rigid and semi-flexible docking, and NAMD for molecular dynamics simulations.

A streamlined version of the AMBER simulation package based MM-PBSA (molecular mechanics with Poisson-Boltzmann surface area) protocol was implemented by Brown and Muchmore [21] of Abbott Laboratories for high-throughput calculations of protein-ligand binding affinities on an enterprise grid utilizing the distributed computing middleware Condor. Bullard et al. [91] from Anadys Pharmaceuticals established an enterprise grid, "Hydra" for utilization of spare cycles from users' desktop computers along with a set of dedicated computers inside the company. A homogeneous environment of Linux operating systems was created by use of coLinux as a service to run Linux in the background of Windows operating systems. Scientific applications such as Omega v2.1, ROCS v2.2, Fred, Szybki v1.4, and GOLD v3.3 were implemented with Sun Grid Engine (SGE) as basis grid technology.

1.3 Aim of the Thesis

The primary scientific goal of this thesis was to develop a university wide applicable grid system for standard structure- and ligand-based drug discovery workflows using freely available academic software and to test its performance for classification of P-gp substrates and non substrates. Several open research questions should be addressed, such as:

- The challenges and accomplishments related to the establishment of a distributed grid infrastructure, UVieCo (University of Vienna Condor pool).
- The efficiency of the grid by performance measurement experiments using sequential and parallel tasks.
- The relationship between ABC transporters and anticancer compounds by NCI (National Cancer Institute) data mining.
- The identification of compounds exhibiting P-gp substrate like properties by using shape-similarity based virtual screening of the NCI database.
- The identification of novel selective serotonin reuptake inhibitor (SSRI) like compounds for searching the mechanism of apoptosis using shape-similarity and unsupervised machine learning based virtual screening.
- The development of predictive classification models for a dataset of P-gp substrates and non-substrates addressing the problem of imbalanced class distribution.

CHAPTER 2

GRID DEVELOPMENT AND UVIECO – UNIVERSITY OF VIENNA CONDOR POOL

In the first chapter, we discussed various aspects of computational grids and their role in the *in silico* drug discovery process. Grid computing has been implemented in modern computational research to fulfill the increasing demands of the computer resources and to achieve better cost efficiency, thus effectively extracting the value from existing desktop computers. It in turn has enabled to tackle the scientific and high throughput computing challenges in terms of their implementation, resource sharing and management across several administrative domains, complex application formulation and development, networking and security issues.

Several alternative systems were proposed to construct large scale computing infrastructure by utilization of different available hardware. One type consists of traditional clusters which refer to the feature-rich, highly efficient aggregation of multiprocessors, servers and clusters which provide efficient and fast interconnection between the nodes. Another type is desktop grids which combine the wasted computing power of the idle PCs from organizations to perform useful tasks. The most suitable applications for desktop grids are those which do not need to communicate among the tasks and can be executed independently (high computation to communication ratio). For the public volunteer projects, such desktop grids are designed to deal with large applications comprising millions of small tasks contrary to the institutional desktop grids, where the dimensionality of the applications is reduced because of the limited resources and only modestly-sized tasks are accomplished successfully. Another advantage of desktop grids is the mutual cost benefits provided by users (in terms of hardware, internet connection, power supply, etc.) and beneficiary (in terms of networking, server and management services) to each other to exchange massive and unaffordable computational power. Thus, desktop grids prove advantageous not only for the high throughput public computing projects, but also to the academia and industries for harnessing otherwise wasted CPU cycles and storage space from their local PCs. Increase in processor strengths and capacities along with improved networking and communication bandwidths of local desktop computers further strengthen the practical implementation of desktop grids and reduce the costs when compared with dedicated clusters and supercomputers.

In this chapter, we provide the basic introduction to grids and high-throughput computing environments. Then the concepts of our university-wide campus/desktop grid, UVieCo will be presented with more details about its architecture, security, firewall transversal and coLinux concepts. In the final section, the applications implemented on UVieCo are discussed along with their performance measurements experiments.

2.1 Introduction

Use of grids for the research purpose and the drastic improvement in its own development has been seen since the past few years. According to Moore's law, computer power doubles almost every 2 years whereas the price and size reduces in the same manner [92]. Now-a-days, size of the transistors on integrated circuits has reached to such an extent that thousands of them can be arranged near to each other, equal to the thickness of the human hair. Such advancements in computing hardware with improved networking technology have enabled access to computing facilities at much faster rate and at much more affordable cost, making implementation of grid infrastructures possible.

The concept of "grid computing" was first coined by Ian Foster and Carl Kesselman in their book 'The Grid: The Blueprint for a New Computing Infrastructure' in 1998, where they defined a computational grid as *"the hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities"* [93]. This definition hints grid computing more likely as the high performance computing (HPC) in supercomputing facilities than collaborative resource sharing among the various organizations. Thus, after 2 years in 2000 the same authors with Steve Tuecke extended this definition to address social and policy issues, stating that grid computing is *"concerned with coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations (VOs)"* where VO is a group of individuals and/or institutions which follow a certain set of resource sharing rules to perform a joint task [29]. They further elucidate that the nature of sharing is not essentially file exchange but more of a direct access to computers, software, data, and other resources such as scientific instruments, sensors, etc. In 2002, Ian Foster proposed a simple checklist to modify the earlier definition of a grid, according to which a grid is *"a system which coordinates resources that are not subject to centralized control using standard, open, general purpose protocols and interfaces to deliver non-trivial qualities of service"* [94]. The coordinated resources represent the computing infrastructure with identical or different administrative domains addressing security, sharing policy, costs and membership issues following the well set protocols to handle the fundamental aspects of grid computing such as authentication, authorization, locating and accessing available resources. The grid grants different qualities of service (QoS) associated with security, availability, throughput, response time and resource co-allocation to fulfill complex user demands, thus rendering a scenario where utility of a combined grid system is significantly greater than the sum of all the resources when used separately.

The above definition was perceived as too restrictive or too vague by several scientists and proposed different and less technical definitions of grid computing [95-98]. For instance, a general definition is given by Wolfgang Gentzsch: *"A grid is a hardware and software infrastructure that provides dependable, consistent, and pervasive access to resources to enable sharing of computational resources, utility computing, autonomic computing, collaboration among virtual organizations and distributed data processing, among others"* [96]. On the other hand, Andrew Grimshaw proposes that *"a grid enables users to collaborate securely by sharing processing, applications, work flows and processes, and data across heterogeneous systems and administrative domains for collaboration, faster application execution, and easier access to data"* [97]. In a more general way, we can say that grids are fundamentally heterogeneous, loosely coupled and dispersed systems, but can be viewed as a unique virtual environment by the user, providing uniform and remote access to resources and aggregating processing power. Although processors are the most apparent and basic components of the grid, it also encompasses middleware, networking devices, supercomputers, graphics processing units (GPUs), data-storage systems, applications, databases, software and scientific instruments owned and managed by various institutions.

The original idea of a computational grid is analogous to electrical power grids. Several prime-energy resources are used to generate electricity which includes hydro, thermal, wind, solar, coal, gas, oil, nuclear energy and many others. An electric power grid is responsible for its pre-processing, storage, transmission and distribution so that it would be available easily to the end-users who aren't aware of its complexity and reliability. Similarly, computational grids aggregate a variety of computational resources such as clusters, supercomputers, desktop PCs and workstations along with scientific databases, instruments, devices for storage and visualization. The users would not even know which computer performs the computations in the same way that that users of power systems are not aware about the part providing them the electricity they use [99]. The goal of such heterogeneous grid technologies is to provide a transparent secure access to high-end resources irrespective of th location of their physical existence and the location of access. Although these two terminologies are equivalent to each other in many aspects, computational grids are still in their infancy and need a more sophisticated operational model, organized systems transmission to ensure network stability, global resource sharing policies, universalizing the resource properties and ease of use [100].

Although different researchers and groups use grids for various purposes, the main objectives of grid computing can be summarized as follows [101]:

- Provision of consistent load-balancing by utilization of unused computational resources in terms of CPU cycles and storage capacity to employ existing application on different machines.
- Exploitation of multiple processors for CPU intensive grid application using parallel applications via networking.
- Reliable and uniform, inter- and intra- institutional access to remote resources for enhancing collaborations and research opportunities.
- Virtualization of the existing distributed computing resources to enable participation of heterogeneous systems to form a larger grid having individualized policies and priorities for both resources and users.
- Resource balancing in terms of sharing special equipments, software, licenses, and other services. It also includes temporary suspension or even cancellation of low priority tasks to execute the higher priority tasks when the grid is fully used.

2.1.1 Grid Components

While the individual grid projects differ in their architecture, a general collection of components necessary for constructing any grid remains the same and individual projects implement their own version of components accordingly. It ensures the interoperability, code sharing and portability among the participants of virtual organization to distribute heterogeneous resources in dynamic manner. As shown in Figure 2.1, the components required to form basic grid architecture are arranged in distinct layers having their own functionality [102].

The *fabric layer* consists of heterogeneous, geographically distributed resources such as multi-processor computers, physical storage systems and their local resource management systems. Computational power is generated by individual desktops or workstations running a variety of operating systems, or clusters having their own operating systems or resource management systems such as LSF, Condor or PBS. This layer also includes data storage facilities such as databases or file systems as well as different software and scientific instruments such as x-ray crystallography beamer or radio telescope. A grid middleware is required for unhindered interactions within them as all these resources are controlled by different local resource operators with variety of monitoring mechanisms.

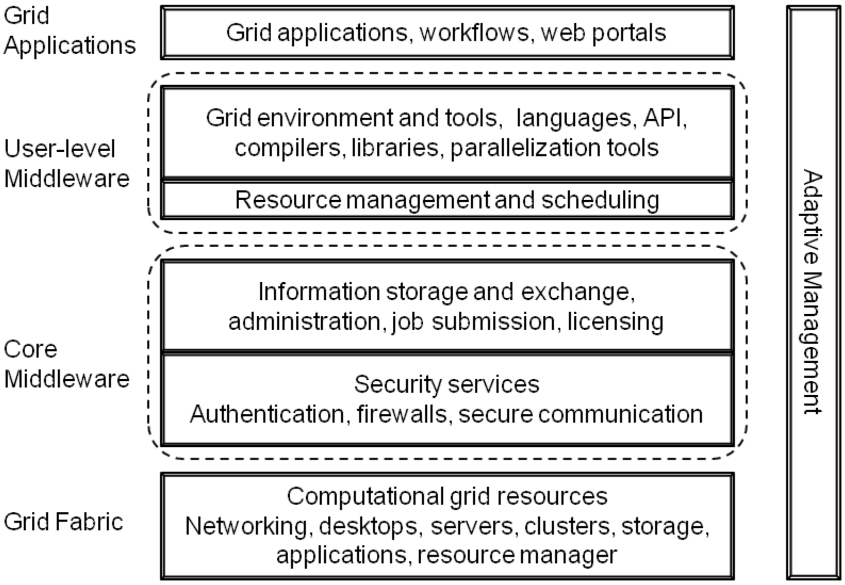

Grid Applications
Grid applications, workflows, web portals
User-level Middleware
Grid environment and tools, languages, API, compilers, libraries, parallelization tools
Resource management and scheduling
Core Middleware
Information storage and exchange, administration, job submission, licensing
Security services Authentication, firewalls, secure communication
Grid Fabric
Computational grid resources Networking, desktops, servers, clusters, storage, applications, resource manager
Adaptive Management

Figure 2.1: The components required to construct a Grid, split into layers containing Applications, Grid Tools, Grid Middleware and Fabric. Modified from [102].

The *core middleware layer* renders important communication and security protocols to grid infrastructure for abstracting complexity and the heterogeneity of the fabric layer. The information services help to overcome the dynamic nature of the grid, i.e. they provide means to register and retrieve information about grid resources, services and their status in a more sophisticated way as their location and availability are constantly changing. Both resource providers and users benefit from resource trading which assists them to formulate strategies, while in turn maximizing their objectives. Security services also play a crucial role in dealing authorization, authentication, privacy, integrity and accountability issues to access cross-boundary, autonomously administered diverse systems.

The *user-level middleware layer* imparts programming frameworks and tools for different types of applications, and resource brokers to chose specific required resources for particular programs. These high level services support standard programming languages, a variety of programming paradigms (e.g. message parsing interfaces (MPIs) or distributed shared memory), compilers, parallelization tools and libraries that allow applications to be developed which can run on the grid. Resource management and scheduling should be user-friendly, self-explanatory and provide opportunity to utilize and manage processor time, memory, network, storage and other resources effectively and proficiently using middlewares.

The *applications layer* is the final and important component of grid architecture. It is developed using grid-enabled programming environments and interfaces that lets end-users to utilize grid services to solve large scale scientific and engineering problems. These grid applications should be easy-to-use for naïve users and should also run on several platforms and operating systems. It also includes software and tools to support application workflow and composition. Web portals are thus emerging as excellent solutions in providing unconditional remote access to the resources and databases to users for their application execution.

This is a simple architectural model of the grid system. A large number of components are required to construct the grid and there exist many complex interactions among them. An agreed set of standards needs to be adopted for developing grids so as to achieve interoperability and acceptance from a wide user community.

2.1.2 Types of Grids

Grids are categorized into various types based on different criteria perceived by computer scientists, research analysts and IT vendors. These are classified based on the types of resources, services, geographic locations, functionality, architecture and built-in-components. Depending on the geographical dispersions of the computers, grids are classified into three categories such as cluster grids, campus grids and global grids [101, 103].

Cluster Grids – These are the localized grids, for example within a department or a research group, possessed and operated by a small number of people. Cluster grids consist of several systems functioning together to allow a single point of access to users. Both high-throughput and high-performance jobs are supported where resources can focus particularly on a smaller set of repetitive tasks, or complex jobs can be executed in a parallel manner.

Campus Grids – These are the localized grids within a department or a research group (e.g. Pharmacoinformatics research group) possessed and operated by a small number of people. Cluster grids consist of several systems functioning together to allow a single point of access to users. Both high-throughput and high-performance jobs are supported where resources can focus particularly on a smaller set of repetitive tasks, or complex jobs can be executed in a parallel manner.

Campus Grids – Campus grids are formed when cluster grids within different departments of an organization or university are merged together to fulfill the demands of larger computation. For example, UVieCo Condor pool of University of Vienna as shown in figure 2.2. Various computing resources along with networking solutions, middleware, software and utilities are shared in a co-operative way within a particular domain to handle multiple tasks and projects.

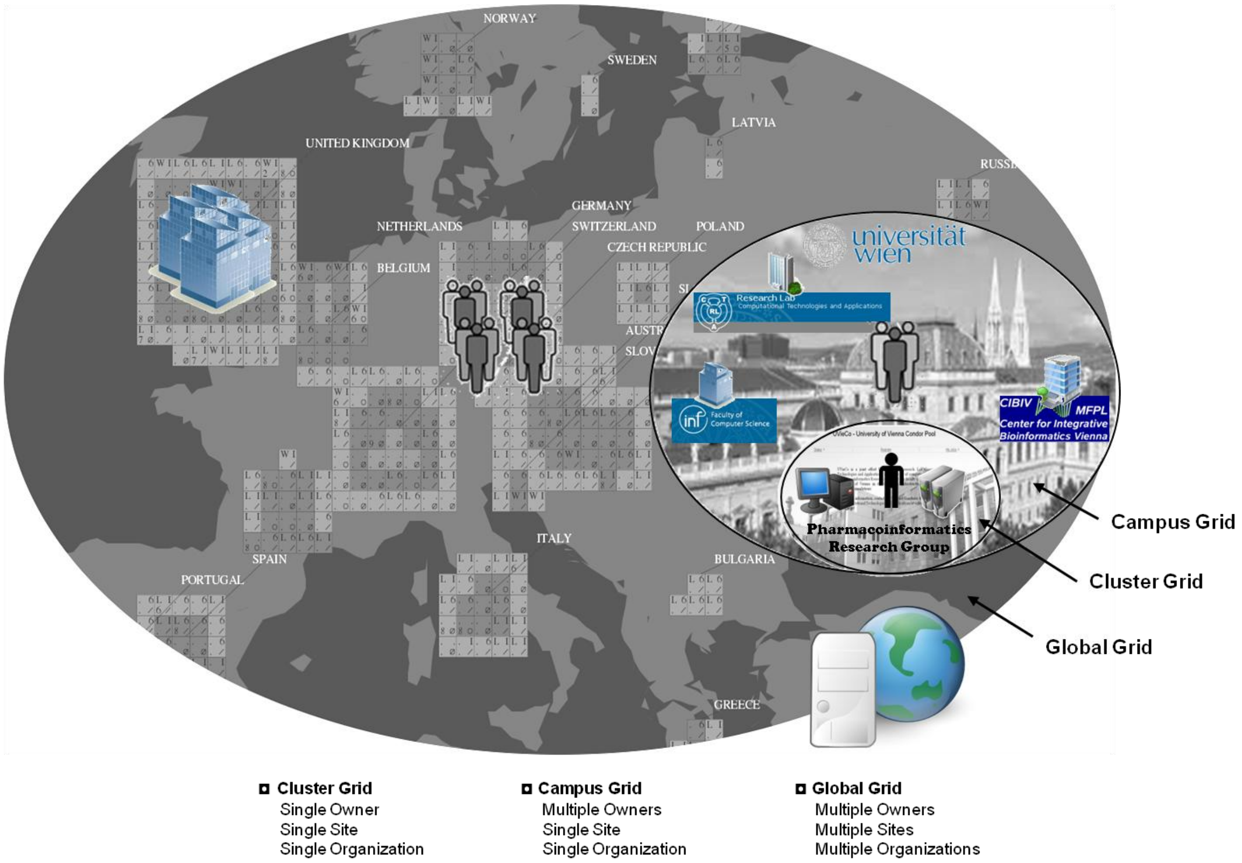


Figure 2.2 : Types of grids based on geographical distribution. Modified from [103]

The three classes of grids are represented in Figure 2.2. In the cluster grid, small groups or organizations own their cluster where user's tasks are performed at a particular site. On the broad spectrum, several such cluster grids come together forming more complex campus grids to perform bigger tasks. Both of them can be a part of a global grid to extend their network worldwide, where jobs can be executed on multiple sites.

With a widespread demand of computational power, most of the research institutions are using shared computing grids which have quite different characteristics than those of traditional supercomputers, clusters, or commercial clouds. The wasted CPU cycles from the idle computers, e.g. when the users are away during the breaks or in the evenings, can be harnessed for doing some fruitful calculations. In other words, campus grid computing is emerging as an innovative strategy to make use of personal computers and disparate clusters, connected by different middlewares for massive amount of calculations.

2.2 UVieCo and High Throughput Computing using Condor

Large numbers of scientific and data intensive calculations require weeks or even months of computation to solve the problem. Establishment of the computational environment to fulfill the massive requirement of computational power is desired to tackle this situation. Concurrent execution of the scientific tasks with available resources serves a faster mean of computing than a single server. This led to emergence of new terminology called as 'high throughput computing (HTC)', where a variety of computing resources are used (in serial as well as in parallel mode) for a longer period of time to complete compute-intensive tasks [104, 105]. The main advantage of HTC is that the costs are kept very low as new computers need not to be acquired and the processing is performed when the available computers would normally be idle.

We developed a university-wide campus grid, UVieCo (University of Vienna Condor pool), composed of the resources from the 'Research Lab Computational Technologies and Applications' of the Faculty of Computer Science and the 'Pharmacoinformatics Research Group' of the Faculty of Life Sciences to develop an infrastructure for large scale distributed computing. The basic idea originated after analyzing the scenario where many computational resources within the university are underutilized by the computer owners. By exploiting the wasted CPU cycles when these computers would normally be idle, high throughput computing can be accomplished while maintaining the very low costs without much investment on the hardware. It is referred to as a Condor pool because several life sciences related applications are executed on different machines within the university network using a public domain middleware, Condor [41], which acts as an interface to the grid. It renders different computation and data storage resources to the university-wide user, thus they can accomplish compute intensive tasks over the high throughput network.

With the implementation of a graphical user interface (GUI) and easy registration policies, this system promotes the users to become a part of our Condor pool. Self extracting executables including Condor and other necessary auxiliary packages were prepared for Linux and Windows platforms which can be downloaded from the portal. These single click installable packages were prepared in such a way that the user machine becomes a part of the pool immediately after its installation and in further runs, as soon as the machine is turned on. The co-operative virtual machine concept (CoLinux) was implemented in a way that Linux-executable jobs can be run inside Windows. Firewall and security issues have been resolved using a virtual private networks setup to guarantee the accessibility of the participating machines among various departments.

In this section, first we briefly describe the basic components of Condor. The detailed information about Condor can also be found in the manual [106]. Then the architecture of the UVieCo and issues related to the firewall transversal, openVPN gateway and CoLinux concept are presented. The applications implemented on the grid are subsequently elaborated along with concluding remarks and future directions.

2.2.1 Condor

Condor is a widely used high throughput grid computing software which was developed with an aim to take full advantage of the available computing resources within the organizational network through distributed ownership [41]. It is a distributed batch system providing some interesting features such as job management mechanisms, queuing and scheduling policies, priority schemes, resource monitoring and resource management [107]. The Condor system comprises a pool of idle workstations, servers, clusters, with each processor of them working as a compute node for submission and execution of the computation jobs. In this way, it performs computer intensive calculations within short times by efficiently exploiting wasted computational power from idle machines. It creates the workflows and helps to organize the scientific tasks in such a manner that huge amounts of computation are performed without much intervention from users. It assists in submitting a large number of jobs based on the policies specified by the users, tracks their progress, gives error notification if there is any and finally intimates the users after completion and returns the results [108]. The main advantage of Condor is that it does not require modifying the source code of the program to be implemented into the Condor system. Several interesting features such as checkpointing and remote system calls are provided by Condor in relation to re-link the codes with its libraries. These aspects along with several important functionalities will be discussed in the following section.

2.2.1.1 Condor Components and Daemons

Condor facilitates the users to run batch jobs on a centrally organized network of computers. This network bears the responsibility of the jobs, searches and schedules the resources, transfers the necessary files and performs rescheduling if the jobs are stopped abruptly on certain machines. The coordinated groups of schedulers and execute machines with condor installations are called as "condor pools" (Figure 2.3) [41].

The central manager (CM), considered as the 'heart' of a Condor pool is a single important machine. It acts as the centralized repository of information which periodically assesses the current status of the pool and attempts to complete the pending tasks by rescheduling them to appropriate resources. As this machine performs the matchmaking process (discussed in next section) in the Condor system, it should be chosen in a way that it should be reliable and likely to be online all the time or at least will be quickly rebooted. Also, in an ideal scenario, it should have a good network connection to the rest of the machines in the pool to achieve better communication. It collects the periodic updates and information from an arbitrary number of other machines joining the pool and negotiates between resources (machines) and resource requests (jobs). These two roles are performed by separate daemons called *condor_collector* and *condor_negotiator* and are considered as the central components/core of the Condor pool.

Besides the CM, there are two additional machine roles that are important for using the Condor system, submit/scheduler and execute roles. The 'submit machine' allows submission of the Condor jobs. It requires large swap space and/or real memory as it generates another process and retains the memory image of the job submitted to the remote execute machine. Additionally, the checkpoint files and the binaries of the submitted jobs are also stored on the local disk of the submit machine. On the other hand, 'execute machines' are configured to execute the Condor jobs. They have a simple configuration and don't need many resources. The Condor pool will not be functional without their service and the more resources (CPU speed, real memory, swap space, etc.) these machines have, the more efficiently the resource requests will be served by them.

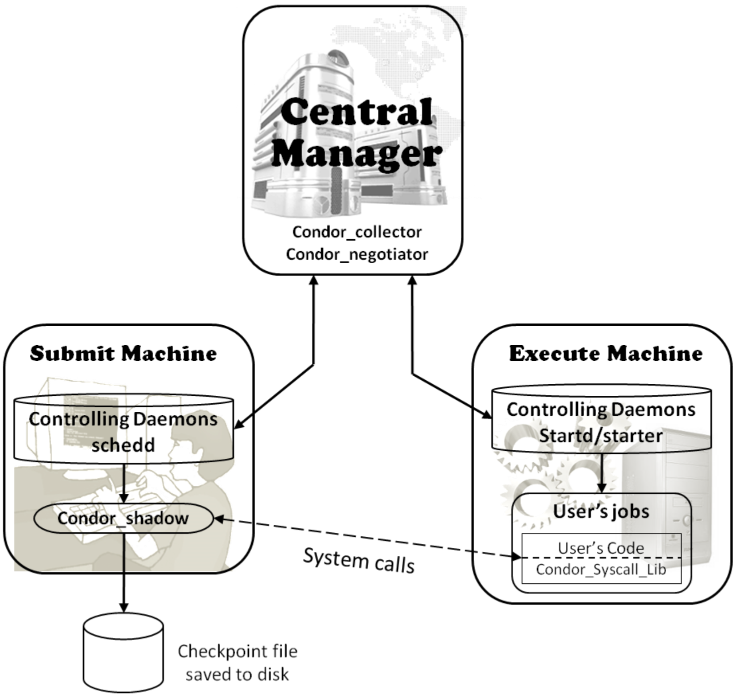


Figure 2.3 Overview of Condor Architecture

The central manager collects the periodic updates and information from an arbitrary number of other machines joining the pool and negotiates between resources (machines) and resource requests (jobs). These two roles are performed by separate daemons called *condor_collector* and *condor_negotiator* respectively and are considered as the central components/core of the Condor pool. The collector daemon monitors the status of all the rest of the machines within a Condor pool in the form of classads. Thus, the users as well as all other daemons can query the collector to obtain the specific information about the different parts of the Condor pool. The negotiator daemon is responsible for match-making process which is initiated by a negotiation cycle where the status of a pool is determined by querying collector, each schedd having awaited resource requests are contacted in priority order and finally meeting the requests with the available resources.

The 'submit' and 'execute' machines are comprised of the *condor_schedd* and *condor_startd* daemons respectively. The schedd daemon represents the resource requests. It allows the user to submit the jobs (using *condor_submit* command) and manages them by viewing (*condor_q*), monitoring (*condor_status*) and manipulating (*condor_rm*) the job queue. It publicizes the information about the jobs in a queue and seizes the available resources to accomplish those tasks. Once the matching resource is found for the particular task, schedd gives rise to *condor_shadow* daemon which acts as a resource manager and allows the transfer of a given request to the remote execute machine. The startd daemon on the other hand corresponds to the resource offerings and advertises the properties of the resources that might prove useful to meet the awaiting resource requests. It comprehends the resource owner's policies and controls the circumstances to start, suspend, resume, vacate or kill the jobs on any remote execute machine. When it is ready to execute a job, it commences a child process in the form of *condor_starter* daemon which actually runs the job on a remote machine. It configures the requisite execution environment, monitors the status of the job and returns the information to the submit machines after the completion.

2.2.1.2 Prominent Characteristics of Condor

ClassAds

A classad (classified advertisements) is a set of unique expressions comprising of specific characteristics and requirements of the machines using Condor to specify the policies required for matching the resource offerings (e.g. machines) or resource requests (e.g. jobs) [109]. This concept was originated from the brief advertisements in the morning newspapers where sellers advertise the details about their products to attract the customers whereas buyers may also publish the features about their proposed necessities. In this way, the constraints from both buyers and sellers are listed and satisfied [108-110].

Classad plays an important role in Condor's matchmaking process where it acts as a schema-free resource allocation mechanism to fulfill requirements from the users about job requirements and job desires. For example, a user would want his job to be run on a machine with at least 128Mbytes of RAM with Linux operating system and specified CPU type and speed or virtual memory size, whereas workstation owner can state his preferences about running jobs from particular department or a set of users. Such job requirements/preferences along with resource availability are described as powerful expressions using classads which help in adapting any form of working policy. They are used for scheduling the jobs, maintained in log files for future statistical and debugging purposes as well as for knowing the current status of the Condor pool. The simple examples of classads describing a submitted job and that of a machine are as follows (Figure 2.4):

Job Classad

```
[
MyType = "Job"
TargetType = "Machine"
Requirements = ((other.Arch =="INTEL"
&& other.OpSys =="LINUX")
&& other.Disk >="10000")
Owner = "griduser"
NumUsers = 0
MaxJobsRunning = 200
StartLocalUniverse = TRUE
TotalIdleJobs = 0
TotalRunningJobs = 0
]
```

Machine Classad

```
[
MyType = "Machine"
TargetType = "Job"
Activity = "Idle"
LoadAvg <= 0.034252
KeyboardIdle > 900 //seconds
Disk = 332542 //Kbytes
State = "Unclaimed"
Arch = "INTEL"
OpSys = "LINUX"
Name = "10-8-1-13.unigrid"
Rank = other. Deptt==self. Deptt
]
```

Figure 2.4: Examples of job classad and machine classad from Condor

Signed Classads

Security with the classads sometimes becomes a prime issue as these are sent in plain text by default and thus may be read and modified by any intruder. The altered policies in the classads which are provided to the central manager from the machines can cause rejection of the services, illegal access of the resources for their own benefits or misuse of the results before returning to the submitter. Condor provides an excellent solution for this problem in the form of "signed classads", where digital signatures are created for executables, arguments, and input data by the submit machine prior contacting to the central manager [111]. The signatures are verified by the central manager, the job is executed with other machine and results are returned to the submitter who checks the signatures on the results and confirms whether the central manager has been consistent with the policy. The signatures are made and checked using X.509 keys and certificates and any alteration in them can be easily detected. Signed classads can play a role in authentication and authorization and keep away intruders from intercepting Condor's communication and harmful modifications in classads.

Job Checkpoint and Migration

The checkpointing is a process in which the information of the program's current state is saved and can be reused for resuming its execution on another machine. This is very much important for long-lasting jobs which take weeks or months for their completion. If the machine running a Condor job becomes unavailable, a checkpoint is created to preserve the already completed computation. This is then used for continuation of the job on other machine after migration, thus enabling Condor's preemptive-resume scheduling policy. Periodic checkpointing is essential when the jobs are executing on machines which become unavailable after every short time. It provides an efficient mechanism of fault tolerance and act as a precautionary measure to save the accumulated computation time in case of system failure such as shutdown or crash of executing machine [108, 112, 113].

Remote System Calls

When a job is submitted to a Condor pool, it is often executed on the remote machine. Despite this external demand, the security and internal local execution environment of the remote machine is preserved by Condor's mechanism called 'remote system calls'. It helps to bypass the needless requirements such as transferring data files or creating a login account on the remote machine before execution of user's programs. The job related input/output files are maintained on the submit machine, thus avoiding their imposition on remote machine [113].

User's Source Code Unchanged

Prior programming knowledge is not necessary for using Condor. Condor can run non-interactive programs efficiently with above mentioned transparent and automatic functionalities. User only needs to re-link the programs with Condor libraries without its modification or recompiling.

Flocking of Condor Pools

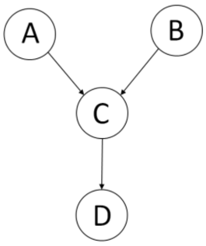
Several Condor pools located at different places can be hooked together which allows submission of a job in one pool and its execution in another. This mechanism is very much flexible and after submitting the jobs, another subset of machines within second pool can set the policies stating which jobs to be executed and which users to allow doing the same.

Priority to Machine Owners

Condor gives prime importance to the machine owner and he has complete priority for using his own resources. The owner allows others to use his machine when it is idle and as soon as he returns, Condor automatically facilitate to regain the control back without any special efforts.

Ordering the Dependencies

Condor considers the dependencies among the jobs and facilitates their execution in sequential manner. DAGMan (Directed Acyclic Graph Manager), a meta-scheduler is used to specify the set of inter-dependent executions of one or more jobs [114]. Here, each node in this graph represents individual job and edges recognizes the dependencies. It presents the programs to Condor in order depicted in DAG and processes the results. Prior to submission of a job, an input file describing DAG is specified alongwith a Condor submit description file for each sub-job in DAF file. A simple DAG containing 4 nodes and its corresponding DAG input file is presented in Figure 2.5.



```
# File name: dagman.dag
JOB A A.condor
JOB B B.condor
JOB C C.condor
JOB D D.condor
PARENT A B CHILD C
PARENT C CHILD D
```

Figure 2.5: Dagman DAG dependancies

DAGMan monitors the log files after submitting the jobs to execute programs in order as stated in DAG. It is also responsible for scheduling of tasks, recovery of the results and reporting about the set of programs presented to Condor. We used this scheduling mechanism for sequential execution of DOCK sub-programs as presented in section 2.2.6.1.

2.2.2 UVieCo Architecture

The very basic architecture of UVieCo is shown in Figure 2.6. This example shows how the resources between the faculties of computer science and life sciences are shared in a single Condor pool. The various components are shown in Figure 2.6, namely the connection manager, the execute machines and the network and firewall setup. Administration and implementation information is provided as appendices 1 and 2.

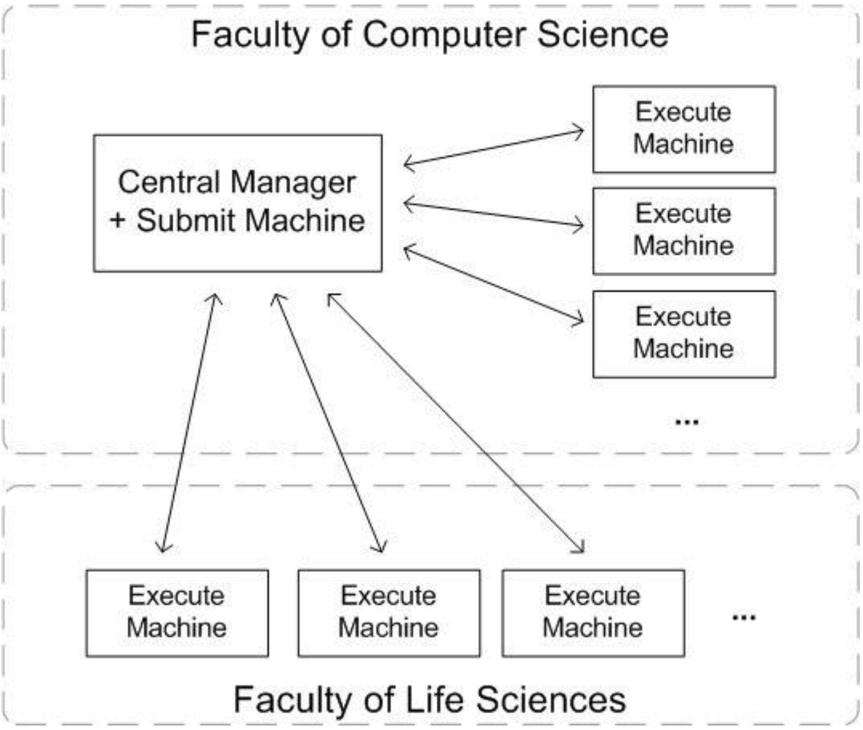


Figure 2.6: Basic Architecture of UVieCo

Central Manager

The central manager is intended to actually have three additional roles apart from being the condor central manager (Condor 7.0.1) [41]: a web interface (Apache Tomcat 6.0) [115], a database server (PostgreSQL 7.4) [116] and a VPN gateway (OpenVPN 2.0.9) [117]. Figure 2.6 shows how these applications work together. This machine is currently running Suse Linux Enterprise Server 10 and owns four dual core AMD Opteron processors with 2GHz each and 24GB of RAM.

Execute Machines

As can be anticipated from Figure 2.7, the execute machines are a collection of heterogeneous computers which can be roughly divided into two categories. 1. Dedicated machines which are intended to be used for the campus grid only or at least to a great extent. Most of them will be running on Linux, while all others run on Windows. 2. Workstations, which on the other hand are pre-existent machines intended for lab work which will be utilized in times of low usage by the owner. Concerning the condor software, an execute machine will be running the startd daemon which will be configured to only accept jobs from the central manager's negotiator and schedd daemons.

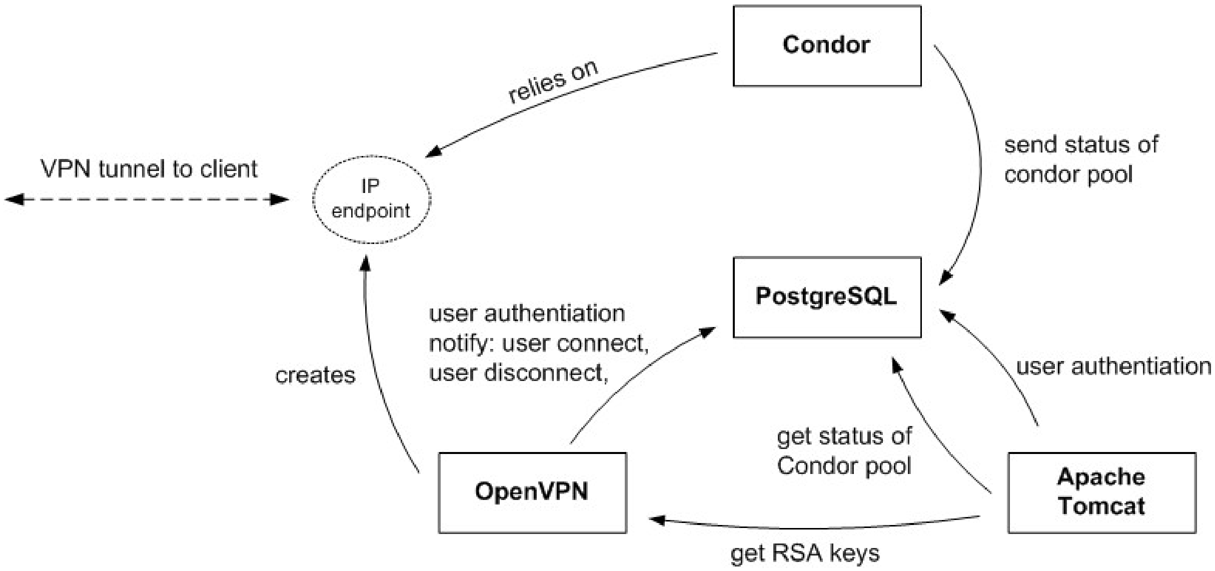


Figure 2.7: Collaboration between central manager components

2.2.3 Firewall Transversal

One of the biggest problems for a condor deployment is to guarantee the accessibility of the execute machines behind firewalls. Usually, the potential client machines on the campus do not possess a global IP address and are hidden behind network address translation (NAT) boxes. Several solutions have been mentioned in the related work section.

2.2.3.1 Firewall within Condor

It has been stated that initially condor was designed to run in a network environment which is both 'symmetric' (i.e. one in which any machine can initiate a connection to any other machine), and in which there are no restrictions on types of network traffic (e.g. firewalls blocking UDP). Now-a-days, in the modern computing environment such an 'open' network environment is increasingly rare. It is thus the case that it can be quite difficult to deploy condor in many current network environments due to the presence of firewalls, private networks (i.e. networks of machines with IP addresses in a specified range) and other circumstances that break the symmetry of the network [118]. Firewall issues related to condor system have been intensively studied by O'Donnell [119], Son et. al [120, 121], Lodygensky et al. [122], Beckles et al. [118], Calleja et al. [123] and Scherp et al. [124].

There are currently two main mechanisms for dealing with firewalls within Condor [125]: 1. Restrict condor to use a specific range of port numbers, and allow connections through the firewall that use any port within the range, and 2. Use Condor Connection Brokering (CCB) or Generic Connection Brokering (GCB). Condor Connection Brokering or CCB allows condor components to communicate with each other when one side is in a private network or behind a firewall. Currently, the functionality of GCB is being replaced by CCB because GCB provides communication between two different private networks whereas CCB only supports communication between nodes with one-directional connectivity. The main reasons why CCB is preferable are: support for all platforms (including Windows), easier configuration and troubleshooting, and ability to restart and reconfigure on the fly. Generic Connection Brokering, or GCB, is a system for managing network connections across private network and firewall boundaries. Although GCB provides numerous advantages over restricting condor to use a range of ports which are then opened on the firewall, it has to be noted that it's also a very complicated system, with major implications for condor's networking and security functionality.

O'Donnell addressed the problem of condor's lack of ability to function through a firewall for the first time in the Wisconsin Computer Science network (cs.wisc.edu) [119]. He considered two approaches: writing a custom proxy server and using an existing standard proxy system, considering the fact that the best solution should not negate the original purpose of a firewall, i.e. security. He finally came up with the SOCKS proxy system which was equipped with security, auditing, management, fault tolerance, and alarm notification at that time.

Son and Livny [120] observed that in grid computing, the pools of hundreds or thousands machines are not necessarily having world-addressable IP addresses and the administrators of those pools would prefer private network configuration as it helped them to manage their clusters easily and also reduced the cost by paying for only several public IP addresses for head nodes instead of hundreds or thousands ones. According to them, these private networks and firewalls damaged internet connectivity, making it asymmetric and difficult or even impossible for peer-to-peer computing. They then correlated this problem with the condor system and came up with two different approaches, DPF (Dynamic Port Forwarding) and GCB (Generic Connection Brokering) which have different characteristics in terms of clusters supported, security, and performance and suggested that the users should choose the better one depending on their policies and situations.

In an another approach, Son et al. [121] presented a middleware firewall traversal system called CODO (Cooperative On-Demand Opening), which provides applications end-to-end connectivity over firewalls/NATs in a secure way along with allowing applications authorized through strong security mechanisms to traverse firewalls/NATs so that authorized applications can communicate through it , while blocking unauthorized applications.

Lodygensky et al. came up with a lightweight grid solution for the deployment of multi-parameters applications by using XtremWeb coordinator to solve problems related to domain administrations and firewalls when connecting different condor pools [122].They demonstrated the usefulness of this approach measuring the performances of a multi-parameters bio-chemistry application deployed on two sites: University of Wisconsin/Madison and Paris South University.

Beckles et al. [118] raised several issue related to condor's pattern of network communication such as machine's role, direction of network communication, network protocols and port usage, administrative overhead, private firewalls, inadequate documentation, unresolved bugs relating to network communication, etc. Then explaining why these are unfriendly to the firewalls and private networks and finally coming up with the solutions / techniques which have been developed to address or mitigate these problems. These solutions include - mitigating the effects of firewalls, altering the pattern of network communication e.g. reducing it to 'few-to-many 'or even to 'one-to-many', firewall/NAT traversal, i.e. traversing the security boundary along with generic connection brokering (GCB) and dynamic port forwarding (DPF).

A grid infrastructure, WISENT has been created using Globus Toolkit and Condor by Scherp et al. [124] to handle large heterogeneous datasets generated in energy meteorology research. Because of the 6 different locations of the partner sites, the construction of the grid infrastructure is hindered by blocking firewalls due to strong firewall policies and the use of network address translation (NAT). To tackle with this problem, after considering possible solutions such as an extra grid-zone, a tunnel via virtual private network (VPN) established with each external project partner and application level gateway (ALG), they came with an approach of using a connection broker which can be used for hole punching to traverse firewalls and NAT systems.

Similarly, Calleja et al. [123] implemented an experimental solution by constructing a dedicated Virtual Private Network (VPN) and flocking the small condor pools across this VPN. They further claimed that VPN enabled to tunnel through departmental firewalls and encrypted traffic across interdepartmental links, allowing nodes with private IP addresses which could join a grid that crosses institutional boundaries. Jobs migrated seamlessly across the flocked Condor pools and there was no noticeable degradation in performance due to the overhead of running across the VPN.

2.2.3.2 Firewall Solution in UVieCo

For these issues, we settled for a different solution, the openVPN package. Calleja et. al [123] have similar implementation of the condor setup as we have in our pool. Their setup differs from ours as they used the VPN to flock their Condor pools whereas we intended to use it to connect the execute machines to the central manager. OpenVPN supports a wide variety of operating systems including Windows 2000/XP/Vista/7, Linux and Mac OS X. Considering a more detailed look at the connection between the central manager and the execute machines, the setup actually looks like it is shown in Figure 2.8.

2.2.4 Security

The detailed description of security implementation in Condor can be found at [126]. We describe here the security solutions implemented with UVieCo.

VPN Tunnel

The VPN software is configured to lease private IP addresses to all authorized clients. Authorization is done via client certificates. The establishment of the VPN tunnel is initiated via UDP from the client-side. The firewall protecting the execute machine therefore has to allow outgoing UDP connections. A brief overview of the execute machine startup is shown in Figure 2.9.

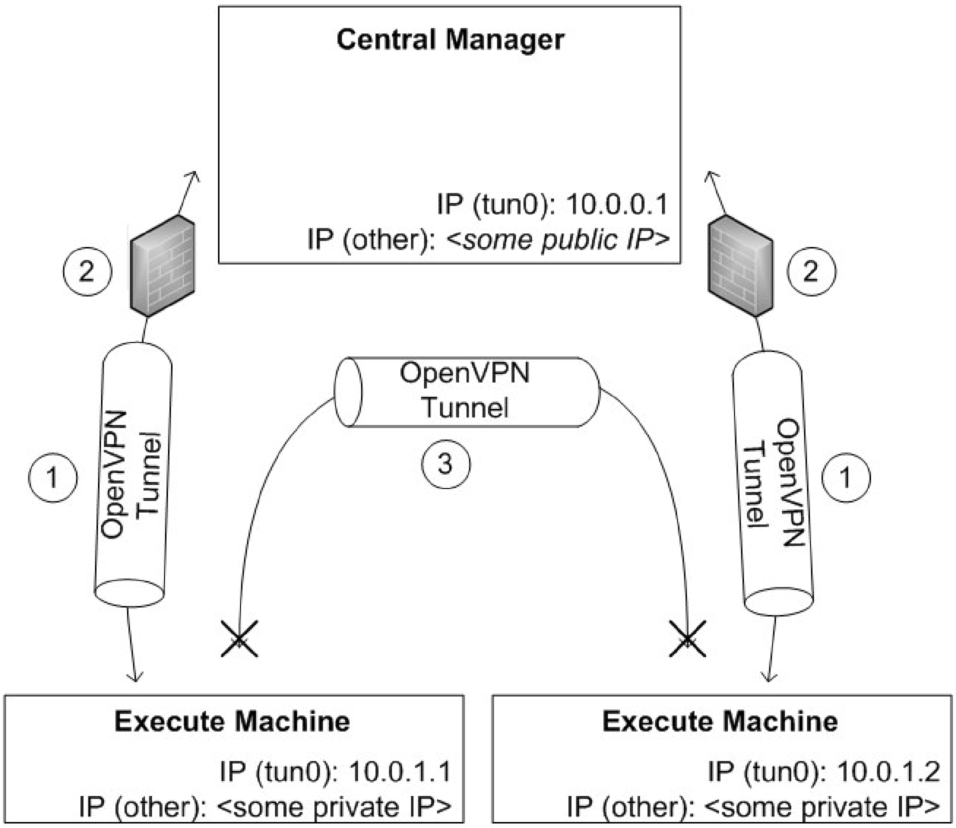


Figure 2.8: Network Setup: 1. VPN tunnel traversing firewalls; 2. Central manager's firewall; 3. VPN client-to-client connections are not possible

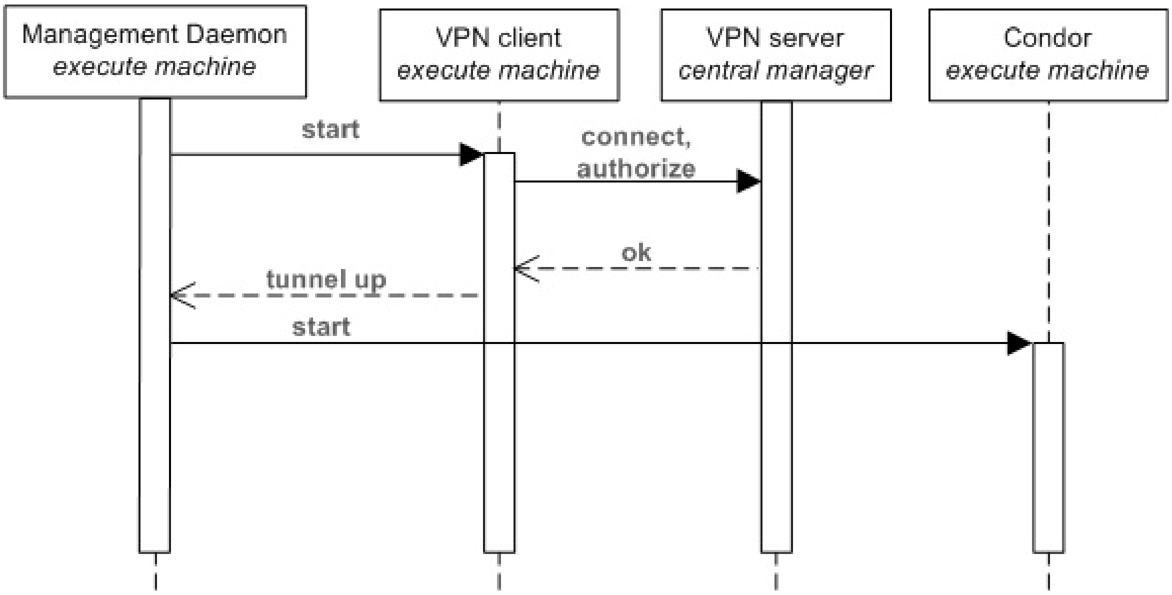


Figure 2.9: Execute machine startup sequence

Central manager's firewall

The central manager is a central point in this network setup as it is able to communicate with all the connected clients through firewalls which were put in place to protect them. We therefore intend to restrict incoming as well as outgoing IP traffic on the central manager to the ports or port ranges used by condor. This has been done via the Linux tool "iptables".

VPN Client-to-client connections

As an additional security measure, the VPN server will be configured to disallow communication between the VPN clients (the execute machines).

The openVPN software offers to fine-tune a wide range of parameters for the tunnel. We have set the most important ones so far, as outlined in the following list.

1. Authentication is done by setting up a public key infrastructure (PKI). The openVPN server as well as each client owns a private and a public key (or "certificate"). Additionally there is a Certificate Authority (CA) public/private key pair which is used to sign the client and server certificates. OpenVPN will be configured to require each client to present a valid CA-signed certificate and allow only one client to connect with the same certificate which may also be revoked manually. Concerning the cipher used to encrypt the condor traffic, the default value "Blowfish" is considered to be enough for our uses. Although there are some recommendations against the use of this cipher, the payload in our case is not precious enough to research more secure possibilities of encryption.

2. IP addresses - Each client connecting to the VPN gateway receives an IP address for its virtual device. In order to prevent conflicts due to the VPN subnet overlapping with the client's other networks we intend to use an unpopular address space such as 10.8.1.0

3. Server port - In order to expose as few potential targets as possible to the outside world, the UDP server port which is responsible for accepting VPN clients will be set to a random number far away from the default value. The same is intended for the SSH port, which, together with the openVPN server port will be the only two ports available to the public world.

As can be seen in the list above, setting up a basic openVPN server is fairly simple and a matter of adjusting a few configuration parameters.

2.2.5 Virtualization and CoLinux Package

Enormous advancements have also been observed in the field of virtualization of computer hardware which allows users to run multiple virtual machines in a single computer, each of which acts as a separate section with a unique instance of an operating system. Several open-source [127-130] as well as commercial [131, 132] packages are available for virtualization.

Use of this concept has also already been demonstrated for establishing condor pools where the implementation of linux-based condor has been carried out in windows computers using coLinux, utilizing different methodologies by several groups pioneered by Sumanth at the University of Nebraska Lincoln [133] and adopted later by Neeman et.al. [134], Severini et al. [135], Santosa et al. [136] and Alexander et al. [137]. For networking, they used the winPcap tool [133, 137], which requires real IP addresses of the machines, resulting in having the same local IP for all coLinux installs. So the suggestion of changing mac-address of the new TAP-Win32 device has been proposed to have real host's IP. Another observation is that these pools have only windows machines as a part of the grid whereas we plan to have a grid comprising of different machines with linux as well as windows operating systems. There are several similar approaches emerging, such as making use of virtual machines as an attractive tool for building campus grids, Pools of Virtual Boxes (POVB) [138], or exploring Virtual Workspace Concepts [139] or the use of virtual network, ViNe [140].

The coLinux installer basically contains a Debian 4.0 root image with all the necessary auxiliary packages and the client .deb package installed. The scripts are therefore essentially the same. One difference, however, when running under windows is for condor to determine if the machine is idle or busy. Under a native Linux, condor may access the CPU status and the keyboard and mouse activity. When running inside coLinux, however, the virtual machine has no access to the windows keyboard and mouse usage. The same applies to the CPU. This problem was solved with a small VBScript running as a windows service ("colinux_monitor") which reports the current performance data to a directory shared by windows and the coLinux virtual machine. Inside coLinux, the script unigrid_monitor.sh, which is periodically called by condor, reads out the windows performance data and acts accordingly.

2.2.6 Applications and Performance Efficiency Measurement

Several molecular modeling software packages have been running on the grid environment using different methodologies [91, 141]. For our implications, we decided to target DOCK, OMEGA, ROCS, NAMD and WEKA. These are free for academics and we did not need to modify their source code for Condor.

2.2.6.1 DOCK

DOCK is developed at the University of California in San Francisco (UCSF) and evaluates chemical and geometric complementarities between a small molecule and a macromolecular binding site [142]. It examines the fitting pattern of the small molecules such as a drug into macromolecules like an enzyme or protein receptor. Compounds that might bind tightly to the target receptor must have complementary chemical and spatial natures. Thus, docking can be seen as a 3D puzzle searching for pieces that will fit into the receptor site [141]. It is important to be able to identify small molecules (compounds), which may bind to a target macromolecule [15]. This is because a compound, which binds to a biological macromolecule, may modulate its function and with further development eventually become a drug candidate.

The sequential connections between the sub-programs in the DOCK suite are depicted in Figure 2.10. The user initially has to prepare some preliminary files using the freely available chimera software [143]. Although a new update of chimera allows the user to build the molecular surface file from the non-hydrogenated pdb file, we used the open source C code of the DMS [144] program from the UCSF Computer Graphics Laboratory for this task. The program sphgen [145] generates spheres from these molecular surfaces which are used to create a negative image of the surface invaginations of the target. A subset of spheres which represents the binding site is chosen using a sphere-select program considering the defined radius around the ligand or the binding pocket residues. Then an interactive program, showbox, is used to visualize and define the location and size of the grid of sphere centers that reflects the actual shape of the active site which is then calculated using the grid program[146, 147]. The spheres (generated by sphgen) are matched with ligand atoms using the dock program and uses scoring (from the software GRID) to evaluate ligand orientations [145, 147].

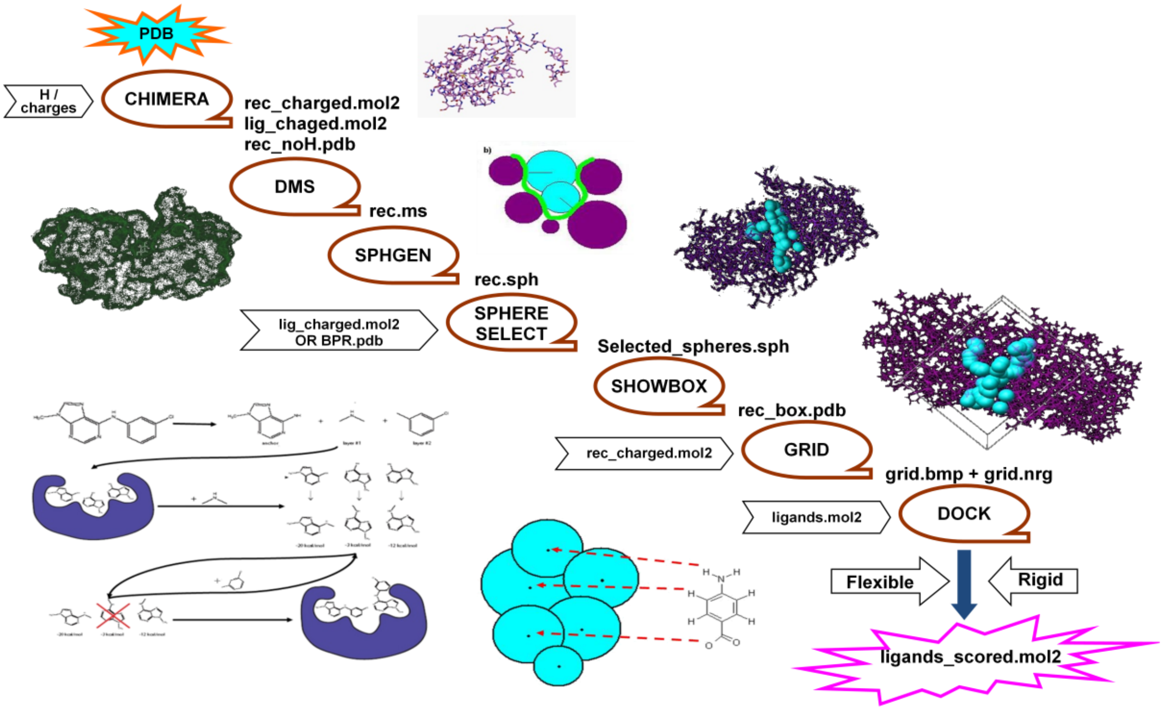


Figure 2.10: The sequential connection between sub-programs in the dock suite

The focus of our work was to create a workflow using condor's DAGMan [114] meta-scheduler and measuring the performance of the docking task when compared with single processor runs. Single compound docking using DOCK is expected to take up to 5 hours of execution time on a desktop computer. We docked 4 compounds with 100 conformations each into 60 homology models of P-glycoprotein using our grid, considering the mixed pool of linux and windows machines.

Two different cases were considered to determine the effect of inter-node communication speed in terms of cpu-time between the departments of computer-science and life-science. The 28 computers chosen are from the informatics labs from both departments, which are accessible to all users.

Case 1 – pool of 50 processors only from life-science department comprising of 22 dual-core linux machines and 6 windows machines with coLinux.

Case 2 – pool of 50 mixed processors where 6 windows machines with colinux nodes were from computer-science department and 22 dual-core linux machines from life-sciences department.

Table 2.1: The statistics of the 60 docking experiments

Parameter	Case 1	Case 2
Average time to complete a job	209.6 min	181.7 min
For 60 jobs on a single machine	(209.6*60) = 12576 min	(181.7*60) = 10902min
Time taken by grid for 60 jobs	366 min	322 min
Average speed-up value of grid	12576/366 = 34.36	10902/322 = 33.86
Average Efficiency	34.36/50 = 0.6872 (68.72%)	33.86/50 =0.6772 (67.72%)

Both pools perform in a similar way for the sequential tasks if we consider speedup values and efficiency. The 'time taken by the grid for 60 jobs' is more when compared to regular docking job as condor's queuing mechanism allows the calculations of the first 50 jobs on 50 processors, keeping 10 jobs idle which will be started after the completion of first 10 jobs in the queue. Also the delays in completion for case 1 might be because of the inter-departmental communication time as our central manager is located at the computer-science department. Similarly, if some users intervene during this measurement then this particular machine becomes unavailable, leading to transfer of the job to another computer, which adds a further delay in completion.

Web server implementation

We also implemented a web based docking job submission portal for DOCK with normalized parameters. To facilitate the docking for new users who are not aware about working with DOCK, we implemented an automated pipeline of our sequential docking method and developed a web server interface. Tomcat/Apache served as a J2EE container for Java Servlet and JSP. A iptables mechanism of Linux was used to store the user account information, uploaded data and docking results. The web server runs on the central manager mentioned earlier and works with both the Microsoft Internet Explorer and Mozilla Firefox browsers. It can be visited at https://goedel.ani.univie.ac.at/unigrid/.

2.2.6.2 NAMD

Molecular dynamics simulations play an important role in modern molecular biology and can be used for different purposes starting from studying the natural dynamics of biomolecules on different timescales with or without solutions or simulating a single molecule with its surroundings for a period of time [148]. It can also be used to determine the bulk properties of fluids and the free energy differences for chemical processes such as ligand binding and to explore the conformational space of a molecule or a complex [149]. These are very computer time-intensive tasks, which limits the physical time and the size of the systems that can be studied. A promising means of overcoming these limitations is through the use of parallel computers.

NAMD is a parallel, object-oriented molecular dynamics program designed for high performance simulation of large biomolecular systems [150-152]. It uses a spatial decomposition scheme to partition the domain in a way that provides maximum scalability coupled with a multithreaded, message-driven design which is shown to scale efficiently to multiple processors. Based on Charm++ parallel objects, NAMD scales to hundreds of processors on high-end parallel platforms. NAMD binaries were compiled for the parallel execution on our condor pool. Version 2.6 of NAMD was used, compiled with the mpi compiler and OpenMPI was used for parallelism.

We used the default files of Ubiquitin protein provided with NAMD installation for our performance measurement studies (running for 100 & 2000 steps). Following three instances were considered and CPU time in seconds was compared as shown in Table 2.2 and Figure 2.11.

1. In the first instance, only the Open-MPI setup was used to make the time-evaluation. This is very fast and takes only a short time to complete the job.

2. The second instance was establishing a local condor pool (using 20 PCs in the informatics lab of the life-science department), taking their real hostname & IP-addresses, establishing one of the PCs as central-manager and remaining 19 as execute PCs to perform the time-evaluation. This works similar to the first instance.

3. The third attempt of performance check has been done using the UVieCo grid using 20 PCs in the informatics lab of the life-science department considering virtual private IPs and a central manager which is in the computer-science department. In this case, the performance until 8 nodes/processors is almost similar to the 2nd instance, but onwards decreases drastically. The reason for this drop in performance might be because of the openVPN, the ethernet connection and inter-node communication time lag.

Table 2.2: The difference in calculation time for the mentioned 3 instances

No. of Processors	Time for 100 steps (sec)			Time for 2000 steps (sec)		
	Instance 1	Instance 2	Instance 3	Instance 1	Instance 2	Instance 3
2	66.4	61.3	125.28	1644	693.45	1625.2
4	46.3	45.5	54.32	706.7	388.76	729.5
8	40.67	36.76	47.6	329.62	242.42	678
12	43.08	39.87	56.6	245.18	228.76	795.4
16	39.24	36.5	54.2	212.48	186.53	994.3
20	37.4	34.54	52.06	230.54	222.68	1098.4
24	36.44	35.42	56.5	225.43	209.54	1064.9
28	35.13	35.2	64.2	224.51	213.4	1315
32	34.2	31.9	231.8	265.39	238.5	1683.6
36	32.48	31.59	268.8	248.07	242.68	1967.7
40	32.11	31.62	446.8	266.56	254.3	2458.8

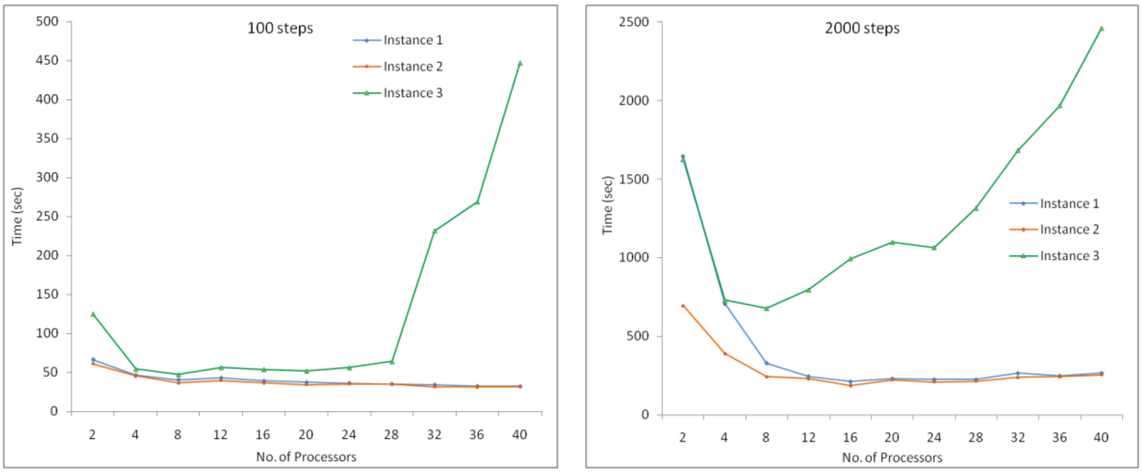


Figure 2.11: CPU time comparison for 3 instances considering 100 steps and 2000 steps of MD simulations using NAMD

2.2.6.3 OMEGA and ROCS

We implemented the shape-based similarity searching approach on our pool using OMEGA and ROCS from OpenEye Inc [153, 154]. We can process these tasks in a sequential manner using condor. But another hurdle came across for this implication, i.e. Execution of similarity-based screening requires huge amount of time on a single processor. Unfortunately, Condor stopped the support for PVM universe, which is needed for openeye's parallelization approach. So, we installed these programs in parallel using the PVM environment, independent of the Condor pool, thus making them efficient. Performance measurement was carried out with both OMEGA and ROCS (using 20 PCs in the informatics lab). For OMEGA analysis, we used 1395 compounds giving finally 100,811 conformers. For ROCS calculations, we screened a small dataset of 130 compounds (with max. 3 conformers of each) against 1395 compounds (with max. 10 conformers of each). Rest of the parameters was kept default for both OMEGA and ROCS. The statistics for different number of processors is shown below in Figure 2.12.

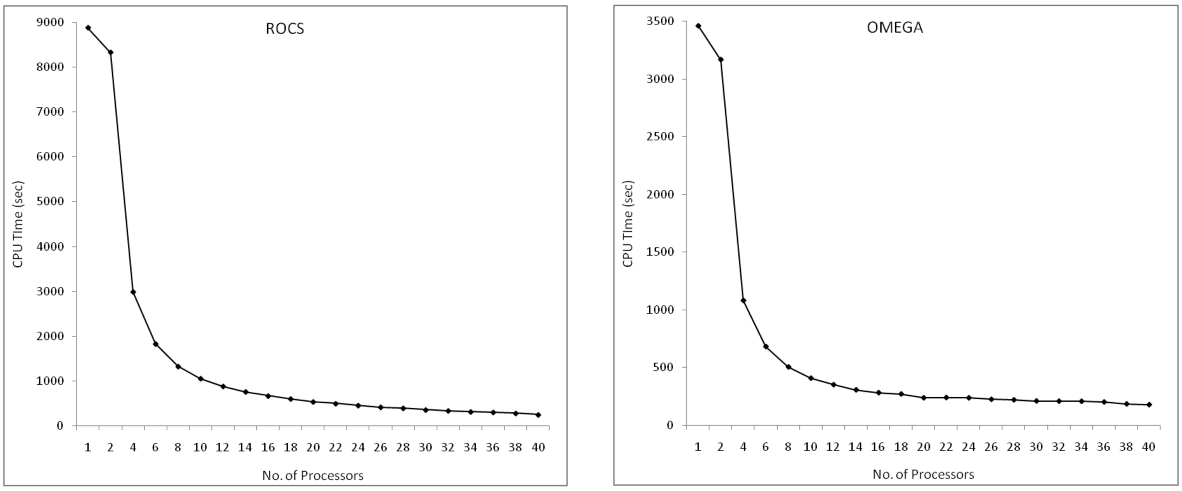


Figure 2.12 CPU time comparisons OMEGA and ROCS

2.2.6.4 WEKA

WEKA is developed at the University of Waikato, New Zealand and is a collection of state-of-the-art machine learning algorithms for data mining tasks [155, 156]. It contains tools for data pre-processing, classification, regression, clustering, association rules, attribute selection and visualization. It is designed to quickly trying out existing methods on new datasets in flexible ways as well as well-suited for developing new machine learning schemes.

Weka was extensively used with our Condor pool to carry out several tasks. For our classification problem, initially we used Weka’s support vector machine (SVM) module via libsvm [157]. In this program, the values of the gamma and cost parameters are needed to be chosen using grid search in such a way so as to get the maximum accuracy. We performed the grid search changing both the values of gamma and cost parameters from 2^{-15} till 2^{15} for various experiments with several feature-selection methodologies. We also tried other classification algorithms, changing several parameters similar to a grid search approach. The jobs were submitted as sequential tasks and accomplished on 20 linux-operated computers. The application of Weka for classification of P-gp substrates and non-substrates is discussed in details in the section 3.3.

2.2.7 Conclusions and Future Tasks

Computational grids facilitated to tackle the data-intensive, large-scale problems efficiently with sharing and aggregation of distributed resources. With UVieCo, we tried to address the computational drug discovery problems via effectively handling the firewall and coLinux issues alongwith implementation of several molecular modeling techniques using the middleware Condor. The use of coLinux to create a homogeneous grid of linux-based computers has proven to be very efficient. The VPN was enabled to tunnel through university-wide firewalls and to encrypt traffic across interdepartmental links, allowing nodes with private IP addresses to join UVieCo that crosses institutional boundaries. The DAGMan workflow manager was employed to automatize the execution of docking jobs. Virtual screening of thousands of compounds was enabled with fast shape similarity methods such as OMEGA and ROCS. Parallel execution of NAMD program was also tested using openMPI in UVieCo to measure the efficiency of grid for parallel tasks. It needed more detailed investigation to minimize the inter-node communication delays. Subsequently, a collection of machine learning algorithms using WEKA program was implemented for data mining and ligand-based QSAR. Web based monitoring is also possible with a separate portal for submission of docking jobs. Finally, other users are also need to be convinced to become part of UVieCo to keep it in use and continuously growing.

CHAPTER 3

NCI-60 DATA MINING AND GRID APPLICATIONS

In the second chapter, the basic architecture, underlying technology and novel concepts implemented in our grid environment, UVieCo were discussed. The important drug discovery techniques in the form of molecular docking, molecular dynamic simulations, shape-base similarity analysis and ligand-based QSAR techniques in the form of classification algorithms are also successfully implemented and reviewed.

In this chapter, we describe the real-life applications and scientific experiments carried out using these implementations. The data used in these studies was the raw data collected from the rich repository of National Cancer Institute.

3.1 NCI-60 Data Mining

Over the last 5 decades, the National Cancer Institute (NCI) has played a crucial role in the development of drugs for the cancer treatment which is evident from the fact that approximately half of the chemotherapeutic agents currently used were discovered and/or developed at NCI [158]. It was established way back in 1937 when it started to screen experimental cancer drugs. The transformation into an official organization (called as Cancer Chemotherapy National Service Center) took place in 1955 to incorporate laboratory resources with clinical facilities. In the early era, leukemic mice were used to test various compounds, but later in 1975, an initiative to screen new compounds with mice bearing murine leukemia P388 cells was introduced. This strategy satisfactorily identified compounds active against leukemias, but could not distinguish those which were effective against solid tumours such as carcinomas.

Thus, NCI developed a replacement approach for this method in terms of a new *in vitro* primary screen based upon a diverse panel of 60 human cancer cell lines (the NCI-60) [159]. The NCI-60 panel represents nine distinct tumour types such as leukemias (6 cell lines), melanomas (8), lung (9), colon (7), kidney (8), breast (8), prostate (2), ovary (6) and central nervous system (6). Since 1990, this panel of 60 human tumor cell lines was used by NCI's Developmental Therapeutics Program (DTP) to screen potential anticancer drugs from more than 70,000 chemical compounds with a defined range of concentrations to determine the relative degree of growth inhibition or cytotoxicity [160-162]. Each compound is tested in individual cell line and a sulforhodamine B assay is performed to assess the growth inhibition from the changes in total cellular protein after 48h of drug treatment. A characteristic profile or a “fingerprint” is generated using a vector of 60 growth inhibition values (50 percent growth inhibitory concentration, GI_{50}), one for each cell line, representing the activity pattern of a compound. The patterns of such 60 GI_{50} values across NCI-60 open a whole broad spectrum of unexpectedly rich, discriminative information about mechanisms of drug action, resistance and modulation [163-165]. Each compound’s pattern is essentially unique among many billions of distinguishable possibilities. The usefulness of this information can be further improved by the correlation of these activity patterns with molecular characteristics (e.g. MDR1 levels and p53 status) of genes/tissue types [165-168] or structural descriptors of the tested compounds [169]. All these experiments and assay procedures led to generation of various information-rich anticancer drug databases which include protein and mRNA expression profiles of molecular targets (genes) in the NCI-60, *in vitro* cell line screening and growth inhibitory profiles of small molecules along with their 2D and 3D structures collected and stored in the NCI repository since its establishment. To make better use of these information-rich databases, this project has also encouraged to develop, utilize and validate important data mining tools, like COMPARE [163, 170]. It analyzes correlations between protein/mRNA expression profiles of molecular targets (genes) and activity patterns of anticancer compounds using Pearson correlation coefficients to understand the possible mechanism of action of a drug and/or to discover potential novel lead compounds. Using COMPARE, promising lead compounds have been successfully identified for numerous molecular targets [171, 172].

Several remarkable characteristics of the cancer cell line data are presented by Wang et al. [173] who emphasize the integrated use of chemical, biological and genomic information from NCI-60 screen and relevance in finding biomarkers from variety of data relating to current research. By provision of a well curated set of tumor-related cellular assay screening results for thousands of compounds, it substitutes the use of high-throughput screening data. The screening program is progressing regularly as the information about the cancer cell line is continuously growing while maintaining the number of cell lines constant (in terms of numbers, assay procedures and comparability of results). The most important thing about this data is that it is provided freely from the DTP Web server which can be used for further research and publications.

3.1.1 Gene expression profiles and drug activity patterns

In the past, assessment of cell characteristics was performed on the basis of one gene, gene product or molecular pathway at a time. Scherf et al. [174] imparted a broad perspective to this work and performed more extensive microarray expression analysis of many thousands of genes simultaneously in the 60 cancer cell lines to generate protein and mRNA expression database [175-177]. These databases are applied for cancer diagnosis, prognosis, prevention and therapy and can been included as useful information to the data mining activity of the NCI compound screening [178].

It has been observed to a large extend that NCI-60 cells lines show common behavior in the gene expression patterns and clustered in the same way as they cluster based on their phenotypic properties (in this case, histological tissue of origin). According to Ross et al. [177], the gene expression profiles illustrate the patterns of phenotypic variation in the NCI-60 cancer cell lines. This approach of analyzing gene expression patterns was extended by Scherf et al. to pharmacologically characterize the drug sensitivity [174]. They correlated the activity patterns of more than 70,000 compounds tested in NCI-60 against mRNA expression levels of 9,703 cDNAs (~8,000 unique genes) in NCI-60 for the interpretation of drug-gene expression profiles. It is well represented in figure 3.1. A subset of 1,376 genes was selected from the initial 9,703 genes based on selective filters of strong patterns of variations and clustering in cell lines. Similarly, from the database of 70,000 compounds, 1,400 compounds tested at least 4 times on all or most of the cell lines were selected for detailed analysis. Cluster analysis was performed using both subsets to organize NCI-60 cell lines on the basis of gene expression patterns and the growth inhibitory activities (GI_{50}). It has been demonstrated that the 60 cell lines clustered well by organ of origin on the basis of gene expression patterns rather than on the basis of the patterns of drug sensitivity. The reason for this difference in clustering was attributed to the ability of certain cell lines (NCI-ADR-RES, ACHN, UO-31 and HCT15) to express the multi-drug resistance gene ABCB1 [179].

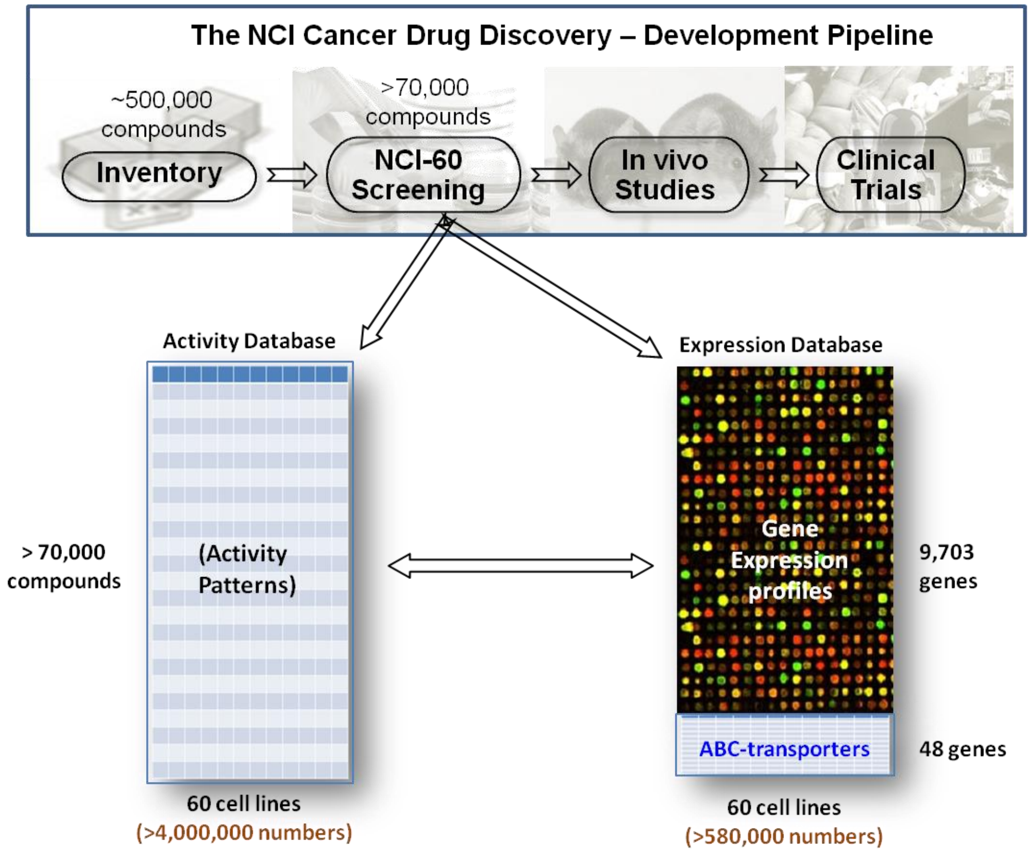


Figure 3.1: Simplified representation of NCI's drug discovery and development program. Each row of the activity database represents the pattern of activity of a particular compound across the 60 cell lines, and each column represents the pattern of sensitivities of a particular cell line to the compounds tested. The gene-expression database contains cDNA microarray measurements on the 60 cell lines. The small gene expression database of 48 ABC transporters was performed by Szakács et al. Modified from Figure 1 of Reference [174].

3.1.2 Expression profiling of ABC transporters in NCI-60

Szakács et al. [179] extended the above mentioned approach and hypothesized that measuring the expression of ABC transporters in NCI-60 will allow to correlate ABC transport function with the structural, molecular, physiological and pharmacological features of the cells. Similarly, analysis of resistance or sensitivity of particular transporter to several classes of anti-cancer agents can be assessed based on relationship between ABC expression levels and sensitivity to drugs. Earlier studies about expression profiling of NCI-60 cell lines from Scherf et al. [174] and Staunton et al. [180] involved only 15 and 11 of 48 ABC transporters using cDNA arrays and affy-metrix Hu6800 oligonucleotide chips, respectively. This led Szakács et al. [179] to perform the transcript expression measurement experiments using the superior method, quantitative real-time RT-PCR, rather than the less sensitive, less specific microarray technology. Improved quantitative correlations were observed between expression and sensitivity which demonstrated the important role of ABC transporters in the drug resistance of cancer cells. The complete RT-PCR results on the 48 ABC transporters were presented by Szakács et al. [179] and we used them to calculate the person correlation coefficients for a larger set of compounds. They showed that the cancer with specific tissue of origin cluster together leading to common pattern of expression profiles.

3.1.3 Correlation of mRNA expression of ABC transporters and drug sensitivity

Since ABCB1 (P-gp/ MDR1) expels the chemicals from the cell, it has been demonstrated that the compounds showing negative correlation between their sensitivity/cytotoxicity and the transporter expression across the NCI-60 are expected to be transported by the transporter and termed as ‘substrates’ [179, 181, 182]. Weinstein et al. [164] plotted the correlations of a set of 118 compounds with known mechanism of action to confirm this hypothesis. They found that substrates showed prominent inverse correlations, whereas non-substrates were noncorrelated or positively correlated. In this study, they also found that two drugs were inversely correlated [(NSC 355644, $r = -0.36$) and Baker's soluble antifol (NSC 139105, $r = -0.3$)], despite no proof of being as ABCB1 substrates. However, Gupta et al. showed that Verapamil, a potent ABCB1 inhibitor reverses the resistance of Baker’s antifol, indicating it as an ABCB1 substrate [183].

To identify which ABC transporters and substrates might play roles in drug resistance of cancer cells, Szakács et al. [179] extended the analysis to a larger data set of 1,429 compounds and calculated Pearson's correlation coefficients for a total of 68,592 relationships (48 genes × 1429 compounds). They emphasize that the real-time RT-PCR database and analytical approaches used by them provided an unbiased method for discovering the substrate specificities of known, as well as yet uncharacterized, members of the ABC superfamily.

3.1.4 Calculation of Pearson correlation coefficients for available compounds

In the present work, we performed the comprehensive analysis of NCI-60 drug activity data using RT-PCR expression results of the 48 ABC transporters from Szakács et al. [179] to compute Pearson correlation coefficients for all available compounds. Here, we were interested in 50% growth inhibition (GI_{50}) which will give a good indication about drug sensitivity, resistance and toxicity and was also used by Szakács et al. [179]. The screening results of the drugs which were screened against NCI-60 for cancer are updated periodically. The GI50 metafile updated in October 2009 ("gi50_oct09.bin") was downloaded in the form of compressed ASCII file from the NCI's Developmental Therapeutics Program (DTP) server (http://dtpws4.ncifcrf.gov). Each row describes eleven different dose response parameters to express the screened drugs alongwith the log activity values for 159 cell lines representing cancers of different origin. They are as follows [184]:

1. NSC number - the NCI's internal ID number (NSC)
2. Concentration unit – either molar, volumentric or µg/mL (CONCUNIT)
3. Log of highest concentration tested (LCONC)
4. Panel name for the cell line (PANEL)
5. Cell line name (CELL)
6. Panel number of the cell line (PANELBR)
7. Cell number of the cell line (CELLBR)
8. Log of the result (NLOGGI50)
9. Number of tests for NSC and cell line (INDN)
10. Maximum number of tests for this NSC (TOTN)
11. Standard deviation for the $\log_{10}$ of the results average across all tests for this NSC and cell line (STDDEV)

The handling and preprocessing of this file was very complicated and time consuming as the word handling programs were unable to process such a huge file of size 185MB with 2.7 million rows. Thus, we spited the big metafile into small sub-files. Several compounds were having more than one concentration (LCONC) values (i.e. the highest concentration tested of the cell-line to achieve 50% growth inhibition). The concentration units are also represented in terms of Molar, μg/ml and volumetric units which are finally presented as LOGGI50 values. Since the developed perl script (Appendix 3) was unable to handle compounds with same NSC but tested at different concentrations, they were renamed based on the concentrations and represented uniquely. The same script was used to process each of the sub-files. The code of which can be represented as follows:

- *Read the sub-file*
- *Present the data about expression of particular ABC transporter under study in the 60 cell lines (taken from Szakács et al. [179])*
- *Split the rows based on comma as delimiter*
- *Read the lines specified for each unique NSC*
- *Check if the cell line is from the 60 cell lines we are interested in.*
- *Consider the NLOGGI50 activity values for these cell lines*
- *Calculate the sum, mean, standard deviation for each NSC based on available NLOGGI50.*
- *Compute covariance between expression values of particular transporter and retrieved NLOGGI50 activity values*
- *Finally compute the Pearson correlation coefficient (PCC)*
- *Iterate the process for all compounds till the end*

The Pearson correlation coefficient (PCC) between the activity of the *ith* compound and the expression level of the ABC gene was calculated based on standard deviation and covariance using following formula:

$$PCC_{AiT} = \frac{C_{AiT}}{S_{Ai}S_T}$$

Where,

PCC_{AiT} = Pearson correlation coefficient of *ith* compound for particular transporter

S_{Ai} and S_T = Standard deviation of the activity of the *ith* compound and the expression of transporter, calculated from the available GI_{50} data out of 60 cell-lines.

C_{AiT} = Covariance between the two variables, viz – activity and expression

The process mentioned in the perl script was iterated several times, searching the new NSC from the top of the file again and again after finishing the calculation of PCC for each NSC. Thus, having bigger file consumes huge amount of computing power and time for processing. It was avoided by splitting the metafile into sub-files providing great advantage.

With this process, 48,758 Pearson correlation coefficients were calculated for 47,622 compounds considering gene expression patterns of each of the 48 ABC transporters in the 60 cell lines and GI_{50} data. Only 40,935 compounds have valid PCCs i.e. 6687 compounds had same GI_{50} activity across all available NCI-60 cell lines, leading to standard-deviation to zero. So, we decided to consider only these compounds which have valid growth-inhibitory information. The updated 2D-structure file ("jan2010_2d.zip") was downloaded from the same DTP server mentioned earlier. It contains structures for 40,375 compounds out of 40,935 compounds in sdf format. The remaining 560 compounds were manually searched on the PubChem Substance database [185]. Out of 560 compounds, structures of only 220 compounds were retrieved as the rest of the compounds were representing proteins, extracts, glycosides, interleukins and polymers and therefore not considered in our study. These were added to earlier dataset containing 40,375 compounds. Thus, finally we have a dataset of 40,595 compounds with valid PCCs for all available 48 transporters.

This dataset was used as a starting point for shape similarity measurements and QSAR analysis in the form of classification studies. In the next sections, we will discuss the utilization of NCI's data mining approach in separation of substrates and non-substrates of ABCB1/ P-gp transporter using 3D-shape similarity approach and classification approach for handling imbalanced data.

3.2 Shape Similarity Approach

Molecular similarity approaches are widely used in the computational drug development studies for various purposes such as virtual screening, chemical property and activity prediction, data mining and selection of potential hits. In these methods, the structural characteristics of the compounds under investigation are examined to measure similarity/dissimilarity among them [186]. Even with the pharmacophore modeling or quantitative structure–activity relationships (QSARs), similarity among the local determinants of activity such as stereochemistry, orientation and conformations of the compounds, bonding patterns, functional groups and related physicochemical properties is considered. According to the 'Molecular similarity principle', structurally similar compounds behave in the same manner having similar physicochemical as well as biological properties than dissimilar ones [187]. Thus, medicinal chemists tend to have measures for precise prediction of similarity or dissimilarity among the molecules to get better insight into the underlying SAR.

Different methods are proposed to assess similarity, ranging from two- /three-dimensional, quantum chemical, molecular field approaches to supervised/unsupervised machine learning methods. The chemical compounds are represented quantitatively using topological, spatial, physicochemical, electronic, quantum chemical descriptors; molecular fingerprints and holograms as well as molecular shape analysis in terms of steric and electronic overlays [188]. Comparison of these numeric values (binary or nonbinary) and their patterns provide an important mean to evaluate structural similarity among the molecules. Similarity coefficients strongly influence the degree of similarity between these properties, thus exhibiting similarity among the compounds themselves. Several forms of these coefficients are proposed and categorized in three types such as association coefficients (for binary data), correlation coefficients (assess degree of correlation among numerical descriptors) and distance coefficients (measure degree of dissimilarity) [189, 190].

Among all these coefficients, tanimoto coefficient based on binary fingerprints is widely used for determination of chemical similarity. It can be represented by following Tanimoto equation

$$\text{Tanimoto coefficient} = nAB / (nA + nB - nAB)$$

Where nA and nB are the number of structural fragments/features in compound A and B respectively and nAB is the number of common features between them.

Fingerprints are the set of commonly used fragment substructures. Presence or absence of structural fragments is expressed as 1s and 0s (bit string) representing a unique binary pattern for individual molecule. Collection of predefined structural features, such as MACCS keys are used to identify fragments contained in a molecule. Comparison of these topological identities proves to be an effective tool in screening large chemical libraries. Despite their fast and efficient processing, the 2D similarity methods incline towards searching compounds of resembling scaffolds to that of reference molecule. Thus, to identify structurally diverse compounds taking into account the stereochemical aspects and geometric constraints, 3D shape similarity techniques are extensively used now-a-days. But the complexity in generation, selection and comparison of 3D representations of chemical compounds make such methods computationally expensive and time consuming. Grid computing and fast hardware/ networking resources prove to be an efficient way to increase the available computational power which has enabled to use fast 3D structural overlay programs, such as ROCS [191] for large database screening.

3.2.1 Identification of P-gp Substrates

In this section, we present a distinct approach of identification and separation of P-gp substrates from non-substrates using three-dimensional shape similarity technique. Two NCI databases (containing 35,692 and 1,429 compounds respectively) with Pearson correlation coefficient (PCC) information were used. 3D shape similarity calculations were performed using the substrates from the small database to screen the large database. Results were analyzed using area under the ROC curve to understand whether this approach was able to identify P-gp substrates.

Materials and Methods

The database containing 3D-strucutres of 42,247 compounds ("cans03sd.bin") which was ready to download from NCI's DTP server was used in this study. From this database, 35,692 compounds with available PCC values were extracted and used for further studies ('NCI_big' in short). The distribution of the PCC values in NCI_big is presented in figure 3.2. A smaller NCI database ('NCI_small' in short) containing 1429 compounds with PCC information for 48 ABC transporters is available as supplementary information from Szakács et al.[179]. For the present study, we considered compounds with PCC values only related to P-gp. In both databases, compounds with PCC less than -0.30 were subsequently annotated as 'substrates' and rest of the compounds as 'non-substrates'. This resulted in 885 and 104 substrates for NCI_big and NCI_small respectively.

ROCS 2.4.2 (Rapid Overlays of Chemical Structures) superposition program was chosen to determine similar compounds based on 3D configurations [153, 191]. It distinguishes compounds with similar 3D shape using atom-centered Gaussians as a measure to compute geometric overlap. Along with the shape based measure of molecular volume overlap (shapeTanimoto score), chemical complementarity of the functional groups (donor, acceptor, hydrophobe, cation, anion, and ring) is critical for biological activity and is considered using color force-field (color score). The combination of both these scores, so-called as "combo score" provides an improved estimate of shape similarity. 3D shape similarity studies often require the reference/query compounds to search related similar compounds from large databases. Computation of conformational space based on the molecular flexibility is also important for both such types of compounds and is taken into account using the OMEGA 2.3.2 program [192].

Conformer databases were created for both these data sets using OMEGA 2.3.2. For each of the 104 substrates from NCI_small, a single low-energy conformation was generated (by setting the parameter maxconfs = 1) which were then used as query molecules. Similarly, maximum 400 conformations were generated for the NCI_big database. Rest of the parameters was kept as defaults. ROCS 2.4.2 has a built-in ImplicitMillsDean color forcefield which includes pKa model based on pH=7 and is used in this work (with the –chemff ImplicitMillsDean flag). All overlays were optimized to maximize color (chemical) overlap after the best shape overlay was located (using the -optchem flag). The hits were ranked on the basis of the sum of their shapeTanimoto and the normalized color score in this optimized overlay (using the -rankby combo flag). Also the combo cutoff was set to 0.3 to cover the whole chemical space of NCI_big (using flag -cutoff = 0.3).

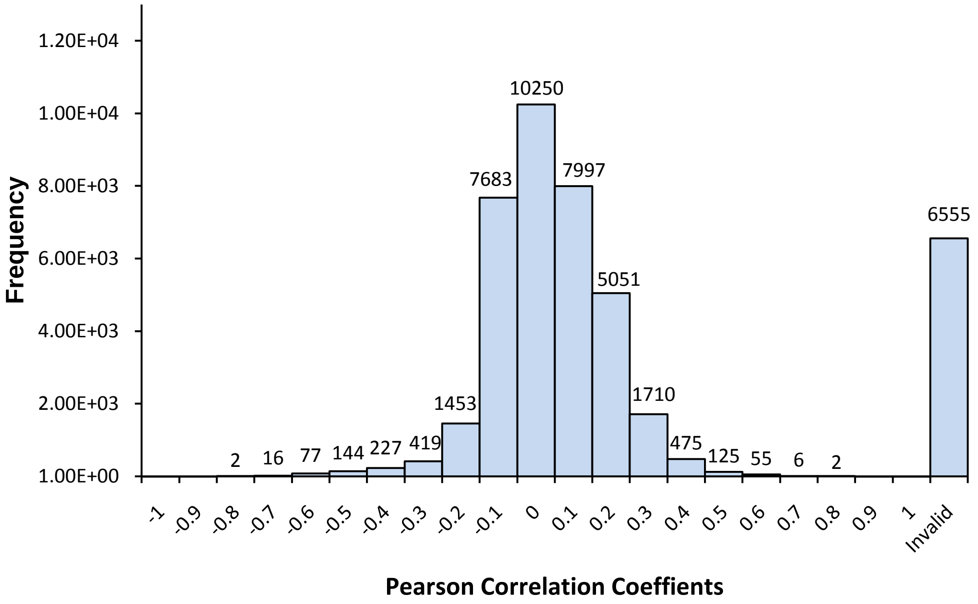


Figure 3.2 Pearson correlation coefficients of the compounds present in NCI-big database

Several quantitative techniques such as calculation of enrichment factor or hit rate were proposed to assess the performance of a particular tool in virtual screening. These metrics solely depend on the ranking-based cutoffs at different intervals, thus minor changes in ranking lead to large variations in the final results. This sensitivity to the small changes in ranking can be avoided to a great extend by screening entire databases and measure the performance using area under the receiver operator characteristic, or ROC [193]. The ROC curve is derived by plotting sensitivity (fraction of false positives) on the x-axis versus 1-specificity (fraction of true positives) on the y-axis. It provides an accurate estimate of given tool's performance by pruning the whole database, unlike enrichment factor or hit rate which examine just specified, initial points of the database. The value of area under a ROC curve is 1.0 (maximum) when virtual screening tool perform perfectly whereas value of 0.5 suggests random performance. In this study, we used all the 3 scores viz shapeTanimoto, color and combo. Area under the ROC curve (AUROC) was calculated, compared, and the performance of virtual screening experiments was assessed.

Results and Discussion

104 substrates from NCI_small were used as queries for 'shape and chemical similarity based screening' of 35692 compounds from NCI_big using ROCS. Each query was processed against each compound from NCI_big database, thus giving ~40,00,000 different of scores. Combo score was used to sort results from highest to lowest. The numbers of unique hits were separated based on multiple thresholds which is the requirement for calculating area under the ROC curve (Table 3.1). Similar separation was performed using shapeTanimoto and color scores (Table 3.2 and 3.3). If the hit is annotated as 'substrate' in NCI_big, it is considered as true positive. Similarly number of false negative, false positive and true negative compounds was determined. Subsequently, pairs of sensitivity and 1-speficity derived for each of these thresholds were used to plot ROC curve (Figure 3.3). Finally, the AUC is manually calculated in Microsoft Excel using Trapezoidal rule which is the sum of the areas of the rectangles below the ROC curve [193]:

$$AUC = \sum_{i} [\frac{(Se_{i+1})(Sp_{i+1} - Sp_i)}{2}]$$

Where Se and Sp denote sensitivity and specificity respectively.

Table 3.1: Cumulative number of hits retrieved per threshold using combo score

Threshold	Hits	NCI_big Substrates (TP)	FN	FP	TN	Sensitivity TP/(TP+FN)	(1-Specificity) FP/(FP+TN)
Combo = 2.0	111	90	795	21	34786	0.10	0.00
Combo = 1.95	194	109	776	85	34722	0.12	0.00
Combo = 1.9	302	127	758	175	34632	0.14	0.01
Combo = 1.85	428	141	744	287	34520	0.16	0.01
Combo = 1.8	570	152	733	418	34389	0.17	0.01
Combo = 1.75	709	162	723	547	34260	0.18	0.02
Combo = 1.7	940	173	712	767	34040	0.20	0.02
Combo = 1.65	1401	179	706	1222	33585	0.20	0.04
Combo = 1.6	2163	196	689	1967	32840	0.22	0.06
Combo = 1.55	3107	214	671	2893	31914	0.24	0.08
Combo = 1.5	4279	231	654	4048	30759	0.26	0.12
Combo = 1.45	5682	250	635	5432	29375	0.28	0.16
Combo = 1.4	7380	278	607	7102	27705	0.31	0.20
Combo = 1.35	9456	308	577	9148	25659	0.35	0.26
Combo = 1.3	12368	344	541	12024	22783	0.39	0.35
Combo = 1.25	16021	392	493	15629	19178	0.44	0.45
Combo = 1.2	19671	439	446	19232	15575	0.50	0.55
Combo = 1.15	22799	494	391	22305	12502	0.56	0.64
Combo = 1.1	25506	565	320	24941	9866	0.64	0.72
Combo = 1.05	27749	639	246	27110	7697	0.72	0.78
Combo = 1.0	29605	693	192	28912	5895	0.78	0.83
Combo = 0.95	30955	734	151	30221	4586	0.83	0.87
Combo = 0.9	31942	762	123	31180	3627	0.86	0.90
Combo = 0.85	32663	785	100	31878	2929	0.89	0.92
Combo= 0.8	33163	807	78	32356	2451	0.91	0.93
Combo = 0.75	33521	818	67	32703	2104	0.92	0.94
Combo = 0.7	33762	828	57	32934	1873	0.94	0.95
Combo = 0.65	33893	838	47	33055	1752	0.95	0.95
Combo = 0.6	33974	841	44	33133	1674	0.95	0.95
Combo = 0.55	34019	842	43	33177	1630	0.95	0.95
Combo = 0.5	34054	843	42	33211	1596	0.95	0.95
Combo = 0.4	34090	845	40	33245	1562	0.95	0.96
Combo = 0.3	34095	845	40	33250	1557	0.95	0.96

Table 3.2: Cumulative number of hits retrieved per threshold using color score

Threshold	Hits	NCI_big Substrates (TP)	FN	FP	TN	Sensitivity TP/(TP+FN)	(1-Specificity) FP/(FP+TN)
Color = 1.0	250	126	759	124	34683	0.14	0.00
Color = 0.95	764	152	733	612	34195	0.17	0.02
Color = 0.9	1145	166	719	979	33828	0.19	0.03
Color = 0.85	1595	178	707	1417	33390	0.20	0.04
Color= 0.8	2324	206	679	2118	32689	0.23	0.06
Color = 0.75	4212	259	626	3953	30854	0.29	0.11
Color = 0.7	8523	345	540	8178	26629	0.39	0.23
Color = 0.65	10623	391	494	10232	24575	0.44	0.29
Color = 0.6	13212	447	438	12765	22042	0.51	0.37
Color = 0.55	18846	572	313	18274	16533	0.65	0.53
Color = 0.5	23959	665	220	23294	11513	0.75	0.67
Color = 0.45	27817	732	153	27085	7722	0.83	0.78
Color = 0.4	29752	766	119	28986	5821	0.87	0.83
Color = 0.35	31313	801	84	30512	4295	0.91	0.88
Color = 0.3	32255	819	66	31436	3371	0.93	0.90
Color = 0.25	33393	836	49	32557	2250	0.94	0.94
Color = 0.2	33722	840	45	32882	1925	0.95	0.94
Color = 0.1	34025	845	40	33180	1627	0.95	0.95
Color = 0.0	34095	845	40	33250	1557	0.95	0.96

Table 3.3: Cumulative number of hits retrieved per threshold using shapeTanimoto score

Threshold	Hits	NCI_big Substrates (TP)	FN	FP	TN	Sensitivity TP/(TP+FN)	(1-Specificity) FP/(FP+TN)
ST = 1.0	114	90	795	24	34783	0.10	0.00
ST = 0.95	267	119	766	148	34659	0.13	0.00
ST = 0.9	968	155	730	813	33994	0.18	0.02
ST = 0.85	3093	198	687	2895	31912	0.22	0.08
ST= 0.8	8401	269	616	8132	26675	0.30	0.23
ST = 0.75	16569	370	515	16199	18608	0.42	0.47
ST = 0.7	24117	453	432	23664	11143	0.51	0.68
ST = 0.65	28713	574	311	28139	6668	0.65	0.81
ST = 0.6	31358	663	222	30695	4112	0.75	0.88
ST = 0.55	32713	733	152	31980	2827	0.83	0.92
ST = 0.5	33322	780	105	32542	2265	0.88	0.93
ST = 0.45	33746	823	62	32923	1884	0.93	0.95
ST = 0.4	33977	839	46	33138	1669	0.95	0.95
ST = 0.3	34095	845	40	33250	1557	0.95	0.96

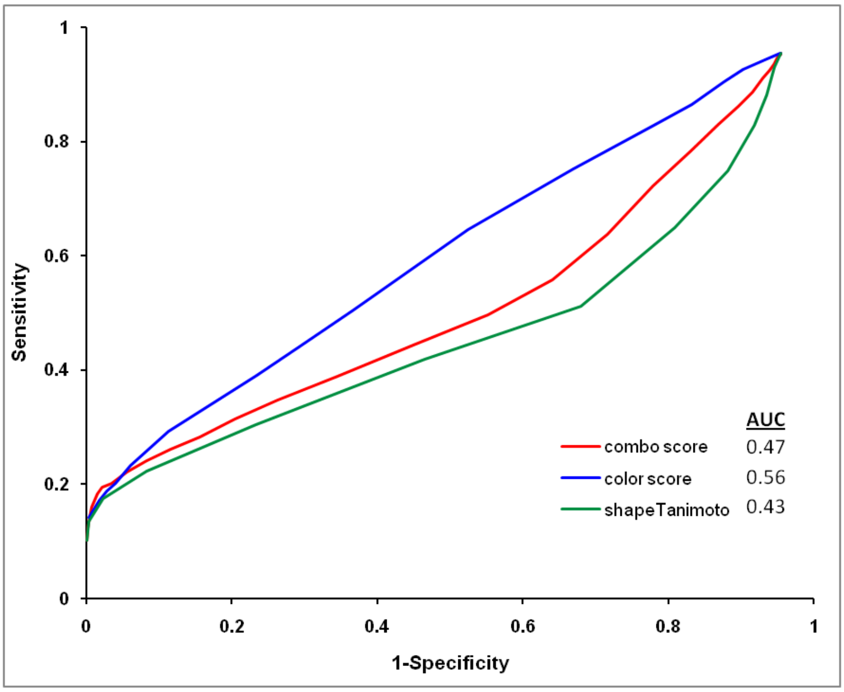


Figure 3.3 Area under the ROC curves obtained using 3 different scores (combo score, color score and shapeTanimoto score) from the ROCS results of NCI_big database using substrates from NCI_small

Various structure-based (e.g. docking) as well as ligand-based methods (e.g. shape-similarity based) are extensively used in identification of hits from vast chemical space. Similarly, they have been proven efficient in distinguishing binders from non-binders, in our case substrates from non-substrates. Generally, most of these methods require the knowledge of bioactive conformation of reference compounds for superior performance. The ligand-oriented method used in this study, ROCS bypasses this prerequisite and uses similarity based on shape, pharmacophoric features and their combination to rank the compounds [194]. In our study, equal weight to the alignment of Gaussian volumes (shape similarity) and chemical complementarity (functional group similarity) in combo score failed to provide superior performance in virtual screening. Rather, the overlap of chemical groups performs better than rest of two similarity measures.

Shape-similarity methods frequently endure a contradictory problem of false positives and false negatives. As both shape and functional group similarity of unknown compounds is considered and matched with that of the query molecules, there are higher chances of false positives. Similarly, compounds with different shape than query compounds could simply escape the screening. In our case, 104 substrates from NCI_small which are used as queries and the compounds in NCI_big are diverse in nature. ROCS algorithm initially overlaps the centers of mass of query and database compounds and then their principal moments of inertia are aligned. Thus, when two diverse compounds having same functional groups are aligned based on moments of inertia, these functional groups are misaligned. This results in low combo score for the compounds. This might be the reason for low AUC values when shapeTanimoto and combo score are used for plotting ROC curves.

Conclusions

In conclusion, three different types of scores viz shapeTanimoto, color and combo, are used to assess the performance of the ROCS program in identifying compounds of interest, here P-gp substrates. Even though the scoring approach based on pharmacophoric features demonstrate superiority over the other two scores, the results could not be considered significant as the area under the ROC curve values are just around random. This suggests that the 3D shape similarity approach using ROCS is insufficient to identify and separate substrates from non-substrates.

3.2.2 Identification of SSRI-like compounds using shape similarity approach

Some selective serotonin reuptake inhibitors (SSRIs) have been described to be responsible for programmed cell death in B-lymphoid originated malignant cells, i.e. cells extracted from Burkitt lymphoma [195]. Current cytotoxic treatment is very expensive to the people where this disease is endemic, i.e. sub-Saharan Africa, Papua New Guinea and northeastern Brazil [196]. Thus, SSRIs can prove as an alternative, less expensive modality for treating Burkitt lymphoma. However, low selectivity is a prime concern for SSRIs since at μM concentration range, they interact with other ubiquitously expressed targets inhibiting proliferation of and killing variety of mammalian cells [197]. In such concentrations, they also act as effective spermicides and can kill protozoa such as *Trichomonas vaginalis* [198]. These higher concentrations needed for exerting toxic action can also cause potentially severe serotonin syndrome. On the other hand, our collaboration partners Freissmuth and co-workers ascertained the relation between affinity to the obvious SSRI target, serotonin transporter (SERT) and extent of apoptosis using [^{3}H]thymidine incorporation assay. The acetylated versions of SSRIs (*N*-acetyl-fluvoxamine and *N*-acetyl-paroxetine) kill the experimental tumor cells in comparable concentrations as those of their original non-acetylated counterparts, demonstrating SERT is not necessary for cell killing [197]. This led us to devise all the virtual screening experiments in search of potent SSRI-like compounds exclusively using ligand-based virtual screening approaches.

Initially, the in-house library of synthesized compounds and later the chemical library from Enamine Ltd. were used as the screening databases to identify novel chemotypes similar to SSRIs. The program ROCS v3.1.2 [191] was used to assess the similarity between the reference compounds (sertraline and paroxetine) and the compounds from the datasets. In the preliminary screening (round 1) of the in-house data, where top-ranked compounds were tested using [^{3}H]thymidine incorporation assay for their ability to induce apoptosis in HEK293 cells, we found a molecule (REM25, 8.81μM) which was more potent than the most active SSRIs. Subsequently, for screening of a large chemical library (round 2), an unsupervised machine learning method, self organizing maps (SOM) [199] was used in order to narrow down the number of hits retrieved from ROCS screening. Out of 8 commercial compounds selected and purchased for biological evaluation, 2 were confirmed as potential hits *in vitro* (25% hit rate). These hits possess chemical scaffolds different from those of the reference compounds. The scaffold-based follow-up screening of the novel chemotypes yielded several additional hits in micromolar range, out of which one compound will be chosen as lead compound for preliminary structure-activity relationship studies. Still there is much room for improvement in the compound potency by modifying the flexible backbone and substituted phenyl moiety. The results demonstrate the effectiveness of *in silico* techniques such as shape similarity approach and unsupervised machine learning methods in identification of new chemotypes.

Materials and Methods

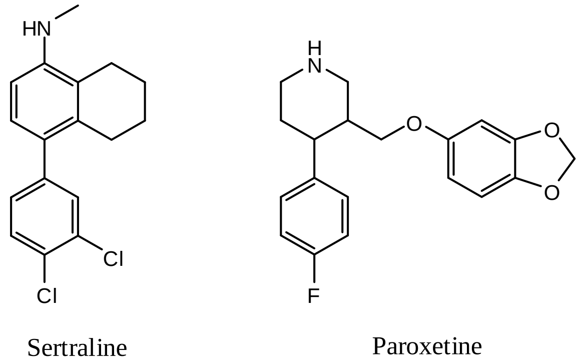
Datasets

The in-house database contained 412 unique compounds either synthesized in our lab or provided by collaboration partners. The other chemical library containing 1,286,460 compounds was obtained from Enamine (Enamine Ltd., version Aug-2009, Kiev, Ukraine, http://www.enamine.net). Counterions, salts and metal containing fragments were removed in MOE v2009.10 [200] before generating the conformations in OMEGA v2.4.6 [192]. 3394 compounds were excluded further during the processing since OMEGA does not consider compounds with unparameterized atoms like Se, Pt, Mn, As, etc. as well as compounds with inappropriate geometry. 1,283,066 compounds in 365,836,769 conformations were finally scored by ROCS. After the preliminary screening, thirteen compounds (REM14 [201], REM25 [201], GE68 [202], REC2219 [203], GPV0389 [204], GPV0385 [205], GPV0865 [206], GPV0189 [203], GPV0186 [203], GPV0825 [207], GPV0896 [207], GPV0442 [208], and GPV0512 [209]) were selected for pharmacological experiments.

In the second round of screening, 18 permanently charged compounds were removed out of 400 ROCS hits (200 each for sertraline and paroxetine). Additionally, a small subset of five known actives (SSRIs such as paroxetine, fluoxetine and hits retrieved from in-house database screening: REM25, REM14 and GPV896) was used for the self-organizing maps based screening. Thus, for the dataset of 387 compounds, the lowest energy conformer of each molecule was retrieved from OMEGA and the energy minimized in MOE using MMFFx94 forcefield.

Shape-based similarity screening

OMEGA v2.4.6 was used to generate the 10 conformers of the reference compounds (**sertraline** and **paroxetine**) and upto 400 conformers of each compound from the in-house as well as enamine databases. Rest of the parameters was kept default. The three-dimensional shape-based similarity screening was performed with the program ROCS v3.1.2. The ComboScore, the sum of ShapeTanimoto score and ScaledColor score, was employed to rank the database compounds. The cutoff value of ComboScore was set to the value 1.



Structures of the chemical compounds used as reference in ROCS screening round 1.

Self-Organizing Map (SOM)

Self-organizing maps (or Kohonen neural networks) [199], a specialized type of artificial neural network, implemented in SONNIA v4.20 software package [210] was used in this study. In this method, higher dimensional objects are visualized into a space of lower dimensionality such as two-dimensional arrangement in a plane. The class labels are not considered while training the maps and thus the self-organizing maps are called as an unsupervised learning method. To begin with the training process, the weights are randomly assigned to each neuron and the number of weights corresponds to the dimensions of the structural representations, i.e. number of descriptors. During training molecules are arbitrarily presented to the network several times. For each molecule, the most similar neuron, so called winning neuron is identified based on the lowest Euclidean distance between its structural descriptors and the neuron weights. The response of the neural network is further improved by adjusting the weights of winning neuron accordingly to the training data. As the training progresses, the degree of weight adjustment and the number of neighborhood neurons is decreased. When the training is finished, overall network response is computed and the molecules are projected from high dimensional space into 2D plane. This leads to map similar compounds in adjacent or even in the same neurons. It depends on mainly the size of the network, the heterogeneity in the data set and the number of compounds.

Seven types of descriptors, two network topologies (rectangular and toroidal) and 3 different network sizes (15x15, 19x12 and 20x20) were used to train the models of SOMs and later utilized for identifying co-localization of the Enamine compounds with known actives. Descriptors (Topological (2D), VSA, VSURF, MACCS, Physicochemical properties, Topological (2D) autocorrelation and Spatial (3D) autocorrelation vectors [211]) were calculated using the software packages such as MOE v2009.10 and ADRIANA.code v2.0 [210]. For using in SONNIA, the descriptors were first autoscaled (mean centered and scaled to unit variance) to yield the final set.

Hit selection was performed as outlined in the Results and Discussion section. The sizes of the maps were chosen in a way to prevent over-crowding of the neurons with high number of co-localizations. Table 3.4 gives the number of Enamine compounds co-localizing with 5 actives obtained with the seven models and three different network sizes.

Table 3.4: SOM Models and number of co-localizations

SOM Model	Type of Descriptors	No. of Descriptors	Dimension	Topology	No. co-localizations
model 1a	MOE 2D	146	15 x 15	Rectangular	0
model 1b	MOE 2D	146	19 x 12	Rectangular	0
model 1c	MOE 2D	146	20 x 20	Rectangular	0
model 1d	MOE 2D	146	15 x 15	Toroidal	5
model 1e	MOE 2D	146	19 x 12	Toroidal	0
model 1f	MOE 2D	146	20 x 20	Toroidal	0
model 2a	2D-autocorrelation	88	15 x 15	Rectangular	7
model 2b	2D-autocorrelation	88	19 x 12	Rectangular	4
model 2c	2D-autocorrelation	88	20 x 20	Rectangular	9
model 2d	2D-autocorrelation	88	15 x 15	Toroidal	14
model 2e	2D-autocorrelation	88	19 x 12	Toroidal	7
model 2f	2D-autocorrelation	88	20 x 20	Toroidal	2
model 3a	3D-autocorrelation	96	15 x 15	Rectangular	1
model 3b	3D-autocorrelation	96	19 x 12	Rectangular	3
model 3c	3D-autocorrelation	96	20 x 20	Rectangular	5
model 3d	3D-autocorrelation	96	15 x 15	Toroidal	0

model 3e	3D-autocorrelation	96	19 x 12	Toroidal	7
model 3f	3D-autocorrelation	96	20 x 20	Toroidal	6
model 4a	VSURF	75	15 x 15	Rectangular	4
model 4b	VSURF	75	19 x 12	Rectangular	6
model 4c	VSURF	75	20 x 20	Rectangular	17
model 4d	VSURF	75	15 x 15	Toroidal	3
model 4e	VSURF	75	19 x 12	Toroidal	12
model 4f	VSURF	75	20 x 20	Toroidal	10
model 5a	VSA	32	15 x 15	Rectangular	10
model 5b	VSA	32	19 x 12	Rectangular	12
model 5c	VSA	32	20 x 20	Rectangular	5
model 5d	VSA	32	15 x 15	Toroidal	4
model 5e	VSA	32	19 x 12	Toroidal	6
model 5f	VSA	32	20 x 20	Toroidal	10
model 6a	MACCS	166	15 x 15	Rectangular	13
model 6b	MACCS	166	19 x 12	Rectangular	19
model 6c	MACCS	166	20 x 20	Rectangular	10
model 6d	MACCS	166	15 x 15	Toroidal	1
model 6e	MACCS	166	19 x 12	Toroidal	8
model 6f	MACCS	166	20 x 20	Toroidal	9
model 7a	Physicochemical	11	15 x 15	Rectangular	14
model 7b	Physicochemical	11	19 x 12	Rectangular	3
model 7c	Physicochemical	11	20 x 20	Rectangular	4
model 7d	Physicochemical	11	15 x 15	Toroidal	2
model 7e	Physicochemical	11	19 x 12	Toroidal	10
model 7f	Physicochemical	11	20 x 20	Toroidal	1

Results and Discussion

Ligand-based virtual screening was performed in two cycles based on the “similarity principle” and unsupervised machine learning methodology. The most active SSRIs from the earlier apoptosis experiments, sertraline and paroxetine were used as the starting point for both the screenings. The whole screening process is illustrated in Figure 3.4.

1. In a first cycle of virtual screening, the database of in-house compounds was virtually screened by the ROCS v3.1.2 program, a fast 3D shape comparison program using sertraline and paroxetine as reference compounds. The ComboScore, sum of ShapeTanimoto score and ScaledColor score representing shape and chemical feature overlay respectively, was employed to rank the database compounds. The top 10 database compounds with high ComboScore were selected for each reference compounds. Four hits

from each of the hit-list of reference compounds (compound 1-4 for sertraline and compound 5-8 for paroxetine, Table 3.5) were selected. These retrieved 8 compounds were tested using 3H-Thymidine incorporation assay for HEK293 cells killing (apoptosis) and EC_{50} values were determined. All the compounds exhibit the phenomenon of programmed cell death at various concentrations. The best hits (REM25 and REM14) have different scaffold architecture than their corresponding reference compound, sertraline indicating successful scaffold-hops using ROCS. REM25 found to be more potent (EC_{50}= 8.81μM) than the reference compounds (Sertraline EC_{50} = 9μM and paroxetine EC_{50} = 11μM). Furthermore, qualitative structure-activity relationship was performed to assess importance of sulfur, hydroxyl, propyl substituents and size of the linkage within the same series. Additional 5 compounds (9-13, Table 3.5) were selected and screened using 3H-Thymidine incorporation assay.

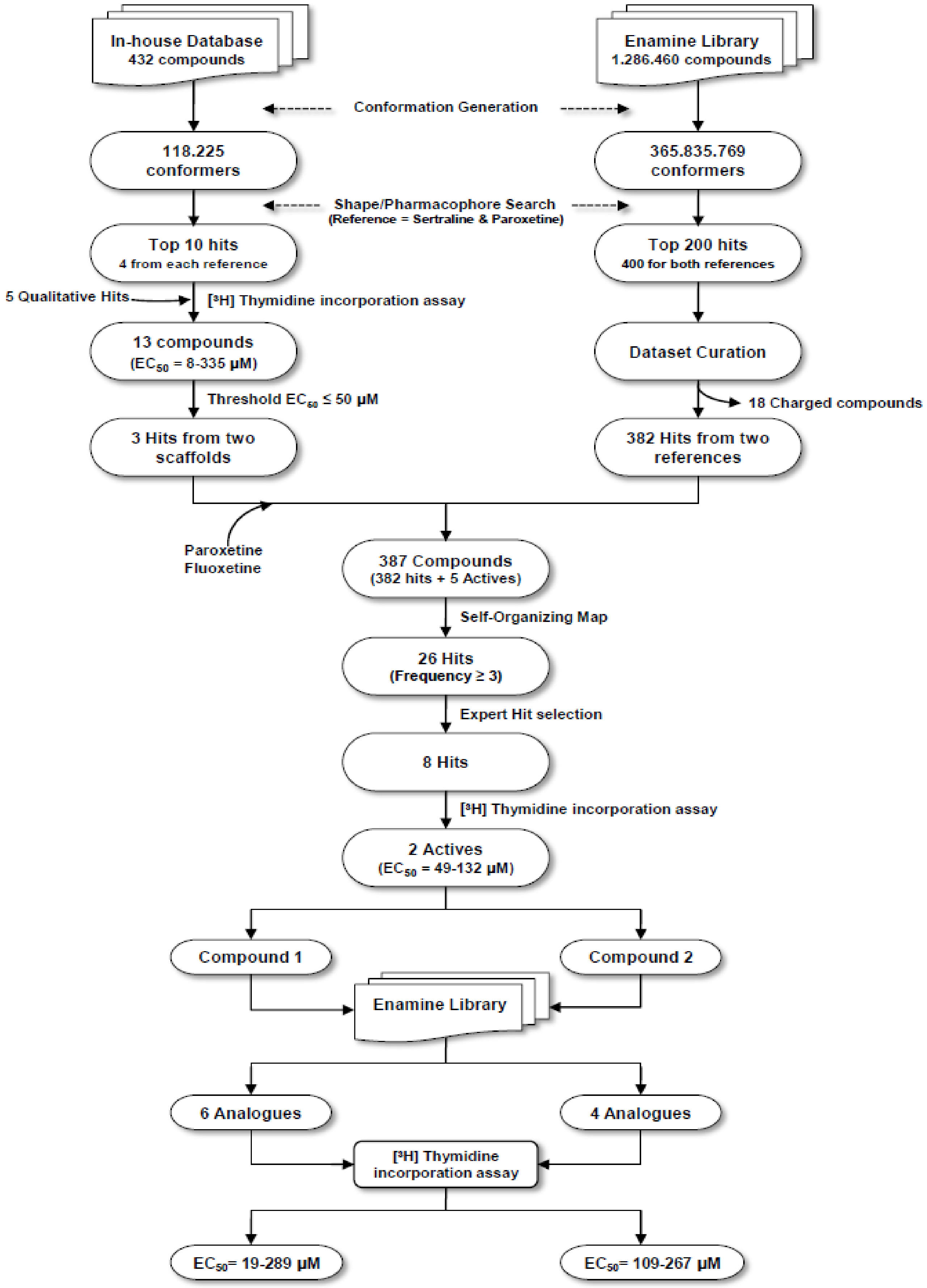
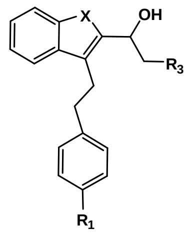
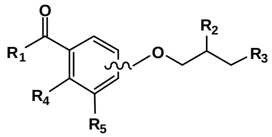
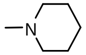
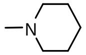
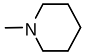
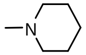
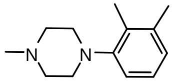
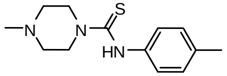
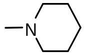


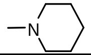
Figure 3.4: Overall workflow of the screening process

Table 3.5: Compounds used in screening round 1

A


B


Compound	Scaffold	Position	R$_1$	R$_4$	R$_5$	R$_2$ =Scaffold A X = Scaffold B	R$_3$	EC$_{50}$ (μM)
1 (REM25)	A	-	Me	-	-	S	NH(*i*-Pr)	8.81
2 (REM14)	A	-	H	-	-	S	NH(*i*-Pr)	15.23
3 (REC2219)	B	ortho	CH$_2$CH$_2$Ph	H	H	H	NH(CH$_3$)	129.5
4 (GPV0442)	B	meta	Ph	H	-CH3	-OH	NH(*n*-Pr)	120.4
5 (GPV0865)	B	para	Et	H	H	-OH		260.3
6 (GPV0512)	B	meta	R1-R4 = -CH$_2$-CH$_2$		H	-OH		281.5
7(GPV0389)	B	meta	Me	H	H	-OH		306.8
8 (GPV0385)	B	para	Me	H	H	-OH		335.2
9 (GE68)	A	-	H	-	-	O	NH(*n*-Pr)	25.18
10 (GPV0896)	B	meta	Ph	H	H	-OH		36.31
11 (GPV0825)	B	meta	Ph	H	H	-OH		55.07
12 (GPV0189)	B	meta	CH$_2$CH$_2$Ph	H	H	H		65.47

13 (GPV0186)	B	meta	CH_2CH_2Ph	H	H	H		70.54

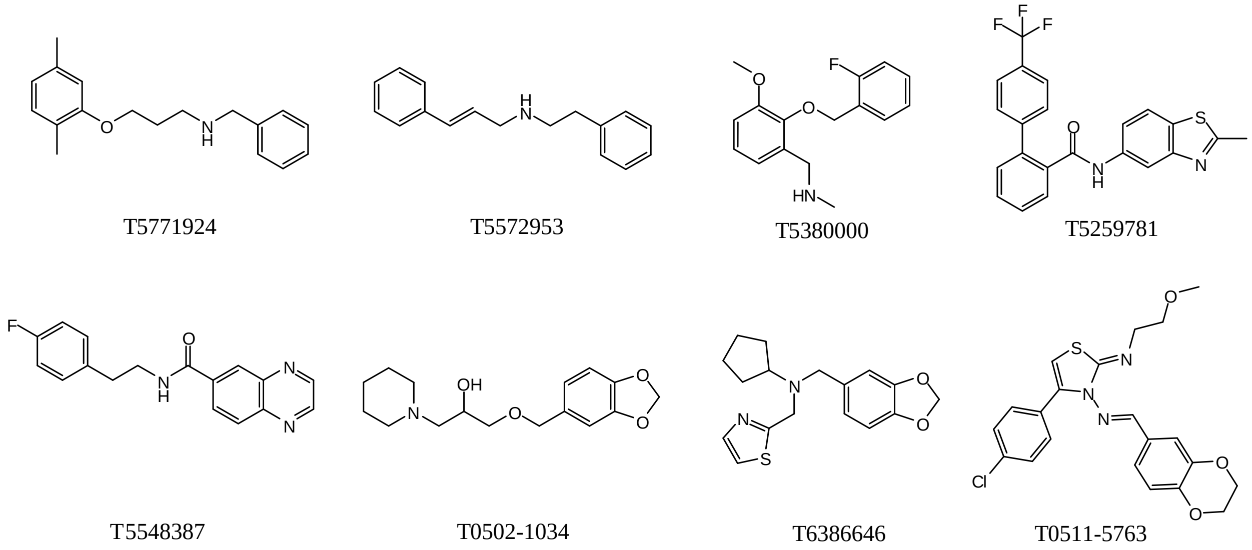
2. In the second cycle, the same two reference compounds (sertraline and paroxetine) were used in ROCS to screen Enamine database. The same virtual screening approach as implemented in step 1 was used except that the top ranked 200 compounds were selected from each query. 18 hits out of 400 hits contained permanent charges and were thus removed from further studies. For identification of new lead compounds, 382 hits were merged with a small subset of actives (SSRIs such as paroxetine, fluoxetine and hits retrieved from in-house database screening: REM25, REM14 and GPV896) and subjected to the processing with self-organizing maps (SOM). The idea behind this approach was to analyze the hits co-localizing with active compounds leading to identification of highly active hits [212]. The characteristic phenomenon demonstrated by SOM is its ability to depict a multidimensional space into 2D form by agglomerating compounds with similar properties. Hits from the 42 models were collected and those with co-localization frequency ≥ 3 were retrieved giving rise to a set of 26 compounds (Table 3.6). Finally, eight of out of 26 compounds were cherry-picked based on the frequency of co-localization, corresponding actives, types of descriptors and scaffold resemblance (figure 3.5). When pharmacologically tested, two out of these eight compounds, T5771924 (EC_{50}=49μM) and T5572953 (EC_{50}=132μM) were shown to be active in [^{3}H]thymidine incorporation assay (Figure 3.6a and Figure 3.6b).

Table 3.6: Hits found at least 3 times in all 42 SOMs

Compound	Corresponding co-localized actives	Descriptor types	Frequency
T5771924_S	Flu(8)	m, 11	8
T5380000_S	Flu(5), Par(1)	vu, v, m, 11	6
T5984498_S	Flu(5)	2a,3a,m	5
T6451571_P	Par(5)	vu, v, m	5
T5259781_P	GPV(3), Rem14-25	2a, 2m, v, 11	5
T5379981_S	Flu(5)	m, 11	5
T5380001_S	Flu(5)	m, 11	5
T5380002_S	Flu(5)	m, 11	5
T5532533_P	Par(4), Rem14	2a, v	4
T5546523_P	GPV(4)	3a	4
T5833735_P	Par(4)	2a, m	4
T6412799_S	Flu(4)	vu, m	4
T0502-1769_P	Par(3), GPV	v, m, 11	4
T6386646_P	Par(2), Rem14-25	2a, v	4

T0502-1034_P	Par(2), GPV	v, m, 11	3
T0502-1719_P	Flu, Par, GPV	vu, v, m	3
T0502-1749_P	Par(2), GPV	v, m	3
T0511-5763_P	GPV(3)	vu	3
T5247275_P	Flu, GPV, Rem25	2a, v, 11	3
T5438183_P	Par(2), Flu	2a, 3a, m	3
T5548387_P	Par, GPV, Rem14-25	3a, vu, m	3
T5572953_S	Flu, Rem14-25	m	3
T5822995_S	Flu(3)	vu	3
T5984503_S	Flu(3)	2a, m	3
T6174731_S	Flu(3)	m	3
T6486556_P	Par, GPV, Rem25	2a, 2m, m	3

Note: *_S, _P: in Compound column represent the corresponding reference compound (sertraline or paroxetine) from which this hit was identified. Corresponding co-localized actives: Flu- fluoxetine, Par- paroxetine, GPV – GPV896, Rem14-25 – both Rem14 and Rem25. Descriptor types: m- MACCS fingerprints, 11 – 11 Physicochemical properties, vu- VSURF descriptors, v- VSA descriptors, 2m- 2DMOE descriptors, 2a- 2D autocorrelation descriptors, 3a- 3D autocorrelation descriptors. The compounds highlighted with grey color were ordered from Enamine Ltd. for pharmacological screening.*



T5771924

T5572953

T5380000

T5259781

T5548387

T0502-1034

T6386646

T0511-5763

Figure 3.5: Structures of the 8 compounds ordered and pharmacologically tested using [^{3}H]thymidine incorporation assay

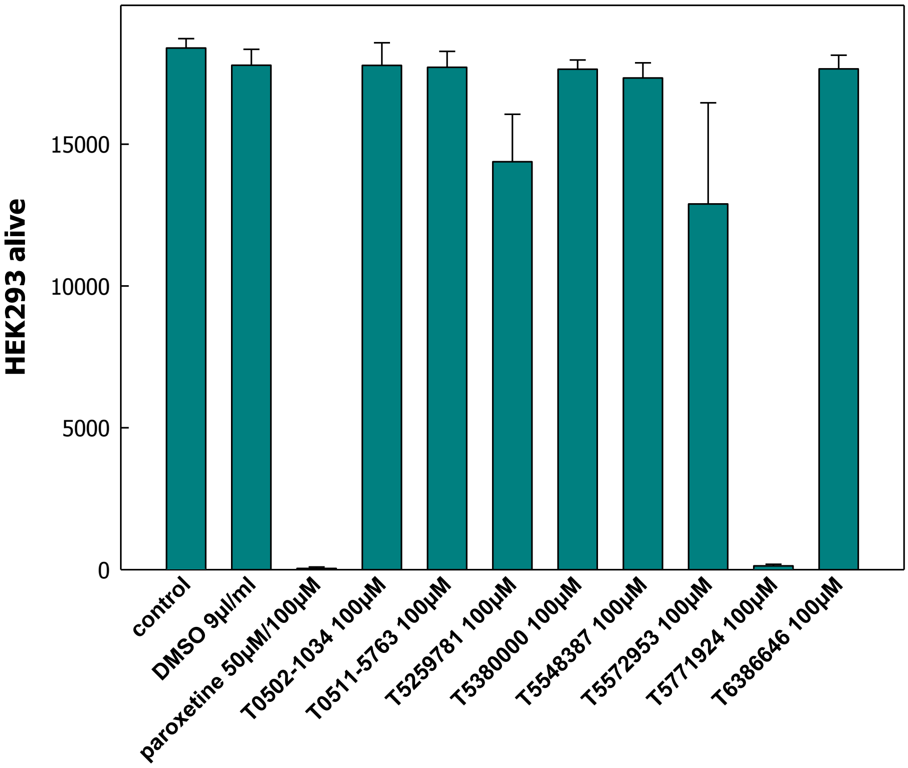


Figure 3.6a: Population of live HEK293 cells analyzed by flow cytometry after treatment with 8 compounds ordered from Enamine Ltd. Paroxetine was used as standard compound. Compounds, T5771924 and T5572953 show good activity

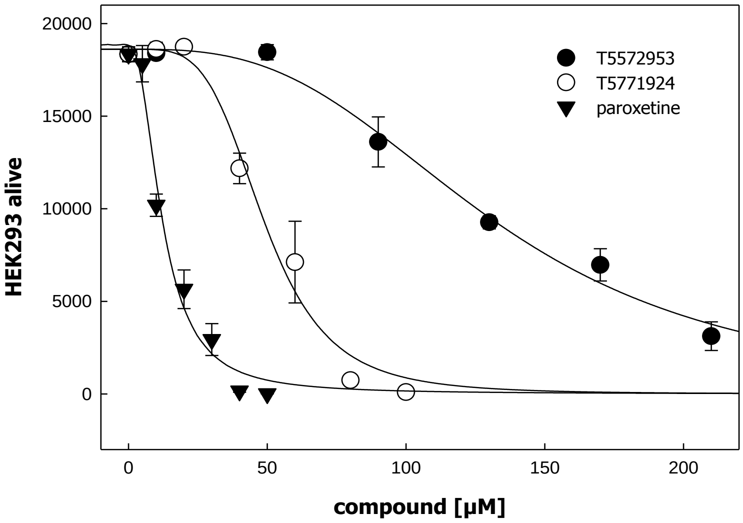


Figure 3.6b: Effect of paroxetine and two hits from Enamine on viability of HEK293 cells 24 h after exposure to drug.

As a follow-up study, small subsets of structural analogs for the two hits **T5771924** (compounds 1-6 in Table 3.7) and **T5572953** (compounds 7-10 in Table 3.7) were selected from the Enamine database and biologically tested (Figure 3.7). In the case of the quite active T5771924, two out of six compounds tested showed pharmacological activity below 100μM (Table 3.7). In case of the weak active T5572953, none of the four compounds tested showed significant pharmacological activity. These results confirmed that indeed new scaffolds can be identified by this approach

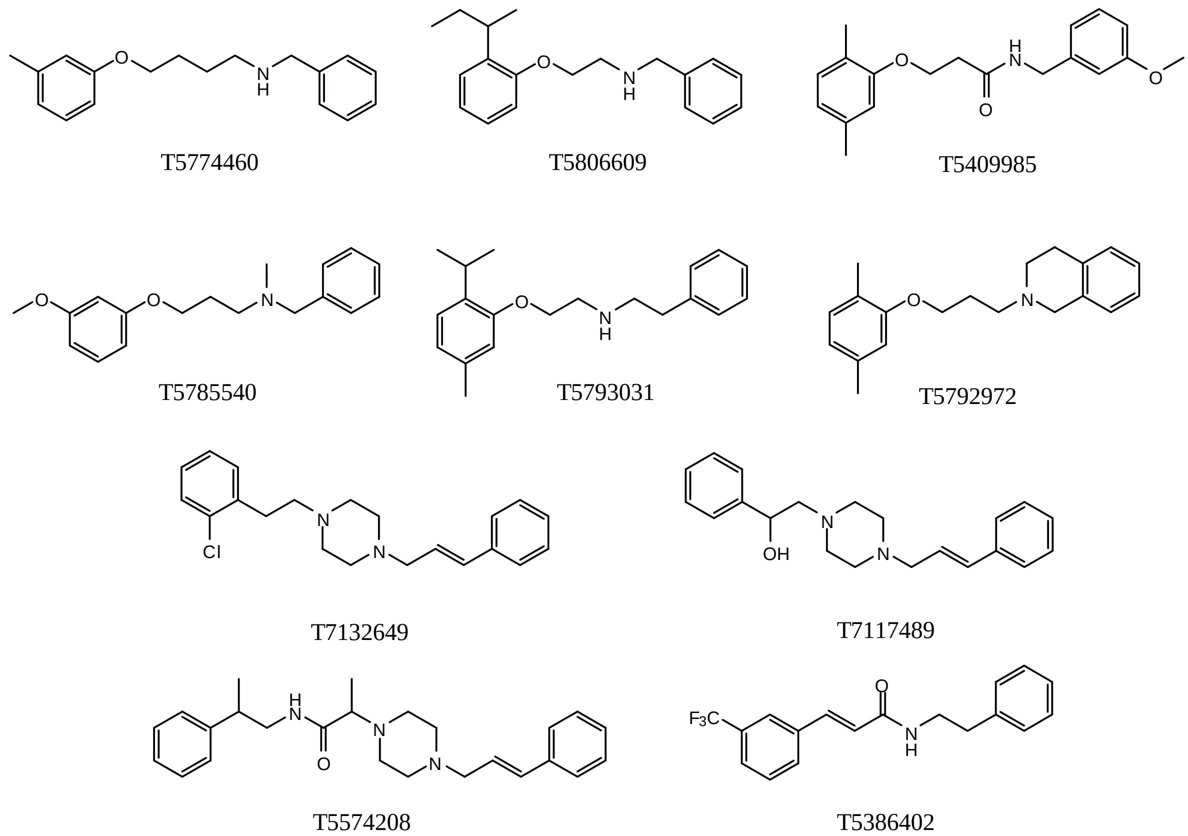


Figure 3.7: 10 compounds (follow-up) ordered and tested using [^{3}H]thymidine incorporation assay

Table 3.7: Compounds ordered in follow-up study

#	Compound	EC50 (μM)
1.	T5793031	19
2.	T5806609	64
3.	T5774460	106
4.	T5792972	161
5.	T5409985	215
6.	T5785540	289
7.	T7132649	109
8.	T5574208	170
9.	T5386402	253
10.	T7117489	267

A multi-step ligand-based virtual screening approach was implemented to discover seven and three novel hits ($EC_{50} \leq 100\mu M$) both from our in-house chemical library and from the

Enamine database, respectively. The preliminary Enamine database screening yielded two hits (T5771924 and T5572953) from a total of eight tested compounds, implying a hit rate of 25%. Both these hits were imparting vacuole formation in the cells revealing their role in apoptosis. Subsequent hit follow-up process considered the structural analogues of both hits and six and four compounds respectively (Figure 3.7) were ordered and pharmacologically tested. This study provided very potent hit (T5793031) which will be used for further lead optimization studies.

Conclusions

A virtual screening protocol for identification of novel SSRI like compounds has been presented. The two-step strategy consisting of shape and pharmacophore based similarity screening followed by self-organizing maps was employed. By utilizing this methodology, several active compounds were identified which represented novel scaffolds than SSRIs. The discovery of REM25 and REM14 is a proof-of-concept for the directed inclusion of benzothiophene scaffolds to mimic programmed cell death activity in the search of potential candidates.

3.3 Imbalanced Data Classification Problem

Now-a-days, tremendous amount of data regarding the effect of small molecules on the biological systems is becoming available via public archives such as PubChem, ChEMBL, DrugBank, NCI, etc. General drawbacks with such pharmaceutical, biological and medical data are small number of observations/data points, incomplete and missing data, or most importantly, unbalanced class distribution [213]. A small example is provided in Table 3.8 where the imbalanced ratio of actives and inactives for several important targets from PubChem is presented. Also in our case, the data set used is extremely imbalanced with e.g. 776 substrates of P-gp and 36,064 non-substrates. The highly skewed class distribution usually creates difficulties in the binary data analysis where the data with non-desired property (majority class or negatives samples) always exceed the data with desired property (minority class or positive samples) by a significant amount. Most of the machine learning methods and standard classifiers do not take this data distribution into consideration and focus either on minimizing global quantities such as error rate or increasing accuracy, thus maximizing overall performance of the resulting models [214]. In other words, they are inclined towards the majority class and thus usually showing substandard performance on the minority class (lack of generality). If we consider a simple example where the majority class contains 99% of the data and the remaining 1% belongs to the minority class, any conventional classifier predicting all entities as belonging to the majority class still gives 99% accuracy, thus producing quite misleading results.

A number of solutions have been proposed to deal with the class-imbalance problems, mainly comprising sampling and cost-sensitive learning approaches [215, 216]. The sampling techniques handle only the training data related issues whereas cost-sensitive methods look up into the algorithms and their modifications to tackle the imbalance. Random oversampling and undersampling are the non-heuristic methods that aim to balance class distribution so as to equalize the composition of the dataset without any specified rules. Random removal of majority class examples during undersampling lead to loss of valuable information from the data whereas replicating the minority class data randomly in oversampling might lead to overfitting as multiple instances of certain examples become tied to each other [215].

Table 3.8: Class imbalance of actives and inactives for some important targets in PubChem

#	PubChem AID	Target	Number of Compounds		Active : Inactive (Imbalance Ratio)	% Actives in whole dataset
			Active	Inactive		
1	602	ABCB1	35	85,167	1 : 2433.34	0.04
2	1326	ABCB1	130	193,674	1 : 1489.80	0.07
3	1325	ABCG2	200	194,393	1 : 971.97	0.10
4	807	ABCC1	10	9522	1 : 952.20	0.10
5	601	ABCB1	12	9977	1 : 831.42	0.12
6	2676	RXFP1	1084	357,384	1 : 329.69	0.30
7	2648	hERG	1083	304,532	1 : 281.19	0.35
8	2247	TRPC4	1189	302,815	1 : 254.68	0.39
9	1511	hERG	1552	304,061	1 : 195.92	0.51
10	567	5HT1a	366	64,541	1 : 176.34	0.56
11	799	ABCC1	843	137,879	1 : 163.56	0.61
12	612	5HT1a	416	61,189	1 : 147.09	0.68
13	1024	CYP2C9	1368	95,860	1 : 70.07	1.41
14	844	ABCC1	42	761	1 : 18.12	5.23
15	1451	ABCB1	18	255	1 : 14.17	6.59
16	741	ABCB1	29	267	1 : 9.21	9.80
17	397743	hERG	15	113	1 : 7.53	11.72
18	778	CYP2C19	20295	95,899	1 : 4.73	17.47
19	434978	TRPC4	541	1726	1 : 3.19	23.86
20	1835	hERG	749	1656	1 : 2.21	31.14

In case of directed/informed sampling, no new examples are created but the choice of samples to be replaced or eliminated is informed rather than random. It also includes synthetic sampling with data generation, sampling with data cleaning, editing or clustering techniques, and combinations of the above techniques [217].

Synthetic minority oversampling technique (SMOTE) is a very effective method for handling imbalanced data in various applications [218]. The SMOTE algorithm creates new artificial instances rather than merely replicating the existing ones of the minority class (as in random oversampling). For each positive instance, its nearest positive neighbors were identified based on Euclidean distance and new positive instances were created and placed randomly in between the instance and its neighbors.

Cost sensitive learning methods [219] consider the misclassification costs and target the imbalanced learning problem by using different cost matrices describing costs of misclassifying particular data examples. A strong connection between cost sensitive learning and imbalanced data has been shown [220] and also proved its superiority over sampling methods in some specific imbalanced learning application domains [221, 222]. Thus, cost-sensitive techniques are seen as an important alternative to sampling methods for imbalanced learning domains.

Kernel-based learning is centered on the theories of statistical learning and Vapnik-Chervonekis dimensions [223]. Support Vector Machines (SVMs) are representative of it and give good classification results with imbalanced data [224] when combined with general sampling and ensemble techniques [225, 226]. It has been used for chemoinformatics data mining problem, suggesting that the assay data should be carefully selected from PubChem to avoid problems when using imbalanced data [227]. SVM also has been combined with maximum entropy methods for cost sensitive classification of highly imbalanced CYP450 data of drugs [228].

Although there are a lot of efforts focused on two-class imbalanced problems, multiclass imbalanced learning problems also exist and are of equal importance. Several approaches based on cost-sensitive boosting algorithm [229], min-max modular network to decompose multiclass imbalanced problem into series of small two-class subproblems [230], the ensemble knowledge for imbalance sample sets (eKISS) method [231] are some of the few to tackle the multiclass imbalanced learning problems.

3.3.1 Prediction of P-Glycoprotein Substrates and Non-Substrates from Highly Imbalanced Dataset using Cost-Sensitive Machine Learning Methods

Introduction

P-glycoprotein (P-gp/ABCB1) is encoded by the highly conserved multidrug resistant (MDR) type 1 genes [232, 233]. It plays an important physiological role in the protection of cells and tissues from harmful xenobiotics (including drugs and chemicals) by extruding them out of cells. However, in the context of the cancer cells, this protective mechanism becomes destructive, leading to the expulsion of a wide variety of structurally and functionally diverse cytotoxic drugs from tumor cells and this phenomenon called as "Multi Drug Resistance" (MDR), which is one of the major reason for failure of chemotherapy to the cancer treatment [234]. Several strategies including development of P-gp inhibitors have emerged initially to tackle the problem of MDR in the cancer treatment [235]. However, because of the regulatory role of ABCB1 transporters in pharmacokinetics, toxicity and drug-drug interactions, it is more advised to identify and characterize the substrates in early phase of drug discovery and development process [236-240].

For this reason, various in vivo and in vitro screening assays have been developed to classify P-glycoprotein substrates [241]. However, these assay procedures are expensive, laborious and time consuming, which results in use of smaller datasets by them. This leads to the need of *in silico* methods, which can handle large amount of data and provide rapid and efficient screening for P-gp substrates and inhibitors [242]. For this purposes, several QSARs [243, 244], rule-based models [245] and pharmacophore models [203, 246, 247] have developed. Very recently various machine learning methods have also been successfully applied for early prediction of ADMET properties in particular prediction of P-gp substrates and inhibitors [248-250].

---

* - This section should be seen as a manuscript to be submitted in 'Journal of Chemical Information and Modeling'.

In the present study, we have used about 7000 compounds to build models for P-gp substrates and non-substrates using various machine learning methods such as support vector machine, random forest, kappa nearest neighbor, decision tree and naïve bayes for a test set containing about 29500 compounds. Predictive classification models were developed from highly imbalanced dataset (2% of substrates and 98% of non-substrates) which often gives poor or biased results when conventional machine learning methods were used. Several feature selection techniques were implemented and analyzed. In order to further validate our models, all the developed models were applied to an additional external dataset of 322 drug compounds. In addition, an applicability domain experiment was performed using a descriptor range approach. All these rigorous and time-consuming calculations were performed on our grid-platform. This is to our knowledge the first time that the information from such a highly imbalanced dataset is used to generate and validate *in silico* models to predict P-gp substrates and non-substrates.

Computational Methods

Dataset Preprocess and Curation

As shown in the Figure 3.8, a set of 47622 compounds was retrieved from the NCI's DTP tumor cell line data [251]. We found that 6687 compounds have same GI_{50} values across available NCI-60 cell lines, which leads to standard-deviation zero, thus giving invalid PCC. Chemical structures were searched for the remaining 40935 compounds in drug information system of DTP, which resulted in retrieval of a set of 40375 compounds and the remaining 560 compounds were manually searched in PubChem [185]. Out of 560 compounds, only 220 compounds structure were obtained as the remaining 340 compounds could not been represented by structural features, i.e. they were proteins, polymers, extracts or interleukins. From the 40595 remaining compounds, the counter ions, salts, mixtures and racemates were removed by using MOE software (Molecular Operating Environment) [200]. Filter program [252] from Openeye software Inc. was then used to remove compounds having other than H, C, N, O, F, P, S, Cl, Br, I elements. Subsequently, structures were preprocessed in Standardizer module from ChemAxon Ltd. [253] to have uniform representations. In this step, stereochemistry was cleared alongwith neutralizing and mesomerizing the structures leading to final dataset of 36840 compounds.

Since P-gp extrudes the molecules from the cell, it has been shown [179, 181, 182] that the compounds showing negative correlation between transporter expression and their activity/cytotoxicity across the 60 tumor cell lines are expected to be transported by the transporter. Compounds with PCC less than -0.3 (≤ -0.3) were assigned as substrates whereas compounds with >-0.3 were assigned as non-substrates. From the dataset of 36840 compounds, 776 compounds were assigned as substrates and the remaining 36064 compounds as non-substrates. In addition to this, other activity thresholds were also explored.

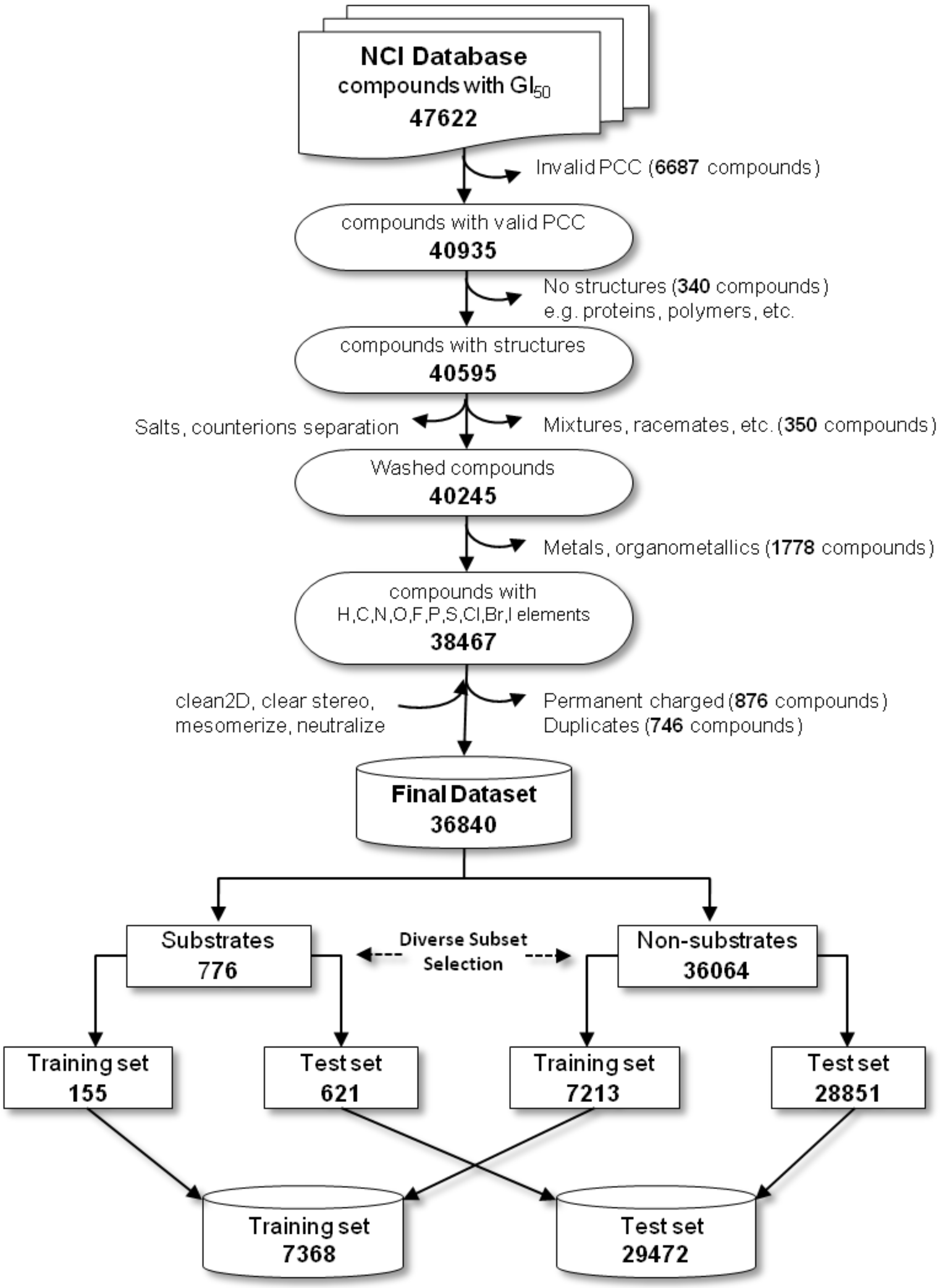


Figure 3.8: Flowchart of Dataset Preprocess

Molecular Descriptors

In this study, three different classes of molecular descriptors were computed from the 2D structures of the compounds in MOE software [200]. A set of 152 2-dimensional ('2DMOE', in short) descriptors was calculated. These numerical molecular features were derived from the connection table representing a molecule which includes the physical properties, atom and bond counts, adjacency and distance matrix descriptors containing BCUT and GCUT descriptors, Kier & Hall connectivity, kappa shape indices, subdivided surface areas, partial charges and pharmacophoric features. In addition, a set of 32 VSA (Van der Waals Surface Area) descriptors was calculated. VSA descriptors focus on incremental surface area for logP, molar refractivity and partial charge properties. A set of 166 MACCS structural keys [254] was computed from MOE. Each "key" represents a small substructure consisting of about one to ten non-hydrogen atoms and enumerates very simple features, but when used in combination can prove to be very specific and useful in distinguishing the characteristics among the small molecules.

Selection of Training and Test set

All the computed descriptors and fingerprints were merged with binary biological activity (1 for substrates, 0 for non-substrates) of 36840 compounds. As mentioned in the previous section, final dataset has an imbalance ratio of 1/45 (substrate/non-substrate) i.e. 2 % substrates and 98% of non-substrates. To avoid the overfitting and to improve generalization of the models, the dataset was divided into a training (n=7368; 155 substrates and 7213 non-substrates) and a test set (n=29472; 621 substrates and 28851 non-substrates) using MOE's diverse subset selection approach. This process starts with selection of reference compound and then the compounds most structurally different from each other are identified using generally MACCS/Tanimoto similarity till the desired number of diverse compounds is chosen.

Machine Learning Methods and Feature Selection

Various commonly used machine-learning methods such as support vector machine (SVM), decision tree (DT), kappa nearest neighbor (kNN), random forest (RF) and Naïve bayes (NB) were applied. A detailed account of the theory behind these methods can be found elsewhere [156, 255]. These methods are based on the different concept and representative of most of classification methods. All the classification models were constructed using Weka data mining software [256], which provides a set of classification and regression methods, variable selection methods. For the high-dimensional imbalanced datasets, it is very important to select features, which distinguish between the two classes and can capture the high skewness in the class distribution. In the present study, various automatic feature selection procedures were applied such as CfsSubsetEval (correlation-based feature subset selection evaluator) with BestFirst search method and feature ranking methods viz Chisquared, InfoGain, ReliefF and GainRatio methods with Ranker search method. All feature selection was performed in Weka data mining software [256].

Cost-Sensitive Bagging Approach (CSB)

Most classifiers assume that the active and inactive compounds are evenly distributed in the dataset with equal misclassification costs. They are designed with the goal to maximize the accuracy (or to minimize the error rate). This in turn led to classification of all instances as negative (Minority class or positive class predicted as negative class) in a highly imbalanced dataset. In order to solve this imbalanced problem, many approaches have been proposed [215], which includes sampling techniques and cost-sensitive learning approaches (CS). Cost-sensitive learning considers misclassification error and assigns high cost to the misclassification of the minority class. It has recently being used in machine learning classification problem, where the class distributions in the dataset are highly imbalanced [228].

CS learning was first introduced by Elkan [219] and Zadrozny [257] and later it was simplified by Ling et al [258]. In brief, CS demonstrates the importance of misclassification cost in the different cost-sensitive algorithms. For example, a simple binary cost matrix can be represented as shown in Table 3.9. Misclassification of actual positive instance into negative instance is much more expensive than an actual negative instance classified into positive instance, i.e. value of C (0,1) or FN > value of C(1,0) or FP. For example, from the pharmaceutical point of view, P-gp substrate is the positive class (minority) and non-substrate is the negative (majority) class. Failing to correctly predict substrates is a more serious problem than predicting non-substrates as substrates (FP). P-gp substrates being predicted as non-substrates would cause serious problem later in the drug development process.

Table 3.9: Cost matrix for two-class classification

	Actual Positive	Actual Negative
Predicted Positive	C (1,1) or TP	C (1,0) or FP
Predicted Negative	C (0,1) or FN	C (0,0) or TN

Note: C (i, j) represents the cost of incorrectly labeling an instance from its actual class j into a predicted class I. Here 1 and 0 denote Positive and negative classes respectively.

Assume that x is the compound under investigation and this compound is classified either as active or inactive by probability. In the cost matrix, this compound is classified to a class having low expected cost and this can be expressed as follows

$$R(i|x) = \sum_j P(j|x).C(i,j) \qquad \text{(Eqn. 1)}$$

where $R(i|x)$ is the expected cost of compound x classified into class i (predicted), $P(j|x)$ is the probability of compound x to be classified into class j (actual). Compound x is classified as active by classifier, if expected cost of compound x classified into active is smaller than expected cost of compound x into inactive class. i.e. $P(0|x)C(1,0)+P(1|x)C(0,0) \leq P(0|x)C(0,0) +P(1|x)C(0,1)$.

Probability of compound x classified as inactive can be written as $P(0|x)=1-P(1|x)$. Compound x is classified as active by the classifier, if $P(1|x)$ is equal or greater than the threshold p^*. Threshold p^*can be calculated from the misclassification of active (FN) and inactive (FP) as shown in Eqn. 3.

$$P^* = \frac{C(1,0)}{C(1,0)+ C(0,1)} = \frac{FP}{FP+FN} \qquad \text{(Eqn. 2)}$$

In addition to the cost-sensitive learning method, we used the bagging algorithm as an ensemble classifier [259, 260], as it is implemented in the Weka data mining tool [256]. Bagging (bootstrap aggregating) [260] creates an ensemble of classifiers where the same size of new training sets are created (called as "bags") by sampling with replacement from the original training data either by deletion or by replication of compounds. This sampling procedure is repeated several times and bagging chooses those prediction class where the majority of classifiers having most votes.

Model Evaluation and Performance Measurement

All the constructed models were evaluated in terms of predictability of test set and n-fold cross validation, for clarity only 5-fold cross validation and test set predictions are provided. Most often model performance is measured using overall accuracy and error rate of the model, however, these measurement perform poorly when learning from imbalanced datasets. In order to overcome this problem, we used performance measures which effectively address the imbalanced learning problems, such as sensitivity (true positive rate), specificity (true negative rate), G-mean, precision, recall, accuracy and Matthews correlation coefficient (MCC).

Sensitivity (TPR) or Recall = $\frac{\mathrm{TP}}{\mathrm{TP+FN}}$ (Eqn. 3)

Specificity (TNR) = $\frac{\mathrm{TN}}{\mathrm{TN+FP}}$ (Eqn. 4)

Geometric-mean (G-mean) = $\sqrt{Sensitivity\ .Specificity}$ (Eqn. 5)

Precision = $\frac{\mathrm{TP}}{\mathrm{TP+FP}}$ (Eqn. 6)

Accuracy = $\frac{\mathrm{TP+TN}}{\mathrm{TP+FP+TN+\ FN}}$ (Eqn. 7)

Matthews correlation coefficient = $\frac{(TP\ .\ TN)\ -(FP\ .\ FN)}{\sqrt{(TP+FP).(TP+FN).(TN+FP).(TN+FN)}}$ (Eqn. 8)

Considering a basic two-class problem, all these parameters are calculated from true positive (TP), true negative (TN), false positive (FP) and false negative (FN). The geometric mean (G-mean) evaluates the degree of inductive bias in terms of a ratio of positive accuracy and negative accuracy. Precision can be seen as a measure of exactness or fidelity (i.e. correctly labeled positive examples among all positives) whereas recall (sensitivity) is a measure of completeness (i.e. correctly labeled positive examples among all positive class). Matthews's correlation coefficient (MCC) indicates the degree of the correlation between the actual and predicted classes. It ranges from -1 to +1 and is generally regarded as a good measure of the quality of the binary classification. In the machine learning classification study, MCC values above 0.4 are considered to be predictive [261].

In this study, we explored several feature selection approaches considering different base classifiers in combination with wide range of costs for cost-sensitive experiments. As our dataset is quite large and classification with weka requires much more time, large computational power was needed to tackle this situation. This led us to implement weka within our distributed computing grid environment, UVieCo [262].

Results and Discussion

Dataset Characterization and Physicochemical Properties of P-gp Substrates

The drug-likeness of a dataset compounds was accessed by calculating the Lipinski's "Rule-of-Five" descriptors. The result shows that 69% compounds in the dataset did not violate any of the four rules. Approximately 27.4% of the compounds in the dataset violate one or two rules and 120 compounds violate all four rules. Inspecting of these compounds revealed that there was no structural similarity among them which indicates that p-glycoprotein substrates are structurally diverse as mentioned in the introduction.

The dataset was divided into training and test set according to diverse subset selection method. This resulted in a training set of 7368 compounds and a test of 29472 compounds. A principal component analysis (PCA) was performed to check possible presence of clusters, outliers, similarities or dissimilarities. First two principal components explain 79% of variance in the data set. The score plot from PCA shows (Figure 3.9) that the diversity of the dataset is satisfactorily reflected in the training set and there are no distinct clusters in the dataset.

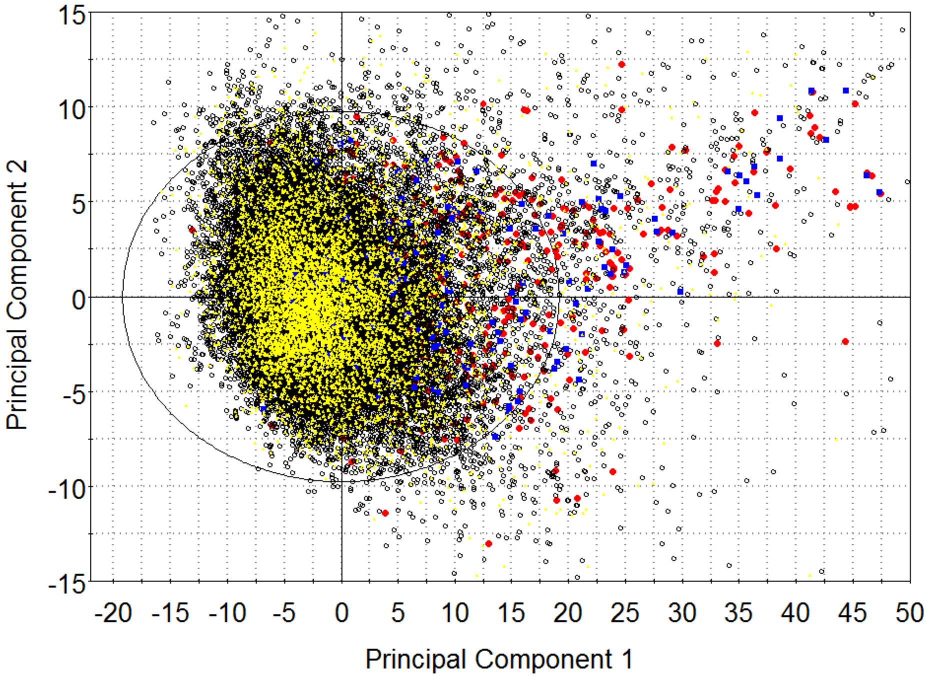


Figure 3.9: Score plot from principal component analysis of NCI dataset (first two principal components are shown). Compounds are colored as follows, Blue dot: Training active, Red dot: Test active; Black circle: Test inactive and yellow dot: Training inactive.

It was observed from the loading plot that most of the substrates are highly influenced by the topological polar surface area, molecular weight, hydrophobic parameters such as LogP(o/w), number of aromatic atoms and number of rings. This reveals that non-substrates generally seem less hydrophobic than substrates. This is also in good agreement with the general idea of P-glycoprotein substrates, which first penetrate into the hydrophobic membrane and then bind to P-gp. In addition to PCA, we analyzed the chemical space of substrates and non-substrates from a set of 36840 compounds using simple molecular properties such as molecular bulkiness (molecular weight, molar refractivity, number of rings, number of rotatable bonds and van der Waals volume), hydrogen bonding ability (total number of nitrogen and oxygen atoms), polarity (topological polar surface area) and hydrophobicity (logP(o/w)). It was observed that most of the chemical properties have relative impact in discriminating substrates from non-substrates (Figure 3.10). For instance, majority of the substrates have large TPSA, molecular weight, more hydrogen bonds and high molecular reflectivity.

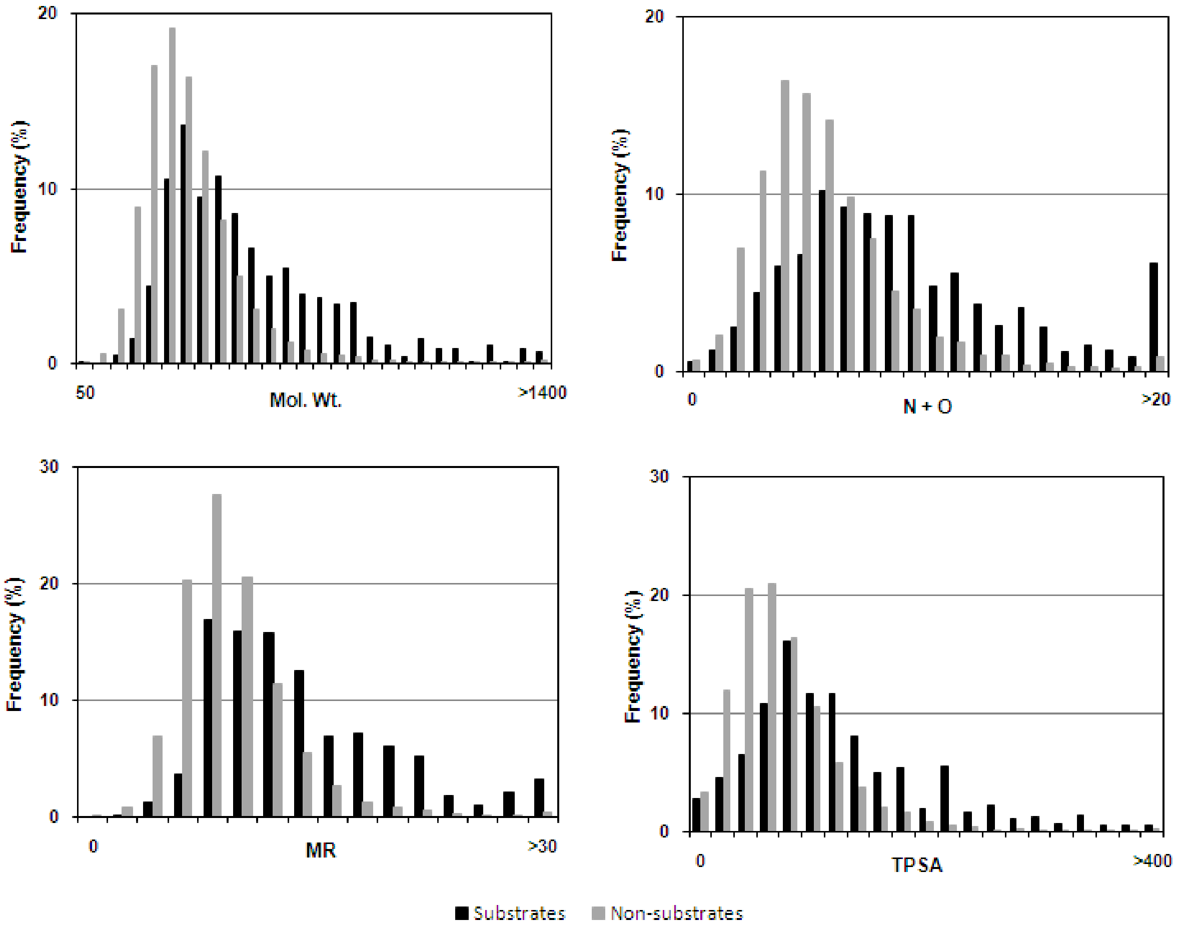


Figure 3.10: Properties distribution of P-gp substrates and non-substrates.

Construction of Classification Models by Machine Learning Methods

Various machine learning classification models for P-gp substrates and non-substrates were developed using 9 set of descriptors to check which combination of descriptors might lead to a good classification model. The model's performance was compared in terms of various statistical parameters of 5-fold cross-validation runs and a test set (Table 3.10), (for clarity only Matthew's correlation coefficient (MCC), geometric-mean and overall accuracy are given). In general, all 9 set of descriptors perform well except ReliefF descriptors. The overall accuracy for the test set ranges from 55-73%. Notably, the quality of the models was good when using all 350 descriptors and 152 2D-MOE descriptors compared to other set of descriptors. From the feature selection methods, Chisquared method performs better than other methods and provides comparable statistics to initial 2 descriptor sets. These three models have reasonably high sensitivity (75-88 %) and specificity (60-69%) and these models have the ability to correctly predict both substrates and non-substrates, which is reflected in MCC (0.44-0.50) and G-Mean (72-73).

Table 3.10: Comparison of model performance using various feature selection methods

Descriptors	5-fold CV				Test Set		
	N	MCC	Accuracy	G-Mean	MCC	Accuracy	G-Mean
ALL	350	0.34	0.67	0.67	0.50	0.74	0.73
2D-MOE	152	0.32	0.66	0.66	0.49	0.74	0.73
MACCS	166	0.06	0.53	0.53	0.29	0.62	0.55
VSA	32	0.31	0.66	0.66	0.41	0.70	0.70
CfsSubsetEval	12	0.28	0.64	0.64	0.43	0.71	0.70
Chisquared	12	0.31	0.66	0.66	0.44	0.72	0.72
GainRatio	12	0.31	0.65	0.65	0.42	0.71	0.71
Infogain	12	0.27	0.63	0.63	0.41	0.70	0.69
ReliefF	12	0.08	0.54	0.54	0.11	0.55	0.55

Note: N: Total number of descriptors used for model building; 5-fold CV: Five-fold cross validation; MCC: Matthew's correlation coefficient.

Although models developed using all 350 descriptors and 152 2D-MOE descriptors perform better than other models, the number of descriptors used in Chisquared models were several folds less than these two giving almost similar predictions. Therefore we considered 13 Chisquared descriptors as starting point for our classification model building and model optimization. List of 12 Chisquared descriptors is given in Table 3.11. These mainly constitute the simple descriptors such as count of hydrophobic atoms, carbon atoms, and heavy atoms. Similarly, the bond information between heavy atoms is also considered important in this study. Number of hydrogen donating atoms and relative positive partial charges along with descriptors representing van der Waals surface areas based on octanol/water distribution coefficient, partial charge and hydrophobic atoms show significant contribution. The importance of these properties towards P-gp specificity has been reported in previous observations [263, 264].

Table 3.11: List of 12 Chisquared descriptors used in present classification analysis

Chisquared Des.	Definition
a_heavy	Number of heavy atoms
a_hyd	Number of hydrophobic atoms
a_nC	Number of carbon atoms
b_heavy	Number of bonds between heavy atoms.
Kier1	First kappa shape index
lip_don	The number of OH and NH atoms
PEOE_RPC+	Relative positive partial charge: the largest positive partial charge divided by the sum of the positive partial charges (PEOE partial charges)
VAdjEq	Vertex adjacency information (equality): $-(1-f)\log_2(1-f) - f \log_2 f$ (where $f = (n^2-m)/n^2$, n is the number of heavy atoms and m is the number of heavy-heavy bonds)
VAdjMa	Vertex adjacency information (magnitude): $1 + \log_2 m$
vsa_hyd	Approximation to the sum of VDW surface areas of hydrophobic atoms (Å2)
PEOE_VSA-1	Sum of van der Waals surface area where partial charge is in the range [-0.10,-0.05)
SlogP_VSA0	Sum of van der Waals surface area such that the contribution to logP(o/w) is <= -0.40

Chisquared descriptors were used for machine learning classification using support vector machine (SVM), kappa nearest neighbor (kNN), naïve bayes (NB), random forest (RF) and decision tree (DT).

Table 3.12: Classification models for P-gp substrates and non-substrates using machine learning methods

Method	**Prediction (%)**				**Sensitivity**	**Specificity**	**G-mean**	**Precision**	**MCC**	**Accuracy**
	TP	**FP**	**TN**	**FN**						
RFa	1	0	100	99	0.01	1.00	0.11	0.95	0.07	0.51
DTa	0	0	100	100	0.00	1.00	0.00	0.00	-0.02	0.50
NBa	26	8	92	74	0.26	0.92	0.49	0.75	0.23	0.59
kNNa	6	2	98	94	0.06	0.98	0.25	0.74	0.10	0.52
SVMa	0	0	100	100	0.00	1.00	0.00	-	-	0.50
RFb	0	0	100	100	0.00	1.00	0.00	0.00	-0.01	0.50
DTb	0	0	100	100	0.00	1.00	0.00	-	-	0.50
NBb	51	29	71	49	0.51	0.71	0.60	0.64	0.22	0.61
kNNb	8	2	98	92	0.08	0.98	0.28	0.80	0.14	0.53
SVMb	0	0	100	100	0.00	1.00	0.00	-	-	0.50

Note: a) 5-fold cross validation prediction, b) Test set prediction, RF: Random forest, DT: Decision tree, NB: Naïve bayes, kNN: kappa nearest neighbor, SVM: Support vector machine, TP: True positive, TN: True negative, FP: False positive, FN: False negative, MCC: Matthews correlation coefficient.

Overall the quality of the models was poor, overall accuracy and MCC were below 65% and 0.25 (Table 3.12) respectively. However, correct prediction of non-substrates was quite high (specificity is >90%), which might be due to the fact that the dataset exhibits a high number of non-substrates, which dominate the classification model to a great extent. Here, the classifiers assume that the substrates and the non-substrates are distributed equally in the dataset, which leads to high substrate (minor class) misclassification rate and low non-substrate (major class) misclassification rate. All classification models poorly predicted the substrates in the test set, with an overall prediction of the best model (NB) of 61% and a sensitivity of 51%. This imbalanced prediction is also reflected with the poor value of G-mean (60%) and MCC (<0.40). All these observations argue us to develop cost-sensitive machine learning models, which consider misclassification error.

Cost-Sensitive Bagging Models

Cost-sensitive bagging (CSB) based machine learning classification methods were investigated. Costs (weights) for each method were optimized and selected a best classification model based on the MCC, G-Mean and overall accuracy. Overall, the different methods provide models of similar quality. From the 29472 compounds in the test set, 18034 to 21306 (64–74%) are correctly predicted in which 394 to 505 (63-81%) substrates and 17539 to 20832 (61-72%) non-substrates were predicted correctly. More importantly, sensitivity and specificity are 61-81% and MCC is >0.4 (Table 3.13). Support vector machine, random forest and naïve bayes methods give best models while other methods show reasonable performance. These methods significantly perform well in classifying substrates, 76%, 81% and 75% respectively. However, for non-substrates, SVM model achieves better prediction (72%) compared to other classifiers. Overall, RF and SVM correctly predict more than 72% of the test set and their performance is also reflected in high G-Mean (0.72-0.74), precision (0.71-0.73) and MCC (0.44-0.49). Five-fold cross-validation of the training set shows slightly less prediction compared to test set prediction. Here, NB and RF models perform significantly higher. The sensitivity (63-72%) and specificity (63-68%) are well in balance with overall accuracy is >65%.

Table 3.13: Cost-sensitive bagging-machine learning classification models

Method	Cost	Prediction (%)				Sensitivity	Specificity	G-mean	Precision	MCC	Accuracy
		TP	FP	TN	FN						
RFa	110	63	32	68	37	0.63	0.68	0.66	0.66	0.31	0.66
DTa	75	56	26	74	44	0.56	0.74	0.64	0.68	0.30	0.65
NBa	1000	72	37	63	28	0.72	0.63	0.67	0.66	0.35	0.67
kNNa	110	54	35	65	46	0.54	0.65	0.59	0.61	0.19	0.60
SVMa	48	52	24	76	48	0.52	0.76	0.63	0.68	0.29	0.64
RFb	110	75	31	69	25	0.75	0.69	0.72	0.71	0.44	0.72
DTb	75	70	33	67	30	0.70	0.67	0.68	0.68	0.37	0.68
NBb	1000	81	44	68	19	0.81	0.61	0. 70	0.67	0.43	0.71
kNNb	110	63	40	72	37	0.63	0.64	0.64	0.64	0.28	0.64
SVMb	48	76	31	81	24	0.76	0.72	0.74	0.73	0.49	0.74

The models for the test set were also compared using area under the receiver operating characteristic curve (ROC). As seen from Figure 3.11, SVM, RF and NB have highest enrichment values around 0.80.

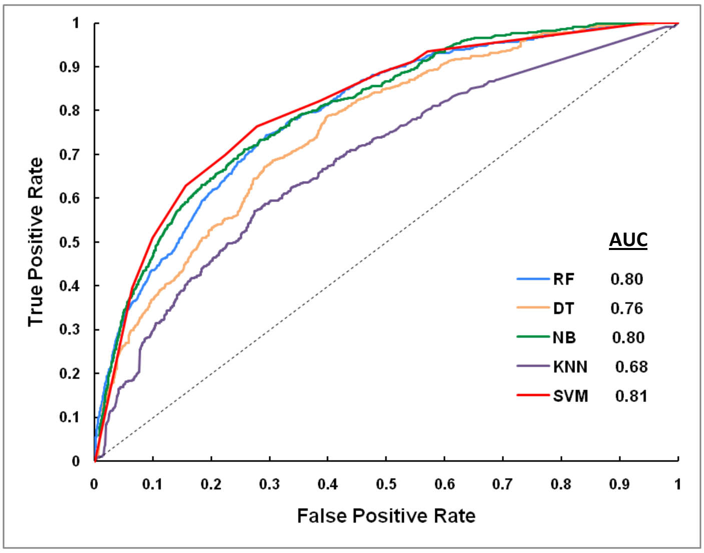


Figure 3.11: Area under the Receiver Operating Characteristic Curve (AUC) for test set

Furthermore, cost sensitive bagging models (CSB) were compared with non-cost-sensitive bagging models (Figure 3.12) and performance is evaluated in terms of G-Mean, MCC and overall accuracy. It is noted that cost-sensitive bagging models performs significantly higher than non-CSB model. Therefore, it can be suggested that cost-sensitive machine learning methods are suitable for highly imbalanced large dataset.

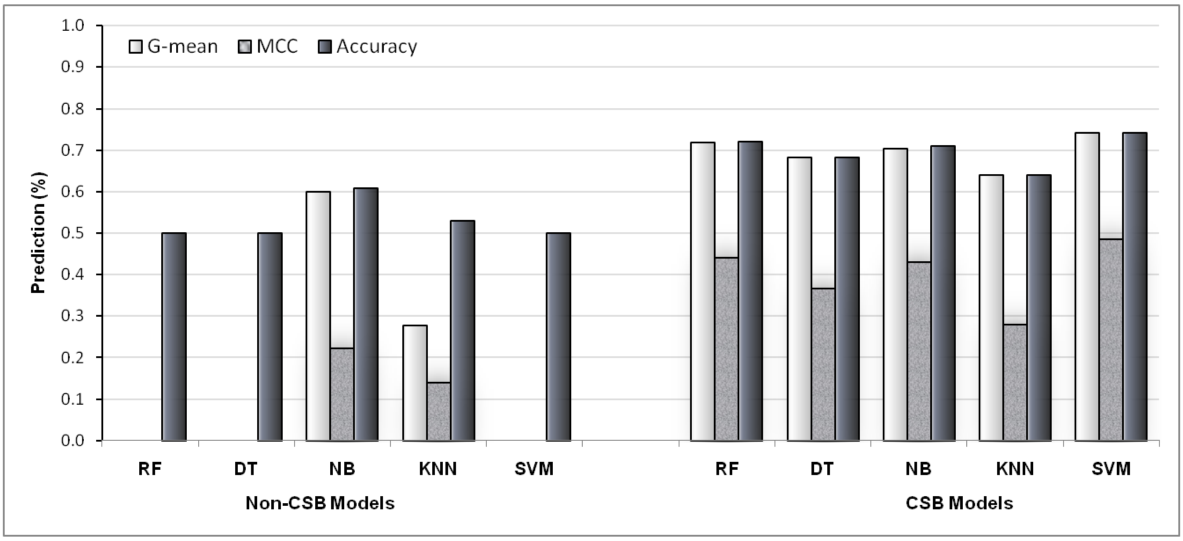


Figure 3.12: Comparison of cost-sensitive bagging (CSB) models with non-CSB model

Construction of Classification Models from Different Activity Thresholds

As mentioned in the method section, the classification was performed based on the PCC value, which correlates transporter mRNA expression and compound activity or cytotoxicity across the 60 cancer cell lines. The problem was to set the right PCC for substrates and non-substrates. According to Szakács et al. we assigned compounds with PCC ≤ -0.3 as substrates (negative correlation) and compounds with PCC >-0.3 were classified as non-substrates. An additional experiment was performed using different activity thresholds to check how the model changes. In order to do this, the whole data set was used for classification and 5- and 10- fold cross validations were performed. In total, six models were developed using CSB-RF (Results of three-activity thresholds of 5-fold and 10-fold cross validation are provided in Table 3.14). As seen from the table 3.14, predictions significantly improve when tightening the activity threshold for substrates i.e. from -0.25 to -0.40. Substrate prediction is substantially improved at threshold -0.40 (Model-3). Sensitivity and specificity are equally good (85%), which is also reflected in high MCC (0.70).

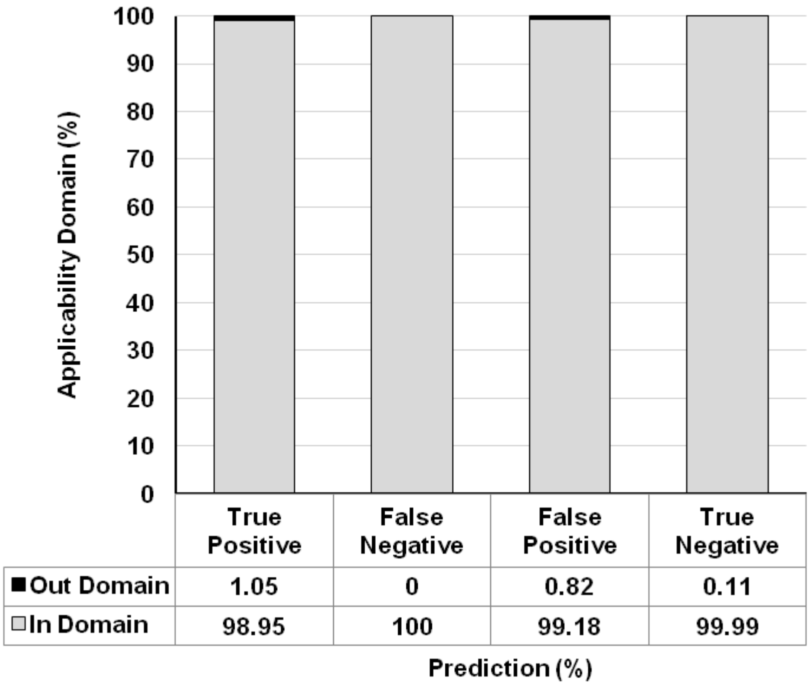
Table 3.14: Comparison of Random forest models (n-fold cross validation) with different activity threshold for P-gp substrates and non-substrates

Models	Prediction (%)					Sensiti-vity	Specifi city	G-mean	Precis ion	MCC	Accuracy
	Cost	TP	FN	TN	FP						
Model-1a)	62	72	27	73	28	0.72	0.73	0.73	0.45	0.73	0.73
Model-2a)	140	78	23	77	22	0.78	0.77	0.77	0.54	0.77	0.77
Model-3a)	580	83	19	81	17	0.83	0.81	0.82	0.64	0.82	0.82
Model-1b)	62	72	72	28	72	28	0.72	0.72	0.72	0.44	0.72
Model-2b)	140	78	76	24	76	24	0.76	0.76	0.76	0.53	0.76
Model-3b)	580	86	82	19	81	18	0.82	0.81	0.81	0.63	0.81

Note: Model 1a) & 1b) 10-fold and 5-fold cross validation with threshold ≤-0.25 (Substrate)/ >-0.25 (Non-substrate), Model 2a) & 2b) 10-fold and 5-fold cross validation with threshold ≤-0.30 (Substrate)/ >-0.30 (Non-substrate), Model 3a) & 3b) 10-fold cross validation with threshold ≤-0.40 (Substrate)/ >-0.40 (Nonsubstrate). TP: True positive, TN: True negative, FP: False positive, FN: False negative, BCR: Balanced classification rate, MCC: Matthews correlation coefficient

Applicability Domain Investigation

Applicability domain (AD) analysis has emerged as a hot-topic in recent ligand-based modeling such as quantitative structure activity/property relationships (QSARs/QSPRs), machine learning classification studies [265, 266]. AD is primarily used to check whether a given compound can be reliably predicted or not based on the chemical space of training set and this provides a first structural alert. Many theories have been proposed and each has their own pros and cons [265]. One of the well-known AD approach is "physicochemical descriptor ranges" which is estimated from the descriptors range of training compounds (preferably, descriptors from final model). These ranges will be used for applicability checking of the test set. In the present study, AD was estimated using Ambit Discovery v0.04 software [267]. Ambit Discovery preprocesses the given dataset by principal component analysis in order to eliminate co-linearities among the descriptors. Subsequently, AD is estimated based on a physicochemical descriptor range approach. The applicability domain for the test set is provided in Figure 3.13. As can be seen, the majority of the test set compounds (99.65%) are within the chemical space of the training set compounds and only a small fraction were out of domain (0.35%).



	True Positive	False Negative	False Positive	True Negative
■ Out Domain	1.05	0	0.82	0.11
□ In Domain	98.95	100	99.18	99.99

Figure 3.13: Applicability domain analysis of true and false predictions from SVM model for test set using 12 Chisquared descriptors

Analyzing these compounds, 5/474 true substrates (TP) and 98/8019 false substrates (FP) were out of the chemical space and only 0.11% of true non-substrates (32/20882) were out of the domain. To extent this experiment, we analyzed a set of 1252 FDA approved drugs from DrugBank and to check whether our CSB models can reliably predict also drug compounds. Surprisingly, more than 98% (1229/1252) of the drugs are within the chemical space of our training set. This investigation suggests that our classification model could be applicable to very diverse compounds.

Comparison with Available Models

It is not appropriate to directly compare our models with previously reported models, which have been developed with different datasets (size of training and test set) and assay protocol. All the developed models were compared with previously reported P-gp substrate and non-substrate classification models [243, 248, 268-271], which have overall accuracies of 63-90% and correctly predict substrate (sensitivity 53-96%) and non-substrate (specificity 67-89%) (Figure 3.14).

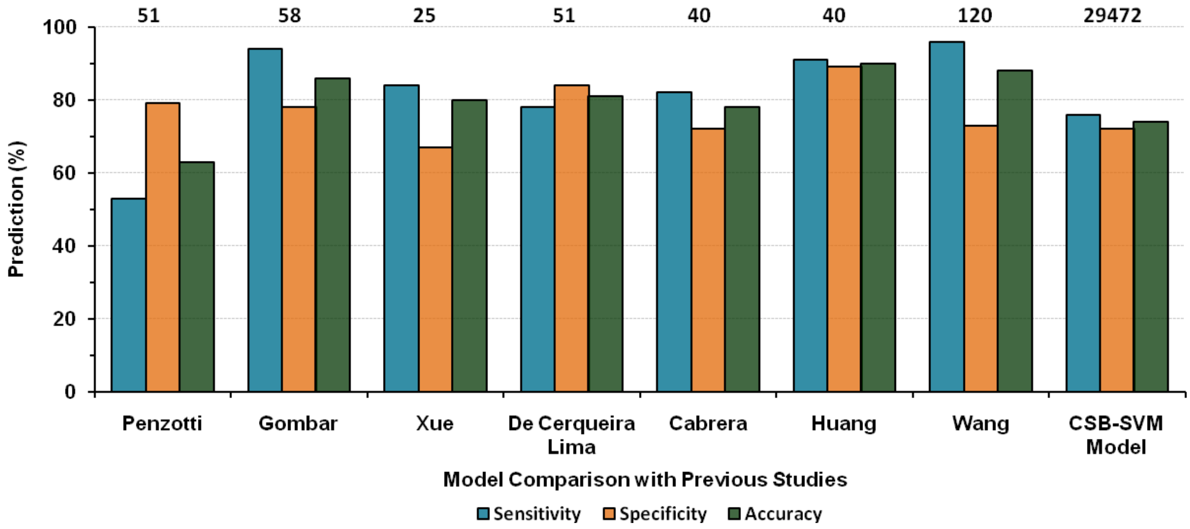


Figure 3.14: Comparison of our CSB models with previously published models for P-gp substrate-nonsubstrate classification (Upper row represents the number of compounds used in the test set)

Our models were trained using a highly imbalanced dataset with about 7000 compounds in training test and about 29000 compounds in test set. We achieved best models with support vector machine (overall accuracy is 74%, sensitivity: 76%, specificity: 72%) and random forest (overall accuracy is 72%, sensitivity: 75%, specificity: 69%) using only 12 2D descriptors from MOE using Chisquared feature selection method. Compared to previous models, our models correctly predict about 74% of the test set, which is slightly lower than the reported values. However, it can be emphasized that our dataset is two hundred times larger than previously reported datasets (n=25-120).

During the preparation of this manuscript, a similar study has been published by Wang et al. [272] where they developed support vector machine models for P-gp substrates and non-substrate using a set of 332 compounds, which include drugs. Wang et al. showed that a support vector machine correctly classify 96% of substrates and 73% of non-substrates, with an overall accuracy of 88% in the test set using 23 ADRIANA-Code and MOE descriptors. In order to validate our classification models further, we used datasets from Wang et al. as additional external validation datasets. Before investigation, applicability domain for these compounds was checked with our training set, which revealed only 4 compounds being out of our training set chemical space. Furthermore, we found 3 duplicate entries in Wang et al. dataset and were skipped from our studies. Validation was performed in two scenarios: a whole dataset (n=325) was used as external test set in scenario 1 and only test compounds (n=119) were used for validation in scenario 2. CSB classification models were developed using Chisquared descriptors and the result is provided in Table 3.6. In general, majority of the models perform good and the overall accuracy is 63-68%. In scenario 1, random forest (RF) and support vector machine (SVM) performs significantly better than other classifiers, both methods correctly predict >70% of substrates. Naïve bayes is good for predicting substrates (93%) but poorly predicts non-substrates (39%), thus low overall predictive ability (G-mean 57% and accuracy 62%). Random forest moderately predicts the non-substrates (63%). Overall RF performs significantly better than other methods, which is also reflected in G-mean (0.68) and MCC (0.36). In the second scenario, we used Wang et al.'s test set as a validation set. Similar quality of models was obtained as scenario 1 and again RF and SVM perform better than other classifiers with good MCC values (>30). Both show balanced classification predictions for substrates (70-78 %) and non-substrate (53-67%). Similar to Zhi Wang et al. SVM model, our SVM model also performs significantly better for substrate prediction (78%) but it predicts only 53% of non-substrates correctly (overall accuracy 66%). The moderate prediction of our best models (CSB-RF and CSB-SVM) to model from Wang et al. dataset might be due to the fact that the annotation of substrates/non substrates is based on different experimental procedures.

Table 3.6: External dataset prediction of P-gp substrates and non-substrates

Methods	Cost	Prediction (%)				Sensitivity	Specificity	G-mean	Precision	MCC	Accuracy
		TP	FP	TN	FN						
RFa)	110	73	37	63	27	0.73	0.63	0.68	0.66	0.36	0.68
DT a)	75	76	51	49	24	0.76	0.49	0.61	0.60	0.26	0.62
NB a)	1000	93	69	31	7	0.93	0.31	0.54	0.57	0.30	0.62
kNN a)	110	66	53	47	34	0.66	0.47	0.56	0.55	0.13	0.56
SVM a)	48	83	52	48	17	0.83	0.48	0.63	0.61	0.32	0.65
RF b)	110	70	33	67	30	0.70	0.67	0.68	0.68	0.37	0.68
DT b)	75	74	49	51	26	0.74	0.51	0.62	0.60	0.26	0.63
NB b)	1000	88	62	38	12	0.88	0.38	0.58	0.59	0.30	0.63
kNN b)	110	68	40	60	32	0.68	0.60	0.64	0.63	0.28	0.64
SVM b)	48	78	47	53	22	0.78	0.53	0.65	0.63	0.33	0.66
SVM c)	-		-	-	-	0.96	0.73	-	-	0.73	0.88

3.4.4 Conclusions

In the present study, we reported the application of machine learning methods to predict P-gp substrate and non-substrates from highly imbalanced dataset consisting of 36840 compounds. Models obtained from CSB-Random forest and support vector machine perform significantly better than the other CSB-classifiers. CSB models show better performance when compared to models obtained from conventional machine learning classifiers. The CSB-SVM model correctly predicts 74% of the overall compounds in test set. Most importantly, a balanced prediction model was obtained, which is able to correctly predict 76% of P-gp substrates and 72% of non-substrates. In addition to this, the classification models were further validated with a set of 325 drug compounds. An additional applicability domain experiment was performed and reveals that our model not only predicts NCI compounds, but it can also be applied to drug-like molecules. Models developed in this study are relatively simple and precise enough to be applicable for virtual screening of large chemical libraries for early identification of compounds, which are being transported by P-glycoprotein. An early identification of P-gp substrates can be potentially useful to remove compounds of poor ADMET properties.

Bibliography

1. DiMasi, J.A., R.W. Hansen and H.G. Grabowski, The price of innovation: new estimates of drug development costs. *Journal of Health Economics*, 2003. 22(2): 151-185.
2. Czerepak, E.A. and S. Ryser, Drug approvals and failures: implications for alliances. *Nat Rev Drug Discov*, 2008. 7(3): 197-198.
3. Heilman, R.D., Drug development history," overview," and what are GCPs? *Quality assurance (San Diego, Calif)*, 1995. 4(1): 75.
4. Prentis, R.A., Y. Lis and S.R. Walker, Pharmaceutical innovation by the seven UK-owned pharmaceutical companies (1964-1985). *British journal of clinical pharmacology*, 1988. 25(3): 387.
5. Venkatesh, S. and R.A. Lipper, Role of the development scientist in compound lead selection and optimization. *Journal of pharmaceutical sciences*, 2000. 89(2): 145-154.
6. Kola, I. and J. Landis, Can the pharmaceutical industry reduce attrition rates? *Nat Rev Drug Discov*, 2004. 3(8): 711-716.
7. Bleicher, K.H., H.J. Bohm, K. Muller and A.I. Alanine, Hit and lead generation: beyond high-throughput screening. *Nature reviews Drug discovery*, 2003. 2(5): 369-378.
8. Kennedy, T., Managing the drug discovery/development interface. *Drug Discovery Today*, 1997. 2(10): 436-444.
9. Augen, J., The evolving role of information technology in the drug discovery process. *Drug Discovery Today*, 2002. 7(5): 315-323.
10. Terstappen, G.C. and A. Reggiani, In silico research in drug discovery. *Trends in pharmacological sciences*, 2001. 22(1): 23-26.
11. Rauwerda, H., M. Roos, B.O. Hertzberger and T.M. Breit, The promise of a virtual lab in drug discovery. *Drug Discovery Today*, 2006. 11(5-6): 228-236.
12. Imming, P., C. Sinning and A. Meyer, Drugs, their targets and the nature and number of drug targets. *Nature reviews Drug discovery*, 2006. 5(10): 821-834.
13. Drews, J., Drug discovery: a historical perspective. *Science*, 2000. 287(5460): 1960.
14. Drews, J., Drug discovery today-and tomorrow. *Drug Discovery Today*, 2000. 5(1): 2-4.
15. García-Lara, J., M. Masalha and S.J. Foster, Staphylococcus aureus: the search for novel targets. *Drug Discovery Today*, 2005. 10(9): 643-651.
16. Singh, S., B.K. Malik and D.K. Sharma, Molecular drug targets and structure based drug design: A holistic approach. *Bioinformation*, 2006. 1(8): 314.
17. Lipinski, C. and A. Hopkins, Navigating chemical space for biology and medicine. *Nature*, 2004. 432(7019): 855-861.
18. Shoichet, B.K., Virtual screening of chemical libraries. *Nature*, 2004. 432(7019): 862-865.
19. Li, J., C.W. Murray, B. Waszkowycz and S.C. Young, Targeted molecular diversity in drug discovery: integration of structure-based design and combinatorial chemistry. *Drug Discovery Today*, 1998. 3(3): 105-112.
20. Breton, V., D. Kim and G. Rastelli, WISDOM: A Grid-Enabled Drug Discovery Initiative Against Malaria. *Grid Computing: Infrastructure, Service, and Applications*, 2008. 1: 353-381.
21. Brown, S.P. and S.W. Muchmore, High-throughput calculation of protein-ligand binding affinities: modification and adaptation of the MM-PBSA protocol to enterprise grid computing. *Journal of chemical information and modeling*, 2006. 46(3): 999-1005.
22. Nicolas, J. and V. Breton, *In silico drug discovery services in computing grid environments against neglected and emerging infectious diseases*, in *Bioinformatics*. 2006, Universite Blaise Pascal: Clermont-Ferrand. p. 172.

23. Chien, A., I. Foster and D. Goddette, Grid technologies empowering drug discovery. *Drug Discovery Today*, 2002. 7(20): s176-s180.
24. Boutaba, R., W. Golab, Y. Iraqi, T. Li and B.S. Arnaud, Grid-controlled lightpaths for high performance grid applications. *Journal of Grid Computing*, 2003. 1(4): 387-394.
25. Gray, J., Distributed computing economics. *Queue - Object-Relational Mapping*, 2008. 6(3): 63-68.
26. Simeonidou, D., R. Nejabati, N. Ciulli, L. Battestilli, G. Carrozzo, et al. Grid optical burst switched networks (GOBS). 2005.
27. Rio, M., A. Di Donato, F. Saka, N. Pezzi, R. Smith, S. Bhatti and P. Clarke, Quality of service networking for high performance grid applications. *Journal of Grid Computing*, 2003. 1(4): 329-343.
28. Birrell, A.D. and B.J. Nelson, Implementing remote procedure calls. *ACM Transactions on Computer Systems (TOCS)*, 1984. 2(1): 39-59.
29. Foster, I., C. Kesselman and S. Tuecke, The anatomy of the grid: Enabling scalable virtual organizations. *International Journal of High Performance Computing Applications*, 2001. 15(3): 200-222.
30. Foster, I., C. Kesselman, G. Tsudik and S. Tuecke. A security architecture for computational grids. 1998.
31. Rosenberry, W., D. Kenney and G. Fisher, Understanding DCE. 1992: O'Reilly & Associates, Inc.
32. Pope, A.L.M., The CORBA reference guide: understanding the common object request broker architecture. 1998: Addison-Wesley Longman Publishing Co., Inc.
33. Hartman, D., Unclogging distributed computing. *Spectrum, IEEE*, 1992. 29(5): 36-39.
34. Vinoski, S., CORBA: Integrating diverse applications within distributed heterogeneous environments. *Communications Magazine, IEEE*, 1997. 35(2): 46-55.
35. Baker, M. and A. Apon, Middleware. *International Journal of High Performance Computing Applications*, 2001. 15(2): 136-142.
36. The Globus Toolkit. 2011 (http://www.globus.org/toolkit).
37. Wilson, R.J., The European DataGrid Project. *Institute de Fisica d'Altes Energies and Colorado State University, Barcelona, Spain and Fort Collins, Colorado, USA*, 2001.
38. Catlett, C. The philosophy of TeraGrid: building an open, extensible, distributed TeraScale facility. 2002.
39. Deelman, E., C. Kesselman, G. Mehta, L. Meshkat, L. Pearlman, et al., GriPhyN and LIGO, building a virtual data grid for gravitational wave scientists. 2002.
40. Johnston, W.E. and S.C. Center The DOE Science Grid. 2003 (http://doesciencegrid.org/).
41. Condor Project. Condor project official website. (http://www.cs.wisc.edu/condor).
42. UNICORE - Distributed computing and data resources. 2011 (http://www.unicore.eu/).
43. gLite - Lightweight Middleware for Grid Computing. 2011 (http://glite.cern.ch/).
44. Jones, D.M., J.W. Fenner, G. Berti, F. Kruggel, R.A. Mehrem, W. Backfrieder, R. Moore and A. Geltmeier. The GEMSS Grid: an evolving HPC environment for medical applications. *In Proceedings of HealthGrid*. 2004. Clermont-Ferrand, France.
45. Legion: A Worldwide Virtual Computer. 2011 (http://legion.virginia.edu/).
46. Abramson, D., I. Foster, J. Giddy, A. Lewis, R. Sosic, R. Sutherst and N. White, The Nimrod computational workbench: A case study in desktop metacomputing. *Australian Computer Science Communications*, 1997. 19: 17-26.
47. Chien, A., B. Calder, S. Elbert and K. Bhatia, Entropia: architecture and performance of an enterprise desktop grid system. *Journal of Parallel and Distributed Computing*, 2003. 63(5): 597-610.

48. Univa UD: Univa UD, PCs Grid solution: Grid MP. (http://www.univaud.com/hpc/products/grid-mp/).

49. Oracle Grid Engine. 2011 (http://www.oracle.com/technetwork/oem/grid-engine-166852.html).

50. Wilson, R.J. The European DataGrid Project. 2001 (http://www.snowmass2001.org/e7/papers/wilson.pdf , http://eu-datagrid.web.cern.ch/eu-datagrid/).

51. Catlett, C. The philosophy of TeraGrid: building an open, extensible, distributed TeraScale facility. *In Proceedings of IEEE International Symposium on Cluster Computing and Grid*. 2002. Berlin, Germany.

52. Buyya, R., K. Branson, J. Giddy and D. Abramson, The Virtual Laboratory: a toolset to enable distributed molecular modelling for drug design on the World Wide Grid. *Concurrency and Computation: practice and Experience*, 2003. 15(1): 1-25.

53. Peitsch, M.C. Grid computing in drug discovery. 2006.

54. Richards, W.G., Virtual screening using grid computing: the screensaver project. *Nature Reviews Drug Discovery*, 2002. 1(7): 551-555.

55. Stevens, R., R. McEntire, C. Goble, M. Greenwood, J. Zhao, A. Wipat and P. Li, myGrid and the drug discovery process. *Drug Discovery Today: BIOSILICO*, 2004. 2(4): 140-148.

56. Krieger, E. and G. Vriend, Models@ Home: distributed computing in bioinformatics using a screensaver based approach. *Bioinformatics*, 2002. 18(2): 315.

57. Levesque, M.J., K. Ichikawa, S. Date and J.H. Haga, Design of a grid service-based platform for in silico protein-ligand screenings. *Computer methods and programs in biomedicine*, 2009. 93(1): 73-82.

58. Zhang, W., J. Zhang, Y. Chang, S. Chen, X. Du, F. Liu, F. Ma and J. Shen. Drug discovery grid. *U.K. e-Science All Hands Meet*. 2005. Nottingham, U.K.

59. Zhang, W., X. Du, F. Ma, J. Zhang and J. Shen. DDGrid: Harness the Full Power of Supercomputing Systems. *Fifth International Conference on Grid and Cooperative Computing Workshops*. 2006. Hunan.

60. Talbi, E.G. and A.Y. Zomaya, Grid computing for bioinformatics and computational biology. 2008: Wiley-Interscience.

61. FightAIDS@home. (http://fightaidsathome.scripps.edu/).

62. Chang, M.W., W. Lindstrom, A.J. Olson and R.K. Belew, Analysis of HIV wild-type and mutant structures via in silico docking against diverse ligand libraries. *Journal of chemical information and modeling*, 2007. 47(3): 1258-1262.

63. Folding@home Distributed Computing. 2000 (http://folding.stanford.edu/).

64. The Cancer Screensaver Lifesaver Project. 2007.

65. Anthrax Research Project. 2002 (http://www.chem.ox.ac.uk/anthrax/).

66. Smallpox Protection Project. 2003 (http://www.chem.ox.ac.uk/smallpox/).

67. Glick, M., G.H. Grant and W.G. Richards, Pinpointing anthrax-toxin inhibitors. *Nature biotechnology*, 2002. 20(2): 118.

68. Docking@Home. (http://docking.cis.udel.edu/).

69. GPUGrid. (http://www.gpugrid.net).

70. Rosetta@home. (http://boinc.bakerlab.org/rosetta/).

71. Drug Design and Optimization Lab (D2OL). (www.d2ol.com ; http://en.wikipedia.org/wiki/D2OL).

72. Sengent Inc. . (www.sengent.org).

73. Zhang, W., J. Zhang, Y. Chang, S. Chen, X. Du, F. Liu, F. Ma and J. Shen. Drug Discovery Grid. *Fourth UK e-Science All Hands Meeting*. 2005. Nottingham, UK.

74. Jacq, N., J. Salzemann, F. Jacq, Y. Legré, E. Medernach, et al., Grid-enabled virtual screening against malaria. *Journal of Grid Computing*, 2008. 6(1): 29-43.

75. Kasam, V., M. Zimmermann, A. Maa, H. Schwichtenberg, A. Wolf, N. Jacq, V. Breton and M. Hofmann-Apitius, Design of new plasmepsin inhibitors: a virtual high throughput screening approach on the EGEE grid. *Journal of chemical information and modeling*, 2007. 47(5): 1818-1828.

76. Kasam, V., J. Salzemann, M. Botha, A. Dacosta, G. Degliesposti, et al., WISDOM-II: Screening against multiple targets implicated in malaria using computational grid infrastructures. *Malaria journal*, 2009. 8(1): 88.

77. Kasam, V.K., J. Salzemann, V. Breton and N. Jacq. WISDOM-II: a large in silico docking effort for finding novel hits against malaria using computational grid infrastructure. 2007.

78. Gagliardi, F., B. Jones, F. Grey, M.E. Bégin and M. Heikkurinen, Building an infrastructure for scientific Grid computing: status and goals of the EGEE project. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 2005. 363(1833): 1729.

79. AuverGrid. (www.auvergrid.fr).

80. E-Science grid facility for Europe and Latin America (EELA). (http://www.eu-eela.eu/).

81. EUChinaGrid. (http://www.euchinagrid.org/).

82. EUMedGrid. (http://www.eumedgrid.eu/).

83. Milanesi, L. BioInfogrid: BioInformatics Simulation and Modeling Based on Grid. 2007.

84. BRIDGE (Bilateral Research and Industrial Development Enhancing and Integrating GRID Enabled Technologies). (http://www.bridge-grid.eu/).

85. Darvas, F., Á. Papp, I. Bágyi, G. Ambrus and L. Ürge. OpenMolGRID, a GRID based system for solving large-scale drug design problems. 2004.

86. Podvinec, M., S. Maffioletti, P. Kunszt, K. Arnold, L. Cerutti, et al. The SwissBioGrid Project: Objectivse, Preliminary Results and Lessons Learned. 2006.

87. Eerola, P., B. Kónya, O. Smirnova, T. Ekelöf, M. Ellert, et al. The NorduGrid production Grid infrastructure, status and plans. 2003.

88. Vangrevelinghe, E., K. Zimmermann, J. Schoepfer, R. Portmann, D. Fabbro and P. Furet, Discovery of a potent and selective protein kinase CK2 inhibitor by high-throughput docking. *Journal of medicinal chemistry*, 2003. 46(13): 2656-2662.

89. Ghemtio, L., E. Jeannot and B. Maigret, Efficiency of a hierarchical protocol for high throughput structure-based virtual screening on GRID5000 cluster grid. *Open Access Bioinformatics*, 2010. 2: 41-53.

90. Cappello, F., E. Caron, M. Daydé, F. Desprez, Y. Jégou, et al. Grid'5000: A large scale and highly reconfigurable grid experimental testbed. 2005.

91. Bullard, D., A. Gobbi, M.A. Lardy, C. Perkins and Z. Little, Hydra: a self regenerating high performance computing grid for drug discovery. *Journal of chemical information and modeling*, 2008. 48(4): 811-816.

92. Schaller, R.R., Moore's law: past, present and future. *Spectrum, IEEE*, 1997. 34(6): 52-59.

93. Kesselman, C. and I. Foster, The grid: blueprint for a new computing infrastructure. 1998: Morgan Kaufman.

94. Foster, I., What is the grid? a three point checklist. *GRID today*, 2002. 1(6).

95. Stockinger, H., Defining the grid: a snapshot on the current view. *The Journal of Supercomputing*, 2007. 42(1): 3-17.

96. Gentzsch, W., Response to ian foster's" what is the grid?" *GRID today*, 2002. 1(8): 5.

97. Grimshaw, A.S. and A. Natrajan, Legion: Lessons learned building a grid operating system. *Proceedings of the IEEE*, 2005. 93(3): 589-603.

98. Bote-Lorenzo, M.L., Y.A. Dimitriadis and E. Gómez-Sánchez, Grid characteristics and uses: a grid definition, in *Grid Computing*. 2004, Springer: Berlin / Heidelberg. 291-298.

99. Berman, F., G. Fox and A.J.G. Hey, Grid computing: making the global infrastructure a reality. Vol. 2. 2003: John Wiley & Sons Inc.

100. Chetty, M. and R. Buyya, Weaving computational Grids: How analogous are they with electrical Grids? *Computing in Science & Engineering*, 2002. 4(4): 61-71.

101. Ault, M. and M. Tumma, Oracle 10g Grid & Real Application Clusters: Oracle 10g Grid Computing with RAC. 2004: Rampant TechPress.

102. Yeo, C.S., M.D. De Assuncao, J. Yu, A. Sulistio, S. Venugopal, M. Placek and R. Buyya, Utility computing and global grids. *Arxiv preprint cs/0605056*, 2006.

103. What Is Grid Computing, Sun Grid Engine. 2009 (http://wikis.sun.com/display/GridEngine/What+Is+Grid+Computing).

104. Montero, R.S., E. Huedo and I.M. Llorente, Benchmarking of high throughput computing applications on grids. *Parallel Computing*, 2006. 32(4): 267-279.

105. Overview of the Condor High-Throughput Computing System. (http://www.cs.wisc.edu/condor/overview/).

106. Condor Project. Condor administrator manual v7.0. (http://www.cs.wisc.edu/condor/manual/v7.0/).

107. Tannenbaum, T., D. Wright, K. Miller and M. Livny, Condor- A distributed job scheduler, in *Beowulf Cluster Computing with Linux*, T. Sterling, Editor. 2001, MIT Press: Cambridge, MA, USA. 307–350.

108. Thain, D., T. Tannenbaum and M. Livny, Distributed computing in practice: The Condor experience. *Concurrency and Computation: Practice and Experience*, 2005. 17(2 4): 323-356.

109. Raman, R., M. Livny and M. Solomon. Matchmaking: Distributed resource management for high throughput computing. *The Proceedings of the Seventh International Symposium on High Performance Distributed Computing*. 1998. Chicago, IL , USA

110. Raman, R., *Matchmaking Frameworks for Distributed Resource Management*, in *Deaprtment of Computer Science*. 2001, University of Wisconsin-Madison: Madison, Wisconsin, US.

111. Alderman, I.D., *A Security Framework for Distributed Batch Computing*, in *Department of Computer Science*. 2010, University of Wisconsin-Madison: Madison, Wisconsin, US.

112. Condor's Checkpoint Mechanism 2011 (http://research.cs.wisc.edu/condor/manual/v7.2.1/4_2Condor_s_Checkpoint.html).

113. Condor's Exceptional Features. 2011 (http://research.cs.wisc.edu/condor/manual/v7.2.1/1_3Exceptional_Features.html).

114. DAGMan (Directed Acyclic Graph Manager). 2008 (http://research.cs.wisc.edu/condor/dagman/).

115. Apache Software Foundation: Apache tomcat. 2008 (http://tomcat.apache.org/).

116. PostgreSQL. 2008 (http://www.postgresql.org/).

117. OpenVPN - Open Source VPN. 2008 (http://openvpn.net/).

118. Beckles, B., S.C. Son and J. Kewley. Current methods for negotiating firewalls for the Condor system. *Proceedings of the Fourth UK e-Science All Hands Meeting*. 2005.

119. O'Donnell, C. Condor and Firewalls. 1998 (www.cs.wisc.edu/condor/firewall/firewall.doc).

120. Son, S. and M. Livny. Recovering internet symmetry in distributed computing. *Third IEEE International Symposium on Cluster Computing and the Grid (CCGrid'03)*. 2003. Tokyo, Japan.

121. Son, S., B. Allcock and M. Livny. Codo: Firewall traversal by cooperative on-demand opening. *14th IEEE Symposium on High Performance Distributed Computing (HPDC14)*. 2005.

122. Lodygensky, O., G. Fedak, F. Cappello, V. Neri, M. Livny and D. Thain. XtremWeb & Condor: sharing resources between Internet connected Condor pool. *Proceedings of 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid*. 2003.

123. Calleja, M., B. Beckles, M. Keegan, M.A. Hayes, A. Parker and M.T. Dove. CamGrid: Experiences in constructing a university-wide, Condor-based grid at the University of Cambridge. *Proceedings of the UK e-Science All Hands Meeting*. 2004.
124. Scherp, G., W. Hasselbring and J. Ploski. The WISENT Grid Architecture: Coping with Firewalls and NAT. *German e-Science Conference*. 2007. Baden.
125. Condor User's Manual - Version 7.5.3. . (http://www.cs.wisc.edu/condor/manual/v7.5/condor-V7_5_3-Manual.pdf).
126. Condor's Security. 2011 (http://research.cs.wisc.edu/condor/manual/v7.0/3_6Security.html).
127. The Xen virtual machine monitor. (http://www.cl.cam.ac.uk/research/srg/netos/xen).
128. Aloni, D. Cooperative linux. *Proceedings of the Linux Symposium*. 2004. Ottawa, Canada.
129. Whitaker, A., M. Shaw and S.D. Gribble, *Denali: Lightweight virtual machines for distributed and networked applications*. 2002, Citeseer.
130. Lawton, K. plex86: an i80x86 virtual machine. *Proceedings of the 4th Annual Linux Showcase & Conference*. 2000. Atlanta, USA.
131. VMware. (http://www.vmware.com/).
132. Parallels: Cost-effective, High-Performance Virutalization Made Easy. (http://www.parallels.com/files/upload/Parallels_Intel_whitepaper_on_VT_Technology.pdf).
133. Sumanth, J. Running condor in a virtual environment with colinux. 2006 (http://www.cs.wisc.edu/condor/CondorWeek2006/presentations/sumanth_condor_colinux.ppt).
134. Neeman, H., H. Severini and D. Wu, Supercomputing in plain english: teaching cyberinfrastructure to computing novices. *ACM SIGCSE Bulletin*, 2008. 40(2): 27-30.
135. Severini, H., H. Neeman, C. Franklin, J. Alexander and J.V. Sumanth. Implementing Linux-enabled Condor in windows computer labs. *Nuclear Science Symposium Conference*. 2008. Dresden, Germany
136. Santosa, M. and A. Schaefer Build a heterogeneous cluster with coLinux and openMosix. 2005 (http://www.ibm.com/developerworks/linux/library/l-colinux).
137. Alexander, J. and C. Franklin Implementing linux-enabled condor in windows computer labs. 2008 (https://twiki.grid.iu.edu/pub/Education/GTGS2008Syllabus/Implementing_Condor_in_Windows.pdf).
138. Herzfeld, D.J., L.E. Olson and C.A. Struble. Pools of virtual boxes: building campus grids with virtual machines. *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*. 2010. Chicago, Illinois.
139. Lewis, Q. Exploring Virtual Workspace Concepts in a Dynamic Universe for Condor. (http://www2.cs.uh.edu/~qslewis/cosc7360/Dynamic%20Condor%20Universe.doc).
140. Tsugawa, M. and J.A.B. Fortes. A virtual network (ViNe) architecture for grid computing. *Proceedings 20th IEEE International Parallel & Distributed Processing Symposium*. 2006.
141. Buyya, R., K. Branson, J. Giddy and D. Abramson, The Virtual Laboratory: a toolset to enable distributed molecular modelling for drug design on the World-Wide Grid. *Concurrency and Computation: Practice and Experience*, 2003. 15(1): 1-25.
142. Ewing, T.J.A., S. Makino, A.G. Skillman and I.D. Kuntz, DOCK 4.0: search strategies for automated molecular docking of flexible molecule databases. *Journal of computer-aided molecular design*, 2001. 15(5): 411-428.
143. Lang, P.T. Preparing Molecules for DOCKing. 2009 (http://dock.compbio.ucsf.edu/DOCK_6/tutorials/struct_prep/prepping_molecules.htm).
144. Huang, C. DMS - calculate a molecular surface. 2008 (http://www.cgl.ucsf.edu/chimera/current/docs/UsersGuide/midas/dms1.html#dmsformat).

145. Kuntz, I.D., J.M. Blaney, S.J. Oatley, R. Langridge and T.E. Ferrin, A geometric approach to macromolecule-ligand interactions. *Journal of Molecular Biology*, 1982. 161(2): 269-288.
146. Aynechi, T. and P.T. Lang Generating the Grid. (http://dock.compbio.ucsf.edu/DOCK_6/ tutorials/grid_generation/generating_grid.html).
147. Shoichet, B.K., I.D. Kuntz and D.L. Bodian, Molecular docking using shape descriptors. *Journal of Computational Chemistry*, 1992. 13(3): 380-397.
148. Karplus, M. and J.A. McCammon, Molecular dynamics simulations of biomolecules. *Nature Structural & Molecular Biology*, 2002. 9(9): 646-652.
149. Hansson, T., C. Oostenbrink and W.F. van Gunsteren, Molecular dynamics simulations. *Current opinion in structural biology*, 2002. 12(2): 190-196.
150. Nelson, M.T., W. Humphrey, A. Gursoy, A. Dalke, L.V. KalÃ©, R.D. Skeel and K. Schulten, NAMD: a parallel, object-oriented molecular dynamics program. *International Journal of High Performance Computing Applications*, 1996. 10(4): 251.
151. Phillips, J.C., R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, et al., Scalable molecular dynamics with NAMD. *Journal of Computational Chemistry*, 2005. 26(16): 1781.
152. Phillips, J.C., G. Zheng, S. Kumar and L.V. Kale. NAMD: Biomolecular simulation on thousands of processors. *Proceedings of the 2002 ACM/IEEE conference on Supercomputing*. 2002.
153. Rush Iii, T.S., J.A. Grant, L. Mosyak and A. Nicholls, A shape-based 3-D scaffold hopping method and its application to a bacterial protein-protein interaction. *Journal of medicinal chemistry*, 2005. 48(5): 1489-1495.
154. Grant, J.A., M.A. Gallardo and B.T. Pickup, A fast method of molecular shape comparison: A simple application of a Gaussian description of molecular shape. *Journal of Computational Chemistry*, 1996. 17(14): 1653-1666.
155. Frank, E., M. Hall, L. Trigg, G. Holmes and I.H. Witten, Data mining in bioinformatics using Weka. *Bioinformatics*, 2004. 20(15): 2479.
156. Witten, I.H. and E. Frank, Data Mining: Practical machine learning tools and techniques. 2 ed. 2005, San Francisco: Morgan Kaufmann.
157. Chang, C.C. and C.J. Lin, LIBSVM: a library for support vector machines. 2001.
158. Drug Discovery at the National Cancer Institute: Fact Sheet. 2006 (http://www.cancer.gov/cancertopics/factsheet/NCI/drugdiscovery).
159. Shoemaker, R.H., The NCI60 human tumour cell line anticancer drug screen. *Nature Reviews Cancer*, 2006. 6(10): 813-823.
160. Boyd, M.R. and K.D. Paull, Some practical considerations and applications of the National Cancer Institute in vitro anticancer drug discovery screen. *Drug Development Research*, 1995. 34(2): 91-109.
161. Grever, M.R., S.A. Schepartz and B.A. Chabner, The National Cancer Institute: cancer drug discovery and development program. *Seminars in Oncology*, 1992. 19: 622-638.
162. Monks, A., D. Scudiero, P. Skehan, R. Shoemaker, K. Paull, et al., Feasibility of a high-flux anticancer drug screen using a diverse panel of cultured human tumor cell lines. *Journal of the National Cancer Institute*, 1991. 83(11): 757.
163. Paull, K.D., R.H. Shoemaker, L. Hodes, A. Monks, D.A. Scudiero, L. Rubinstein, J. Plowman and M.R. Boyd, Display and analysis of patterns of differential activity of drugs against human tumor cell lines: development of mean graph and COMPARE algorithm. *Journal of the National Cancer Institute*, 1989. 81(14): 1088.
164. Weinstein, J.N., K.W. Kohn, M.R. Grever, V.N. Viswanadhan, L.V. Rubinstein, et al., Neural computing in cancer drug development: predicting mechanism of action. *Science*, 1992. 258(5081): 447.
165. Weinstein, J.N., T.G. Myers, P.M. O'Connor, S.H. Friend, A.J. Fornace, et al., An information-intensive approach to the molecular pharmacology of cancer. *Science*, 1997. 275(5298): 343.

166. O'Connor, P.M., J. Jackman, I. Bae, T.G. Myers, S. Fan, et al., Characterization of the p53 tumor suppressor pathway in cell lines of the National Cancer Institute anticancer drug screen and correlations with the growth-inhibitory potency of 123 anticancer agents. *Cancer research*, 1997. 57(19): 4285.

167. Alvarez, M., K. Paull, A. Monks, C. Hose, J.S. Lee, J. Weinstein, M. Grever, S. Bates and T. Fojo, Generation of a drug resistance profile by quantitation of mdr-1/P-glycoprotein in the cell lines of the National Cancer Institute Anticancer Drug Screen. *Journal of Clinical Investigation*, 1995. 95(5): 2205.

168. Lee, J.S., K. Paull, M. Alvarez, C. Hose, A. Monks, M. Grever, A.T. Fojo and S.E. Bates, Rhodamine efflux patterns predict P-glycoprotein substrates in the National Cancer Institute drug screen. *Molecular pharmacology*, 1994. 46(4): 627.

169. Shi, L.M., T.G. Myers, Y. Fan, P.M. Oâ€™Connor, K.D. Paull, S.H. Friend and J.N. Weinstein, Mining the National Cancer Institute Anticancer Drug Discovery Database: cluster analysis of ellipticine analogs with p53-inverse and central nervous system-selective patterns of activity. *Molecular pharmacology*, 1998. 53(2): 241.

170. Zaharevitz, D.W., S.L. Holbeck, C. Bowerman and P.A. Svetlik, COMPARE: a web accessible tool for investigating mechanisms of cell growth inhibition. *Journal of Molecular Graphics and Modelling*, 2002. 20(4): 297-303.

171. Shao, L., N.E. Lewin, P.S. Lorenzo, Z. Hu, I.J. Enyedy, et al., Iridals are a novel class of ligands for phorbol ester receptors with modest selectivity for the RasGRP receptor subfamily. *Journal of medicinal chemistry*, 2001. 44(23): 3872-3880.

172. Zaharevitz, D.W., R. Gussio, M. Leost, A.M. Senderowicz, T. Lahusen, C. Kunick, L. Meijer and E.A. Sausville, Discovery and initial characterization of the paullones, a novel class of small-molecule inhibitors of cyclin-dependent kinases. *Cancer research*, 1999. 59(11): 2566.

173. Wang, H., J. Klinginsmith, X. Dong, A.C. Lee, R. Guha, Y. Wu, G.M. Crippen and D.J. Wild, Chemical data mining of the NCI human tumor cell line database. *Journal of chemical information and modeling*, 2007. 47(6): 2063-2076.

174. Scherf, U., D.T. Ross, M. Waltham, L.H. Smith, J.K. Lee, et al., A gene expression database for the molecular pharmacology of cancer. *nature genetics*, 2000. 24(3): 236-244.

175. Myers, T.G., N.L. Anderson, M. Waltham, G. Li, J.K. Buolamwini, D.A. Scudiero, K.D. Paull, E.A. Sausville and J.N. Weinstein, A protein expression database for the molecular pharmacology of cancer. *Electrophoresis*, 1997. 18(3): 647-653.

176. DeRisi, J., L. Penland, P.O. Brown, M.L. Bittner, P.S. Meltzer, M. Ray, Y. Chen, Y.A. Su and J.M. Trent, Use of a cDNA microarray to analyse gene expression patterns in human cancer. *nature genetics*, 1996. 14(4): 457-460.

177. Ross, D.T., U. Scherf, M.B. Eisen, C.M. Perou, C. Rees, et al., Systematic variation in gene expression patterns in human cancer cell lines. *nature genetics*, 2000. 24(3): 227-235.

178. Blower, P.E., C. Yang, M.A. Fligner, J.S. Verducci, L. Yu, S. Richman and J.N. Weinstein, Pharmacogenomic analysis: correlating molecular substructure classes with microarray gene expression data. *The pharmacogenomics journal*, 2002. 2(4): 259-271.

179. Szakács, G., J.P. Annereau, S. Lababidi, U. Shankavaram, A. Arciello, et al., Predicting drug sensitivity and resistance:: Profiling ABC transporter genes in cancer cells. *Cancer Cell*, 2004. 6(2): 129-137.

180. Staunton, J.E., D.K. Slonim, H.A. Coller, P. Tamayo, M.J. Angelo, et al., Chemosensitivity prediction by transcriptional profiling. *Proceedings of the National Academy of Sciences*, 2001. 98(19): 10787.

181. Shoemaker, R.H., Genetic and epigenetic factors in anticancer drug resistance. *Journal of the National Cancer Institute*, 2000. 92(1): 4.

182. Türk, D., M.D. Hall, B.F. Chu, J.A. Ludwig, H.M. Fales, M.M. Gottesman and G. Szakács, Identification of compounds selectively killing multidrug-resistant cancer cells. *Cancer research*, 2009. 69(21): 8293.
183. Gupta, R.S., W. Murray and R. Gupta, Cross resistance pattern towards anticancer drugs of a human carcinoma multidrug-resistant cell line. *British journal of cancer*, 1988. 58(4): 441.
184. DTP Human Tumor Cell Line Screen. 2010 (http://dtp.nci.nih.gov/docs/cancer/cancer_data.html).
185. PubChem Project. Accessed in August 2010 (http://pubchem.ncbi.nlm.nih.gov).
186. Willett, P., Similarity methods in chemoinformatics. *Annual Review of Information Science and Technology*, 2009. 43(1): 1-117.
187. Johnson, M.A. and G.M. Maggiora, Concepts and applications of molecular similarity, ed. J.W. Sons. Vol. 67. 1990.
188. Nikolova, N. and J. Jaworska, Approaches to measure chemical similarity - a review. *QSAR & Combinatorial Science*, 2003. 22(9-10): 1006-1026.
189. Ellis, D., J. Furner-Hines and P. Willett, Measuring the degree of similarity between objects in text retrieval systems. *Perspectives in Information Management*, 1993. 3(2): 128-149.
190. Khalifa, A.A., M. Haranczyk and J. Holliday, Comparison of Nonbinary Similarity Coefficients for Similarity Searching, Clustering and Compound Selection. *Journal of chemical information and modeling*, 2009. 49(5): 1193-1201.
191. ROCS, Openeye Scientific Software, Santa Fe, NM. 2010 (http://www.eyesopen.com/rocs).
192. OMEGA, Openeye Scientific Software, Santa Fe, NM.2010 (http://www.eyesopen.com/omega).
193. Triballeau, N., F. Acher, I. Brabet, J.P. Pin and H.O. Bertrand, Virtual screening workflow development guided by the receiver operating characteristic curve approach. Application to high-throughput docking on metabotropic glutamate receptor subtype 4. *Journal of Medicinal Chemistry*, 2005. 48(7): 2534-2547.
194. Hawkins, P.C.D., A.G. Skillman and A. Nicholls, Comparison of shape-matching and docking as virtual screening tools. *Journal of medicinal chemistry*, 2007. 50(1): 74-82.
195. Serafeim, A., M.J. Holder, G. Grafton, A. Chamba, M.T. Drayson, et al., Selective serotonin reuptake inhibitors directly signal for apoptosis in biopsy-like Burkitt lymphoma cells. *Blood*, 2003. 101(8): 3212-3219.
196. Rochford, R., M.J. Cannon and A.M. Moormann, Endemic Burkitt's lymphoma: a polymicrobial disease? *Nature Reviews Microbiology*, 2005. 3(2): 182-187.
197. Schuster, C., N. Fernbach, U. Rix, G. Superti-Furga, M. Holy, M. Freissmuth, H.H. Sitte and V. Sexl, Selective serotonin reuptake inhibitors--a new modality for the treatment of lymphoma/leukaemia? *Biochemical pharmacology*, 2007. 74(9): 1424-1435.
198. Kiran Kumar, S., V.L. Sharma, M. Kumar, P.K. Shukla, P. Tiwari, et al., Synthesis of benzenepropanamine analogues as non-detergent spermicides with antitrichomonas and anticandida activities. *Bioorganic & medicinal chemistry*, 2006. 14(19): 6593-6600.
199. Kohonen, T., The self-organizing map. *Proceedings of the IEEE*, 1990. 78(9): 1464-1480.
200. MOE (Molecular Operating Environment), Version 2009.10, Chemical Computing Group Inc, 1010 Sherbrooke Street West, Suite 910, Montreal, Canada. 2009 (http://www.chemcomp.com/).
201. Rems, L., *Propafenon-Analoga mit Benzothiophen-Struktur: Enantiomerentrennung mit Hilfe eines neuen chiralen Lactols*, in *Pharmazeutische Chemie*. 1999, Westfalischen Wilhelms-University Munster. p. 181.
202. Ecker, G., W. Fleishhacker and C.R. Noe, New benzofuran-type antiarrhythmic compounds related to propafenone. *Heterocycles*, 1994. 38(6): 1247-1256.

203. Langer, T., M. Eder, R.D. Hoffmann, P. Chiba and G.F. Ecker, Lead identification for modulators of multidrug resistance based on in silico screening with a pharmacophoric feature model. *Archiv der Pharmazie*, 2004. 337(6): 317-327.

204. Chiba, P., S. Burghofer, E. Richter, B. Tell, A. Moser and G. Ecker, Synthesis, pharmacologic activity, and structure-activity relationships of a series of propafenone-related modulators of multidrug resistance. *Journal of medicinal chemistry*, 1995. 38(14): 2789-2793.

205. Rastogi, S.N., N. Anand, P.P. Gupta and J.N. Sharma, Agents acting on the central nervous system. 19.(+-)-1-(o-and m-Alkanoylphenoxy)-3-(N4-arylpiperazinyl) propan-2-ols as local anesthetics, hypotensives, and tranquilizers. *Journal of medicinal chemistry*, 1973. 16(7): 797-804.

206. Rastogi, S.N., N. Anand and C.R. Prasad, Agents acting on the central nervous system. 14. 1-(p-alkanoylphenoxy)-3-(N4-arylpiperazinyl) propan-2-ols. New class of antidepressants. *Journal of medicinal chemistry*, 1972. 15(3): 286-291.

207. Jabeen, I., K. Pleban, U. Rinner, P. Chiba and G.F. Ecker, Structure Activity Relationships, Ligand Efficiency, and Lipophilic Efficiency Profiles of Benzophenone-Type Inhibitors of the Multidrug Transporter P-Glycoprotein. *Journal of medicinal chemistry*, 2012. 55(7): 3261-3273.

208. Ecker, G.F., E. Csaszar, S. Kopp, B. Plagens, W. Holzer, W. Ernst and P. Chiba, Identification of Ligand-Binding Regions of P-Glycoprotein by Activated-Pharmacophore Photoaffinity Labeling and Matrix-Assisted Laser Desorption/Ionization-Time-of-Flight Mass Spectrometry. *Molecular pharmacology*, 2002. 61(3): 637-648.

209. Salem, M., E. Richter, M. Hitzler, P. Chiba and G. Ecker, Studies on propafenone-type modulators of multidrug resistance. Part 8. Synthesis and pharmacological activity of indanone analogs. *Scientia Pharmaceutica*, 1998. 66(3): 147-158.

210. ADRIANA Code, Molecular Networks GmbH: Erlangen, Germany. (http://www.molecular-networks.com).

211. Bauknecht, H., A. Zell, H. Bayer, P. Levi, M. Wagener, J. Sadowski and J. Gasteiger, Locating biologically active compounds in medium-sized heterogeneous datasets by topological autocorrelation vectors: dopamine and benzodiazepine agonists. *Journal of chemical information and computer sciences*, 1996. 36(6): 1205-1213.

212. Kaiser, D., L. Terfloth, S. Kopp, J. Schulz, R. de Laet, P. Chiba, G.F. Ecker and J. Gasteiger, Self-organizing maps for identification of new inhibitors of P-glycoprotein. *Journal of medicinal chemistry*, 2007. 50(7): 1698-1702.

213. Yang, P., L. Xu, B.B. Zhou, Z. Zhang and A.Y. Zomaya, A particle swarm based hybrid system for imbalanced medical data sampling. *BMC genomics*, 2009. 10(Suppl 3): S34.

214. Molinara, M., M.T. Ricamato and F. Tortorella. Facing imbalanced classes through aggregation of classifiers. *14th International Conference on Image Analysis and Processing*. 2007.

215. He, H. and E.A. Garcia, Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 2008: 1263-1284.

216. Chawla, N.V., N. Japkowicz and A. Kotcz, Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 2004. 6(1): 1-6.

217. Kubat, M. and S. Matwin. Addressing the curse of imbalanced training sets: one-sided selection. *Proceedings of 14th Interantinoal Conference on Machine Learning*. 1997.

218. Chawla, N.V., K.W. Bowyer, L.O. Hall and W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 2002. 16(1): 321-357.

219. Elkan, C. The foundations of cost-sensitive learning. *Proceedings of the seventeenth international joint conference of artificial intelligence*. 2001. Seattle, Washington.

220. Maloof, M.A. Learning when data sets are imbalanced and when costs are unequal and unknown. *Proceedings of the ICML'03 Workshop on Learning from Imbalanced Data Sets*. 2003.

221. McCarthy, K., B. Zabar and G. Weiss. Does cost-sensitive learning beat sampling for classifying rare classes? *Proceedings of the 1st International Workshop on Utility-Based Data Mining*. 2005.

222. Liu, X.Y. and Z.H. Zhou. The influence of class imbalance on cost-sensitive learning: an empirical study. *Proceedings of the Sixth International Conference on Data Mining*. 2006.
223. Vapnik, V.N., The nature of statistical learning theory. 1995: Springer-Verlag New York Inc.
224. Japkowicz, N. and S. Stephen, The class imbalance problem: A systematic study. *Intelligent data analysis*, 2002. 6(5): 429-449.
225. Akbani, R., S. Kwek and N. Japkowicz, Applying support vector machines to imbalanced datasets. *Proceedings of the 15th European Conference on Machine Learning*, 2004: 39-50.
226. Wang, B.X. and N. Japkowicz. Boosting support vector machines for imbalanced data sets. *Proceedings of the 17th international conference on Foundations of intelligent systems*. 2008.
227. Weis, D.C., D.P. Visco Jr and J.L. Faulon, Data mining PubChem using a support vector machine with the Signature molecular descriptor: Classification of factor XIa inhibitors. *Journal of Molecular Graphics and Modelling*, 2008. 27(4): 466-475.
228. Eitrich, T., A. Kless, C. Druska, W. Meyer and J. Grotendorst, Classification of highly unbalanced CYP450 data of drugs using cost sensitive machine learning techniques. *Journal of chemical information and modeling*, 2007. 47(1): 92-103.
229. Abe, N., B. Zadrozny and J. Langford. An iterative method for multi-class cost-sensitive learning. *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2004.
230. Chen, K., B.L. Lu and J.T. Kwok. Efficient classification of multi-label and imbalanced data using min-max modular classifiers. *International Joint Conference on Neural Networks*. 2006.
231. Tan, A., D. Gilbert and Y. Deville, Multi-class protein fold classification using a new ensemble machine learning approach. *Genome Informatics*, 2003. 14: 206-217.
232. Thiebaut, F., T. Tsuruo, H. Hamada, M.M. Gottesman, I. Pastan and M.C. Willingham, Cellular localization of the multidrug-resistance gene product P-glycoprotein in normal human tissues. *Proceedings of the National Academy of Sciences*, 1987. 84(21): 7735.
233. Gottesman, M.M., T. Fojo and S.E. Bates, Multidrug resistance in cancer: role of ATP-dependent transporters. *Nature Reviews Cancer*, 2002. 2(1): 48-58.
234. Gottesman, M.M., I. Pastan and S.V. Ambudkar, P-glycoprotein and multidrug resistance. *Current opinion in genetics & development*, 1996. 6(5): 610-617.
235. SzakÃ¡cs, G., J.K. Paterson, J.A. Ludwig, C. Booth-Genthe and M.M. Gottesman, Targeting multidrug resistance in cancer. *Nature reviews Drug discovery*, 2006. 5(3): 219-234.
236. Giacomini, K.M., S.M. Huang, D.J. Tweedie, L.Z. Benet, K.L.R. Brouwer, et al., Membrane transporters in drug development. *Nature reviews Drug discovery*, 2010. 9(3): 215-236.
237. Zhang, L., Y.D. Zhang, P. Zhao and S.M. Huang, Predicting Drug-Drug Interactions: An FDA perspective. *The AAPS journal*, 2009. 11(2): 300-306.
238. Sugano, K., M. Kansy, P. Artursson, A. Avdeef, S. Bendels, et al., Coexistence of passive and carrier-mediated processes in drug transport. *Nature reviews Drug discovery*, 2010. 9(8): 597-614.
239. Szakács, G., A. Váradi, C. Özvegy-Laczka and B. Sarkadi, The role of ABC transporters in drug absorption, distribution, metabolism, excretion and toxicity (ADME-Tox). *Drug Discovery Today*, 2008. 13(9-10): 379-393.
240. Yu, D.K., The contribution of P-glycoprotein to pharmacokinetic drug-drug interactions. *The Journal of Clinical Pharmacology*, 1999. 39(12): 1203.
241. Demel, M.A., R. Schwaha, O. KrÃ¤mer, P. Ettmayer, E.E.J. Haaksma and G.F. Ecker, In silico prediction of substrate properties for ABC-multidrug transporters. 2008.
242. Ecker, G.F., T. Stockner and P. Chiba, Computational models for prediction of interactions with ABC-transporters. *Drug Discovery Today*, 2008. 13(7-8): 311-317.

243. Gombar, V.K., J.W. Polli, J.E. Humphreys, S.A. Wring and C.S. Serabjitâ€Singh, Predicting P-glycoprotein substrates by a quantitative structure-activity relationship model. *Journal of pharmaceutical sciences*, 2004. 93(4): 957-968.

244. Pajeva, I. and M. Wiese, Molecular modeling of phenothiazines and related drugs as multidrug resistance modifiers: a comparative molecular field analysis study. *Journal of medicinal chemistry*, 1998. 41(11): 1815-1826.

245. Seelig, A., A general pattern for substrate recognition by P-glycoprotein. *European Journal of Biochemistry*, 1998. 251(1-2): 252-261.

246. Pearce, H.L., A.R. Safa, N.J. Bach, M.A. Winter, M.C. Cirtain and W.T. Beck, Essential features of the P-glycoprotein pharmacophore as defined by a series of reserpine analogs that modulate multidrug resistance. *Proceedings of the National Academy of Sciences*, 1989. 86(13): 5128.

247. Cianchetta, G., R.W. Singleton, M. Zhang, M. Wildgoose, D. Giesing, A. Fravolini, G. Cruciani and R.J. Vaz, A pharmacophore hypothesis for P-glycoprotein substrate recognition using GRIND-based 3D-QSAR. *Journal of medicinal chemistry*, 2005. 48(8): 2927-2935.

248. Xue, Y., C.W. Yap, L.Z. Sun, Z.W. Cao, J.F. Wang and Y.Z. Chen, Prediction of P-glycoprotein substrates by a support vector machine approach. *Journal of chemical information and computer sciences*, 2004. 44(4): 1497-1505.

249. Wang, Y.H., Y. Li, S.L. Yang and L. Yang, Classification of substrates and inhibitors of P-glycoprotein using unsupervised machine learning approach. *Journal of chemical information and modeling*, 2005. 45(3): 750-757.

250. Sakiyama, Y., The use of machine learning and nonlinear statistical tools for ADME prediction. *Expert Opin Drug Metab Toxicol*, 2009. 5(2): 149-169.

251. Developmental Therapeutics Program. Accessed August 2010 (http://dtp.nci.nih.gov/).

252. FILTER, Openeye Scientific Software, Santa Fe, NM. 2010 (http://www.eyesopen.com/filter).

253. Chemaxon Standardizer, JChem 5.8.2, Budapest, Hungary. 2012 (http://www.chemaxon.com).

254. MACCS (Molecular ACCess System) Structural Keys; Symyx Software: San Ramon, CA. 2005.

255. Fox, T. and J.M. Kriegl, Machine learning techniques for in silico modeling of drug metabolism. *Current Topics in Medicinal Chemistry*, 2006. 6(15): 1579-1591.

256. Hall, M., E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I.H. Witten, The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 2009. 11(1): 10-18.

257. Zadrozny, B. and C. Elkan. Learning and making decisions when costs and probabilities are both unknown. *Proceedings of the seventeenth international conference on knowledge discovery and data mining*. 2001.

258. Ling, C.X. and V.S. Sheng, *Cost-sensitive learning and the class imbalance problem*. 2008, Springer: Encyclopedia of machine learning.

259. Sheng, V. and C. Ling, Roulette sampling for cost-sensitive learning. *Machine Learning: ECML 2007*, 2007: 724-731.

260. Breiman, L., Bagging predictors. *Machine learning*, 1996. 24(2): 123-140.

261. Chohan, K.K., S.W. Paine, J. Mistry, P. Barton and A.M. Davis, A rapid computational filter for cytochrome P450 1A2 inhibition potential of compound libraries. *Journal of medicinal chemistry*, 2005. 48(16): 5154-5161.

262. Aher, Y.D., G. Stampfel, W. Gansterer and G.F. Ecker, *UVieCo-A distributed computing grid for life sciences applications*. 2010 (Poster presentation).

263. Didziapetris, R., P. Japertas, A. Avdeef and A. Petrauskas, Classification analysis of P-glycoprotein substrate specificity. *Journal of drug targeting*, 2003. 11(7): 391-406.

264. Seelig, A., How does P-glycoprotein recognize its substrates? *International journal of clinical pharmacology and therapeutics*, 1998. 36(1): 50-54.

265. Jaworska, J., N. Nikolova-Jeliazkova and T. Aldenberg, QSAR applicability domain estimation by projection of the training set in descriptor space: A review. *ATLA Alternatives to laboratory animals*, 2005. 33(5): 445-459.
266. Ellison, C.M., R. Sherhod, M.T.D. Cronin, S.J. Enoch, J.C. Madden and P.N. Judson, Assessment of Methods To Define the Applicability Domain of Structural Alert Models. *Journal of chemical information and modeling*, 2011.
267. Ambit Discovery, Ideaconsult Ltd., Sofia, Bulgaria. (http://ambit.acad.bg).
268. Penzotti, J.E., M.L. Lamb, E. Evensen and P.D.J. Grootenhuis, A computational ensemble pharmacophore model for identifying substrates of P-glycoprotein. *Journal of medicinal chemistry*, 2002. 45(9): 1737-1740.
269. de Cerqueira Lima, P., A. Golbraikh, S. Oloff, Y. Xiao and A. Tropsha, Combinatorial QSAR modeling of P-glycoprotein substrates. *Journal of chemical information and modeling*, 2006. 46(3): 1245-1254.
270. Cabrera, M.A., I. Gonzalez, C. Fernandez, C. Navarro and M. Bermejo, A topological substructural approach for the prediction of P-glycoprotein substrates. *Journal of Pharmaceutical Sciences*, 2006. 95(3): 589-606.
271. Huang, J., G. Ma, I. Muhammad and Y. Cheng, Identifying P-glycoprotein substrates using a support vector machine optimized by a particle swarm. *Journal of chemical information and modeling*, 2007. 47(4): 1638-1647.
272. Wang, Z., Y. Chen, H. Liang, A. Bender, R. Glen and A. Yan, P-glycoprotein Substrate Models Using Support Vector Machines Based on a Comprehensive Dataset. *Journal of chemical information and modeling*, 2011. 51(6): 1447–1456.

APPENDICES

Appendix 1 (UVieCo Administration)

In this section, the main administration tasks of central manager goedel.ani.univie.ac.at for the UVieCo pool are such as user management, starting and stopping the various applications, and activating the firewall rules.

1. Directory Structure

All the necessary files are located under the directory /usr/local/unigrid which is owned by the user gridadmin. There are several subdirectories:

bin contains various scripts for starting and stopping the various needed server applications
condor contains the Condor application in its version 7.0.1
keys the RSA keys necessary for accessing the VPN are generated and stored in this directory
logs contains the generated logfiles
openvpn contains OpenVPN version 2.0.9 built from source
pkg_factory contains all the necessary files and scripts needed for creating client packages (including user-specific RSA keys) on-the-fly
postgresql contains PostgreSQL version 7.4 built from source
tomcat6 contains Apache Tomcat version 6.0 built from source

2. User Management

User data is stored in the PostgreSQL database running on goedel. The database may be modified by logging into goedel as user gridadmin, starting the command line client, and issueing SQL statements directly. The command line client for the PostgreSQL DBMS is called psql and its usage is shown in the following lines:

```
gridadmin@goedel:~> psql -U gridadmin mydb
Password:
Welcome to psql 7.4.19, the PostgreSQL interactive terminal.
Type: \copyright for distribution terms
\h for help with SQL commands
\? for help on internal slash commands
\g or terminate with semicolon to execute query
\q to quit
mydb=> select * from users;
user_name | user_pass | role_name | submit_privilege | department |
----------------+---------------+---------------+-----------------------+-----------------+-
Yogesh       | *** | member | t | Pharma    | yogesh.aher
PhI_seminar  | *** | member | f | Pharma    | yogesh.aher
test         | *** | member | f | TEST      | test
Gerald       | *** | member | t | RLCTA     | gerald.stam
(4 rows)
```

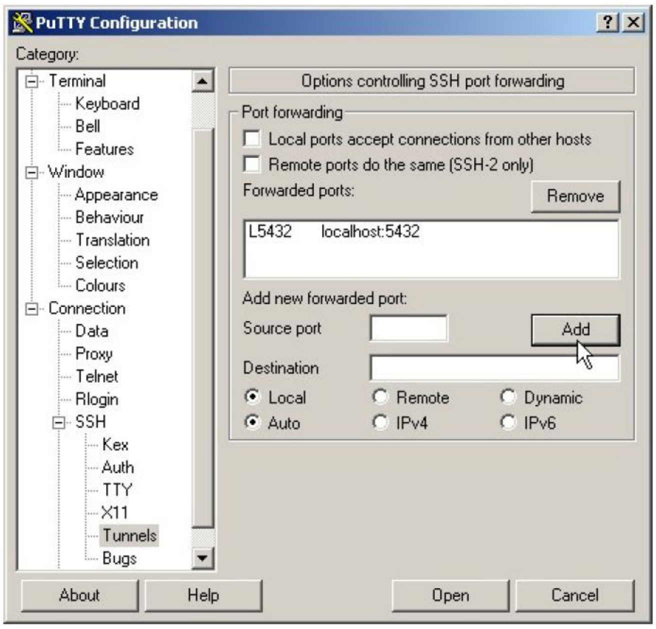
Techincal implementation of UVieCo was performed by Gerald Stampfel, Deptt. of Computer Sciences.

As can be seen in the listing above, the PostgreSQL database is called *mydb*. See the SQL documentation[1] for details on the available commands.

A more user-friendly way, however, is to use a graphical client. The procedures throughout this document are explained for the Windows client *pgAdmin*[2] which is freely available. Other clients, although, should work very similar.

As PostgreSQL is (for security reasons) configured to only accept connections stemming from the local host, it is not possible to connect to the DB without taking additional measures. Therefore, a SSH tunnel is (easily) set up using the Windows SSH client *Putty*[3] (see Figure A1.1). TCP port 5432 is the one used by PostgreSQL.

After setting up the tunnel, the database can be access using the address localhost:5432 in pgAdmin (see Figure 2).



Options controlling SSH port forwarding

Port forwarding

☐ Local ports accept connections from other hosts

☐ Remote ports do the same (SSH-2 only)

Forwarded ports:

L5432 localhost:5432

Add new forwarded port:

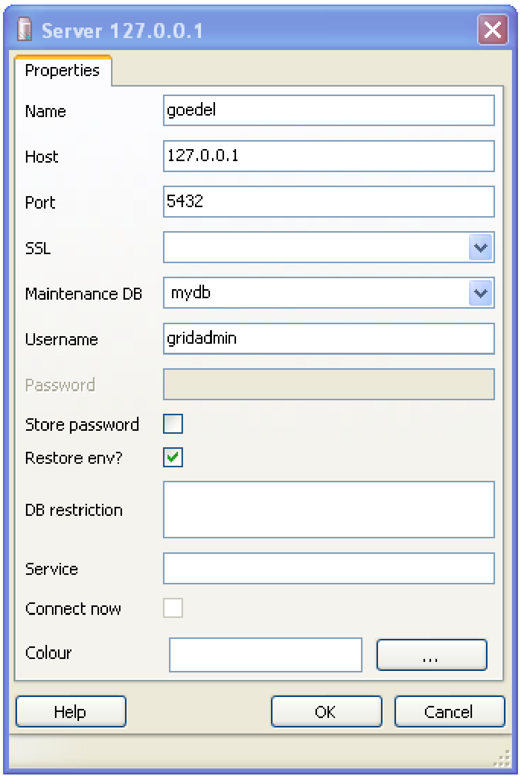
Source port

Destination

● Local ○ Remote ○ Dynamic

● Auto ○ IPv4 ○ IPv6

Figure A1.1: SSH tunnel using Putty



Name: goedel

Host: 127.0.0.1

Port: 5432

SSL:

Maintenance DB: mydb

Username: gridadmin

Password

Store password ☐

Restore env? ☑

DB restriction

Service

Connect now ☐

Colour

Figure A1.2: pgAdmin setup

---

[1] http://www.postgresql.org/docs/7.4/interactive/sql.html

[2] http://www.pgadmin.org

[3] http://www.chiark.greenend.org.uk/ sgtatham/putty/download.html

The important table for user management in the database is “Users” table (figure A1.3). The details of this table are as follows:

3.1 Create

In order to create a new user, a new row has to be created for the user table. This is simply done by filling the last line in the "Edit Data" dialog (see Figure A1.4). In the table, values for the following columns have to be specified:
OID: left empty, PostgreSQL internal data
User_name: name of the user (as he will be displayed on the status web site)
User pass: clear-text password
Role_name: always has to be "member", this is for the basic authentication in Tomcat which checks every access to restricted web sites against this table
Submit privilege: is the user allowed to submit jobs to the condor pool?
Department: university department the user is part of
Email: e-mail address

Note: Changing a user's name will also lock this user out of the VPN. OpenVPN checks the keys presented by the user against this database and if the supplied user name does not exist, access is denied. Therefore, changing the name requires the user to fetch new keys.
Also note that changing *submit_privilege* for a specific user also requires this user to fetch new keys.

3.2 Delete

Deleting a user is as simple as deleting the corresponding row in the "EditData" dialog (see Figure 4).

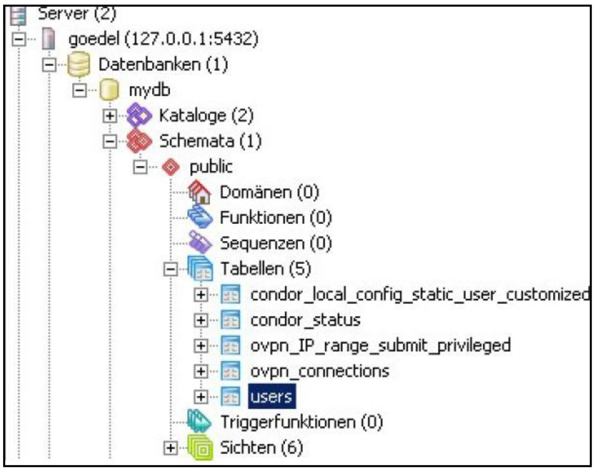
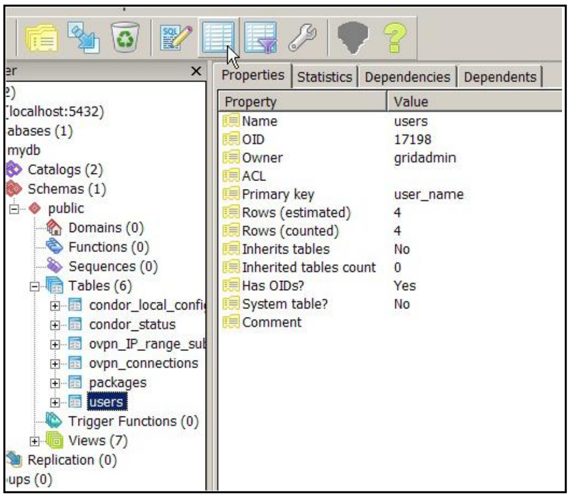


Figure A1.3: Users Table



Property	Value
Name	users
OID	17198
Owner	gridadmin
ACL	
Primary key	user_name
Rows (estimated)	4
Rows (counted)	4
Inherits tables	No
Inherited tables count	0
Has OIDs?	Yes
System table?	No
Comment	

Figure A1.4: Edit Data

4 Server Daemon Control

The machine goedel acts (in the context of UVieCo) as a Condor central manager, VPN gateway, database server, web server, and firewall is constantly running the following applications: Condor, OpenVPN, PostgreSQL, and Apache Tomcat.
The applications have to be stopped in the right order: Tomcat and Condor first, then OpenVPN, and PostgreSQL last. Consequently, this order has to be reversed when starting the applications.

For controlling the applications, one has to login as gridadmin and use the start/stop scripts located in */usr/local/unigrid/bin*: *startTC.sh, stopTC-.sh* for Tomcat; *startCONDOR.sh, stopCONDOR.sh* for Condor; *startPSQL.sh, stopPSQL.sh* for PostgreSQL; *startOVPN.sh, stopOVPN.sh* for OpenVPN. *startTC.sh* and *stopTC.sh* ask for the root password, this is necessary for Tomcat to be able to bind to TCP port 443. Once started, Tomcat drops the root privileges and proceeds as user gridadmin.

5 Firewall

A bash script called *fwProtect.sh* which sets up the necessary iptables rules is located in */usr/local/unigrid/bin*. This script identifies three types of pool participants: the central manager, submit machines (in the 10.8.1.0/24 subnet), and execute-only machines (10.8.0.0/24 subnet). The iptables rules basically enable communication between the central manager and the other two parties. Furthermore, all submit machines are able to ping each other and the execute machines, but execute machines are disallowed to "see" each other in the VPN.

All the denied packets are recorded in */var/log/firewall*.

All allowed communication paths (except for ICMP naturally) are restricted regarding their source and destination ports.

6 Crontab

The crontab of user gridadmin should look like this:

```
gridadmin@goedel:~> crontab -l
* * * * * /usr/local/unigrid/bin/update_status_db.sh
&> /dev/null
0 4 * * *
/usr/local/unigrid/pkg_factory/deletePkgTempDir.sh &> /dev/null
```

update_status db.sh pastes the output of condor status into the PostgreSQL database. These lines are then parsed by the JSP pages and displayed accordingly.

deletePkgTempDir.sh deletes the client packages which were prepared for downloading in */usr/local/unigrid/tomcat6/webapps/unigrid/download/tmp.**. As soon as a user logs on to the web interface's download site, his personal keys are packaged together with the UVieCo client scripts. These package file are useless once a user has downloaded them, therefore, this cron task deletes these temporary directories.

Appendix 2 (UVieCo Implementation)

This section describes the software setup of the *central manager* and the *client machines*. The detailed configuration of the applications Condor, OpenVPN, Apache Tomcat, and PostgreSQL are explained. Additionally, the bash scripts on the client and server side are shortly described.

A2.1. Central Manager

Before going into the details of configuration, a few more words on the VPN and authentication follow: For a machine to participate in the VPN, authentication has to be conducted. This is done via public/private key pairs. Consequently, a client machine has to show the OpenVPN server that it holds a valid keypair before being allowed access to the VPN. Moreover, this client-specific key pair has to be somehow brought to the client in a secure fashion.

The approach chosen here is to create and store the RSA keys on a password-protected SSL web server (Figure A2.1). This method is sufficiently secure and user-friendly.

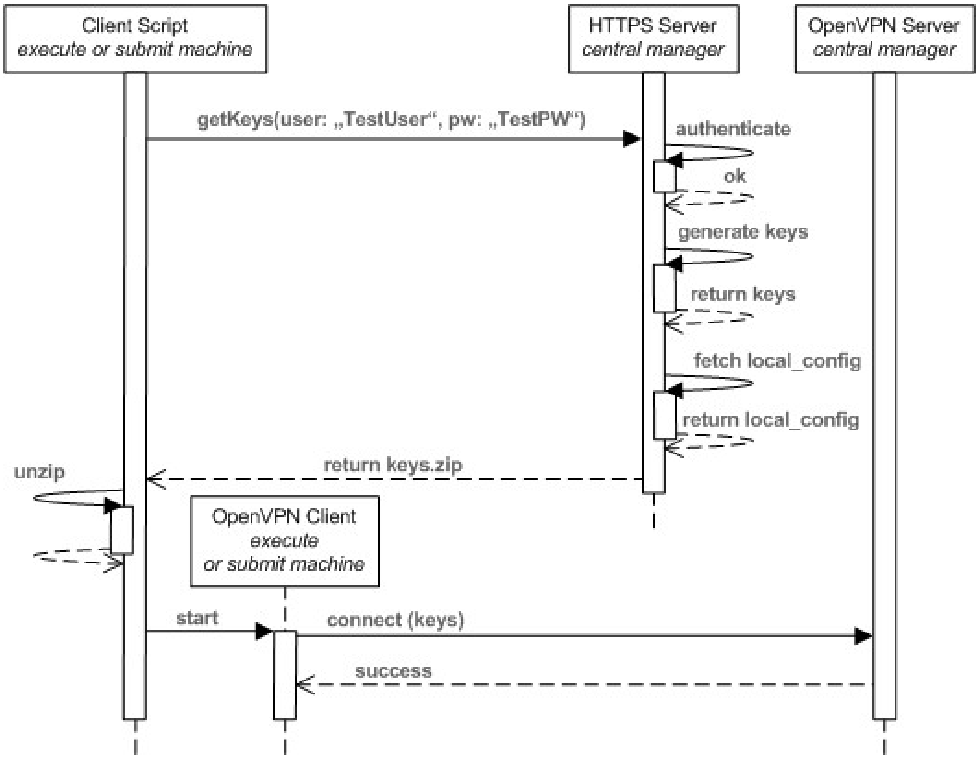


Figure A2.1: Successful connect sequence for UVieCo.

As can be seen in Figure A2.1, the client also receives a “local config”. This refers to a small file which specifies additional Condor configuration rules specific for that user.

A2.1.1 Directory Structure

All the necessary files are located under the directory */usr/local/unigrid* which is owned by the user gridadmin. There are several subdirectories:

bin : Contains various scripts for starting and stopping the server applications
condor: Contains the Condor application in its version 7.0.1
keys: The RSA keys necessary for accessing the VPN are generated and stored in this directory
logs: Contains the generated log files
openvpn: Contains OpenVPN
pkg_factory: Contains all the necessary files and scripts needed for creating the client packages (including user-specific RSA keys) on-the-fly
postgresql: Contains PostgreSQL version 7.4
tomcat6: Contains Apache Tomcat version 6.0

A2.1.2 OpenVPN

We are using OpenVPN version 2.0.9 which was built from source.

```
# openvpn --help
OpenVPN 2.0.9 x86_64-suse-linux [SSL] [LZO] [EPOLL]
built on Apr 20 2008
(more)
```

A2.1.2.1 Setup

For the setup of the PKI infrastructure as described in the OpenVPN documentation *(http://openvpn.net/howto.html#pki)*, OpenSSL had to be installed:

```
# openssl version
OpenSSL 0.9.8a 11 Oct 2005
```

OpenSSL was obtained as a binary. Following the OpenVPN documentation ("Generate the master Certificate Authority (CA) certificate & key"), a certification authority was set up:

```
. ./vars
./clean-all
./build-ca
./build-dh
```

When asked various questions by the build-ca script, the following data was provided:
C=AT, ST=Vienna, L=Vienna, O=University of Vienna,
CN=Unigrid-CA/emailAddress=gerald.stampfel@gmail.com

After this, a key pair and a certificate for the server were created:

```
./build-key-server server
```

When asked various questions by the build-key-server script, the following data was provided:

C=AT, ST=Vienna, O=University of Vienna,
CN=Unigrid OpenVPN Server/emailAddress=gerald.stampfel@gmail.com

All the generated keys, certificates, and the *vars* were placed in */unigrid/keys*.

Next, the OpenVPN configuration file and its most important statements are described:

File name : server.conf
Location : /usr/local/unigrid/openvpn/etc
Used by : OpenVPN
Contents **:**

```
cd /usr/local/unigrid
daemon
server 10.8.0.0 255.255.255.0
port 1194
proto udp
dev tun

ca keys/ca.crt
cert keys/server.crt
key keys/server.key
dh keys/dh1024.pem

ifconfig-pool-persist openvpn/logs/ipp.txt
route 10.8.1.0 255.255.255.0
push "route 10.8.1.0 255.255.255.0"
client-to-client
duplicate-cn
keepalive 10 100
comp-lzo
user stampfel
group nogroup
verb 3
status openvpn/logs/openvpn-status.log

persist-key
persist-tun
log openvpn/logs/openvpn.log
client-connect bin/ovpn_clientconnect.sh
client-disconnect bin/ovpn_clientdisconnect.sh
```

The cd directive tells OpenVPN its working directory, which is the location of the *unigrid* directory in this case. The *server* directive defines that IPs of the 10.8.0.0/24 subnet should be leased to connected clients. *ca, cert, key, dh* point to the OpenSSL keys, certificates and other files we generated and placed under *unigrid/keys*. The *route* and *push route* commands are needed for telling the clients that also the subnet 10.8.1.0/24 (used by privileged clients) is accessible through the VPN tunnel and has to be entered in their routing tables.

In order to allow each client connected to the VPN to communicate with each other client, the directive *client-to-client* has to be stated. Client-to-client communication although is very restricted by our firewall rules, see Section A2.8. *duplicate-cn* allows two clients to connect with the same key. *keepalive* 10 100 tells the client to ping each server every 10 seconds and to disconnect those not responding after 100 seconds. OpenVPN has to be started as root because some operations such as changes to the routing table or networking interface configuration need these privileges. After initialization, although, the OpenVPN process changes its user ID to the one passed by *user*. After each connect or disconnect of a client, the scripts *bin/ovpn_clientconnect.sh* and *bin/ovpn_clientdisconnect.sh* are called by OpenVPN. These are further described in the next section.

The OpenVPN server is started using *sudo openvpn unigrid/openvpn/etc/server.conf*. The command sudo has to be used in order to start OpenVPN with root-privileges which is necessary for various network configurations.

A2.1.2.1 Scripts

build-key A modified version of the build-key script supplied with OpenVPN is used to generate the RSA keys non-interactively:

Script name : build-key
Location : /usr/local/unigrid/keys
Description : Generate RSA keys for OpenVPN
Workflow : Generate and sign RSA keys using OpenSSL → Fetch Condor local config from PostgreSQL → Pack everything into .zip
Used by : Apache Tomcat

ovpn_clientconnect.sh The script *ovpn_clientconnect.sh* is responsible for deciding if a connecting user is someone with submit privileges and should therefore receive a 10.8.1.0/24 IP or an execute-only user in the normal IP subnet of 10.8.0.0/24.

Script name : ovpn-clientconnect.sh
Location : /usr/local/unigrid/bin
Description : Authentication of VPN clients and insert of client data into database
Workflow : Query DB if supplied username does exist → Query DB if connecting user has submit privileges → Tell OpenVPN to assign an IP based on the outcome of the previous query → Insert client IP and connect time into DB
Used by : OpenVPN
Relies on : PostgreSQL

ovpn_clientdisconnect.sh This is a simple script which deletes the user entry from the table ovpn connections which was inserted upon the execution of *ovpn_clientconnect.sh*

Script name : ovpn-clientdisconnect.sh
Location : /usr/local/unigrid/bin

Description : Deletes user entry from DB.
Used by : OpenVPN
Relies on : PostgreSQL

A2.1.3 PostgreSQL

We are using PostgreSQL version 7.4.19 which was built from source. The build was installed to the directory *unigrid/postgresql*.

```
# postgres -V
postgres (PostgreSQL) 7.4.19
```

The tool pgAdmin was very valuable during for the management of the PostgreSQL database.

A2.1.3.1 Setup

After building (*./configure --prefix=/usr/local/unigrid/postgresql*), adatabase was created via

```
postgresql/bin/initdb -D postgresql/data
```

The file *postgresql/data/pg hba.conf* was edited to allow connections from other hosts without supplying a password. This temporary measure was necessary in order to remote administer the database via pgAdmin:

```
# cat postgresql/data/pg_hba.conf
(more)
host all all XX.YY.ZZ.AA 255.255.255.255 trust
(more)
```

Also the configuration fie *postgresql/data/postgresql.conf* needed a few (self-explanatory) modifications:

```
tcpip_socket = true
virtual_host = '127.0.0.1'
log_statement = true
```

Afterwards, the database was started using

```
# postgresql/bin/pg_ctl
-D /usr/local/unigrid/postgresql/data
-l /usr/local/unigrid/postgresql/logs/log start
```

Using the tool pgAdmin and authenticating as user postgres (without password), a new user “gridadmin” and various tables and views were created by this user. These tables include the following:

users: A list of all users and their passwords, e-mail addresses, departments and a boolean value telling if they are privileged for submitting jobs to the Condor pool.

ovpn_connections: A snapshot of the current OpenVPN connection pool. This table is used by the scripts *bin/ovpn_clientconnect.sh* and *bin/ovpn_clientdisconnect.sh* described in Section A2.1.2.

ovpn_IP_range_submit_privileged: A list of all the possible IP addresses in the 10.8.1.0/24 subnet. OpenVPN needs two IP addresses for each client, 10.8.1.1 and 10.8.1.2 for example.

condor_local_config_static_user_customized: Not yet used.

condor_status: An up-to-date dump of the command condor status. This data is used by the web interface to display the current status of the pool.

Additionally, the following views have been created:

Unused_privileged_submit_IPs: This view joins the tables - ovpn_connections and ovpn_IP_range_submit_privileged to find out which IP addresses out of the submit privileged subnet are not yet assigned. Used by *bin/ovpn_clientconnect.sh*.

condor_local_config: This view's output are two columns, a username and a Condor local config which is stored and archived by the script *keys/build_key.sh*. In its archived form it is transported to the client via HTTPS and there extracted to the Condor directory of the user. This is therefore a very comfortable way of making additions to clients' local Condor configuration files.

This view integrates the local config directives from the views condor_local_config_submit_privileged_users, condor_local_config_submit_non-privileged_ users and condor_local_config_depts.

condor_local_config_submit_privileged_users: Outputs Condor config directives for submit privileged pool members such as for starting the Condor scheduler.

condor_local_config_submit_non-privileged_users:

condor_local_config_depts: This view pastes the information about a user's department from the table users and transforms it into a Condor configuration directive in order for Condor to know which machine belongs to which department.

users_online_time: This view joins the tables - users and ovpn_connections and additionally displays since when each user is online in seconds. This data is used by the web sites to display the current status of the pool.

A2.1.4 Tomcat

We are using Apache Tomcat version 6.0.16 which was built from source. The build was installed to the directory *unigrid/tomcat6*. A Java Virtual Machine version 1.5 was obtained from Sun Inc. *(www.sun.com).*

```
# java -showversion
java version "1.5.0_15"
Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_15-b04)
Java HotSpot(TM) 64-Bit Server VM (build 1.5.0_15-b04, mixed mode)
(more)
```

A2.1.4.1 Setup

In order to setup a SSL web server, the instructions from the manual were followed *(http://tomcat.apache.org/tomcat-6.0-doc/ssl-howto.html)*

```
# $JAVA_HOME/bin/keytool -genkey -alias tomcat -keyalg RSA
-keystore /usr/local/unigrid/keys/tomcat6/.keystore
```

The keystore password has to be remembered for the following step. In order to configure Tomcat to start a SSL server, the following lines were added to *tomcat6/conf/server.xml*:

```
<Connector port="443" protocol="HTTP/1.1" SSLEnabled="true"
maxThreads="150" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS"
keystoreFile="/usr/local/unigrid/keys/tomcat6/.keystore"
keystorePass="PASSWORD"
/>
```

As mentioned before, the SSL server which is used to access users' RSA keys may only be accessed with the right username and password entered into the browser. We therefore configured tomcat to authenticate against the PostgreSQL database which is easier to handle than a text file. The following lines were added to *tomcat6/conf/server.xml*:

```
<Realm className="org.apache.catalina.realm.JDBCRealm" debug="99"
driverName="org.postgresql.Driver"
connectionURL="jdbc:postgresql://127.0.0.1/mydb?user=gridadmin
&amp;password=PASSWORD"
userTable="users" userNameCol="user_name" userCredCol="user_pass"
userRoleTable="users" roleNameCol="role_name"
/>
```

The Tomcat server also needs the PostgreSQL JDBC .jar to connect to the database. Thus, *postgresql-8.3-603.jdbc3.jar10* (obtained from: *http://jdbc.postgresql.org/download.html*) was placed under *tomcat6/lib*.

A2.1.4.2 JSP Scripts - Webapps

certs. The RSA generated by OpenSSL are fetched from a SSL web server. The JSP page responsible for providing the RSA keys is *certs/index.jsp*.

Script name : certs/index.jsp
Location : /usr/local/unigrid/tomcat6/webapps/
Description : Builds (by calling the *build-key* script, see Section A2.1.2) and returns the RSA keys for an authenticated user
Workflow : Call *build-key* → return generated .zip file
Used by **:** Various client scripts such as *fetchKeys.sh* (Linux) and *unigrid_fetchkeys.exe* (Windows)

unigrid. This web application is the user's interface to the Condor pool.

Script name : unigrid/*.jsp
Location **:** /usr/local/unigrid/tomcat6/webapps/
Description **:** Displays pool status, provides client package downloads, job submission, additional information regarding UVieCo, etc.

A2.1.5 Condor

We are using Condor version 7.0.1 which was obtained in binary form (*http://www.cs.wisc.edu/condor/).* The build was installed to the directory *unigrid/condor*.

A2.1.5.1 Setup

The *condor/etc/condor_config* contains a multitude of settings with a lot of them unchanged from the default config. Thus, only the modified ones are provided here.

File name : condor config
Location : /usr/local/unigrid/condor/etc
Description : Condor configuration file
Contents :

```
RELEASE_DIR                     = /usr/local/unigrid/condor
LOCAL_DIR                       = $(RELEASE_DIR)
LOCAL_CONFIG_FILE               = $(RELEASE_DIR)/etc/condor_config.local
REQUIRE_LOCAL_CONFIG_FILE       = TRUE

NETWORK_INTERFACE               = 10.8.0.1
CONDOR_HOST                     = 10.8.0.1

UID_DOMAIN                      = $(FULL_HOSTNAME)
FILESYSTEM_DOMAIN               = $(FULL_HOSTNAME)

COLLECTOR_NAME                  = “UNIVIE Condor Pool”
DEFAULT_DOMAIN_NAME             = unigrid
NO_DNS                          = True

FLOCK_FROM                      =
FLOCK_TO                         =
FLOCK_NEGOTIATOR_HOSTS           = $(FLOCK_TO)
FLOCK_COLLECTOR_HOSTS            = $(FLOCK_TO)
HOSTALLOW_ADMINISTRATOR          = $(CONDOR_HOST)
HOSTALLOW_OWNER                  = $(FULL_HOSTNAME),

$(HOSTALLOW_ADMINISTRATOR)
HOSTALLOW_READ                   = 10.8.*
HOSTALLOW_WRITE                  = $(CONDOR_HOST), 10.8.1.*
HOSTALLOW_CONFIG                 = $(FULL_HOSTNAME)
HOSTALLOW_WRITE_COLLECTOR        = $(HOSTALLOW_WRITE),
$(FLOCK_FROM), 10.8.0.*
HOSTALLOW_WRITE_STARTD           =$(HOSTALLOW_WRITE),
$(FLOCK_FROM)
HOSTALLOW_READ_COLLECTOR         =                 $(HOSTALLOW_READ),
$(FLOCK_FROM)
HOSTALLOW_READ_STARTD            =                 $(HOSTALLOW_READ),
$(FLOCK_FROM)
HOSTALLOW_NEGOTIATOR             = $(CONDOR_HOST)
HOSTALLOW_NEGOTIATOR_SCHEDD = $(CONDOR_HOST),
$(FLOCK_NEGOTIATOR_HOSTS)
```

RELEASE_DIR and *LOCAL_DIR* tell Condor where its software resides. We need Condor only to bind to the VPN interface 10.8.0.1 (*NETWORK_INTERFACE*) as it is not intended for machines outside the VPN to join the pool. We are expecting our clients to be a heterogeneous mixture of operating systems and usage patterns such as dedicated servers or workstations. Therefore, we will not have common UID domain and set *UID_DOMAIN* to $(FULL_ HOSTNAME). This ensures that Condor will not think for any pair of clients to share the same UID domain. The same applies to *FILESYSTEM_DOMAIN*, no common filesystem is operated.

We also do not operate a shared DNS server (*NO_DNS*) and our default hostnames will be IP.unigrid (*DEFAULT_DOMAIN_NAME*), e.g. "10-8-0-1.unigrid".

Flocking is disabled (FLOCK_* directives). Everybody in the VPN is allowed to read from the Collector (*HOSTALLOW_READ*), but only the central manager and the privileged subnet 10.8.1.0/24 is by default allowed to perform write operations such as submitting jobs (*HOSTALLOW_WRITE*). Additionally, the 10.8.0.0/24 subnet is allowed to write to the collector *HOSTALLOW WRITE COLLECTOR*.

File name : condor_config.local
Location : /usr/local/unigrid/condor/etc
Description : Condor local configuration file
Contents :

```
DAEMON_LIST = COLLECTOR, MASTER, NEGOTIATOR, SCHEDD
JAVA = /home/stampfel/jdk1.5/bin/java
JAVA_MAXHEAP_ARGUMENT = -Xmx
CONDOR_ADMIN=gerald.stampfel@gmail.com
LOWPORT=19000
HIGHPORT=20050
```

A2.1.6 Logging

The log files of all the applications running in the server distribution are gathered under *unigrid/logs* as symbolic links. The script *tailLogs.sh* can be used to monitor all these files simultaneously which has proved very useful during debugging.

A2.1.7 Sudo

In order for a user to issue commands with root-privileges, this user either has to be root or use *sudo*. In our situation it is more convenient to operate non-interactively and use *sudo* which enables root-privileges without entering the root password. The following line has to be added (by root) to */etc/sudoers* for this to work:

```
gridadmin ALL= (root) NOPASSWD:
/usr/local/unigrid/openvpn/sbin/openvpn
```

This can be done using the command *visudo*.

A2.1.8 Firewall

Only communication between the Condor daemons running on central manager, submit, and execute machines should be allowed to pass through the VPN, all other traffic (with the exception of a few ICMP cases) is denied. The Condor daemons use defined port ranges (both TCP and UDP):

Central manager : 19000-20050 and 9618 for condor collector
Submit machines : 19000-20200
Execute machines : 19000-19200

The required size for these port ranges depending on the machine type has been calculated using the Condor manual *(http://www.cs.wisc.edu/condor/manual/v7.0/)*.
Traffic inside the VPN is also filtered according to the involved IP addresses. The following scheme is used:

Central manager : 10.8.0.1
Submit machines : 10.8.1.*
Execute machines : 10.8.0.*

Based on the above port and IP restrictions, iptables rules have been set up to deny all other traffic:

Script name : fwProtect.sh
Location : /usr/local/unigrid/bin
Description : Call iptables
Used by : None. Has to be called manually

A2.1.9 Environment Variables

Environmental variables used by scripts under *unigrid/bin/* are set in the file *bin/vars*.

Script name : vars
Location : /usr/local/unigrid/bin
Description : The script sets various environment variables which are subsequently used by scripts in *bin/*
Used by : scripts in /usr/local/unigrid/bin and Condor

A2.1.10 Client Package Generation

A2.1.10.1 Linux Packages

The client packages containing all the necessary files to connect a client (execute or submit) machine to the Condor pool are generated on goedel. This is done in the directory */usr/local/unigrid/pkg_factory*. The subdirectories debian+condor and debian-condor contain the files for each distribution form. The script *preparePackage.sh* is used to compress these archives into .deb Debian/Ubuntu Packages:

Script name : preparePackage.sh
Location : /usr/local/unigrid/pkg factory
Description : Creates .deb packages of the scripts required to connect to the Condor pool for a single user
Workflow : Call build-key script → Call *createDeb.sh* script which prepares the directories and creates the archives → Copy the generated packages to a temporary directory from where they can be picked up by Tomcat → Store the path of the temporary directory in the DB so Tomcat knows where to find them
Used by : Apache Tomcat

In order to generate the packages for all users in the *DB*, *prepareAllPackages.sh* has to be called.
The client package comes in two flavors: With and without Condor. Because the client scripts are the same for both packages it would be unnecessary to have the same scripts in debian+condor and debian-condor. Therefore, the scripts only exist in debian-condor and are copied to debian+condor by createDeb.sh:

Script name : createDeb.sh
Location : /usr/local/unigrid/pkg_factory
Description : Prepares the *debian+condor* and *debian-condor* directories and calls dpkg-deb
Workflow : Copy from debian-condor to *debian+condor* → Chmod and chown the package directories → Create archives using the commands dpkg-deb and ar
Used by : fakeCreateDeb.sh

A2.1.10.2 Windows (coLinux) Package

For the Windows installation, a modified version of the coLinux installer including a custom root image was used. The Nullsoft Scriptable Install System (NSIS), open source software, was used to generate the client package. The .nsis script used to generate the package is located in */usr/local/unigrid/NSIS* on goedel.

A2.2 Client Machines

A2.2.1 Directory Structure

All the necessary files are located in the home directory of the user griduser (*/home/griduser/unigrid*). There are several subdirectories:

bin: Contains various scripts for starting and stopping the OpenVPN client and Condor
condor: Contains the Condor application or the Condor configuration files only (depending on the type of installation package used)
keys: The RSA keys necessary for accessing the VPN are stored in this directory
logs: Contains the generated logfiles

Each of these directories will be described in the following sections.

A2.2.2 Condor

We are using Condor version 7.0.1.

The file *condor/etc/condor_config* is essentially the same as the one for the central manager described in Section A2.1.5. *condor/etc/condor_config.local* is obtained from the central manager together with the keys upon every connection. *condor/etc/condor_config.local.additional* is an additional configuration file untouched by the various scripts and is therefore an occasion to define persistent machine-specific settings not defined in *condor/etc/condor_config.local*.

A2.2.3 OpenVPN

We are using OpenVPN 2.0.9.

```
# openvpn --help
OpenVPN 2.0.9 i486-pc-linux-gnu [SSL] [LZO] [EPOLL]
built on Sep 20 2007
(more)
```

A2.2.4 Connection Establishment

The client connection process is initiated with the script *bin/unigrid connect.sh* which starts the Condor application once the VPN is successfully established.

Script name : unigrid_connect.sh
Location : /home/griduser/unigrid/bin
Description : Connect to the VPN and start Condor
Workflow : Check if running inside coLinux → Check if RSA keys are present → Check if OpenVPN and Condor are already running and kill them if they are → Start *unigrid_keepalive.sh* in a new thread and tell it to keep OpenVPN running → Check if the script got called at bootup and exit if it was → Otherwise, check if the connection is coming up and tell the user what may be wrong if it does not!

The OpenVPN binary is started with various parameters. The parameter *--user* tells OpenVPN to change its process owner to *$OVPN USER* after the connection is established. *--ca, --cert,* and *--key* tell the client where to find the RSA keys. *--route-up* tells OpenVPN to start Condor as soon as the VPN connection is established.

The OpenVPN client is started via *sudo*. The command *sudo* has to be used in order to start OpenVPN with root-privileges which are necessary for various network configurations.

Once OpenVPN has successfully established a connection, *unigrid_startCondor.sh* is executed.

Script name : unigrid_startCondor.sh
Location : /home/griduser/unigrid/bin
Description : Start Condor
Used by : unigrid_connect.sh

Another important script in this context is *unigrid_keepalive.sh*.

Script name : unigrid_keepalive.sh
Location : /home/griduser/unigrid/bin
Description : In an endless loop: Start OpenVPN (which in turn, once successfully launched) calls *unigrid_startCondor.sh*
Workflow : Check if /dev/null is ready (important at bootup) → keep restarting OpenVPN unless it fails in less than two seconds (which would mean something is broken)
Used by : unigrid_connect.sh

Another important use case of the client package is to fetch RSA keys from the SSL web server. This is done via *unigrid_fetchKeys.sh*.

Script name : unigrid_fetchKeys.sh
Location : /home/griduser/unigrid/bin
Description : Ask the user for a username and password and fetch new keys and Condor local config from the web server
Workflow : Ask for a username and password → Fetch the keys using wget → Extract the keys → Write to file *unigrid_user*

Finally, *unigrid_user*:

Script name : unigrid_user
Location : /home/griduser/unigrid
Description : A textfile holding exactly one line which contains the username supplied to *unigrid_fetchKeys.sh*; this is necessary for *unigrid_connect.sh* to know which keys to use
Used by : unigrid_connect.sh

A2.2.5 coLinux Package

The coLinux installer basically contains a Debian 4.0 root image with all the necessary auxiliary packages and the client .deb package installed. The scripts are therefore essentially the same.

One difference, however, when running under Windows is for Condor to determine if the machine is idle or busy. Under a native Linux, Condor may access the CPU status and the keyboard and mouse activity. When running inside coLinux, however, the virtual machine has no access to the Windows keyboard and mouse usage. The same applies for the CPU.

This problem was solved with a small VBScript script running as a Windows service (“*colinux_monitor*") which writes the current performance data to a directory shared by Windows and the coLinux virtual machine. Inside coLinux, the script *unigrid_monitor.sh*, which is periodically called by Condor, reads out the Windows performance data.

Script name : unigrid monitor.sh
Location : /home/griduser/unigrid/bin
Description : Reads the files *non_condor_load.out* and *idle_ticks.out* in */mnt/wininfo*.
Used by : Condor

Appendix 3

Perl script to calculate Pearson correlation coefficient for P-gp transporter

The gene expression values for NCI's 60 cell lines in the beginning of the script were taken from Szakács et. al. For calculating PCC values of rest of the 47 transporters, these values need to be replaced from the same supplementary table for respective gene.

```
%abcb1=(      'MCF7' => -2.299487776,
              'MDA-MB-231/ATCC' => -1.703005883,
              'HS 578T' => 0.129259872,
              'MDA-N' => -1.136008003,
              'BT-549' => -0.620474003,
              'T-47D' => -1.869259753,
              'SNB-19' => -1.98375456,
              'SNB-75' => -3.051216652,
              'U251' => -2.091767493,
              'SF-268' => -1.937680806,
              'SF-295' => 2.472240998,
              'SF-539' => -1.91340836,
              'HT29' => -1.863403683,
              'HCC-2998' => -2.134894428,
              'HCT-116' => -2.389089161,
              'SW-620' => 3.193448512,
              'COLO 205' => -1.951550775,
              'HCT-15' => 11.08019765,
              'KM12' => -1.544163917,
              'CCRF-CEM' => 1.916771756,
              'K-562' => 0.006278353,
              'MOLT-4' => -1.741896161,
              'HL-60(TB)' => 0.534639648,
              'RPMI-8226' => -0.731035406,
              'SR' => -1.87549267,
              'MDA-MB-435' => -1.256649441,
              'LOX IMVI' => -1.591674614,
              'MALME-3M' => -1.134403827,
              'SK-MEL-2' => -0.600740128,
              'SK-MEL-5' => 1.310883616,
              'SK-MEL-28' => -1.20765274,
              'M14' => 2.743252037,
              'UACC-62' => -1.159035406,
              'UACC-257' => -1.667526634,
              'NCI-H23' => -1.792915897,
              'NCI-H522' => -0.602394263,
              'A549/ATCC' => -1.082718529,
              'EKVX' => 3.57602469,
              'NCI-H226' => -2.463504366,
              'NCI-H322M' => -1.822665326,
              'NCI-H460' => 3.050689884,
              'HOP-62' => -1.96839288,
              'HOP-92' => -2.029790915,
              'NCI/ADR-RES' => 12.27898868,
              'OVCAR-3' => -2.635960506,
              'OVCAR-4' => -2.631633021,
              'OVCAR-5' => -2.387480974,
              'OVCAR-8' => 1.940847809,
              'IGROV1' => 1.90702192,
              'SK-OV-3' => -1.476238266,
              'PC-3' => -2.498270448,
              'DU-145' => -0.599140669,
              'UO-31' => 4.680844634,
              'SN12C' => -2.15677922,
              'A498' => 4.313310008,
              'CAKI-1' => 8.236607264,
              'RXF 393' => -1.222590269,
                '786-0' => 2.842396738,
                'ACHN' => 3.689268862,
                'TK-10' => -1.077225108 );

@cl_abcb1=keys(%abcb1);

open (FI,"c:\\yogesh\\nci-rawdata_ids.txt") || die "err file 1\n";
while ($line=<FI>)
{
        $line =~s/\s+$//;
        $mol_id=$line;


        open (FI2,"c:\\yogesh\\nci-rawdata.txt") || die "err data file\n";
        while ($line=<FI2>)
        {
                $line =~s/\s+$//;
                @line=split(/,/,$line);

                $id=$line[0];
                chop $id;

                if ($id eq $mol_id)
                {
                        $ct++;
                        $cl=$line[4];
                        $conc=$line[7];

                        chop $cl;
                        chop $conc;

                        $mol_conc{$cl}=$conc;

                }
        }
        close FI2;

        $x=0;

        foreach $k(keys(%mol_conc))
        {
                foreach(@cl_abcb1)
                {
                        if ($_ eq $k)
                        {
                                $mol_data[$x]=$mol_conc{$k};
                                $abcb1_data[$x]=$abcb1{$k};

                                $x++;
                        }
                }
        }

        %mol_conc=();

        $tot=@mol_data;


        $sum_mol=0;
        $sum_abcb1=0;
        $prod1=0;

        for($j=0;$j<@mol_data;$j++)
        {
                $sum_mol=$sum_mol+$mol_data[$j];
                $sum_abcb1=$sum_abcb1+$abcb1_data[$j];
                $prod1=$prod1+$mol_data[$j]*$abcb1_data[$j];
        }

        $mean_mol=$sum_mol/$tot;
```

```
$mean_abcb1=$sum_abcb1/$tot;
$prod1=$prod1/$tot;

$prod2=$mean_mol*$mean_abcb1;

$covar=$prod1-$prod2;

$dev_mol_total=0;
$dev_abcb1_total=0;

for($j=0;$j<@mol_data;$j++)
{
        $dev_mol=$mean_mol-$mol_data[$j];
        $dev_mol_total=$dev_mol_total+($dev_mol*$dev_mol);

        $dev_abcb1=$mean_abcb1-$abcb1_data[$j];
        $dev_abcb1_total=$dev_abcb1_total+($dev_abcb1*$dev_abcb1);
}

$stdev_mol=sqrt($dev_mol_total/($tot-1));
$stdev_abcb1=sqrt($dev_abcb1_total/($tot-1));

if (($stdev_mol !=0) and ($stdev_abcb1 !=0))
{
        $pcc=$covar/($stdev_mol*$stdev_abcb1);
}
else
{
        $pcc="Invalid";
}

@mol_data=();
@abcb1_data=();


print "$mol_id\t$pcc\n";

}
```

Abstract

In the current drug discovery process, the identification of new target proteins and potential ligands is very tedious, expensive and time-consuming. Thus, use of *in silico* techniques is of utmost importance and proved to be a valuable strategy in detecting complex structural and bioactivity relationships. Increased demands of computational power for tremendous calculations in scientific fields and timely analysis of generated piles of data require innovative strategies for efficient utilization of distributed computing resources in the form of computational grids. Such grids add a new aspect to the emerging information technology paradigm by providing and coordinating the heterogeneous resources such as various organizations, people, computing, storage and networking facilities as well as data, knowledge, software and workflows.

The aim of this study was to develop a university-wide applicable grid infrastructure, UVieCo (University of Vienna Condor pool) which can be used for implementation of standard structure- and ligand-based drug discovery applications using freely available academic software. Firewall and security issues were resolved with a virtual private network setup whereas virtualization of computer hardware was done using the CoLinux concept in a way to run Linux-executable jobs inside Windows machines. The effectiveness of the grid was assessed by performance measurement experiments using sequential and parallel tasks.

Subsequently, the association of expression/sensitivity profiles of ABC transporters with activity profiles of anticancer compounds was analyzed by mining the data from NCI (National Cancer Institute). The datasets generated in this analysis were utilized with ligand-based computational methods such as shape similarity and classification algorithms to identify and separate P-gp substrates from non-substrates. While developing predictive classification models, the problem of imbalanced class distribution was proficiently addressed using the cost-sensitive bagging approach. Applicability domain experiment revealed that our model not only predicts NCI compounds well, but it can also be applied to drug-like molecules. The developed models were relatively simple but precise enough to be applicable for virtual screening of large chemical libraries for the early identification of P-gp substrates which can potentially be useful to remove compounds of poor ADMET properties in an early phase of drug discovery.

Additionally, shape-similarity and self-organizing maps techniques were used to screen in-house as well as a large vendor database for identification of novel selective serotonin reuptake inhibitor (SSRI) like compounds to induce apoptosis. The retrieved hits possess novel chemical scaffolds and can be considered as a starting point for lead optimization studies.

The work described in this thesis will be useful to create distributed computing environment using available resources within an organization and can be applied to various applications such as efficient handling of imbalanced data classification problems or multistep virtual screening approach.

Zusammenfassung

Im derzeitigen Drug Discovery Prozess ist die Identifikation eines neuen Targetproteins und dessen potenziellen Liganden langwierig, teuer und zeitintensiv. Die Verwendung von in silico Methoden gewinnt hier zunehmend an Bedeutung und hat sich als wertvolle Strategie zur Erkennung komplexer Zusammenhänge sowohl im Bereich der Struktur von Proteinen wie auch bei Bioaktivitäten erwiesen. Die zunehmende Nachfrage nach Rechenleistung im wissenschaftlichen Bereich sowie eine detaillierte Analyse der generierten Datenmengen benötigen innovative Strategien für die effiziente Verwendung von verteilten Computerressourcen, wie z.B. Computergrids. Diese Grids ergänzen bestehende Technologien um einen neuen Aspekt, indem sie heterogene Ressourcen zur Verfügung stellen und koordinieren. Diese Ressourcen beinhalten verschiedene Organisationen, Personen, Datenverarbeitung, Speicherungs- und Netzwerkeinrichtungen, sowie Daten, Wissen, Software und Arbeitsabläufe.

Das Ziel dieser Arbeit war die Entwicklung einer universitätsweit anwendbaren Grid-Infrastruktur - UVieCo (University of Vienna Condor pool) -, welche für die Implementierung von akademisch frei verfügbaren struktur- und ligandenbasierten Drug Discovery Anwendungen verwendet werden kann. Firewall- und Sicherheitsprobleme wurden mittels eines virtuellen privaten Netzwerkes gelöst, wohingegen die Virtualisierung der Computerhardware über das CoLinux Konzept ermöglicht wurde. Dieses ermöglicht, dass unter Linux auszuführende Aufträge auf Windows Maschinen laufen können. Die Effektivität des Grids wurde durch Leistungsmessungen anhand sequenzieller und paralleler Aufgaben ermittelt.

Als Anwendungsbeispiel wurde die Assoziation der Expression bzw. der Sensitivitätsprofile von ABC-Transportern mit den Aktivitätsprofilen von Antikrebswirkstoffen durch Data-Mining des NCI (National Cancer Institute) Datensatzes analysiert. Die dabei generierten Datensätze wurden für liganden-basierte Computermethoden wie Shape-Similarity und Klassifikationsalgorithmen mit dem Ziel verwendet, P-glycoprotein (P-gp) Substrate zu identifizieren und sie von Nichtsubstraten zu trennen. Beim Erstellen vorhersagekräftiger Klassifikationsmodelle konnte das Problem der extrem unausgeglichenen Klassenverteilung durch Verwendung der „Cost-Sensitive Bagging“ Methode gelöst werden. Applicability Domain Studien ergaben, dass unser

Modell nicht nur die NCI Substanzen gut vorhersagen kann, sondern auch für wirkstoffähnliche Moleküle verwendet werden kann. Die entwickelten Modelle waren relativ einfach, aber doch präzise genug um für virtuelles Screening einer großen chemischen Bibliothek verwendet werden zu können. Dadurch könnten P-gp Substrate schon frühzeitig erkannt werden, was möglicherweise nützlich sein kann zur Entfernung von Substanzen mit schlechten ADMET-Eigenschaften bereits in einer frühen Phase der Arzneistoffentwicklung.

Zusätzlich wurden Shape-Similarity und Self-organizing Map Techniken verwendet um neue Substanzen in einer hauseigenen sowie einer großen kommerziellen Datenbank zu identifizieren, die ähnlich zu selektiven Serotonin-Reuptake-Inhibitoren (SSRI) sind und Apoptose induzieren können. Die erhaltenen Treffer besitzen neue chemische Grundkörper und können als Startpunkte für Leitstruktur-Optimierung in Betracht gezogen werden.

Die in dieser Arbeit beschriebenen Studien werden nützlich sein um eine verteilte Computerumgebung zu kreieren die vorhandene Ressourcen in einer Organisation nutzt, und die für verschiedene Anwendungen geeignet ist, wie etwa die effiziente Handhabung der Klassifizierung von unausgeglichenen Datensätzen, oder mehrstufiges virtuelles Screening.

CURRICULUM VITAE

Full Name	Yogesh Aher
Sex	Male
Date of Birth	8th May 1984
Place of Birth	Induri (A'nagar, Maharashtra), India
Marital Status	Single
Nationality	Indian
Languages	English, Hindi, Marathi, German
E-mail Address	yogesh.aher@yahoo.co.in , aheryogi@gmail.com

Educational Background

2008-2012	Doctoral Studies of Natural Sciences / Pharmacy University of Vienna, Faculty of Life-Sciences, Department of Medicinal Chemistry, Pharmacoinformatics Research Group, Vienna, Austria Supervisor: Univ.-Prof. Dr. Gerhard F. Ecker(gerhard.f.ecker@univie.ac.at) Thesis: "Development and Application of Distributed Computing Tools for Virtual Screening of Large Compound Libraries"
2005-2007	M.S. (Pharm.) Pharmacoinformatics (CGPA 8.64 on 10 point scale) National Institute of Pharmaceutical Education and Research (NIPER), Mohali, Punjab, India Supervisor: Assoc.Prof. Dr. Prabha Garg (prabhagarg@niper.ac.in) Thesis: "QSAR Modeling of Biological Activity of CCR5 Receptor Antagonists"
2001-2005	Bachelor of Pharmacy (66.28%) Sinhgad College of Pharmacy, Pune University, Pune, Maharashtra, India
1999-2001	Intermediate Studies (76%), Pune Board, Pune, Maharashtra, India
1989-1999	Matriculation Studies (87%), Pune Board, Pune, Maharashtra, India

Scholarships

2008-2009	Research Fellowship from University of Vienna under the project "CPAMMS - Computing Paradigms and Algorithms for Molecular Modeling and Simulation: Applications in Chemistry, Molecular Biology, and Pharmacy" (FS397001)
2009-2010	Research Fellowship from FWF's "Special Research Programs" (SFB)
2010-2011	Research Fellowship from eTOX project (115002)

Publications

Aher Yogesh D, Vasanthanathan P and Ecker Gerhard F, *Prediction of P-Glycoprotein Substrates and Non-Substrates from Highly Imbalanced Dataset using Cost-Sensitive Machine Learning Methods*, 2012, Manuscript prepared for Journal of Chemical Information and Modeling

Aher Yogesh D, Agrawal A, Bharatam PV and Garg P, *3D-QSAR studies of substituted 1-(3, 3-diphenylpropyl)-piperidinyl amides and ureas as CCR5 receptor antagonists*, Journal of Molecular Modeling, 13(4): 519-529, 2007

Aher Yogesh D and Garg P, *QSAR Modeling of CCR5 Receptor Antagonists using Artificial Neural Network*, Proceedings of the International Conference on Artificial Intelligence and Applications (IASTED'07) Innsbruck, Austria, ACTA Press, 192-196, 2007

Contributions to Scientific Conferences

Aher YD, Stampfel G, Demel MA, Stockner T, Gansterer W and Ecker GF, *Condor@Univie- A Distributed System Web Portal for Rigid & Flexible Ligand-Protein Docking*. Poster presentation at the 21st Scientific Congress of the Austrian Pharmaceutical Society, April 16-18, 2009, Vienna. *Abstract in: Sci Pharm 77:200.*

Aher YD, Stampfel G, Demel MA, Stockner T, Gansterer W and Ecker GF, *High-throughput calculations of Ligand-Protein Docking WITH Condor@Univie- A WEB-BASED Distributed COMPUTING Portal.* Poster presentation at the Joint Meeting on Medicinal Chemistry, June 24-27, 2009, Budapest.

Aher YD, Vasanthanathan P and Ecker GF, *Ligand-Based Classification Model for Imbalanced Dataset of P-Glycoprotein Substrates and Non-Substrates*. Poster presentation at the 8th European Workshop in Drug Design, May 22-28, 2011, Siena.

Aher YD, Vasanthanathan P and Ecker GF, *Cost-Sensitive Bagging – A Versatile Approach for Classification of an Imbalanced Dataset of P-gp Substrates and Non-Substrates*. Poster presentation at the European School on Medicinal Chemistry, July 3-8, 2011, Urbino.

Aher YD, Vasanthanathan P and Ecker GF, *Classification of an Imbalanced Dataset of P-Glycoprotein Substrates and Non-substrates using Cost-Sensitive Machine Learning Methods*. Poster presentation at the Joint Meeting of the Austrian and German Pharmaceutical Societies, September 20-23, 2011, Innsbruck.

Vasanthanathan P, Aher YD, Klepsch F, Haider N and Ecker GF, *Ligand and Structure-Assisted Classification Models for Rapid Identification of P-Glycoprotein Inhibition.* Poster presentation at the Joint Meeting of the Austrian and German Pharmaceutical Societies, September 20-23, 2011, Innsbruck.

Vienna, May 31, 2012