UNIVERSITA' DEGLI STUDI DI SIENA
DIPARTIMENTO DI BIOTECNOLOGIE, CHIMICA E FARMACIA

DOTTORATO DI RICERCA IN

SCIENZE CHIMICHE E FARMACEUTICHE

CICLO XXIX

COORDINATORE Prof. Stefano Mangani

Fighting HIV-1 and related diseases using rational medicinal chemistry approaches: from computational chemistry to biological evaluation

SETTORE SCIENTIFICO-DISCIPLINARE:

CHIM/08

DOTTORANDO:

Riccardo Martini

TUTOR:

Prof. Maurizio Botta

ANNO ACCADEMICO: 2015-2016

"Life is a game. The only thing that matters

is whether you're a pawn or a player."

Acknowledgments

I would like to thank my tutor Prof. Maurizio Botta for the opportunity to have worked for the last 3 years on very interesting projects, letting grow myself as person and as scientist.

A special thanks to all Prof. Botta's group, present and past, and in particular to Dr. Cristina Tintori.

I would also like to thank the groups with whom I collaborated to realize this work:

- Department of Pharmaceutical Chemistry, University of Vienna (Austria); Prof. Thierry Langer's group, where I spent three months of my PhD studies.
- Department of Life and Environmental Sciences, University of Cagliari (Italy); Prof. Enzo Tramontano's group.
- Laboratory of Microbiology, San Raffaele Hospital, IRCCS, Milan (Italy); Dr Filippo Canducci's group.
- Institute of Microbiology, Università Cattolica del Sacro Cuore, Rome (Italy); Prof. Maurizio Santuinetti's group.
- Department of Chemistry and Technology of Drugs, University of Rome "Sapienza" (Rome); Prof. Bruno Botta's group.

Ringraziamenti

Vorrei ringraziare il mio tutor Prof. Maurizio Botta per l'opportunità di avermi permesso di lavorare durante gli ultimi 3 anni su progetti molto interessanti, lasciandomi crescere come persona e come scienziato.

Un ringraziamento particolare va a tutto il Gruppo Botta, presente e passato, ed in particolare alla Dr. Cristina Tintori

Vorrei inoltre ringraziare i gruppi di ricerca con i quali ho collaborato per la realizzazione di questo lavoro:

- Dipartimento di Chimica Farmaceutica dell'Università di Vienna, Vienna (Austria); gruppo del Prof. Thierry Langer, dove ho trascorso tre mesi nell'ambito del dottorato.

- Dipartimento di Scienze della Vita e dell'Ambiente dell'Università di Cagliari (Italia); gruppo del Prof. Enzo Tramontano.

- Laboratorio di Microbiologia dell'Ospedale San Raffaele, Milano (Italia); gruppo del Dr. Filippo Canducci.

- Istituto di Microbiologia dell'Università Cattolica del Sacro Cuore, Roma (Italia); gruppo del Prof. Maurizio Sanguinetti.

- Dipartimento di Chimica e Tecnologie del Farmaco dell'Università di Roma "Sapienza", Roma (Italia); gruppo del Prof. Bruno Botta.

Table of Contents

SUMMARY

HIV is undoubtedly one of the viruses that have characterized the last half century in human history. This virus, found to be the AIDS etiological cause, is still one of the biggest challenges for drug therapy, since there are still neither a vaccine nor a definitive cure. Therefore, despite more than 25 marketed drugs administered in several combinations in HAART (Highly Active Anti-Retroviral Therapy), there is still the need of molecules active through new mechanisms, in order to overcome the many resistances developed by the virus to counter the pharmacological treatment. The main purpose of this thesis is to identify new active molecules able to inhibit viral replication with innovative mechanisms. Also, considering that the AIDS patients easily develop secondary infections, another part of the work is focused on the study of antifungal agents previously developed in our laboratories.

The first chapter contains the research carried out directly on the HIV-1 enzyme Integrase.

The second chapter is focused on the study of molecules with antifungal activity able to block candida infections, which easily occur in immunocompromised patients.

The third and last chapter is focused on the development of a new part of code for the program KNIME, a data-pipelining software used in CADDD (Computer-Aided Drug Design and Development).

RIASSUNTO

L'HIV è senza dubbio uno tra i virus che maggiormente hanno segnato l'ultimo mezzo secolo della storia dell'umanità. Tale virus, causa eziologica dell'AIDS, è ancora oggi una delle maggiori sfide da un punto di vista del trattamento farmacologico, in quanto non esistono ancora né un vaccino né una cura definitiva. Quindi, nonostante più di 25 farmaci ad oggi in commercio somministrati in associazione nella terapia HAART (Highly Active Anti-Retroviral Therapy), vi è ancora necessità di molecole che siano attive grazie a nuovi meccanismi, ai fini di superare le numerose resistenze messe in atto dal virus per contrastare la terapia farmacologica. Lo scopo di questa tesi è principalmente quello di individuare nuove molecole attive nell'inibire la replicazione virale con meccanismo innovativo. Inoltre, considerando che i pazienti malati di AIDS sviluppano in maniera estremamente semplice infezioni secondarie, un'altra parte del lavoro è stata incentrata sullo studio di molecole ad attività antifungina sviluppate in precedenza nei nostri laboratori.

All'interno del primo capitolo è riportata la ricerca svolta a colpire direttamente l'HIV, mediante la ricerca di una inibizione allosterica dell'enzima virale Integrasi.

Il secondo capitolo è invece dedicato allo studio di molecole ad attività antifungina capaci di bloccare infezioni sistemiche di candida che facilmente si presentano in pazienti immunodepressi.

Il terzo ed ultimo capitolo è invece incentrato sullo sviluppo di una nuova parte di codice per il programma KNIME, un data-pipelining software utilizzato nel CADDD (Computer-Aided Drug Design and Development).

List of Abbreviations

ABC → APT-binding cassette

ALLINIs → Allosteric Integrase Inhibitors

AST → Zidovudine

db → Database

CADDD → Computer-Aided Drug Design and Development

CDR → Candida Drug Resistance

ECM → Extracellular Matrix

GAFF → Generalized Amber Force Field

HBF → High Biofilm Former

HIV → Human Immunodeficiency Virus

HTVS → High Throughput Virtual Screening

IBD → IN Binding Domain

IDE → Integrated Development Environment

ILKE → Inte:Ligand LigandScout Knime Extension

IN → Integrase

INSTIs → Integrase-Strand Transfer Inhibitors

KNIME → Konstanz Information Miner

LBF → Low Biofilm Former

MD → Molecular Dynamic

MDR → Multi-Drug Resistance

min → minutes

MINI → Multimeric IN Inhibitors

NAC → Non-Albicans Candida

NNIBP → Non Nucleoside Inibitor-Binding Pocket

nm → Nano-meters

PDB → Protein Data Bank

PIC → Pre-Integration Complex

PR → Protease

RDDP → RNA-Dependent DNA Polymerase

RNase H → Ribonuclease H

RT → Reverse Transcriptase

SBP → Sucrose Binding Pocket

SDK → Software Development Kit

ST → Strand Transfer

TOA → Time Of Addition

wt → Wild type

CHAPTER 1

Fighting HIV-1 virus starting from a computational approach

Tintori C, Esposito F, Morreale F, Martini R, Tramontano E, Botta M.
Bioorg Med Chem Lett. ***2015***. *3013-3016. doi: 10.1016/j.bmcl.2015.05.011.*

Esposito F, Tintori C, Martini R, Christ F, Debyser Z, Ferrarese R, Cabiddu G, Corona A, Ceresola ER, Calcaterra A, Iovine V, Botta B, Clementi M, Canducci F, Botta M, Tramontano E.
Chembiochem. ***2015***. *2507-2512. doi: 10.1002/cbic.201500385.*

Martini R, Esposito F, Corona A, Ferrarese R, Ceresola ER, Visconti L, Tintori C, Barbieri A, Calcaterra A, Iovine V, Canducci F, Tramontano E, Botta M.
Chembiochem. ***2016***. *doi: 10.1002/cbic.201600592.*

1.1 Introduction

The Acquired Immune Deficiency Syndrome (AIDS), is a very complex disease, which includes a series of manifestation caused by the progressive deterioration of the immunity system.

The syndrome was observed for the first time in 1981 in the United States, when Michael Gottlieb, professor of immunology at UCLA (University of California, Los Angeles), noted a wired correspondence among few cases he observed in some clinics around the USA. The symptoms were a considerable weight loss, oral candidiasis and fever, often associated with *Pneumocystis carinii* pneumonia and a rare, and usually not fatal, Sarcoma Kaposi. [1] In a very short time, the number of persons affected by severe diseases and opportunistic infections increased considerably. The only common aspect was a dramatic decrease in the number of lymphocytes T helper ($CD4^+$), and this characteristic inspired the physicians to name it AIDS. The syndrome was observed among homosexuals, and in people undergoing blood transfusion or those who were used to take drugs by intra venous administration. The deduction was that the pathologic agent responsible for the disease was transmitted by blood contact or sexual intercourse. Two years later, in 1983, the Professor Luc Montagner at the institute Pasteur was able to isolate from lymph nodes of a patient a retrovirus, later identified to be the etiopathological agent of the disease. [2] Few time later, in the United States, Robert Gallo at the National Cancer Institute, announced the discovery of a particular virus isolated from lymph nodes of two patients having a particular predilection for lymphocytes. [3] The virus isolated in France and in USA was, indeed, the same. In 1984, the Center for Disease Control and Prevention, main American authority for the control of the public health, declared the virus as the cause of AIDS. Only two years later it was named Human Immunodeficiency Virus (HIV). In the same year, another strain was identified to be spread among people in the west Africa, and was named HIV-2. The first drug to gain the approval from FDA as treatment against HIV was the AZT in 1987. This virus, during the last 30 years, has been the focus of intense research, allowing the discover

of an extraordinary complexity. Nowadays more than 25 molecules are approved for the treatment of HIV-1 infections. [4] These compounds are used in the therapy as combination of molecules in the so called HAART (Highly Active Anti-Retroviral Therapy). This chemotherapeutic approach consists in the use of three or more molecules in combination, belonging to at least two different classes. It has been adopted since the last 20 years, inasmuch is more effective against drug resistance development and has low toxicity respect to a monotherapy treatment. [5] Unfortunately, despite the capability of the HAART to reduce the number of AIDS-related deaths, is not able to eradicate the infection. Due to the high mutation rate of the viral proteins and the lifelong duration of HAART, resistance is still a severe problem, and research of new molecules or new mechanism of action, for example to target different key events in the virus replication process, is dramatically needed. [6]

1.1.1 HIV-VIRUS

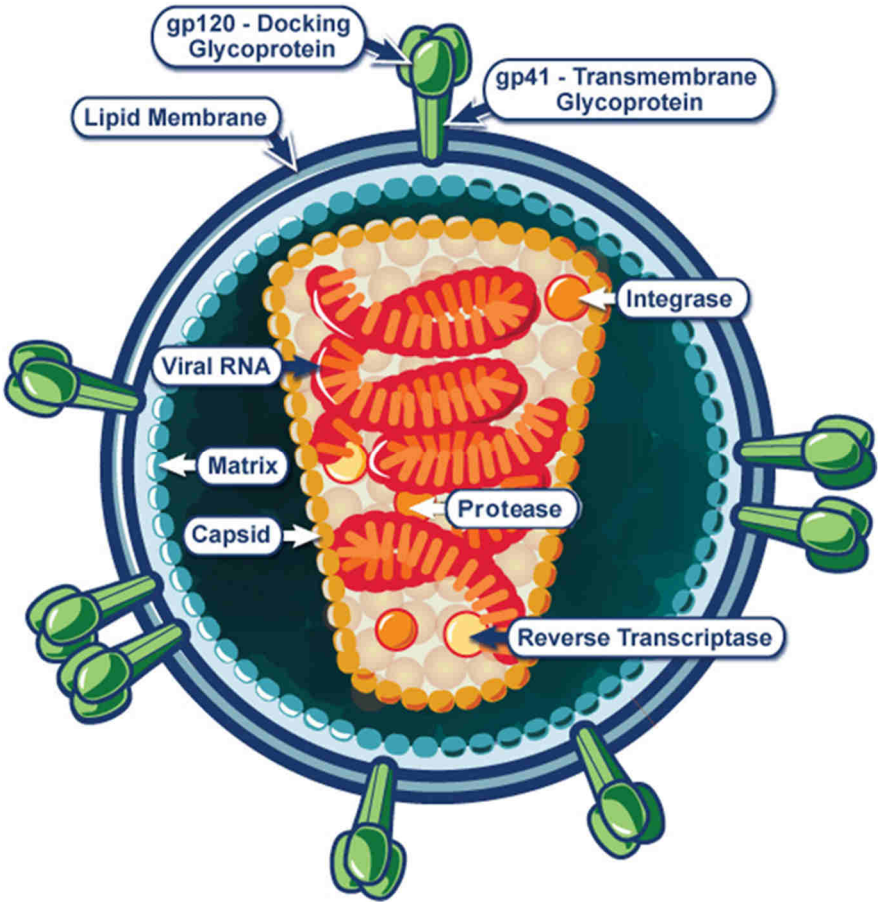
Nevertheless the origins of the virus are still uncertain, it has a genetic similarity of 98% with the Simian Immunodeficiency Virus (SIV) and probably HIV is only a zoonosis. Being the community of which HIV was originated small and isolated in the Equatorial Africa, the diffusion of the disease remained relatively limited until the first half of the last century. However, the advent of colonialism, urbanization, drug addiction and the tragic phenomenon of sex tourism, has favoured the spread of the virus throughout the rest of the world. The WHO (World Health Organization) estimates that in 1981 people killed by AIDS were more than 25 million, making the disease one of the deadliest pandemics of the past half century. Moreover, the WHO estimates that at the end of 2015, people living with HIV are about 36,7 million. However, the estimation from the same year for people who are on antiretroviral therapy are 17 million, including 2 million people who started treatment in 2015. [7]

HIV-1 is a single-strand, positive-sense, enveloped RNA Retrovirus the family of Lentiviruses. Once entered into the cell, the viral RNA is converted

into double-stranded DNA by the reverse transcriptase (RT) which is virally encoded and transported along with the genome in the viral particle. The neo formed viral DNA is then imported into the nucleus and integrated into the host genome by another virally encoded enzyme, Integrase (IN), together with host co-factors. [8] From this point the virus may become latent in the cell in order to avoid the detection by the immune system, or can be actively transcribed. In this latter case, new viral RNA and viral proteins are produced, packaged and released from the host cell. The infection is transferred through blood, vaginal fluid, semen or breast milk, in which fluids the virus is present in both forms: as viral particles and as virus within infected immune cells. The preferred cells are helper T cells, macrophages and dendritic cells, all of them vital element for a working immune system. When the number of white cells decline below a critical threshold, cell-mediated immunity is lost, resigning the body at the mercy of opportunistic infections.

1.1.2 HIV STRUCTURE

The HIV-1 virion (Figure 1-1) has an icosahedral form, with a diameter ranging from 100 to 120 nm. Is formed by a phospholipidic double layer that hosts transmembrane viral glycoproteins. [9] The main representative elements are gp120 and gp41. [10]



Credit: National Institute of Allergy and Infectious Diseases.

Figure 1-1: Schematic representation of HIV structure. The most important viral components are highlighted.

A matrix shell formed by the protein p17 lines the inner surface of the viral membrane, while the protein p24 forms a conical capsid core located in the center of the virus. As previously mentioned, the genetic material used by this virus is the RNA, which is associated to two basic nucleocapsid proteins: p7 and p9. The ribonucleoprotein complex is present in the core together with three vital viral enzymes: RT, IN and Protease (PR). Other key factors are also present in the viral particle, like proteins Nef, Vif and Vpr, and some are not packaged, see Rev, Tat and Vpu. [11]

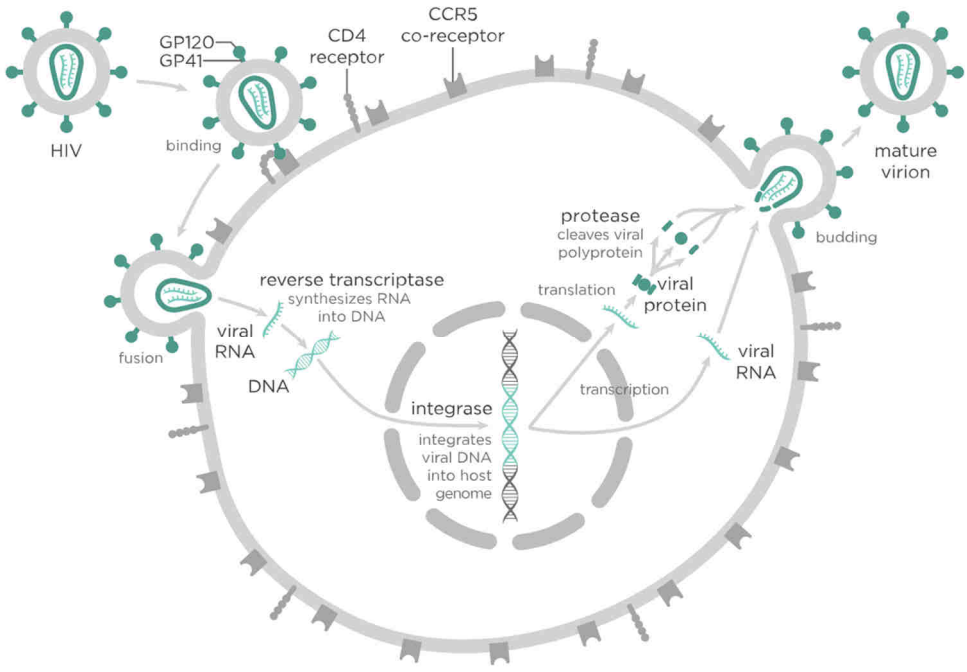
1.1.3 HIV REPLICATION CYCLE

The replicative cycle of HIV-1 is mainly characterized by the following steps:

1. Viral particle adsorption on the host cell: The viral glycoprotein gp120 binds the CD4 receptor present on the host cell. [12]
2. Fusion of the virus with the host cell membrane: After gp120-CCR5 coreceptor binding, also the interaction with CXCR4 coreceptor occurs. This interaction induces the exposition of another glycoprotein, gp41, which consent the fusion with the cellular membrane. [13]
3. Viral particle disassembly: Once inside the cell host, the viral capsid is degraded by cellular enzymes, releasing the RNA and the viral proteins.
4. Reverse transcription: RT converts the viral RNA into double strand DNA, which immediately associates to viral protein like IN to form the Pre-Integration Complexes (PICs). This complexes are actively transported into the nucleus. [14]
5. Viral DNA integration into host genome (this step will be more deeply treated in the following paragraphs): PICs arrive in the nucleus and are directed toward zones with high transcription rate, mainly due to the interaction with the Lens Epithelium Derived Growth Factor (LEDGF) [15] which tether the host DNA and the tetramer of IN bounded to the viral DNA. [16]
6. Proviral DNA replication: After the viral genome integration the viral DNA can stay silent and the infection remains latent, or the viral DNA can be actively transcribed by the RNA Polymerase II and the infection become productive. Considering that the viral DNA behaves like a normal cellular gene, its replication is influenced by the regular cell regulatory factors.

7. Transcription of proviral DNA into viral Messenger RNA (mRNA): a part of mRNA is not processed because serves as genetic material for the new virions.
8. Translation of viral RNA into viral protein precursors: the mRNA is translated into protein precursors.
9. Maturation of the precursors: PR processes the precursor and frees the individual mature proteins.
10. Assembly and release: The single viral proteins assembly together enclosing the viral RNA forming the virions, which are subsequently coated with a lipoproteic envelope during the budding process.

At this point the viral particles are ready to infect new cells and start with a new replicative cycle.



Thomas Splettstoesser (www.scisetyle.com)

Figure 1-2: Schematic representation of HIV life cycle.

1.2 Current Therapy

As already mentioned, the HIV-1 infection has no definitive cure at present. This means that drug therapy, once started, must be continued for the entire life of the patient. Accordingly, there is the needed of nontoxic molecules for the use in long-term therapy. Moreover, during the HIV-1 replication a lot of errors are made, and together with the rapid replication of the virus in patients, these characteristics allow the virus to escape the host's immune system and develop resistances. [17] After the monotherapy treatment used at the beginning, we are moved to a multidrug regimen, which is much more able to prevent resistance, and allows a reduction of the dose of the single drug.

The use of the combination of three or more drugs from two or more different inhibitor classes is termed HAART. Currently five classes of drugs are approved by FDA, which target almost all the most important steps in viral replication: entry, fusion, reverse transcription, integration and protease cleavage. [18]

Entry inhibitors are represented by two molecules, Enfuvirtide, [19] which binds to gp41 preventing the successful docking on the CD4, and Maraviroc, [20] which binds to CCR5 coreceptor leading a conformational change that prevent the interaction with the viral protein gp120.

Considering the key role of RT in the replicative cycle, this enzyme has become one of the main target for the development of anti-HIV drugs. [21] Two classes of RT inhibitors are currently employed in AIDS therapy: the nucleoside (NRTIs) and the non-nucleoside (NNRTIs) inhibitors. The firsts one are the oldest anti-HIV drugs on the market, with the Zidovudine (1987). [22] They are nucleoside analogues that act as chain terminator, because of the lack of the 3'-OH group in the ribose ring, which is essential for the continuation of the DNA. However, due to their nature, the therapeutic effectiveness of these drugs is limited mainly by toxicity (incorporation in the host DNA), unfavourable pharmacokinetic, and emergence of resistant viral strains. The other class, NNRTIs, consists of very different molecule from a chemical point of view. All of them have a

unique specificity for HIV-1, and bind to an allosteric site called NNIBP (Non Nucleoside Inibitor-Binding Pocket) of the RT. [23] Nevirapine was the first NNRT approved by FDA [24] in 1996; it must be used in combination with other drugs.

Another key enzyme is the PR, which if inhibited prevents the formation of mature and infective viral particles. The PR inhibitors are basically peptide analogues, which mimic the cleavage sequences of the natural substrate of the enzyme during the transition state of the cleavage reaction. Saquinavir was the first approved PR inhibitor by FDA in 1995. [25]

Integrase inhibitors are one of the most recent class of anti-HIV drugs. Raltegravir and Elvitegravir, approved by FDA in 2007 and 2009 respectively, are able to block the enzyme once in complex with the activated DNA just before the integration in the host genome. They are also called Integrase-Strand Transfer Inhibitors (INSTIs), Figure 1-3. This kind of molecules are today guideline-preferred agents as part of an antiretroviral regimen for treatment-naive patients. [26] Recently (2013) Dolutegravir, a third generation of IN inhibitors, was also approved. [27]

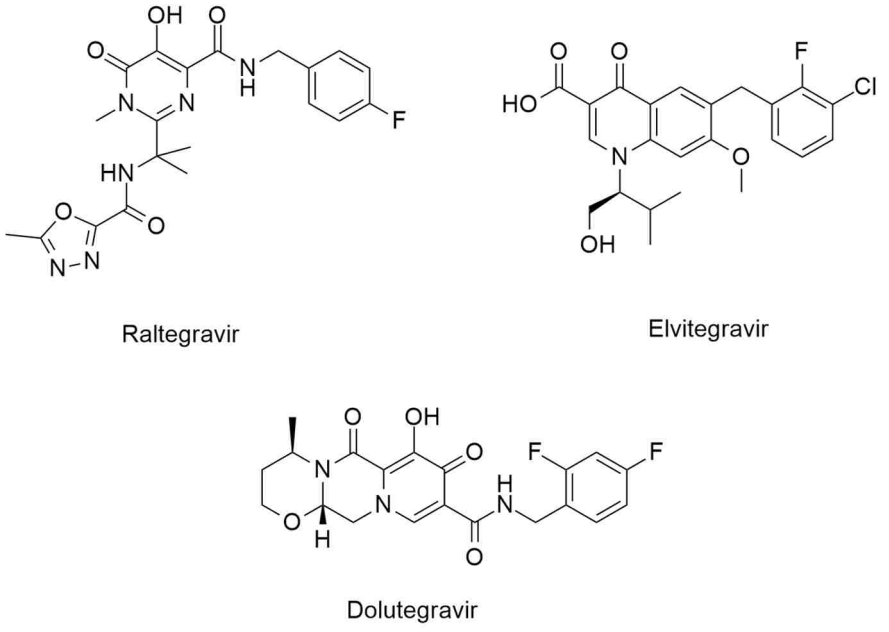


Figure 1-3: Structures of the three commercially available anti-IN drugs.

1.3 New prospectives and targets to treat HIV-1 infections

Despite the HAART, HIV-1 virus is still able to replicate itself in the patients treated with this multidrug regimen. Thus, considering the high replication error rate, drug resistant strains frequently emerge thwarting the therapy. Thus, innovative molecules able to overcome the dramatic problem of the resistance are needed, ideally targeting steps of the replicative cycle that are still unexploited, or being able to bind to allosteric pocket of already know targets. Accordingly, many research efforts were done in both directions.

1.3.1 INTEGRASE AS ATTRACTIVE TARGET FOR NEW MOLECULES DEVELOPMENT

1.3.1.1 STRUCTURE

IN is a protein formed by 288 amino acids, coded by the last part of the *Pol* gene. Is synthetized in the Gag-Pol polypeptide, and freed after proteolytic cleavage actuated by PR. It is divided in three parts:

- NTD (N-Terminal Domain): Zinc-finger is present. This part helps the multimerization [28] and the integration process. [29]
- CCD (Catalytic Core Domain): The core of the protein. It contains the catalytic amino acids (Asp64-Asp116-Glu152), which, together with two Magnesium ions, is able to integrate the viral DNA into the host genome. The current IN-inhibitors bind in the cavity of the active site.
- CTD (C-Terminal -Domain): This part is needed for the interaction with the target DNA. [30]

Despite efforts, a crystal structure of wild type IN is not yet available. Only the constitutive parts of it have been resolved, or at most a couple of them (NDT-CCD, or CCD-CTD). Moreover, all these structures have at least one mutation, because of the unsuitable characteristic of the wt protein to be crystallised.

1.3.1.2 FUNCTIONS

Integrase is a crucial enzyme for the HIV-1 virus. Its function of viral DNA integration into host genome is unique for the virus, and a similar function is not present in mammalian cells. It is composed by two phases, spatially and temporally distinct:

- *3'-end processing*: in the cytoplasm an IN dimer activates the viral DNA, forming than an IN dimer-DNA complex.
- *Strand-transfer*: occurs after the transfer into the nucleus of the previous complex. This step is characterized by the interaction of the IN complex, now in the form of tetramer, with the cellular cofactor LEDGF/p75, which results to be crucial for the correct replication process. [31] In this latter step, the tetramer of IN integrate the viral DNA into the genome.

1.3.1.3 LEDGF/P75-IN INHIBITION

Lens Epithelium-Derived Growth Factor/p75 is a transcriptional coactivator required from HIV-1 IN to tether and correctly integrate the viral genome into the chromatin. The protein-protein interaction IN-LEDGF is found to be essential for a correct viral replication, [32] and many authors qualified it as innovative and druggable target for the HIV therapy. Since crystal structure of the IN Binding Domain (IBD) of LEDGF in complex with the dimer of the CCD of IN was resolved, [33] researchers focused on short peptides derived from IBD. [34] Considering that peptides are not suitable for targeting intracellular targets, small molecules have been designed to prevent the LEDGF/p75-IN interaction (Figure 1-4).

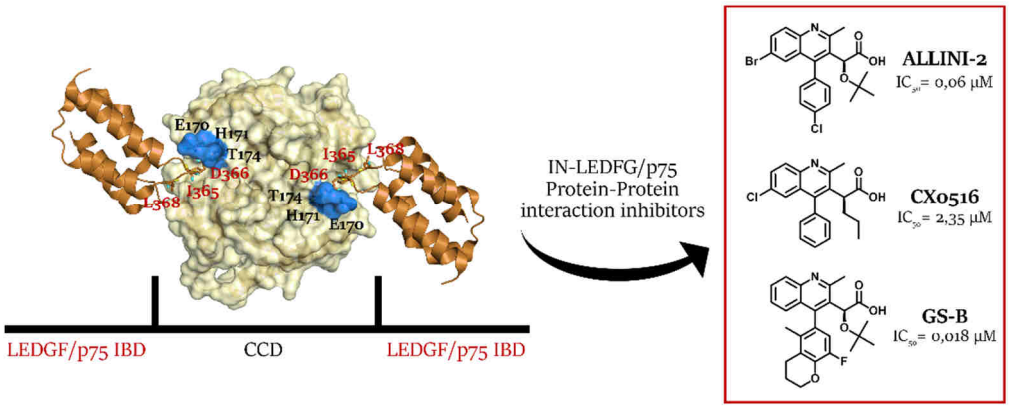


Figure 1-4: PDB code: 2B4J. [35] *In pale yellow surface the CCD of IN in complex with two molecules of LEDGF/p75 IBD (orange cartoons) The key residues for the protein-protein interactions are labelled. The activity and the structure of three ALLINIs are reported.* [36] [37]

These molecules, called ALLINIs (Allosteric Integrase Inhibitors) are designed to bind the pocket defined at the interface of IBD. [38] A graphical representation is shown in Figure 1-5.

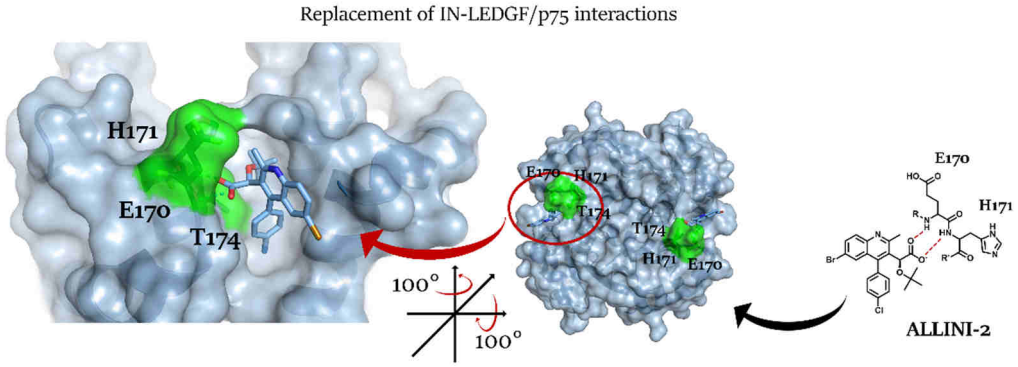


Figure 1-5: Binding mode of ALLINI-2 in 4GW6 crystal structure. [37]

Many classes of ALLINIs have been described, even having an high variability in their biological activity, they are characterized by the same dual mechanism of action: potent inhibition of the LEDGF/p75-IN interaction and allosteric inhibition of the integrase catalytic activity. [39] The multimodal mechanism makes them promising candidate for the treatment of HIV/AIDS.

1.3.1.4 MULTIMERIZATION INHIBITION

As already mentioned, IN is an enzyme which is subjected to a multimeric equilibrium. As showed in Figure 1-6, two are the active forms: the dimer and the tetramer. The first one is required in the cytoplasm for the *3'-processing*, while the second one is necessary in the nucleus for the *strand-transfer*.

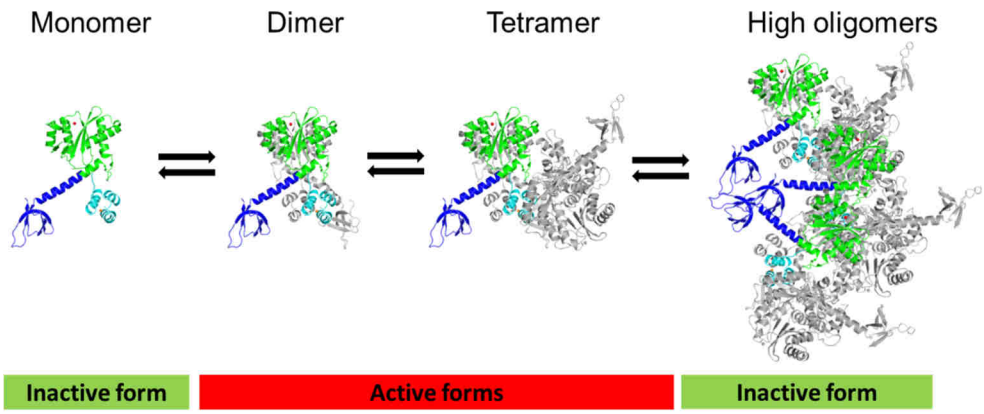


Figure 1-6: Multimerization equilibrium of HIV-1 Integrase.

Thus, in order to disrupt the integration process two different strategies can be applied:

- Dimer formation inhibition: to avoid the passage from the inactive monomer to the first active form (dimer).
- Stabilization of one form: to disrupt the delicate oligomeric equilibrium, for instance promoting the formation of inactive high oligomeric forms.

1.3.1.4.1 Dimerization inhibitors

Since IN dimerizes after the interaction between two CCDs, peptides derived from the dimer interface (namely INH1 and INH5) [40] showed enzyme inhibition by disrupting the monomer-monomer complex

formation. Considering the unsuitability of proteins to develop new drugs with an intracellular target, small molecules have been studied.

In the last few years, new molecules able to prevent the initial dimerization were found in my research group by the successful combination of different computational techniques. In one of our previous work, [41] the interface between two CCD was examined through molecular dynamic experiments and *PocketPicker* [42] analysis, finding a putative binding pocket showed in Figure 1-7.

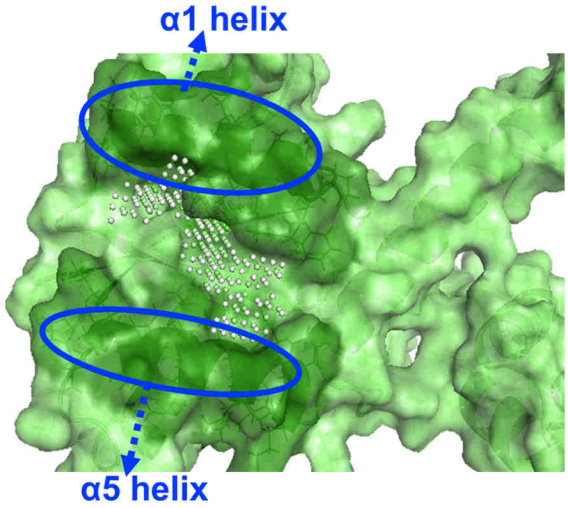
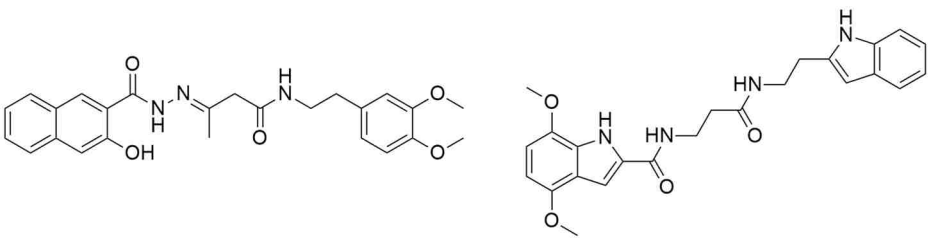


Figure 1-7: Binding site located between the α1 and α5 helices at the monomer-monomer interface. Grid points identified by PocketPicker *are represented as white spheres.*

Virtual screening was then applied on this site, eventually leading to two molecules. Compound 4 and 11 (Figure 1-8) were able to inhibit the dimerization process at 100 µM of 83% and 73% respectively in an AlphaScreen Assay. [43]



Compound 4 *Compound 11*

Figure 1-8: These molecules showed significant activity with inhibition at 100 µM of 83% (Compound 4) and 73% (Compound 11).

1.3.1.4.2 Dimer stabilizers

Recently, another class of molecules that act on the multimeric equilibrium of IN have been discovered and developed by Kvaratskhelia et al. [44] This new class, named MINI (Multimeric IN Inhibitors), is directly derived from ALLINIs. Both molecule series bind at the IBD pocket and share a common interaction network, but they substantially differ in the mode of action. While ALLINIs exert their action with a dual mechanism, impairing the IN-LEDGF interaction and promoting aberrant IN multimerization with a similar IC_{50}, [45] MINIs acts following only the second mode-of-action. In fact, they have only a small effect on the LEDGF-dependent HIV-1 integration in host chromosomes, indicating that the binding inhibition of LEDGF-IN is not crucial for their activity. KF116, one of the MINIs, has an EC_{50} = 86 nM on the full replication cycle, further demonstrating the possibility of developing an efficient compound to target the oligomeric equilibrium of IN.

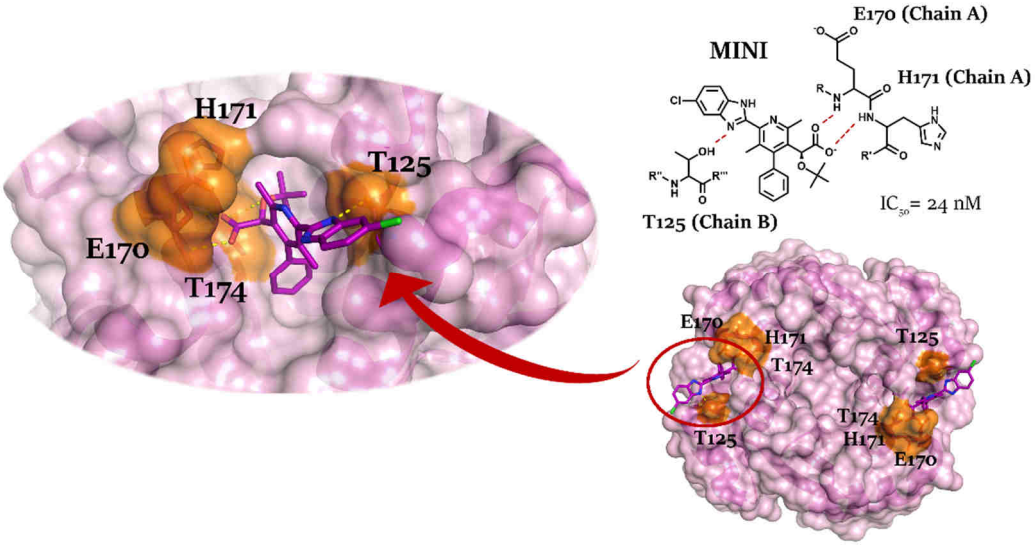


Figure 1-9: The activity and the structure of the MINI KF116 is reported. The binding mode in 4O55 crystal structure is also shown. [44]

1.4 Aim of the work

As already discussed, integrase represents a very interesting target to develop allosteric inhibitors. I've started my work considering two important steps discussed above:

- The multimerization process. Inasmuch it is a crucial step that can be targeted in several ways, using molecules which can stabilize or destabilize one or more of the complexes.
- The interaction between IN and the cellular cofactor LEDGF/p75. This can be inhibited with molecules that bind the dimer of IN in the IBD cavity, impairing the tethering of the IN-LEDGF/p75 complex with the host genome.

Considering this, the first part of the work was focused on the analysis, from a computational point of view, of the dimeric form of the IN CCD. In particular, it was centred on the study of a pocket located at the CCD interface that can allocate a molecule of sucrose (SUC), originally originating from the cryoprotectant in the X-ray experiment. [46] Taking into account the very low affinity of SUC for IN, and the proximity of the ALLINIs' binding site, we wanted anyway to verify the action of the sugar in a system already containing stabilizing molecules such as ALLINIs. From the analysis of the monomer-monomer interaction energy of the various complexes, we obtained a decrease of the mutual free energy suggesting a stabilizing effect for the sucrose molecule. Once confirmed our initial idea with biological experiment, I've continued my work pursuing the study of the Sucrose Binding Pocket (SBP), performing a virtual screening. The procedure was applied in order to identify molecules that could establish a network interaction similar to sucrose into the pocket, determining at the same time an increasing of the binding free energy between the two interacting CCDs. Screening virtually a small *in-house* library of natural products, the molecule Kuwanon-L emerged as the best candidate for biological investigation. Then, the same biological experiments used for the previous study revealed that the natural molecule was also able to increase the IN multimerization rate, similar to what happen with ALLINIs.

Enzymatic tests on HIV-1 IN revealed an IC_{50} of 42 µM and 34 µM in a LEDGF dependent and LEDGF independent tests respectively. Moreover, a cellular test showed also an EC_{50} value of inhibition on a fully replicating HIV-1 laboratory strain NL-4-3 of 1,9 µM. In order to investigate more deeply the mode of action of the natural product, a Time Of Addition (TOA) was performed, highlighting a particular behaviour compatible with multiple target binding. In particular, the hypothesis was that it might bind to two HIV-1 key enzymes: IN and RT. This idea was supported from enzymatic experiments on RT, besides the results already obtained with the IN, in which Kuwanon-L is able to inhibit both functions of the enzyme associated functions: the RNA-dependent DNA polymerase (RDDP) activity, and the Ribonuclease H (RNase H) activity.

1.5 Exploitation of an allosteric binding site on HIV-1 Integrase: The Sucrose Binding Pocket

The research of allosteric binding sites on the Integrase enzyme has led, during several years, to the discovery of various pocket which can be ideally used to impair the regular IN function. Studying the literature, one in particular has attracted our attention. Looking at the catalytic core dimer interface there is a pocket able to host small molecules. [47] [48] [49] Interestingly, a molecule of sucrose, originating from the cryoprotectant, has been observed occupying this IN allosteric binding site in the crystal structure 3L3V deposited on the Protein Data Bank. [46] This symmetrical pocket is lined by residues Glu87, Val88, Ile89, Pro90, Glu96, Tyr99, Phe100, Lys103 and Lys173 of both IN monomers. Due to its symmetry, the sucrose can be found in two overlapping, alternate orientations (Figure 1-10) which, according to crystallographers who realized the X-ray, binds at 50% occupancy based on difference maps.

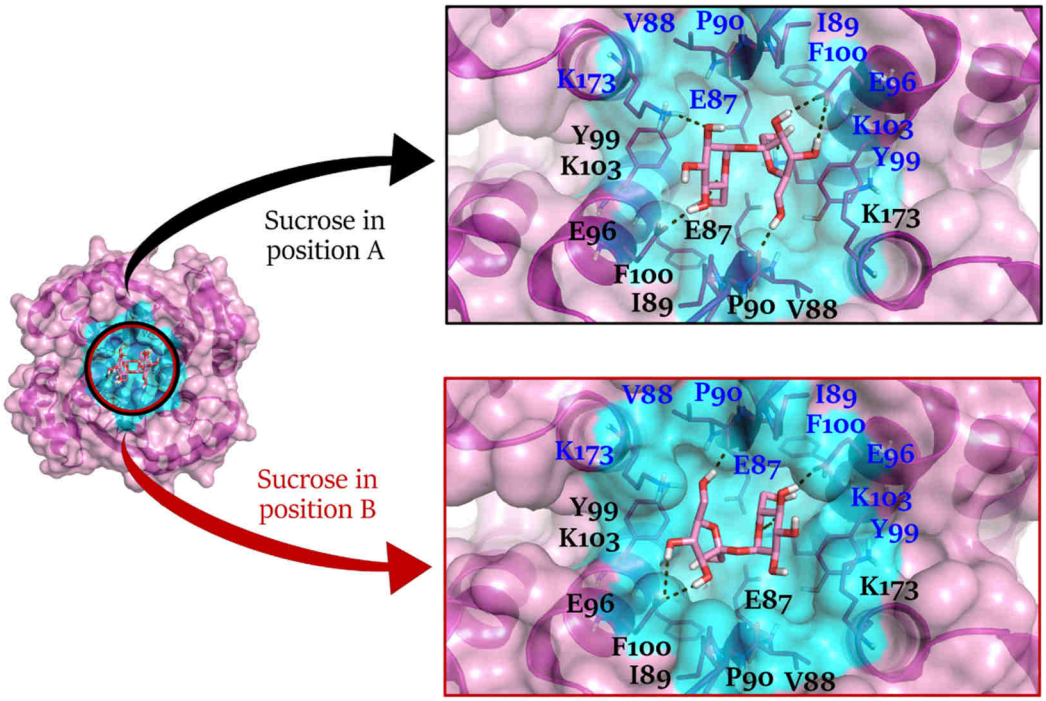


Figure 1-10: 3L3V crystal structure. The two alternate positions of sucrose are splitted and represented separately in the right part of the figure.

Position A and B are referred to the two different poses of sucrose found in 3L3V X-ray.

The ALLINIs' binding site is located 16 Å from the SBP if we measured the distance between the centroids of the two pockets However, there are only 6,5 Å between the two closest atoms of sucrose and ALLINI (Figure 1-11). [49]

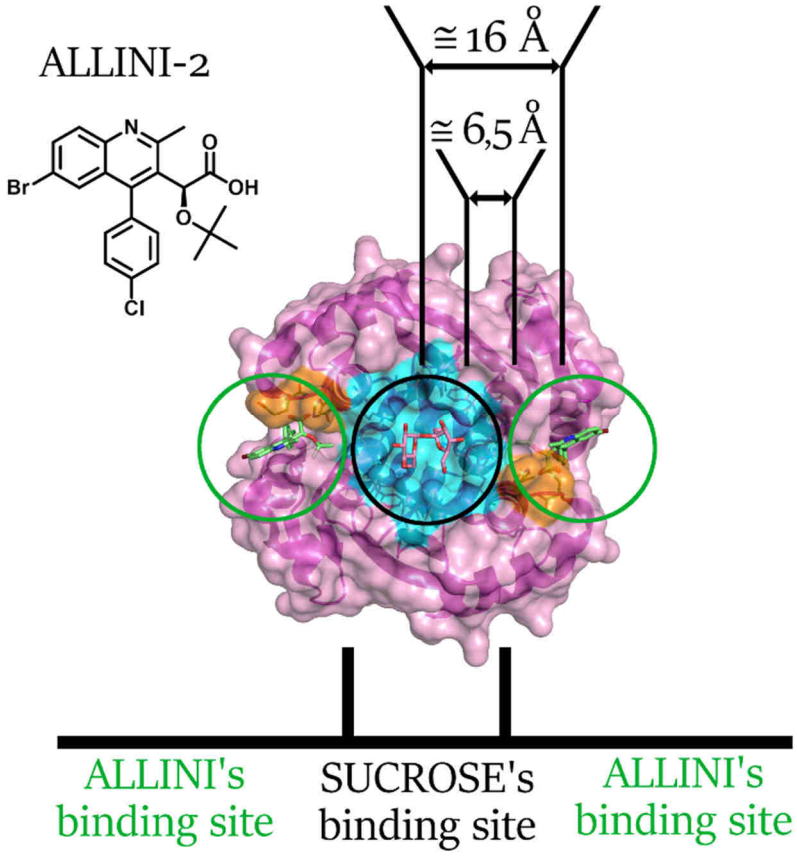


Figure 1-11: Distances between Sucrose Binding Pocket and ALLINIs binding pockets are reported.

1.5.1 COMPUTATIONAL STUDY

Starting from the crystal structure with cocrystalized sucrose (PDB code 3L3V), we wanted to investigate the effect of this molecule on the oligomeric IN equilibrium in presence of a well know IN stabilizer. Such a system should have shown an increase of monomer-monomer binding

energy if SUC acted as dimer stabilizer, and a decrease if it acted as disrupter. To study the system, we applied the computational procedure described below.

1.5.1.1 PROTEIN PREPARATION

The Integrase CCD crystal structure of the complex containing the sucrose molecule was downloaded from the Protein Data Bank, PDB code: 3L3V. The atomic coordinates were extracted from the complex. Whereas the same percentage of occupation for the two sucrose possible orientations, we have maintained only the first one, and named position A. A Magnesium ion was placed in place of the Cadmium ion coordinating residues Glu92 and Asp116 in both chains. For gap filling, a homology modelling was performed with the software Prime. [50] This procedure also let us to mutate residues Asp131 to Trp and His185 to Phe to restore the original wt amino acids. Two crystallographic complexes, specifically 3L3V and 1BL3, [51] were used as template and the FASTA sequence of IN (Figure 1-12) as query.

```
FLDGIDKAQEEHEKYHSNWRAMASDFNLPPVVAKEIVASCDKCQLKGEAM
HGQVDCSPGIWQLDCTHLEGKVILVAVHVASGYIEAEVIPAETGQETAYF
LLKLAGRWPVKTVHTDNGSNFTSTTVKAACWWAGIKQEFGIPYNPQSQGV
IESMNKELKKIIGQVRDQAEHLKTAVQMAVFIHNFKRKGGIGGYSAGERI
VDIIATDIQTKELQKQITKIQNFRVYYRDSRDPVWKGPAKLLWKGEGAVV
IQDNSDIKVVPRRKAKIIRDYGKQMAGDDCVASRQDED
```

Figure 1-12: FASTA sequence of IN. [52]

The structure contained residues (Gln95 orientation B of the ChainA and Glu170 of the ChainB) with an incompatible position with the binding of ALLINIs. Considering the numerous orientation such residues are able to assume in other crystals, we have rotated CB-CG bound by 180 degrees for Glu170 ChainB and we have deleted orientation B for Gln95 ChainA, in

order to obtain better docking performance. The protein was then prepared using the Protein Preparation Wizard workflow [53] from Schrodinger Suite. In particular all hydrogen atoms were minimized, and partial charges were assigned. In addition, the ionization and tautomeric states of His, Asp, Glu, Arg, and Lys were adjusted to match a pH of 7.4. Next, optimization of the hydrogen-bonding network was obtained by reorienting hydroxyl and thiol groups, amide groups of Asn and Gln, and His ring. The procedure to assemble the tetramer was the same used by Tintori et al. in a work of 2010. [54]

1.5.1.2 LIGAND PREPARATION

ALLINI-2, CX0516 and GS-B (Figure 1-13) present in X-ray structures with PDB code 4GW6, [37] 3LPU [55] and 4E1N, [56] respectively, were drawn and minimized using Maestro 9.6 [57] and Macromodel. [58] Furthermore, Ligprep [59] was used to predict ionization and tautomeric states for the ligands at a pH of 7.4 ± 0.3.

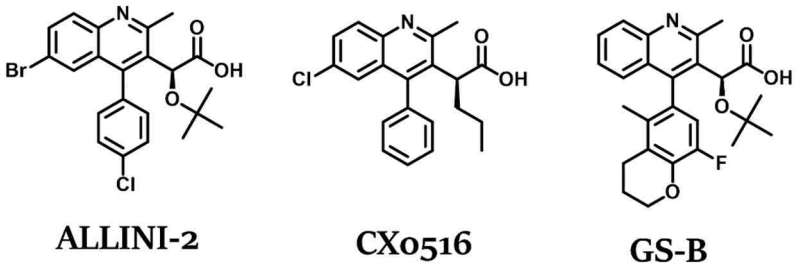


Figure 1-13: 2D structures of the three analysed ALLINIs.

1.5.1.3 DOCKING STUDIES

In the symmetric CCD of Integrase, two ALLINI's binding pockets are present. Docking simulations were performed using Glide [60] within both ALLINI's binding sites. The prepared protein system was then used to generate two receptor grids centred on each site, and no scaling was done for van der Waals radii of nonpolar receptor atoms. No constraints were

included during grid generation, while the rotation of the hydroxyl groups was allowed. After grid preparation, compounds were flexibly docked and scored using the Glide standard-precision (SP) mode, treating the protein as rigid and setting all parameters as default.

1.5.1.4 PREPARATION OF THE STARTING COMPLEXES FOR MD

A total of eleven complexes were prepared and subjected to molecular dynamic simulation.

Three constructs to understand if sucrose itself can promote or disrupt dimer stabilization:

- apo: 3L3V crystal structure after removing sucrose ligand.
- 3L3V_A_SUC: 3L3V crystal structure after removing only sucrose in position B.
- 3L3V_B_SUC: 3L3V crystal structure after removing only sucrose in position A.

Six complexes created to understand if sucrose is able to act synergistically with ALLINIs:

- Three CCD dimers each one with two docked ALLINIs (2x ALLINI-2, 2x CX0516 and 2x GS-B).
- Three complexes equal to the previous ones with the addition of the sucrose ligand in the SBP (A_SUC + 2x ALLINI-2, A_SUC + 2x CX0516 and A_SUC + 2x GS-B).

Two tetrameric structures to evaluate if ALLINIs themselves are able to stabilize a higher oligomeric form of IN:

- The tetramer of IN in its apo form.
- The complex between four molecules of GS-B and the IN tetramer (4x GS-B).

1.5.1.5 MD SIMULATIONS

Molecular dynamics simulations were performed for all the systems previously described. Antechamber suite [61] of AMBER12 [62] was used to assign to the ligand the partial charges, while the parameters were assigned by General Amber Force Field (GAFF). [63] The forcefield used to describe the protein parameters was the standard AMBER12 ff99SB forcefield for bio-organic systems; the carbohydrates specific forcefield GLYCAM_06h was also used for complexes containing SUC molecule. Complexes were solvated by placing them in a 10 Å octahedral box of TIP3P water molecule. The appropriate number of Na+ ions as counterions was subsequent added, inasmuch was necessary to reach the electrostatic system neutrality. Two steps of energy minimization were performed to remove bad contacts before MD simulations. The protein was kept fixed with a constraint of 500 kcal/mol during the first minimization step, allowing the minimization only of the water molecules. During the second stage a constraint of 10 kcal/mol on the α-carbon was applied, in order to minimize the entire system. Each minimization consisted of 15000 steps in which the first 1000 were steepest descent and the last were conjugate gradient. The minimized structures were used as starting point for MD simulations. The first nanosecond of the MD simulations was performed at constant volume and rising temperature from 0 to 300 K using the Langevin dynamics method. Then, 300 K with constant pressure was used in three following steps of 1 ns each. A decreasing harmonic force constraint of 10, 5, and 1 kcal/mol·Å was applied during these steps, respectively. Finally we performed a simulations without restraints, at constant temperature of 300K and a constant pressure of 1 atm for all the complexes (50 ns for the dimer and 100 ns for the tetramer). A 10-Å cutoff value was used for the nonbonded interactions, and a time step of 2 fs was used for the simulations.

1.5.1.6 ANALYSIS OF MD TRAJECTORIES

To analyse trajectories, we used the module Ptraj implemented in AMBER12. In particular, the root-mean-square deviation (RMSD) was calculated excluding flexible loops, thus only for residues forming α-helices and β-sheet of each residue on the production stage. RMSDs for ALLINIs and SUC molecules were also calculated.

1.5.1.7 MM/GBSA ANALYSIS

To evaluate the binding free energy, the MM/GBSA method was used. The MM/GBSA approach estimates the ΔG binding as the free energy difference between the complex (PL), the receptor (P) and the ligand (L).

$$\Delta G_{bind} = G(PL) – G(P) - G(L)$$

To compare the energy between the monomers, one of them was treated as ligand (L), the other as receptor (P) and the dimer as complex (PL). Binding free energies were calculated on systems where water molecules, counterions used during the simulation, ligands (when present) were stripped off. For ALLINIs and sucrose ΔG calculation, only counterions and waters were removed, the protein was treated as receptor (P) and small molecules as ligand (L). 167 frames were extracted from the last 20 ns of the MD and used for monomer-monomer ΔG calculation. The equation showed above was used on each snapshot to estimate the binding free energy (ΔG_{bind}).

1.5.2 RESULTS AND DISCUSSION (I)

1.5.2.1 IN-SILICO

Despite the lack of activity of the sugar itself, the numerous interactions with the protein prompted us to ask if this molecule could nevertheless have some type of effect on integrase.

The investigation of the effect of sucrose molecule in the SBP of IN started from the X-ray structure 3L3V. The results of the first experiment to evaluate the possible stabilization or destabilization effect of the sucrose molecule carried out with molecular dynamics are shown in Table 1-1. The table expresses the monomer-monomer free binding energy calculated with MM-GBSA of the CCD hosting SUC in conformation A, in conformation B and the CCD apo form.

Table 1-1: Values of binding free energy (kcal/mol) calculated with MM-GBSA technique in presence and absence of sucrose.

Complex	Monomer-monomer binding free energy	Sucrose energy
3L3V A-SUC	-120,48 ± 13,42	-40,33 ± 4,77
3L3V B-SUC	-106,98 ± 7,04	-42,15 ± 5,02
apo	-87,71 ± 9,28	/

Analysing the results of the MM-GBSA on several frames extracted from the trajectories of the experiments conducted in presence or in absence of sucrose, an interesting finding emerges: sucrose, independently from its initial position (A or B), increases the binding free energy between the two interacting monomers of IN, stabilizing the complex.

Then, the docking of ALLINIs was performed in order to generate the complexes by which evaluate if sucrose can or not synergistically interact with them to stabilize CCD. The predicted poses were in perfect agreement with those experimentally determined in the corresponding crystal structures (RMSD calculated on heavy atoms < 0.5Å). As an example, in Figure 1-14 two comparisons are reported.

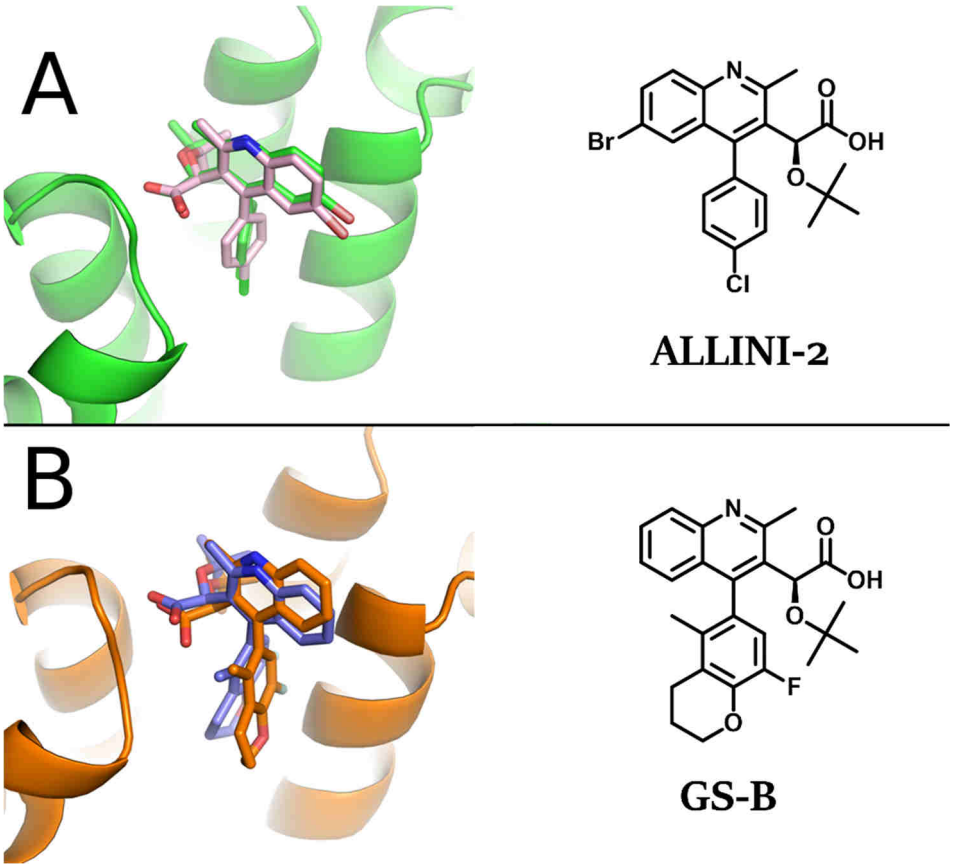


Figure 1-14: Top: In green the crystallographic pose, in pink the docked pose of ALLINI-2. Bottom: In orange the crystallographic pose, in purple the docked pose of GS-B.

Therefore, sucrose position A was arbitrarily chosen as the initial binding pose for creating complexes in which both ALLINIs and sucrose are present.

After performing molecular dynamic simulation experiment on all the six new complexes, the root-mean-square deviations of the α-carbons during the simulation with respect to the starting structures were plotted in order to see when the complexes would have reached the equilibrium (Figure 1-15, Figure 1-16 and Figure 1-17).

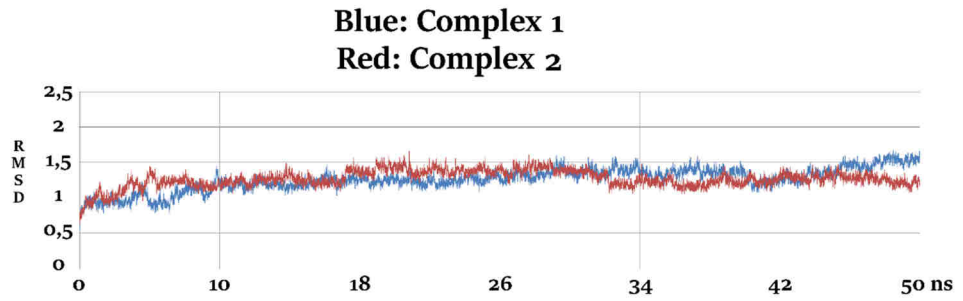


Figure 1-15: RMSD in Å calculated on α-carbons during the production MD of complexes 1 (A-SUC + 2x ALLINI-2) and 2 (2x ALLINI-2) with respect to the starting structure.

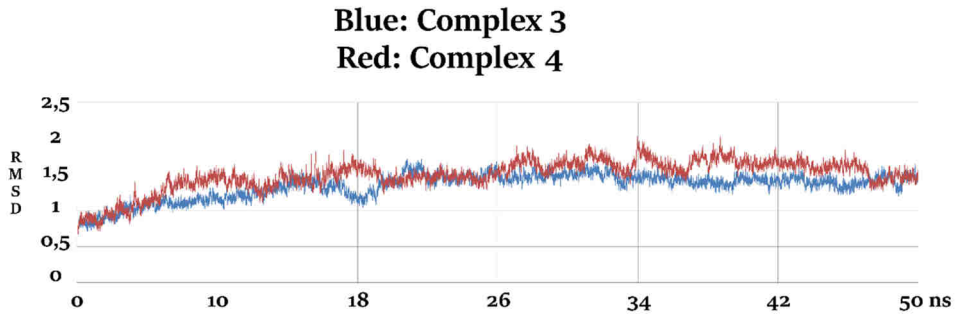


Figure 1-16: RMSD in Å calculated on α-carbons during the production MD of complexes 3 (A-SUC + 2x CX0516) and 4 (2x CX0516) with respect to the starting structure.

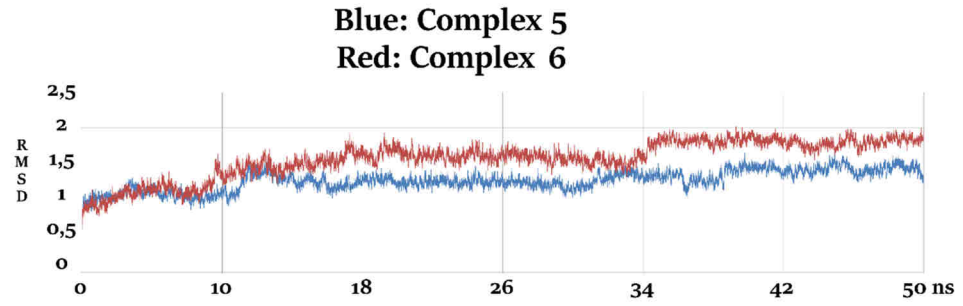


Figure 1-17: RMSD in Å calculated on α-carbons during the production MD of complexes 5 (A-SUC + 2x GS-B) and 6 (2x GS-B) with respect to the starting structure.

As shown in Figure 1-15, Figure 1-16 and Figure 1-17, twenty nanoseconds is the time required by the systems to reach an equilibrium, which is kept for the remaining time of simulation. Moreover, to better understand the trend of the whole system, RMSDs of ALLINIs were plotted

and compared. Analysing results reported in Figure 1-18, Figure 1-19 and Figure 1-20, the RMSDs appear very similar one to each other, regardless the type of ALLINI and the sucrose presence. The shifting observed in ALLINI-2 and GS-B's RMSD are due to the rotation of the C-O bond in their *t-butyl* moiety.

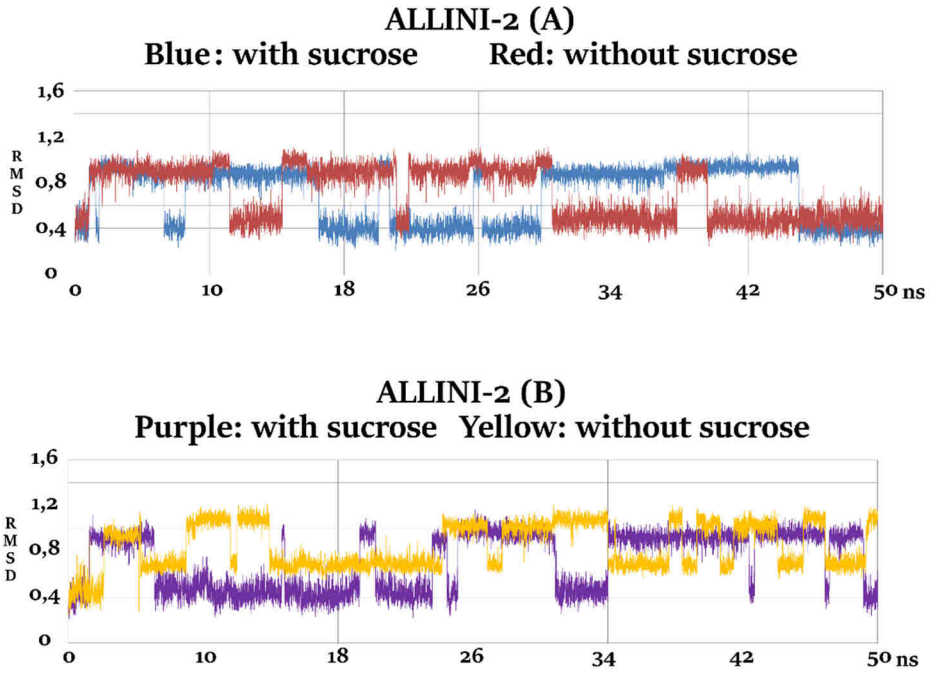


Figure 1-18: Root mean square deviation (Å) calculated on compound ALLINI-2 during the production MD in the presence or absence of sucrose. Two ALLINIs bind the IN at the same time (position A and B).

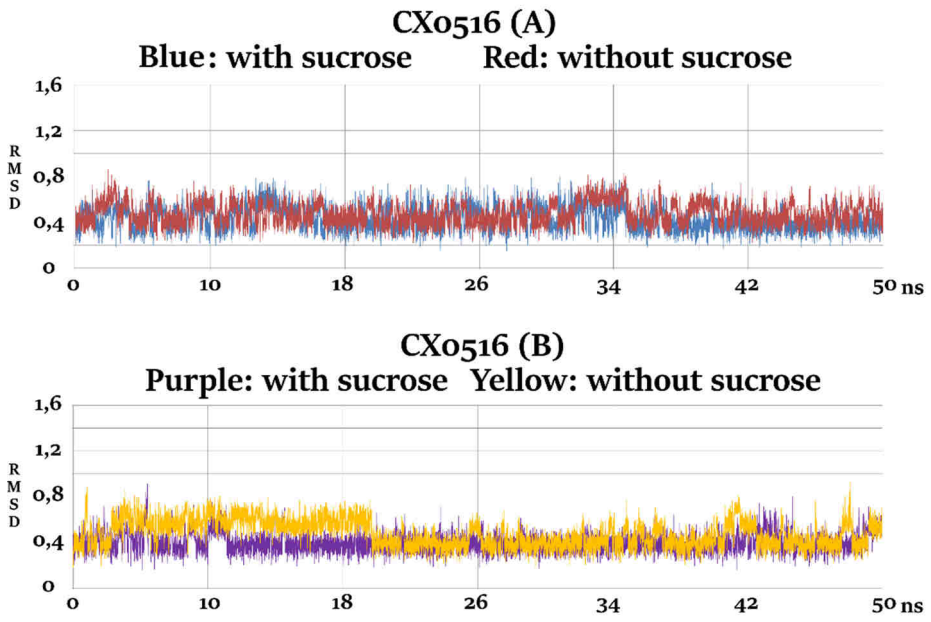


Figure 1-19: Root mean square deviation (Å) calculated on compound CX0516 during the production MD in the presence or absence of sucrose. Two ALLINIs bind the IN at the same time (position A and B).

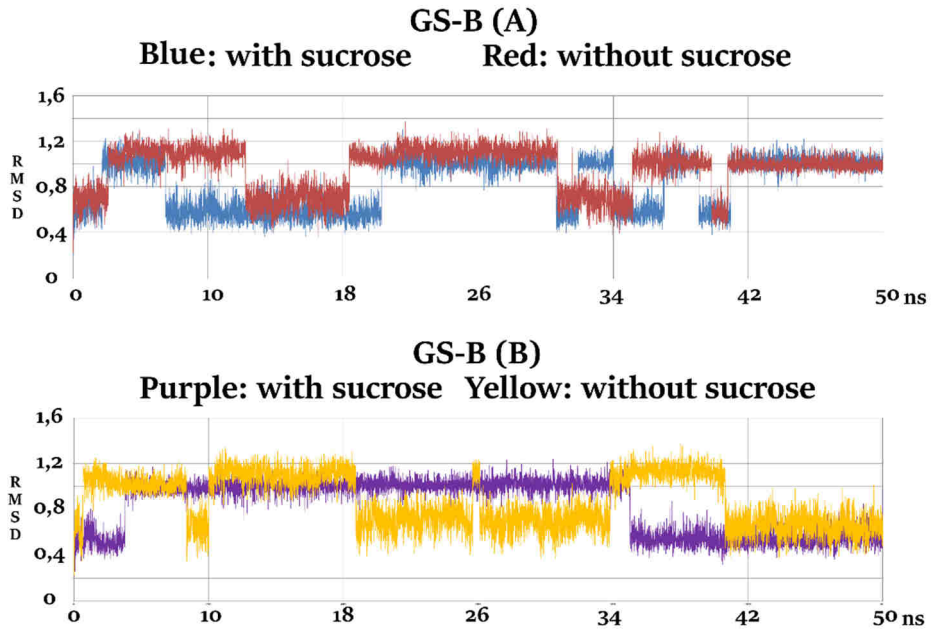


Figure 1-20: Root mean square deviation (Å) calculated on compound GS-B during the production MD in the presence or absence of sucrose. Two ALLINIs bind the IN at the same time (position A and B).

Also the RMSD of the heavy atoms of SUC (Figure 1 13) in each complex have been plotted and compared, showing that the ligand is maintained stable for all the MD.

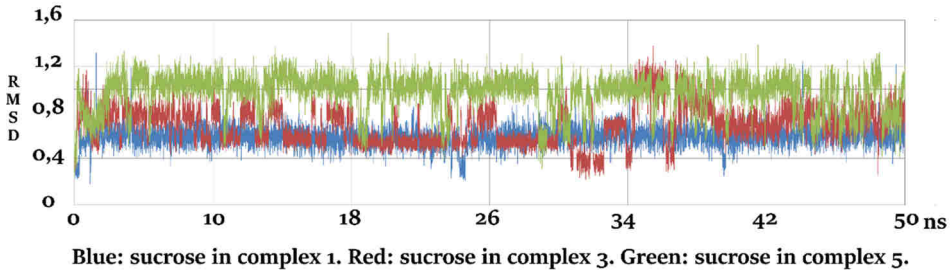


Figure 1-21: Root mean square deviation (Å) calculated on sucrose during the production MD of 1 (A-SUC + 2x ALLINI-2), 3 (A-SUC + 2x CX0516) and 5 (A-SUC + 2x GS-B) complexes.

Then, a total of 167 frames was extracted from 20 ns of each trajectory and used for the energetic analysis, of which results are reported in Table 1-2.

Table 1-2: Values of the binding free energy (kcal/mol) calculated with MM-GBSA technique for all the systems considered.

Ligands in complex with the CCD dimer of IN	Monomer-monomer binding free energy	First ALLINI binding energy	Second ALLINI binding energy	Sucrose energy
A-SUC + 2x ALLINI-2 (**1**)	-116,02 ± 6,38	-37,40 ± 3,42	-37,30 ± 3,11	-36,98 ± 3,61
2x ALLINI-2 (**2**)	-83,64 ± 6,35	-34,75 ± 3,30	-37,01 ± 3,01	/
A-SUC + 2x CX0516 (**3**)	-115,11 ± 6,77	-34,83 ± 3,26	-35,65 ± 2,37	-36,06 ± 5,84
2x CX0516 (**4**)	-82,57 ± 11,21	-31,78 ± 3,48	-33,37 ± 3,53	/
A-SUC + 2x GS-B (**5**)	-115,21 ± 6,07	-42,25 ± 2,93	-44,68 ± 3,09	-36,46 ± 4,20
2x GS-B (**6**)	-76,00 ± 8,02	-44,40 ± 2,90	-43,45 ± 3,47	/

Two main aspects emerge from the Table 1-2:

- Sucrose always led to an increase of the binding energy of about 30 kcal/mol. Compare the complex 1 with 2, 3 with 4 and 5 with 6.
- The binding free energy is not influenced by the presence of ALLINIs respect to the apo form. Compare the apo form of the Table 1-1 with the complexes 2, 4 and 6 in Table 1-2.

The second point seems to be apparently in contrast with the well know stabilizing effect of multimeric form owned by ALLINIs. For this reason, a molecular dynamic experiment was carried out on a full length tetrameric complex of IN, in which also the N-terminal domain and the C-terminal domain were added. Figure 1-21 shows the RMSD for α-carbons of all residues in tetramer complexes. Can be observed that the system containing four GS-B molecules (red line) appears to be more stable than the apo form (blue line), inasmuch has a smaller deviation respect to the initial position.

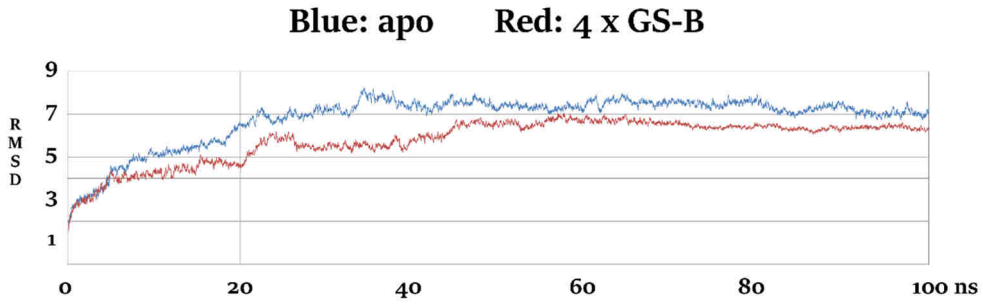


Figure 1-22: RMSD (Å) for α-carbons of all residues in tetramer complexes. Blue line: apo form. Red line: system containing four GS-B molecules.

The results of the MM-GBSA analysis are summarized in Table 1-3.

Table 1-3: Values of the binding free energy (kcal/mol) calculated with MM-GBSA technique for the tetramer in apo form or in complex with GS-B.

IN tetramer systems	Monomer-monomer binding free energy	ALLINI-A binding free energy	ALLINI-B binding free energy	ALLINI-C binding free energy	ALLINI-D binding free energy
apo	-143,16 ± 12,16	/	/	/	/
4x GS-B	-166,99 ± 13,69	-34,76 ± 4,79	-42,30 ± 3,18	-40,43 ± 3,27	-44,32 ± 3,90

The ligand GS-B has demonstrated a significant stabilization effect, increasing the free energy of binding between the two monomers from -143 to -167 kcal/mol. This suggests a sort of synergic effect between the two allosteric modulators involved in the stabilization of IN in its catalytically inactive states.

1.5.2.2 BIOLOGICAL ASSAYS

HTFR assay in presence of LEDGF/p75 was used to validate our theoretical hypothesis, evaluating the improvement of CX0516 inhibitory effect in association with sucrose. [45]

As reported in Figure 1-23, the IN-inhibition activity of the ALLINI alone is dose-dependent, and its IC_{50} is 20 µM. While the sucrose is inactive when testes alone, the addiction of 300 mM of the sugar to CX0516 reveals an increasing of 3 fold in CX0516 potency of IN inhibition assay. In fact, the IC_{50} of the ALLINI tested with SUC was decreased to 6 µM.

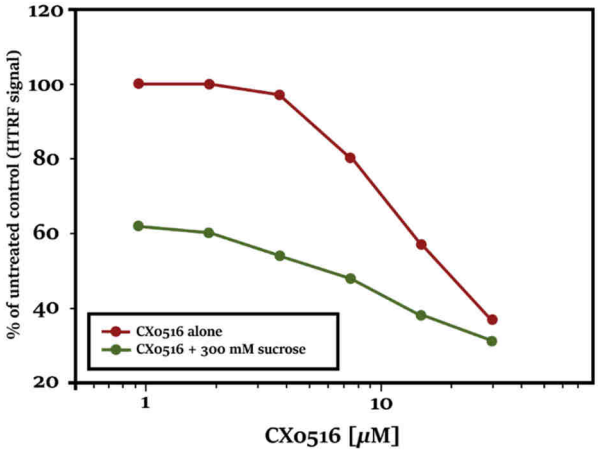


Figure 1-23: HTRF LEDGF/p75 dependent assay. Effects of CX0516 alone (red line) and CX0516 plus sucrose (green line).

The HTRF assay's result confirmed our hypothesis of a synergic effect between the two allosteric modulators. The same experiment was repeated with Raltegravir instead of CX0516, because we supposed that an increasing of the IN catalytic inhibition could be reached by improving the stabilization of IN subunits. In HTRF-based assay, of which results are reported in Figure 1-24, the drug has shown a value of IC_{50} equal to 53 nM when tested alone, while together with sucrose the behaviour was similar to what observed for CX0516, and the IC_{50} decreased to 2.4 nM showing an impressive 22 fold increase in potency.

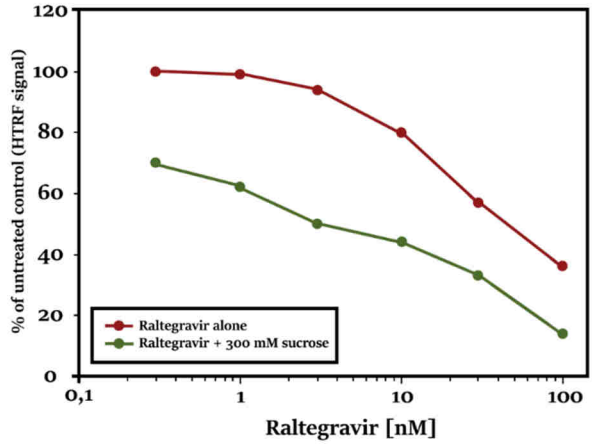


Figure 1-24: HTRF LEDGF/p75 dependent assay. Effects of Raltegravir alone (red line) and Raltegravir plus sucrose (green line).

This effect suggests that sucrose bounded to IN could affect the oligomerization, stabilizing it and leading to a more potent inhibition either of the IN monomer-oligomer equilibrium and the strand-transfer reaction.

1.5.3 CONCLUSIONS (I)

In conclusion, we found that a sucrose molecule bounded to an allosteric pocket located at the CCD interface could increase the potency of IN inhibitors, possibly stabilizing the enzyme in one of its forms, or promoting its oligomerization to inactive multimeric complexes. In particular, we performed biological experiments to confirm our computational findings, obtaining a stabilization of a multimeric form of IN. The catalytic inhibition of the enzymatic activity was demonstrated by the use of sucrose in association of CX0516 in the presence of LEDGF/p75 protein in an in vitro assay. Moreover, sucrose was able to enhance the inhibitory potency of the classic anti-IN drug Raltegravir, cooperatively acting with it. All these findings suggest that the modulators of the HIV-1 IN oligomerization state could be a new way to develop innovative anti-HIV agents. The Sucrose Binding Pocket was proved to be a very interesting site to develop new agents which could act in a cooperative way with both ALLINIs and classic INSTIs.

1.6 Kuwanon-L as a new allosteric HIV-1 integrase inhibitor

The second part of my work is a consequence of two important results obtained from the study carried out on the Sucrose Binding Pocket:

- The mutual binding free energy of the interaction between the two CCD monomers is increased by sucrose, which promotes the IN dimer formation.

- Sucrose enhances the IN inhibition potency of Raltegravir and inhibitors of the IN-LEDGF/p75 protein-protein interaction (ALLINIs), cooperatively acting with them.

Starting from these two main assumptions, we looked for molecules able to bind in the allosteric pocket at the interface of the CCD dimer replacing the sucrose, which is absolutely not drug-like.

1.6.1 COMPUTATIONAL STUDY

1.6.1.1 PROTEIN PREPARATION

A very similar computational procedure used in section 1.5.1.1 was applied to prepare the protein. Briefly, sucrose molecule was removed from 3L3V structure, which was then refined using the software Prime [50] and the structure 1BL3. Cadmium ion was transformed in Magnesium ion; mutated residues were restored to the original wt AAs, and the obtained structure was prepared by the Protein Preparation Wizard workflow. [53] Specifically, water molecules were deleted, hydrogen atoms added, bond orders and charges assigned. After optimization of the protonation state of His residues, also side chains of Asn and Gln residues and the orientation of hydroxyl groups on Ser, Thr and Tyr were optimized. Steric clashes were eased by performing a few minimization steps. However, these steps were not intended to minimize completely the system. The force field used was OPLS, the process was interrupted when 0.30 Å of RMSD of the non-hydrogen atoms was reached.

1.6.1.2 LIGAND PREPARATION

LigPrep from Maestro suite was used to predict the most probable tautomeric and ionization states (pH equal to 7 ± 1) for the library of natural compounds. Only those having more than 0,7 as normalized probability were retained in the final form of the library. OPLS2005 was used to energetically minimize the structures.

1.6.1.3 DOCKING STUDIES

The grid box for the docking was generated using as center the sucrose molecule present in the X-ray crystal structure without applying any kind of constraint, but allowing hydroxy groups rotation. The docking experiment and the scoring were performed using the Glide SP [60] mode.

1.6.1.4 MD SIMULATION, TRAJECTORIES AND MM/GBSA ANALYSIS

Ligands were parametrized by the Antechamber module of the Amber 12 program. [62] Charges were assigned using AM1-BCC method, [64] while atom types for each molecule were assigned using the Generalized Amber Force Field (GAFF). [65] All other operations were conducted using the same procedure already discussed in paragraph 1.5.1.5.

For trajectories analysis see paragraph 1.5.1.6.

For the MM/GBSA analysis see paragraph 1.5.1.7.

1.6.2 RESULTS AND DISCUSSION (II)

1.6.2.1 IN-SILICO

A small *in-house* library (473 molecules) of natural compounds fully characterised and with purity greater than 98% was used for docking experiments conducted on the SBP. Before proceeding, we decided to check the accuracy of the computational protocol evaluating its ability to reproduce the position of the co-crystallized sucrose molecule present in the crystal structure 3L3V. For doing this, the 3D coordinates of X-ray were retrieved from the PDB and prepared for docking. Two docking protocols implemented in Glide [60] were evaluated:

- SP: Single Precision.
- XP: Extra Precision.

The RMSD distance from the docked pose and the crystallographic sucrose was used as yardstick for the evaluation of the best performing protocol. Considering that both methods were able to dock the sugar with an RMSD value less then 1Å, the SP mode was chosen for further calculation because of its lower running time respect to the XP mode. Moreover, is remarkable that both sucrose poses present in the crystal structure (position A and position B) were the first and the second solution in terms of docking score, with values of -5,98 and -5,56 Kcal/mol respectively. The comparison with the crystallographic sucrose (both positions) is reported in Figure 1-25.

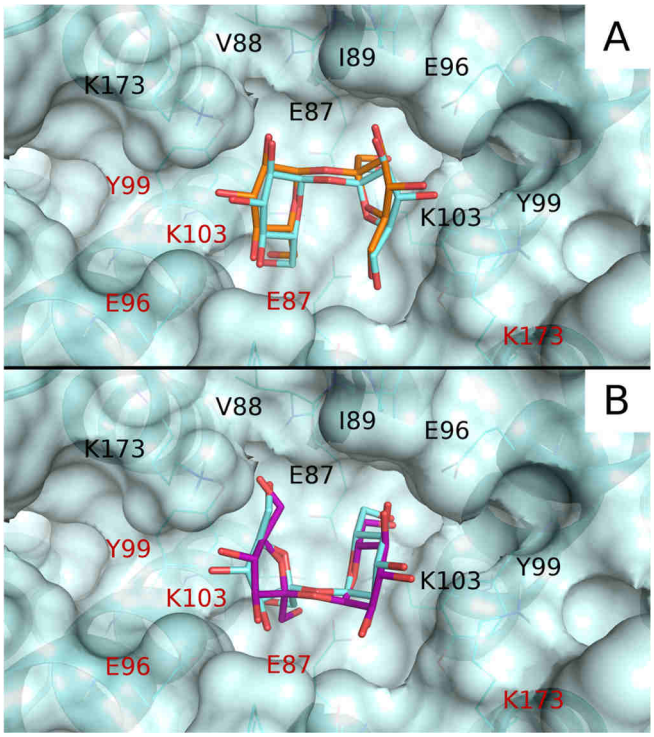


Figure 1-25: A) Comparison of the first position from docking (orange) with crystallographic sucrose in position A. B) Comparison of the second position from docking (purple) with crystallographic sucrose in position B.

Once chosen Glide SP as docking procedure after the validation step, that was applied to the selected library of natural compounds. The result of the docking showed a score range from 0 to -6,48 Kcal/mol. The first 5 molecules of the ranking list are shown in Table 1-4.

Table 1-4: Results of the first run of docking

Rank	Name	Docking Score (Kcal/mol)
1	Kuwanon-L	-6,48
2	Fustin	-6,31
3	Taxifolin	-6,20
4	1,3,5,6-tetra-OH-xanthone	-6,17
5	2,4,6-tri-OH-acetophenone	-6,10
…	…	…

As can be seen from the table, the first position is occupied by the natural compound Kuwanon-L, with a docking score value equal to -6,48 Kcal/mol. The docked pose is reported in Figure 1-26. Similar to sucrose, Kuwanon-L makes polar interactions with the following residues: E87, E96 and K173 of one monomer and E87 of the other monomer, as well as several hydrophobic interactions.

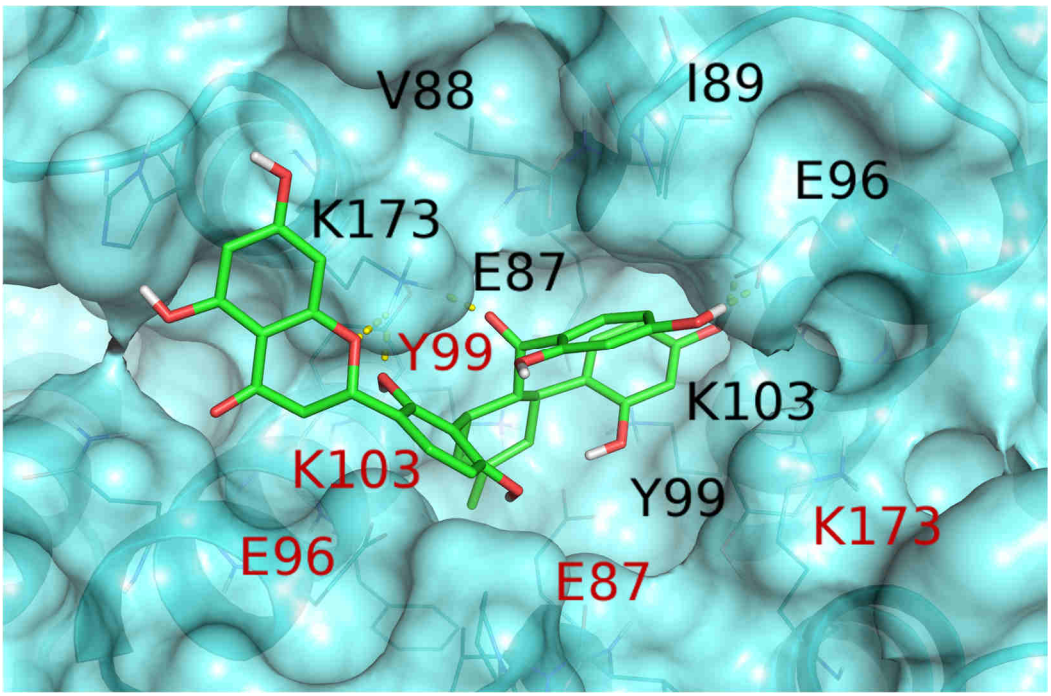


Figure 1-26: Docked pose of Kuwanon-L in SBP, first round. Docking Score -6,48 Kcal/mol.

In order to better analyse the binding capabilities of the library of natural molecules under investigation, a second run of docking was performed. This time a frame extracted from the trajectory of the molecular dynamic simulation of the CCD in complex with sucrose in position A was used as starting conformation for the protein. The frame corresponded to the lowest energy complex of the production trajectory. The complex, once extracted the sucrose molecule, was prepared and used for the docking using Glide SP. The same library of natural compounds showed docking scores ranging from -1,76 Kcal/mol to -8,84 Kcal/mol. Also in this second docking procedure, Kuwanon-L emerged as the best candidate scoring -8,84 Kcal/mol. As shown in Table 1-5, where the first 5 molecules are reported,

Kuwanon-L has a score of about 2 Kcal/mol lower than the second ligand in the ranking list.

Table 1-5: Results of the second run of docking.

Rank	Name	Docking Score (Kcal/mol)
1	Kuwanon-L	-8,84
2	2,4,6-tri-OH-acetophenone_b	-7,14
3	2,4,6-tri-OH-3-Prenilacetophenone_c	-6,86
4	2,4,6-tri-OH-acetophenone	-6,82
5	Fustin	-6,68
…	…	…

The binding mode resulting from this docking procedure is reported in Figure 1-27. In this round, the molecule maintains the interactions found in the previous docking, gaining other polar contacts with E87 and E96 of the second monomer. Moreover, the position of the natural compound is rotated respect to the previous docking and more buried inside the pocket, ensuring a gain in terms of energy interaction.

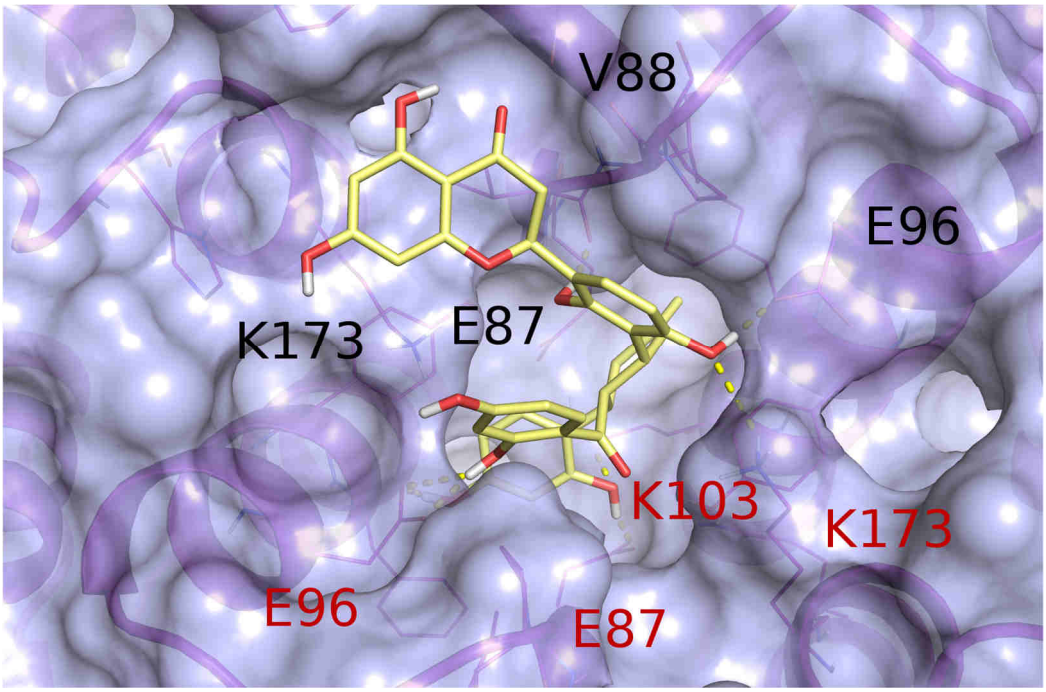


Figure 1-27: Docked pose of Kuwanon-L in SBP, second round. Docking Score -8,84 Kcal/mol.

Considering that Kuwanon-L was the first result in both the docking procedures, the complex from the second round of docking was chosen to undergo to MD simulation. In order to compare its behaviour with the one of the sucrose, the same computational protocol was applied. Briefly, a minimization of the system formed by the complex obtained with the docking inserted in a box of explicit water molecules was performed. Then, after 4 ns of equilibration, 100 ns of production MD simulation were run. The RMSD of the α carbons was calculated through the production of the simulation with respect to the starting complex, in order to check whether the system reached the equilibrium. The graph reported in Figure 1-28 represents it.

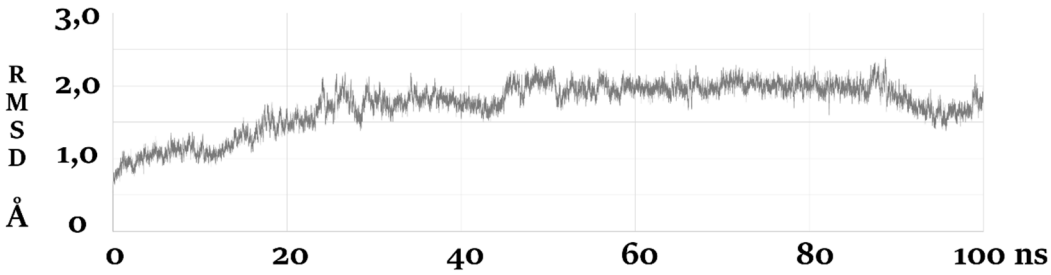


Figure 1-28: RMSD values calculated on α carbons of the system between Kuwanon-L and IN CCD analysed using molecular dynamic simulation.

As the graph shows, the first 25 ns are required from the system to conformationally rearrange the protein loops, then it reaches an equilibrium state around 2 Å of RMSD, which is maintained for the rest of the simulation. Regarding the ligand, also the time evolution of the RMSD values for the Kuwanon-L was calculated, and is reported in Figure 1-29.

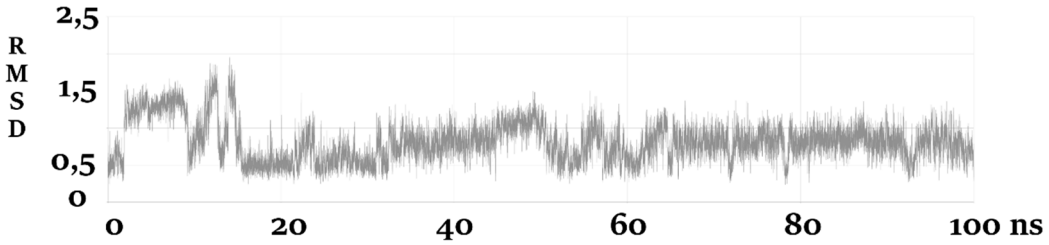


Figure 1-29: RMSD values calculated on all atoms of the Kuwanon-L ligand.

In Figure 1-29, an initial phase of flexibility in which all the system is rearranging can be observed. After that, the docking pose was maintained by the natural compound for the remaining time of simulation, with particular stability in the final stage of the MD. For this reason, the terminal 50 ns of the simulation were used to collect 417 frames to undergo to the MM-GBSA analysis. The value of free energy obtained for Kuwanon-L in the SBP was equal to -37,55 ± 3,67 Kcal/mol. Moreover, performing a residue decomposition to see which amino acids gave the major contribute to this energy, residues 87-90, 99 and 103 of both chains appeared to be the most important for ligand binding. Also the ligand hydrogen bonds were

monitored, and the ones with E87 and V88 were found to have an occupancy greater than 70%.

At the same time, Kuwanon-L promoted the stabilization and increased the binding free energy between the two interacting IN CCD monomers of about 30 Kcal/mol, passing from a value of -87 ± 9,28 Kcal/mol in the apo form to -119,45 ± 7,48 Kcal/mol in the presence of the natural compound. This value was perfectly comparable to that previously found for the sucrose molecule.

1.6.2.2 BIOLOGICAL ASSAYS

As in the experiment with sucrose, also this time the HTRF technique was used to see if Kuwanon-L, like ALLINIs, would have been able to inhibit the HIV-1 IN strand transfer activity both in absence and presence of LEDGF. Results, reported in Table 1-6, showed a similar behaviour for the natural product and CX0516 (taken as representative of ALLINIs class). In fact, both compounds inhibited the IN ST activity with comparable IC_{50}. Moreover, the natural product tested also to evaluate its ability to impair the IN/LEDGF binding, showed an activity very similar to the anti-HIV-1 IN activity. This, together with the proximity of the two sites (SBP and ALLINIs' binding pocket), suggested that Kuwanon-L could exerts the anti-IN activity also through the disruption of the IN-LEDGF binding.

Table 1-6: Effect of Kuwanon-L, CX0516 and Raltegravir on HIV-1 IN LEDGF dependent and independent strand transfer activity and on HIV-1 IN-LEDGF binding.

Compound	HIV-1 IN LEDGF dependent IC_{50} µM[a]	HIV-1 IN LEDGF independent IC_{50} µM[a]	HIV-1 IN–LEDGF Binding IC_{50} µM[b]
Kuwanon-L	43 ± 3	34 ± 0,5	22 ± 0,5
CX0516	9 ± 2	10 ± 0,5	13 ± 4
RAL	0,058 ± 0,01	0,061 ± 0,01	/

[a] Concentration of the compound required to inhibit HIV-1 IN LEDGF dependent and independent strand-transfer activity by 50%.
[b] Concentration of the compound required to reduce IN-LEDGF binding by 50%.

Considering that ALLINIs can modulate the dynamic interplay between IN subunits, also Kuwanon-L was tested to evaluate its ability to promote and stabilize IN multimers. This was tested in a HTRF IN subunit exchange assay in which, when the HTRF signal decreases a compound inhibits IN dimerization and whereas when the HTRF signal increases a compound promotes IN multimerization. The Figure 1-30 was obtained setting to 100% the HTRF signal for 100 µM of concentration of CX0516 (used as control) and Kuwanon-L, respectively. The first showed to promote the IN multimerization with a MI_{50} (concentration needed to inhibit the Multimerization Increase by 50%) equal to 20 µM whereas the second with MI_{50}=38 µM, indicating that both molecules were able to trap the IN in a more rigid multimeric form when bounded to the enzyme. The reduced flexibility led to the inhibition of the catalytic process.

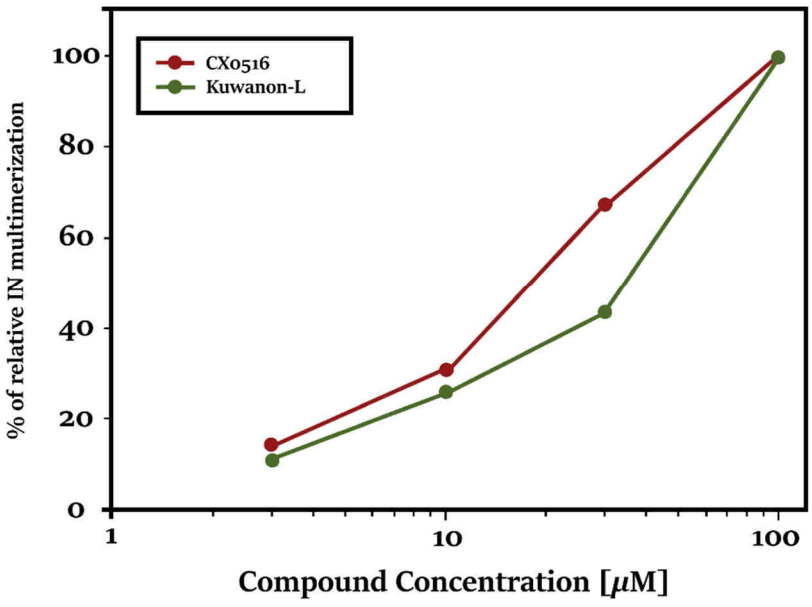


Figure 1-30: Effects of Kuwanon-L and CX0516 on HIV-1 IN multimerization.

The results agreed with the initial hypothesis of binding to SBP for Kuwanon-L, near to ALLINIs binding pocket. Moreover, to further confirm this data, a dual inhibition study was performed, producing an isobologram plot from which can be understood that Kuwanon-L and CX0516 effects

are additive on LEDGF/p75 independent IN inhibition (they don't bind to the same site).

A cellular test on a fully replicating HIV-1 laboratory strain (NL-4-3) was also conducted, to evaluate the ability of Kuwanon-L to inhibit viral replication. The EC_{50} value found for the natural product was of 1,9 μM. Interestingly, no toxic effects were shown at the highest tested concentration (CC_{50} > 20 μM). Figure 1-31 represents the result of the cellular test.

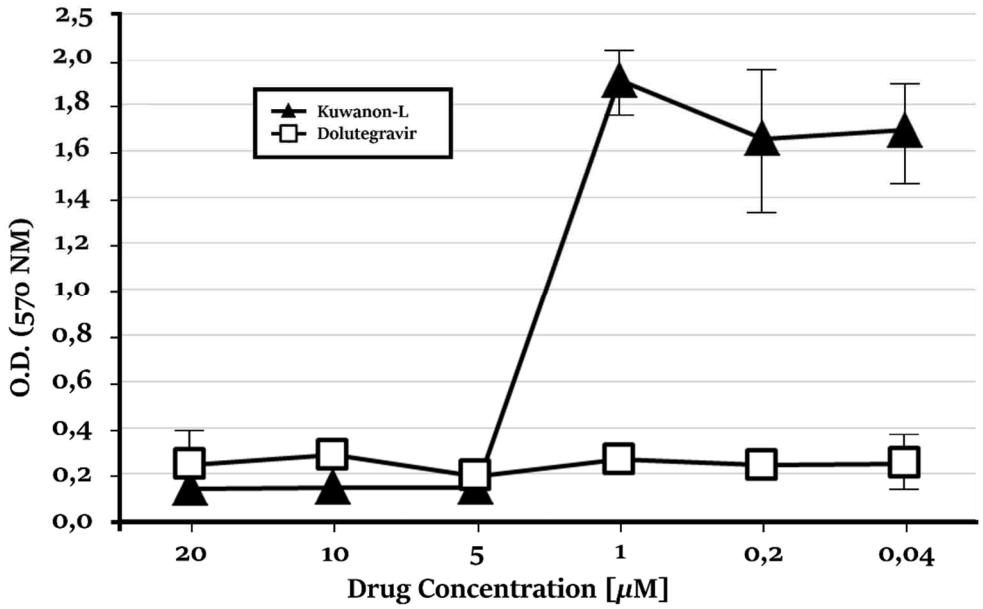


Figure 1-31: Cell culture: effect of Kuwanon-L on HIV-1 viral replication. O.D. (Optical Density) is reported in ordinate. Mean values and standard deviation are shown.

1.6.3 CONCLUSIONS (II)

In conclusion, starting from the investigation of the SBP, a new and very attractive molecule was discovered using different computational techniques: Kuwanon-L. The natural product showed an inhibition profile similar to the one reported for ALLINIs. The strand transfer activity owned by IN was inhibited by Kuwanon-L both in presence and in absence of LEDGF. It also impaired the binding between the enzyme and the cellular cofactor with an IC_{50} value of 22 μM, and promoted and stabilized the

inactive multimeric form of IN. Moreover, testing it in association with CX0516 the effect was additive, suggesting that Kuwanon-L doesn't bind to the same site of ALLINIs, but probably to a close one. Binding to the SBP, it could be able to disrupt enough the ALLINIs' pockets, thus preventing the binding of IN to LEDGF/p75. Finally, Kuwanon-L blocked the HIV-1 replication in cell-based assay, making it a novel and attractive lead with an allosteric mechanism of action targeting this viral enzyme Integrase.

1.7 Natural Product Kuwanon-L Inhibits HIV-1 Replication Through Multiple-Target Binding

The third part of the work on Kuwanon-L was focused on a deeper investigation on its mechanism of action. In this part, the natural compound was explored using a combination of biological experiments, both cellular and enzymatic. In particular, the time of addiction test revealed a peculiar trend from which can be supposed a multiple target binding to two key HIV-1 enzymes: IN and RT. Going more into details, the molecule was tested also against RT, revealing the ability to inhibit the two RT-related enzymatic activities: RDDP and RNase H.

1.7.1 RESULTS AND DISCUSSION (III)

1.7.1.1 BIOLOGICAL ASSAY

In the TOA experiment, the target of the Kuwanon-L was identified testing distinct viral replication steps, by comparing its activity in the time scale of different drugs used as reference.

- Maraviroc: CCR5 coreceptor inhibitor. A great loss of activity is present in the very early stages of viral infection, after the firsts 120 min.
- Lamivudine: RT inhibitor. The loss of activity is present after the reverse transcription step, which occurs just before the 180 min.
- Dolutegravir: IN inhibitor: After the integration step this drug loses the great part of its activity. It happens at 300 min.

The profile for the natural product resulting from the test appeared to be very particular. Analysing the inhibition path (Figure 1-32), the molecule lost its activity in two times. The first step was corresponding to the time in which also Lamivudine loses its activity, consistent with the inhibition of RT. The second one was instead congruent with the inhibition of viral IN (as already known), inasmuch after 300 min can be noticed a loss of activity of Kuwanon-L as well as Dolutegravir. This could be explained by the ability of the natural product to bind different targets during viral infection, namely the RT and the IN. Moreover, in accordance with the subsequent biochemical assays in which was showed an anti-RT activity greater than the anti-IN one, the highest loss of antiviral activity occurred when Kuwanon-L was added after the RT step, while if added after HIV integration was found a significant, but more modest, loss.

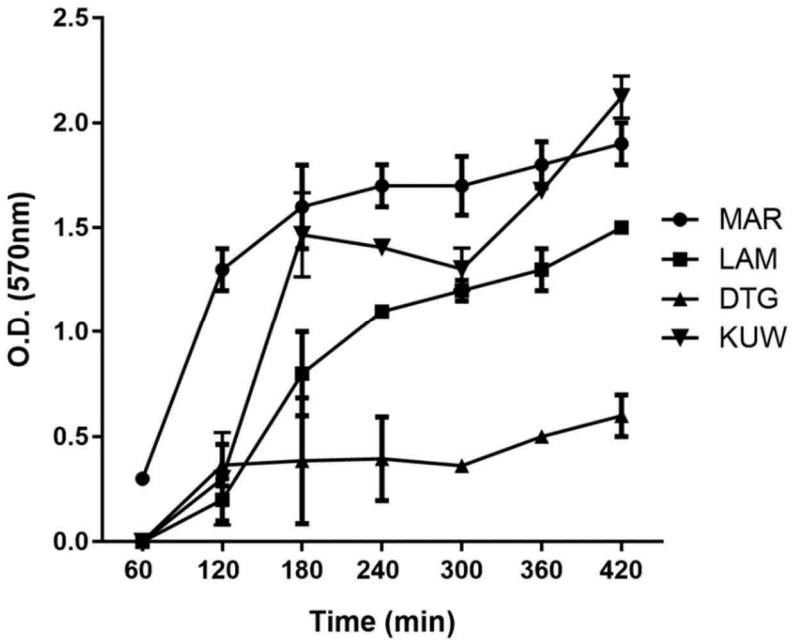


Figure 1-32: Time of Addition assay, inhibition profiles. MAR= Maraviroc; LAM= Lamivudine; DTG= Dolutegravir; KUW= Kuwanon-L; O.D.= optical density.

Following the idea that Kuwanon-L binds both IN and RT, and already having the results of the enzymatic assay on IN, an enzymatic test also on RT was performed, in order to confirm what suggested by TOA. The assays conducted on RT evaluated the ability of Kuwanon-L to inhibit both HIV-1 RDDP and RNase H functions, using efavirenz (EFV) and the RNase H active site inhibitor RDS1759 as reference controls. [66] Results of this enzymatic experiments are reported in Table 1-7, and they indicate that Kuwanon-L has a good potency in the disruption of both HIV-1 RT-associated activities.

Table 1-7: Effect of Kuwanon-L on HIV-1 RT-associated activities

Compound	**HIV-1 RT RDDP IC_{50} μM[a]**	**HIV-1 RT RNase H IC_{50} μM[b]**
Kuwanon-L	0,99 ± 0,25	0,57 ± 0,06
Efavirenz	0,025 ± 0,005	> 50
RDS1759	> 50	7,4 ± 0,2

[a] Concentration of the compound required to inhibit HIV-1 RT- associated RDDP activity activity by 50%.
[b] Concentration of the compound required to inhibit RT-associated RNase H activity by 50%.

Thus, Kuwanon-L inhibits RNase H function with an IC_{50} value of 0,57 µM and the RDDP function with an IC_{50} value of 0,99 µM. This very interesting result suggests that the natural product is able to impair different functions at the same time, in contrast with all other identified drugs and natural extracts that inhibit only one enzymatic function.

1.7.2 CONCLUSIONS (III)

The TOA experiment conducted while investigating the mode of action of the Kuwanon-L has revealed a very particular behaviour for this molecule. A deep analysis of the result prompt us to hypothesize a multiple target binding, because the loss of activity was found at two different times: the first one compatible to RT inhibition, while the second to IN inhibition. While the anti-IN activity was already known from previous experiments (meaning that the loss of activity corresponding to IN inhibition was expected), it was not the same for RT. This fact prompted us to test Kuwanon-L also on isolated RT. The assay revealed the ability to inhibit both the RT-associated activities: RDDP and RNase H, confirming the initial hypothesis of multiple target binding.

In conclusion, Kuwanon-L represents the first compound able to inhibit both HIV-1 RT-associated activities and HIV-1 IN. This molecule can be considered a solid starting point for the multiple target binding strategy. The approach of multiple binding is very interesting, mainly because can let the reduction of the number of drugs administered to HIV-1 positive patients and decrease the selection of drug-resistant strains, as well as possible drug-drug interactions.

1.8 References

1 Control, Centers for Disease. Kaposi's Sarcoma and Pneumocystis Pneumonia among homosexual men in New York City and California. *MMWR Morb Mortal Wkly Rep*, 30, 25 (Jul 1981), 305-308.

2 Barré-Sinoussi, F; Chermann, JC; Rey, F et al. Isolation of a T-lymphotropic retrovirus from a patient at risk for acquired immune deficiency syndrome (AIDS). *Science*, 220, 4599 (May 1983), 868-871.

3 Gallo, RC; Sarin, PS; Gelmann, EP et al. Isolation of human T-cell leukemia virus in acquired immune deficiency syndrome (AIDS). *Science*, 220, 4599 (May 1983), 865-867.

4 FOOD AND DRUG ADMINISTRATION. *http://www.fda.gov/ForPatients/Illness/HIVAIDS/Treatment/ucm118915.htm*. Accessed Sep 2016.

5 Antonelli, G and Turriziani, O. Antiviral therapy: old and current issues. *Int J Antimicrob Agents*, 40, 2 (Aug 2012), 95-102.

6 Tintori, C; Brai, A; Fallacara, AL; Fazi, R; Schenone, S; M, Botta. Protein-protein interactions and human cellular cofactors as new targets for HIV therapy. *Curr Opin Pharmacol*, 18 (Oct 2014), 1-8.

7 WORLD HEALTH ORGANIZATION. *http://www.who.int/hiv/en/*. Accessed Sep 2016.

8 Smith, JA: Daniel, R. Following the path of the virus: the exploitation of host DNA repair mechanisms by retroviruses. *ACS Chem Biol*, 1, 4 (May 2006), 217-226.

9 Turner, BG and Summers, MF. Structural biology of HIV. *J Mol Biol*, 285, 1 (Jan 1999), 1-32.

10 Tiberi, M; Tintori, C; Ceresola, ER et al. 2-Aminothiazolones as anti-HIV agents that act as gp120-CD4 inhibitors. *Antimicrob Agents Chemother*, 58, 6 (Jun 2014), 3043-3052.

11 Turner, BG and Summers, MF. Structural biology of HIV. *J Mol Biol*, 2885, 1 (Jan 1999), 1-32.

12 Deng, H; Liu, R; Ellmeier, W et al. Identification of a major co-receptor for primary isolates of HIV-1. *Nature*, 381, 6584 (Jun 1996), 661-666.

13 Chan, DC and Kim, PS. HIV entry and its inhibition. *Cell*, 93, 5 (May 1998), 681-684.

14 Jayappa, KD; Ao, Z; Yao, X. The HIV-1 passage from cytoplasm to nucleus: the process involving a complex exchange between the components of HIV-1 and cellular machinery to access nucleus and successful integration.Jayappa KD1, Ao Z, Yao X. *Int J Biochem Mol Biol*, 3, 1 (2012), 70-85.

15 Ciuffi, A; Llano, M; Poeschla, E et al. A role for LEDGF/p75 in targeting HIV DNA integration. *Nat Med*, 11, 12 (Dec 2005), 1287-1289.

16 Brown, PO. Integration of retroviral DNA. *Curr Top Microbiol Immunol*, 157 (1990), 19-48.

17 Abram, ME; Ferris, AL; Das, K et al. Mutations in HIV-1 Reverse Transcriptase Affect the Errors Made in a Single Cycle of Viral Replication. *J Virol*, 88, 13 (Jul 2014), 7589-7601.

18 Menéndez-Arias, L. Molecular basis of human immunodeficiency virus type 1 drug resistance: overview and recent developments. *Antiviral Res*, 98, 1 (Apr 2013), 93-120.

19 Dando, TM and Perry, CM. Enfuvirtide. *Drugs*, 63, 24 (2003), 2755-2766.

20 Fadel, H and Temesgen, Z. Maraviroc. *Drugs Today (Barc)*, 43, 11 (Nov 2007), 749-758.

21 Locatelli, GA; Cancio, R; Spadari, S; Maga, G. HIV-1 reverse transcriptase inhibitors: current issues and future perspectives. *Curr Drug Metab*, 5, 4 (Aug 2004), 283-290.

22 Wright, K. AIDS therapy. First tentative signs of therapeutic promise. *Nature*, 323, 6086 (1986), 283.

23 De Clercq, E. Non-nucleoside reverse transcriptase inhibitors (NNRTIs): past, present, and future. *Chem Biodivers*, 1, 1 (Jan 2004), 44-64.

24 Bowersox, J. Nevirapine approved by FDA. Food and Drug Administration. *NIAID AIDS Agenda* (Sep 1996), 10.

25 La Porte, CJ. Saquinavir, the pioneer antiretroviral protease inhibitor. *Expert Opin Drug Metab Toxicol*, 5, 10 (Oct 2009), 1313-1322.

26 Schafer, JJ and Squires, KE. Integrase inhibitors: a novel class of antiretroviral agents. *Ann Pharmacother*, 44, 1 (Jan 2010), 145-156.

27 Lenz, JC and Rockstroh, JK. S/GSK1349572, a new integrase inhibitor for the treatment of HIV: promises and challenges. *Expert Opin Investig Drugs*, 20, 4 (Apr 2011), 537-548.

28 Zheng, R; Jenkins, TM; Craigie, R. Zinc folds the N-terminal domain of HIV-1 integrase, promotes multimerization, and enhances catalytic activity. *Proc Natl Acad Sci U S A*, 93 (1996), 13659-13664.

29 Chiu, TK and Davies, DR. Structure and function of HIV-1 integrase. *Curr Top Med Chem*, 4 (2004), 965-977.

30 Lutzke, RA and Plasterk, RH. Structure-based mutational analysis of the C-terminal DNA-binding domain of human immunodeficiency virus type 1 integrase: critical residues for protein oligomerization and DNA binding. *J Virol*, 72 (1998), 4841-4848.

31 Christ, F; Voet, A; Marchand, A et al. Rational design of small-molecule inhibitors of the LEDGF/p75-integrase interaction and HIV replication. *Nat Chem Biol*, 6, 6 (Jun 2010), 442-448.

32 Christ, F and Debyser, Z. The LEDGF/p75 integrase interaction, a novel target for anti-HIV therapy. *Virology*, 435, 1 (Jan 2013), 102-109.

33 Cherepanov, P; Sun, ZY; Rahman, S; Maertens, G; Wagner, G; Engelman, A. Solution structure of the HIV-1 integrase-binding domain in LEDGF/p75. *Nat Struct Mol Biol*, 12, 6 (Jun 2005), 526-532.

34 Al-Mawsawi, LQ; Christ, F; Dayam, R; Debyser, Z; Neamati, N. Inhibitory profile of a LEDGF/p75 peptide against HIV-1 integrase: insight into integrase-DNA complex formation and catalysis. *FEBS Lett*, 582, 10 (Apr 2008), 1425-1430.

35 Cherepanov, P; Ambrosio, AL; Rahman, S; Ellenberger, T; Engelman, A. Structural basis for the recognition between HIV-1 integrase and transcriptional coactivator p75. *Proc Natl Acad Sci U S A*, 102, 48 (Nov 2005), 17308-17313.

36 Engelman, A; Kessl, JJ; Kvaratskhelia, M. Allosteric inhibition of HIV-1 integrase activity. *Curr Opin Chem Biol*, 17, 3 (Jun 2013), 339-345.

37 Feng, L; Sharma, A; Slaughter, A et al. The A128T resistance mutation reveals aberrant protein multimerization as the primary mechanism of action of allosteric HIV-1 integrase inhibitors. *J Biol Chem*, 288, 22 (May 2013), 15813-15820.

38 Christ, F; Shaw, S; Demeulemeester, J et al. Small-molecule inhibitors of the LEDGF/p75 binding site of integrase block HIV replication and modulate integrase multimerization. *Antimicrob Agents Chemother*, 56, 8 (Aug 2012), 4365-4374.

39 Tsiang, M; Jones, GS; Niedziela-Majka, A et al. New class of HIV-1 integrase (IN) inhibitors with a dual mode of action. *J Biol Chem*, 287, 25 (Jun 2012), 21189-21203.

40 Maroun, RG; Gayet, S; Benleulmi et al. Peptide inhibitors of HIV-1 integrase dissociate the enzyme oligomers. *Biochemistry*, 40, 46 (Nov 2001), 13840-13848.

41 Tintori, C; Demeulemeester, J; Franchi, L; Massa, S; Debyser, Z; Christ, F; Botta, M. Discovery of small molecule HIV-1 integrase dimerization inhibitors. *Bioorg Med Chem Lett*, 22, 9 (May 2012), 3109-3114.

42 Weisel, M; Proschak, E; Schneider, G. PocketPicker: analysis of ligand binding-sites with shape descriptors. *Chem Cent J*, 1 (Mar 2007), 7.

43 Demeulemeester, J; Tintori, C; Botta, M; Debyser, Z; Christ, F. Development of an AlphaScreen-based HIV-1 integrase dimerization assay for discovery of novel allosteric inhibitors. *J Biomol Screen*, 17, 5 (Jun 2012), 618-628.

44 Sharma, A; Slaughter, A; Jena, N et al. A new class of multimerization selective inhibitors of HIV-1 integrase. *PLoS Pathog*, 10, 5 (May 2014), e1004171.

45 Kessl, JJ; Jena, N; Koh, Y et al. Multimode, cooperative mechanism of action of allosteric HIV-1 integrase inhibitors. *J Biol Chem*, 287, 20 (May 2012), 16801-16811.

46 Wielens, J; Headey, SJ; Jeevarajah, D et al. Crystal structure of the HIV-1 integrase core domain in complex with sucrose reveals details of an allosteric inhibitory binding site. *FEBS Lett*, 584, 8 (Apr 2010), 1455-1462.

47 Du, L; Zhao, YX; Yang, LM; Zheng, YT; Tang, Y; Shen, X; Jiang, HL. Symmetrical 1-pyrrolidineacetamide showing anti-HIV activity through a new binding site on HIV-1 integrase. *Acta Pharmacol Sin*, 29, 10 (Oct 2008), 1261-1267.

48 Shkriabai, N; Patil, SS; Hess, S; Budihas, SR: Craigie. R: Burke. TR Jr: Le Grice. SF: Kvaratskhelia. M. Identification of an inhibitor-binding site to HIV-1 integrase with affinity acetylation and mass spectrometry. *Proc Natl Acad Sci U S A*, 101, 18 (May 2004), 6894-6899.

49 Kessl, JJ; Eidahl, JO; Shkriabai, N et al. An allosteric mechanism for inhibiting HIV-1 integrase with a small molecule. *Mol Pharmacol*, 76, 4 (Oct 2009), 824-832.

50 SCHRÖDINGER RELEASE 2013-3. *Prime*. Schrödinger, LLC, New York, NY, 2013.

51 Maignan, S; Guilloteau, JP; Zhou-Liu, Q; Clément-Mella, C; Mikol, V. Crystal structures of the catalytic domain of HIV-1 integrase free and complexed with its metal cofactor: high level of similarity of the active site with other viral integrases. *J Mol Biol*, 282, 2 (Sep 1998), 359-368.

52 UniProt. *http://www.uniprot.org/uniprot/Q76353*.

53 SCHRÖDINGER RELEASE 2013-3. *Schrödinger Suite 2013 Protein Preparation Wizard; Epik version 2.6, Schrödinger, LLC, New York, NY, 2013; Impact version 6.1, Schrödinger, LLC, New York, NY, 2013; Prime version 3.4, Schrödinger, LLC, New York, NY, 2013.*

54 Tintori, C; Veljkovic, N; Veljkovic, V; Botta, M. Computational studies of the interaction between the HIV-1 integrase tetramer and the cofactor LEDGF/p75: insights from molecular dynamics simulations and the informational spectrum method. *Proteins*, 78, 16 (Dec 2010), 3396-3408.

55 Christ, F; Voet, A; Marchand, A et al. Rational design of small-molecule inhibitors of the LEDGF/p75-integrase interaction and HIV replication. *Nat Chem Biol*, 6, 6 (Jun 2010), 442-448.

56 Tsiang, M; Jones, GS; Niedziela-Majka, A et al. New class of HIV-1 integrase (IN) inhibitors with a dual mode of action. *J Biol Chem*, 287, 25 (Jun 2012), 21189-21203.

57 SCHRÖDINGER SUITE 2013-3. *Maestro*. Schrödinger, LLC, New, 2013.

58 SCHRÖDINGER RELEASE 2013-3. *MacroModel*. Schrödinger, LLC, New York, NY, 2013.

59 SCHRÖDINGER RELEASE 2013-3. *LigPrep*. Schrödinger, LLC, New York, NY, 2013.

60 SCHRÖDINGER RELEASE 2013-3. *Glide*. Schrödinger, LLC, New York, NY, 2013.

61 Wang, J; Wang, W; Kollman, PA; Case, DA. Automatic atom type and bond type perception in molecular mechanical calculations. *J Mol Graph Model*, 25, 2 (Oct 2006), 247-260.

62 AMBER. *http://ambermd.org/*. San Diego.

63 Wang, J; Wolf, RM; Caldwell, JW; Kollman, PA; Case, DA. Development and testing of a general amber force field. *J Comput Chem*, 25, 9 (Jul 2004), 1157-1174.

64 Jakalian, A; Jack, DB; Bayly, CI. Fast, efficient generation of high-quality atomic charges. AM1-BCC model: II. Parameterization and validation. *J Comput Chem*, 23, 16 (Dec 2002), 1623-1641.

65 Wang, J; Wolf, RM; Caldwell, JW; Kollman, PA; Case, DA. Development and testing of a general amber force field. *J Comput Chem*, 25, 9 (Jul 2004), 1157-1174.

66 Corona, A; Di Leva, FS; Thierry, S et al. Identification of highly conserved residues involved in inhibition of HIV-1 RNase H function by Diketo acid derivatives. *Antimicrob Agents Chemother*, 58, 10 (Oct 2014), 6101-6110.

CHAPTER 2

Innovative anti-Candida agents

Deodato D, Maccari G, De Luca F, Sanfilippo S, Casian A, Martini R, D'Arezzo S, Bonchi C, Bugli F, Posteraro B, Vandeputte P, Sanglard D, Docquier JD, Sanguinetti M, Visca P, Botta M.
J Med Chem. ***2016***. *3854-3866. doi: 10.1021/acs.jmedchem.6b00018.*

2.1 Introduction

2.1.1 FUNGI

The fungi are a group of eukaryotic microorganisms, some of which can grow as yeasts (unicellular form) or as hyphae (assembled together in an organized group), they produce spores and also other reproductive structures. [1] Fungal cells have an external cell wall made of mannanes, chitin and glucans, which are extensively cross-linked together (Figure 2-1). This structure is crucial for the life of fungi, inasmuch allows the cell to interact with the environment while protecting itself from environmental stresses. [2] The cell wall is very dynamic; is reshaped by hydrolytic enzymes closely associated to the cell wall, like chitinases and glucanases, which are also responsible to digest chitin and glucans during cell division. Considering the unicity of the fungal cell wall, it results to be an excellent target for antifungal treatment. [3] The cellular membrane, made of phospholipids and proteins, is present directly under the cell wall. Here, the integrity and asymmetry of the membrane is maintained by ergosterol, a molecule common to all fungi and not present in human cells. [4] Apart for the membrane and the cell wall, the fungus has all the others typical biostructures of eukaryotic cells.

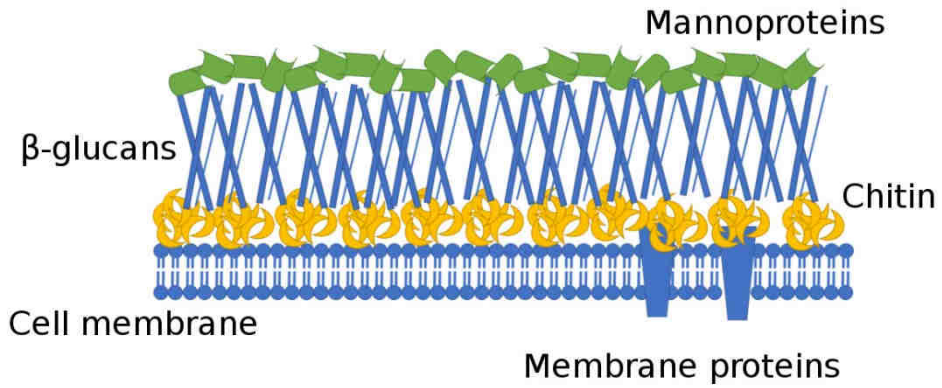


Figure 2-1: Fungal cell wall structure representation. [5]

2.1.2 MYCOSES

During the last years, fungal infections caused by opportunistic fungi have alarmingly increased. This mainly occurs because of the increasing of immunocompromised patients, as a result of more and advanced chemotherapies, transplants, misuse of broad spectrum antibiotics, diseases like AIDS and prolonged hospitalization. [6]

The classification of mycoses depends on the mode of entry into the host and the tissue localization. There are three main categories: [7]

- Superficial mycoses: These infections are mainly localised on the skin, hair and nails. An example is tinea, a fungal infection of the skin that can infect various part of the body: the scalp (*Tinea capitis*), the beard (*Tinea barbae*) the foot (*Tinea pedis*: "athlete's foot") and the groin (*Tinea cruris*). Other types of common superficial mycoses are caused by *Candida albicans*, which infects the vagina or the mouth. This specie is a regular commensal of the flora of the gastrointestinal tract and the vagina, but, during illness or imbalances in the immune system can multiplies and become dangerous causing the disease.
- Subcutaneous mycoses: The dermis, subcutaneous tissue or adjacent structures are the main location for these kinds of infection usually developed following the traumatic implantation of the etiologic agent, like through wounds. However, they are rare and confined mainly to tropical regions. Usually the disease remains localized and then slowly spreads to adjacent tissue. Eumycotic mycetoma, hyalohyphomycosis, phaeohyphomycosis, chromoblastomycosis and sporotrichosis are the most common subcutaneous mycoses. [8] Among them, the most common in Europe was sporotrichosis, which is now become rare.
- Systemic mycoses (primary and opportunistic): These types of infections regard fungi that are entered in the body through a deep focus (lungs, paranasal sinuses or the digestive tract) or

through an internal organ. [9] Such infections can spread all around the body using the blood stream causing disseminated disease. Systemic mycoses can be further divided in two types: primary fungal infections (or endemic respiratory infections) and opportunistic infections. The first types (histoplasmosis, blastomycosis, coccidioidomycosis, paracoccidioidomycosis, and cryptococcosis) occur in previously healthy people and arises through the respiratory route. The seconds instead occur in patient with compromised immune system or in who have undergone surgery; in these cases not pathogenic fungi may become infective. Examples are systemic candidiasis, aspergillosis, cryptococcosis and systemic mucormycosis. [10]

2.1.3 CANDIDA SPECIES AND EPIDEMIOLOGY

Among the three types of mycoses, deep infections are undoubtedly the more dangerous. In developed countries, the treatment of this kind of infection represents a severe issue because of the increasing of population at risk, [11] especially from opportunistic species. In particular, Candida species represent the most frequent clinic isolates. The fungus, as already mentioned, is normally present in specific parts of the human body, but can become opportunistic under certain conditions. This kind of candidiasis (systemic and opportunistic) are associated with a high mortality rate (10%-49%) and an excess length of hospital stay of 3-30 days. [12] In a study from 2014, among 462 candidemia episodes collected, *Candida albicans* had a share of 49,4%, followed by *Candida parapsilosis* with 26,2% and *Candida glabrata* with 10,4%. [13] The main cause of the virulence is its adaptability; *Candida albicans* can relatively quickly morph from yeast to hyphae and can produce biofilm to protect itself to the action of the drugs. Among the other candida species (collectively called Non-Albicans Candida, NAC), *Candida glabrata* results to be highly virulent and is associated with treatment failure due to reduced susceptibility to antifungal agents. [14] Other NAC are represented by *Candida tropicalis*, *Candida parapsilosis*,

Candida krusei, *Candida guilliermondii*, *Candida lusitaniae*, *Candida dubliniensis*, *Candida pelliculosa*, *Candida kefyr*, *Candida lipolytica*, *Candida famata*, *Candida inconspicua*, *Candida rugosa*, and *Candida norvegensis*. [15]

2.1.3.1 VIRULENCE OF CANDIDA ALBICANS

Candida albicans causes every year between 250000 and 400000 deaths worldwide. [16] The reasons of these high numbers are due the capacity of the fungus to adapt its organism in response to environmental changes. One of the key adaptation features is, for example, polymorphism, shifting from budding yeast to hyphae. The last one (hyphae) is much more invasive respect to the first one (yeast). [17] There is adhesion: the capacity to adhere either to host cells or abiotic surfaces (fundamental factor of virulence). [18] Contact sensing: in the moment that *Candida albicans* touches any type of surface (biotic or abiotic), the fungus shift from yeast to hyphal form and starts the invasion. pH sensing: the change of pH is one of the major challenges that a pathogen can encounter. *Candida albicans*, thanks to different protein expression according to the pH, can virtually colonize any organs of the human body, from pHs of <2 to pHs of >10. [19] Secretion of hydrolases: once the adhesion is accomplished, proteases, lipases and phospholipases are expressed and secreted in order to facilitate the penetration. [20] Biofilm formation: The biofilm is an assemblage of microbial cells associated with a surface and enclosed in an exopolysaccharide matrix. [21] The scope of biofilm, as already mentioned, is basically the protection of the inner microorganisms from external factor that could harm them. Is a very powerful defence, because once created it works as a barrier for every kind of pharmacological treatment, inasmuch avoids the possibility for the drugs to reach the living organisms. Thus, with the onset of the biofilm the drug resistance increases, also because of the support of the expression of efflux pumps and others drug-resistance related factors.

2.2 Current Antifungal Therapy

Nowadays only four different classes of molecules are available on the market to treat systemic fungal infections: azoles, polyenes, allylamines and echinocandins.

2.2.1 AZOLES

This class of molecules can be further divided into two subclasses: imidazoles, of which the more representative structures are Clotrimazole, Bifonazole and Ketoconazole, and triazoles, mainly represented by Fluconazole, Voriconazole, Itraconazole and Posaconazole (Figure 2-2). Nowadays the second subclass has replaced the first one in the systemic use, while imidazoles are still used topically.

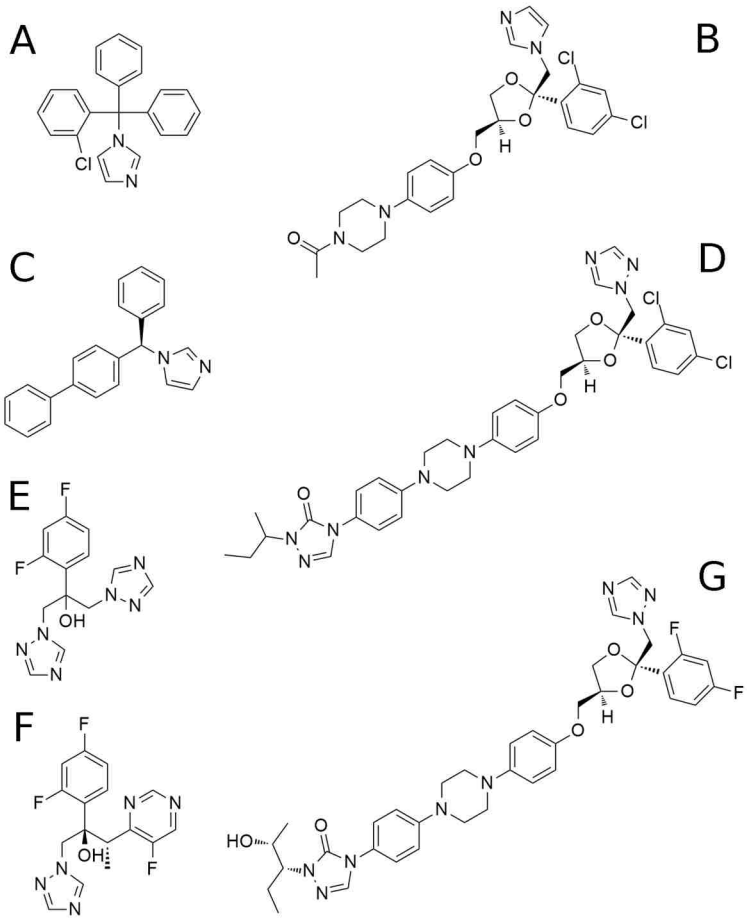


Figure 2-2: 2D structures of azoles antifungals. A: Clotrimazole. B: Ketoconazole. C: Bifonazole. D: Itraconazole. E: Fluconazole. F: Voriconazole. G: Posaconazole.

The mechanism of action is the same for both the subclasses and involves the inhibition of the lanosterol 14-α-demethylase, a cytochrome P450-dependent enzyme, [22] leading to depletion of ergosterol. This conducts to an accumulation of sterol precursors that alter the structure and function of the fungal cell membrane.

2.2.1.1 RESISTANCE TO AZOLES

In the literature, beside the biofilm development, three different mechanisms of resistance are reported for azoles drugs: expression of efflux pumps, alteration of the target site and up-regulation of target enzyme.

- Efflux pumps: The expression of efflux pumps promotes the reduction of the cytoplasmic concentration of the antifungal drugs, which undergo efflux from the cell to the external environment. They are up-regulated when the fungus detects the presence of drugs. In Candida species, these transporters are coded by two gene families: CDR (Candida Drug Resistance) and MDR (Multi-Drug Resistance). [23] The first family exploit ATP to exert its efflux action, in fact belongs to the APT-binding cassette (ABC) superfamily. The two main representative genes that express this class of protein in *Candida albicans* are CDR1 and CDR2. [24] Differently from CDR family, MDR codes for protein that uses the protonic gradient to exert its action. The main example is Mdr1p, which exports a variety of structurally unrelated compounds, of which the most important is the Fluconazole. [25]

- Alteration of the target site: ERG11, the gene encoding for the enzyme lanosterol C14-α-demethylase, can mutate producing a protein able to prevent the binding of azoles to the enzymatic site. [26] Moreover, a specie like *Candida krusei* is intrinsic resistant to Fluconazole, because of the low affinity of its ERG11p for the drug caused by an excess of 80 amino acid substitutions. [27]

- Up-regulation of target enzyme: Under antifungal drugs pressure, some Candida species can amplify the gene, increase the transcription rate or decrease the degradation of the target protein. Although this mechanism contributes less than others to resistance, higher intracellular concentrations of ERG11p were found to be related with reduced susceptibility to azoles. [28]

2.2.2 POLYENES

This class of molecules interact and perturb the fungal cell membrane. Although the exact mechanism of action still remains not fully understood, has been proposed that polyene macrolides can form pores in the fungal cell, allowing the leak of small molecules and ions and inducing thereby the cell death. [29] It has been suggested that polyenes form aqueous pores using 8 molecules hydrophobically interacting between themselves and with membrane sterol. [30] The most important drug of this group is the polyene macrolides Amphotericin B (Figure 2-3). The drug exhibit remarkable differentiation between mammalian cell membranes and fungal one. This is probably due to the relatively high affinity of Amphotericin B towards membranes containing ergosterol compared to those incorporating cholesterol. [31]

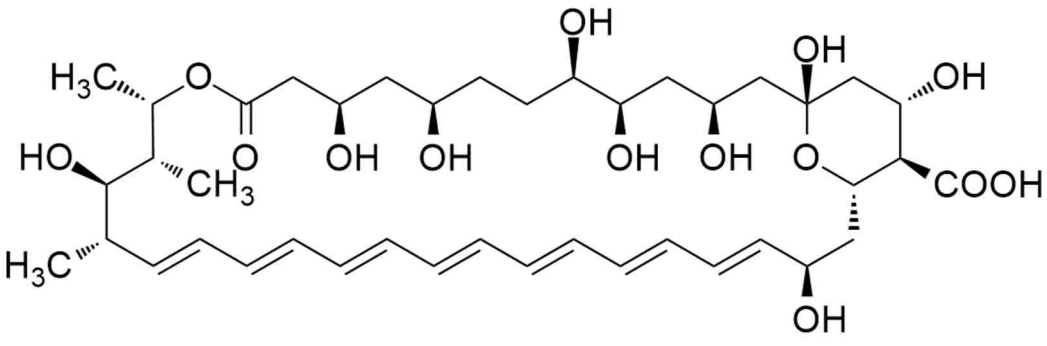


Figure 2-3: Structure of Amphotericin B.

2.2.2.1 RESISTANCE TO POLYENES

The main mechanism of resistance reported against this class of compounds is a defect in the gene involved in the ergosterol biosynthesis ERG3, which leads to accumulation in the membrane of other sterols. As consequence, Candida (and Cryptococcus) isolates with low ergosterol content in the membrane are polyene resistant. [32] Moreover, an increased catalase activity can also lead to a reduction to the oxidative damage, and support the resistance.

2.2.3 ALLYLAMINES

This type of molecules inhibits another key enzyme for the biosynthesis on ergosterol: the squalene epoxidases. [33] Such inhibition promotes the squalene accumulation in the cell membrane, leading to an increased cell permeability and a disruption in cell organization. [34] The resistance to allylamines is mainly due to upregulation of CDR1 and CDR2 genes.

2.2.4 ECHINOCANDINS

Echinocandin molecules inhibit in a non-competitive way the synthesis of a fungal cell wall integral component: 1,3-β-D-glucan. [35] They have a broad spectrum against Candida species, thus, considering also the favourable toxicity profile, they are extensively used in invasive candidiasis. Resistant strains were only recently found, but the mechanism is unclear. Is supposed that a mutation in the site of FKS1 subunit of glucan synthase (the molecular target of this class of molecules) could confer various degree of resistance.

2.3 Main resistance mechanisms in *Candida albicans*

2.3.1 BIOFILM

The biofilm is a complex system formed by a group of microorganisms embedded into an exopolysaccharide matrix, generically named Extra-Cellular Matrix (ECM). Usually, this structure is formed from those microorganisms who are associated to any kind of surface. It produces a particular environment in which the growth rate is reduced, cells can up- and down-regulate specific genes and they also can exchange genetic material. [21] Inside the matrix, cells find protection from external harmful factors, becoming one of the most important resistance mechanism. In fact, any kind of molecule finds difficulty in the diffusion inside the ECM, and thus is hampered in reaching the target fungal cell. Moreover, it acts as glue, holding cell together and associated to the surface.

Biofilms may form on a great variety of surfaces, the most common (and dangerous for humans) of which are: potable water piping, indwelling medical devices and also in living tissues. [21] In particular, fungal biofilms ranging from denture-related stomatitis in the oral cavity to infections located within the airways (both upper and lower) as well as the genitourinary and gastrointestinal tracts until to systemic infections. [36]

These systems are composed of microbial cells and ECM, which may account up to 90% of the total organic carbon of biofilm, [37] with the main component represented by the 1,3-β-glucan. The production of this molecule can be increased by glucoamylases, glucan transferases and exo-glucanase, [38] and decreased by the zinc-responsive transcription factor Zap1. [39] Moreover, also extracellular DNA is present in the matrix, and has been demonstrated that, in addition to being a structural component, acts also as promoter in the biofilm formation. [40]

In biofilm-associated fungal infections, those caused by *Candida albicans* are among the most dangerous, leading to a mortality equal to 63% in some systemic form of the disease. [41] Its formation in this specie is heterogeneous, and has a direct impact upon the pathogenicity and treatment. [42] Depending if the strain is able to form or not a biofilm, the

patient outcome is drastically different. From this point of view, strains can been divided into High Biofilm Former (HBF), with great pathogenicity and low antifungal sensitivity, and Low Biofilm Former (LBF), less pathogenic and with maintained antifungal sensitivity compared to the absence of biofilm. [40]

The presence of biofilm, confers a resistance increased by 1000 folds to azoles, and approximately eight folds to the polyene Amphotericin B. Of course, these values are strictly dependent on the degree of development and the age of the biofilm. [43]

2.3.2 EFFLUX PUMPS

As already mentioned in previous paragraphs, efflux pumps are the major resistance mechanism to antifungal therapy together with biofilm production. In *Candida albicans* the two most important pump types are represented by transmembrane proteins regulated by genes such CDR1 or CDR2, which belong to the ABC family, and MDR1, which is an element of the MFS (Major Facilitator Superfamily) transporter family. [44] They act with an extrusion mechanism, in which the antifungal agent is actively ejected outside the cells. During the biofilm growth, the expression of efflux pumps is increased. However, the disruption of one efflux pump have a minimal impact on the biofilm resistance, suggesting that their role is particularly significant during the early biofilm development phase. [45]

2.3.2.1 CDR1 AND CDR2

This type of ABC transporters is typically expressed as response to antifungal agent exposure. To extrude the antifungal molecule, typically azoles of which efflux pumps are the main resistance mechanism, they consume ATP. The upregulation of CDR1 and CDR2 is due to the presence of the orf19.3188, also called TAC1 (Transcriptional Activator of CDR genes). TAC1 deletion results in a loss of upregulation of transient CDR1 and CDR2. [24]

Cdr1p, encoded by CDR1, in *Candida albicans* is the major drug transporter (its overexpression in clinical isolates coincides with increased drug substrate efflux). [46] Cdr1p is a protein formed by two homologous domains. In addition to the transmembrane domains, a cytoplasmic hydrophilic nucleotide-binding domain (NBD) is also present. While this latter site is dedicated to the hydrolyses of ATP to power drug efflux and is very well conserved, the transmembrane domains are dedicated to drug recognition, and their primary sequences are variable. [47]

2.3.2.2 MDR1

Similar to what happens for CDR1 and CDR2, also this type of transporter is upregulated in presence of antifungal agents. However, differently from the previously described family, instead of using ATP it exploit the protonic gradient across the cytoplasmic membrane to supply energy for transport. [48] The role in azoles resistance of MDR1 was confirmed evaluating the susceptibility to Fluconazole after MDR1 inactivation in such MDR1 overexpressing *Candida albicans* isolates. Deletion of this gene in clinical isolates reversed the drug resistant phenotype. [49] Another difference with respect to CDRs is that in this case MDR-encoded efflux pumps have a narrower spectrum specific for Fluconazole. Regarding the regulatory factors which control the MDR1 expression, to date, they are not yet been identified. [48]

Recently, Sun and co-workers found that the overexpression of these kind of pumps potentiates the antifungal activity of a very particular series of compounds named BQM. They obtained an increased activity for BQM in *Candida albicans* overexpressing the multi-drug-resistant transporter Mdrp1 (regulated by MDR1 gene). [25]

2.4 Antifungal agents developed in our research group

During the last decade, our research group has worked on antifungal agents. In particular, the event with which the work started was the purification and identification of the active components present in the commercially available fungicide: guazatine. [50] This mixture, widely used in agriculture, was found to be composed by more than twenty among polyalkylamines and polyalkylguanidines. After purification, performed using preparative HPCL, each compound was isolated, characterized and tested against a panel of Candida species. Only three of these showed antifungal activity. [50]

2.4.1 DISCOVERY AND DEVELOPMENT OF MACROCYCLIC COMPOUNDS

During the synthesis of the active derivatives found in the guazatine mixture to introduce modifications, a new unknown side compound was discovered. The unknown molecule was characterized by NMR, and the subsequent X-ray analysis confirmed a macrocyclic structure (Figure 2-4). [51] The macrocyclization mechanism was than studied and described. [51]

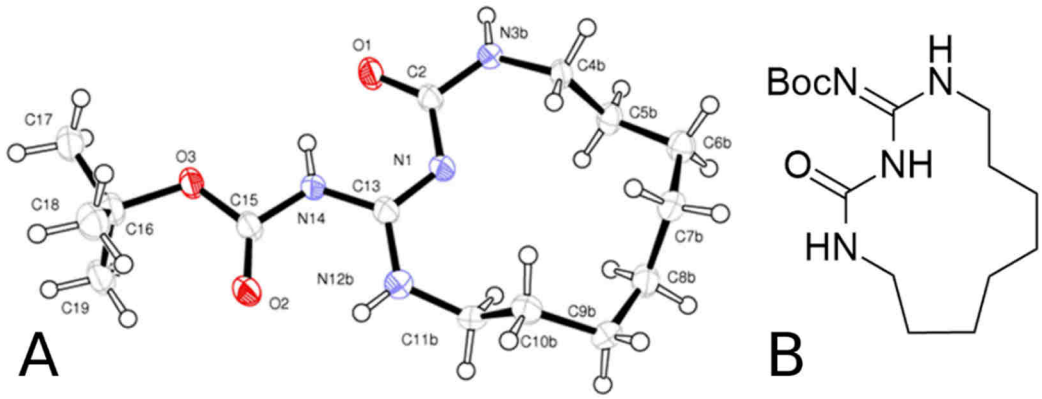


Figure 2-4: A) X-ray structure of the macrocyclic compound. B) 2D structure of the macrocyclic compound.

The first generation of these antifungal compounds, both linear and macrocyclic, was tested against a panel of Candida species. The macrocycles showed higher activity respect to the linear molecules. [52]

Among them, the best compound was the macrocycle BM01, characterized by a thirteen atom core incorporating an amidinoureic moiety and a crotyl moiety on the eight carbons alkylguanidine side chain (Figure 2-5).

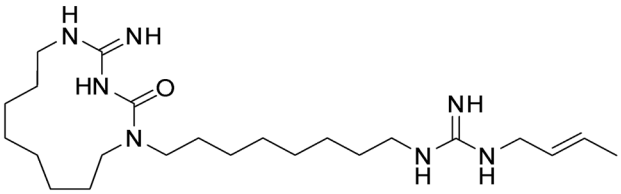


Figure 2-5: 2D Structure of BM01.

Considering the activity found for BM01, a small library of analogues was synthesized, variating the size of the macrocycle, the length of the side chain, the type of the moiety at the end of the side chain, inserting aromatic rings or ester function into the macrocycle(Figure 2-6). [53]

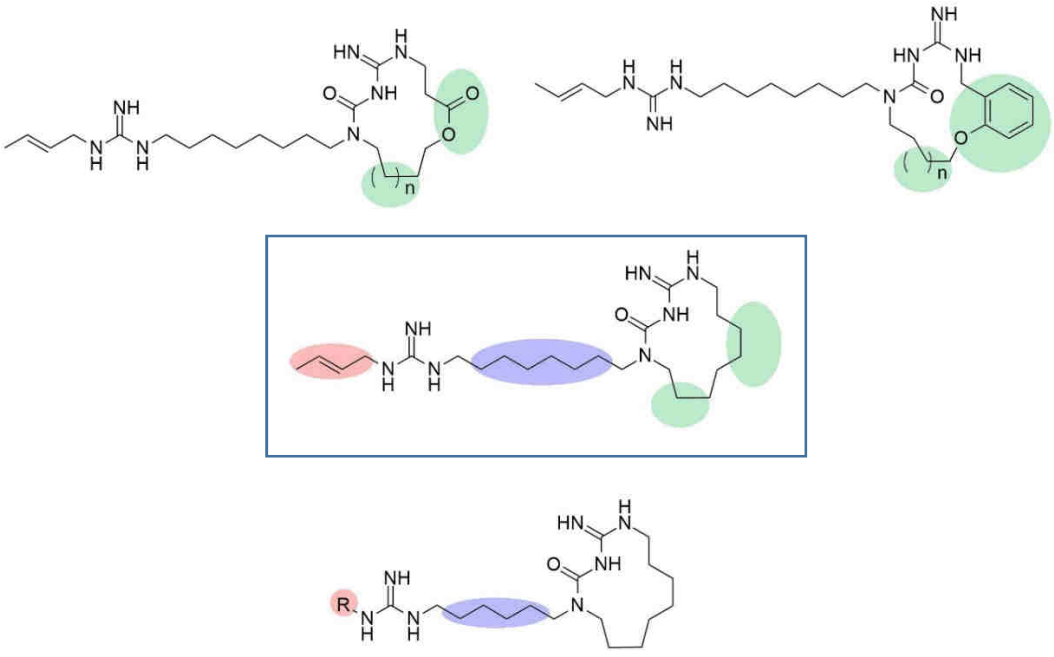
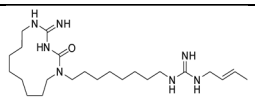
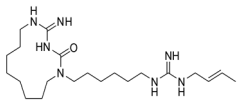
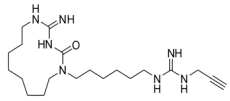
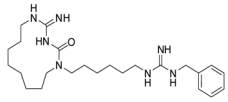
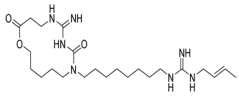
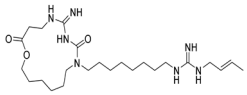
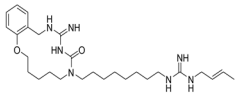
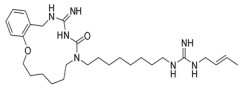


Figure 2-6: Structures of the second generation derivatives. [53] In the center the HIT compound BM01. Up left: Ester moiety introduction and different ring size. Up right: Introduction of a phenyl ring and different ring size. Down: Different length of the lateral chain and various moieties at the end of it.

Also this second generation was tested against different species of Candida. As a result, the introduction of the ester moiety led to a reduction

in the antifungal activity, while the aromatic moiety did not affect the biological profile. Another important aspect for the activity resulted to be the size of the ring, with the greatest activity for the 15 members ring respect to the 13 and 14 of the same class with a phenyl into the cycle (Table 2-1). [53]

Table 2-1: Antifungal activity of some of the macrocyclic compounds.

		MIC90 (μM)[a]					
		Candida species (N° strains tested)					
		Albicans	*Guilliermondii*	*Krusei*	*Parapsilosis*	*Tropicalis*	*Kefyr*
		(22)	*(10)*	*(13)*	*(24)*	*(11)*	*(10)*
BM 01		4	8	4	8	2	1
BM 07		4	2	16	2	2	8
BM 09		16	8	16	8	8	8
BM 12		4	4	8	4	4	8
BM 17		32	32	64	32	32	32
BM 18		16	8	32	8	16	16
BM 20		4	2	8	2	2	4
BM 21		4	2	4	2	1	2

[a] MIC90 values were determined at 24 h both visually and spectrophotometrically and are reported as μg/mL.

However, considering the comparable activity of the derivatives bearing the aromatic ring inside the macrocycle and of BM01, and the easier synthetic route of this latter one, we then decided to proceed with the biological characterization of our first compound: BM01.

2.4.2 EFFICACY AND RESISTANCE PROFILE OF OUR MACROCYCLIC HIT

Journal of Medicinal Chemistry

Article

pubs.acs.org/jmc

Biological Characterization and in Vivo Assessment of the Activity of a New Synthetic Macrocyclic Antifungal Compound

Davide Deodato,[†,×] Giorgio Maccari,[†,×] Filomena De Luca,[‡] Stefania Sanfilippo,[†] Alexandru Casian,[†] Riccardo Martini,[†] Silvia D'Arezzo,[§] Carlo Bonchi,[∥] Francesca Bugli,[⊥] Brunella Posteraro,[#] Patrick Vandeputte,[▽] Dominique Sanglard,[▽] Jean-Denis Docquier,[‡,×] Maurizio Sanguinetti,[⊥,#] Paolo Visca,[∥] and Maurizio Botta*[,†,⊗,×]

[†]Department of Biotechnology Chemistry and Pharmacy, University of Siena, I-53100 Siena, Italy

[‡]Department of Medical Biotechnology, University of Siena, I-53100 Siena, Italy

[§]Istituto Nazionale per le Malattie Infettive "Lazzaro Spallanzani", I-00149 Roma, Italy

[∥]Dipartimento di Scienze, Università Roma Tre, I-00154 Roma, Italy

[⊥]Institute of Microbiology, Università Cattolica del Sacro Cuore, I-00168 Roma, Italy

[#]Institute of Public Health, Università Cattolica del Sacro Cuore, I-00168 Roma, Italy

[▽]Institute of Microbiology, University of Lausanne and University Hospital Center, CH-1011 Lausanne, Switzerland

[⊗]Sbarro Institute for Cancer Research and Molecular Medicine, Temple University, BioLife Science Building, Suite 333, 1900 North 12th Street, Philadelphia, Pennsylvania 19122, United States,

[×]Lead Discovery Siena s.r.l., Via Vittorio Alfieri 31, I-53019 Castelnuovo Berardenga, Italy

Supporting Information

ABSTRACT: We recently identified a novel family of macrocyclic amidinoureas showing potent antifungal activity against *Candida* spp. In this study, we demonstrate the fungicidal effect of these compounds as well as their killing activity in a dose-dependent manner. Transcriptional analysis data indicate that our molecules induce a significant change in the transcriptome involving ATP binding cassette (ABC) transporter genes. Notably, experiments against *Candida albicans* mutants lacking those genes showed resistance to the compound, suggesting the involvement of ABC transporters in the uptake or intracellular accumulation of the molecule. To probe the mode of action, we performed fluorescence microscopy experiments on fungal cells treated with an ad-hoc synthesized fluorescent derivative. Fluorescence microscopy images confirm the ability of the compound to cross the membrane and show a consistent accumulation within the cytoplasm. Finally, we provide data supporting the in vivo efficacy in a systemic infection murine model setup with a drug-resistant strain of *C. albicans*.

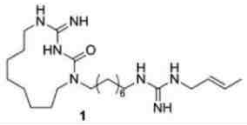
■ INTRODUCTION

In the public opinion, mycoses are often associated with the development of external infections involving the skin or the nails. Less known, but more dangerous diseases, are those caused by systemic fungal infections. In those cases, fungi are systemically spread through the bloodstream and invade internal organs.[1] The large use of antifungal agents, most of them launched in the market more than 20 years ago, has led to the development of drug-resistant or even multi-drug-resistant fungi.[2−6] Multi-drug-resistant strains are responsible for life-threatening acquired infections, which are especially relevant in immunocompromised patients, such as AIDS patients, organ and bone marrow transplant recipients under immunosuppressive therapy, or cancer patients treated with antiproliferative drugs.[7] Among fungal pathogens, *Candida* species represent one of the most common causes of nosocomial bloodstream infections. The mortality rate associated with candidemia is significantly high, ranging between 25% and 50%. Furthermore, therapies for systemic candidosis commonly

Received: January 5, 2016
Published: April 5, 2016

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

A



1

Strain	MIC (μM) 24h	48h	72h
C. albicans ATCC 60193	5	5	5
C. albicans 4T	5	5	5
C. albicans 53T	5	5	5
C. albicans 15T	5	5	5
C. krusei ATCC 14243	10	80	80
C. krusei 193T	10	80	80
C parapsilosis ATCC 34136	10	80	80
C. parapsilosis 64E	20	40	40
C. parapsilosis 81E	20	40	40
C. glabrata 70	>80	>80	>80
C. tropicalis 86E	2.5	5	5
C. neoformans	NG	NG	20
G. silvicola	NG	10	10
G. capitatum	NG	10	10

NG: No growth

B

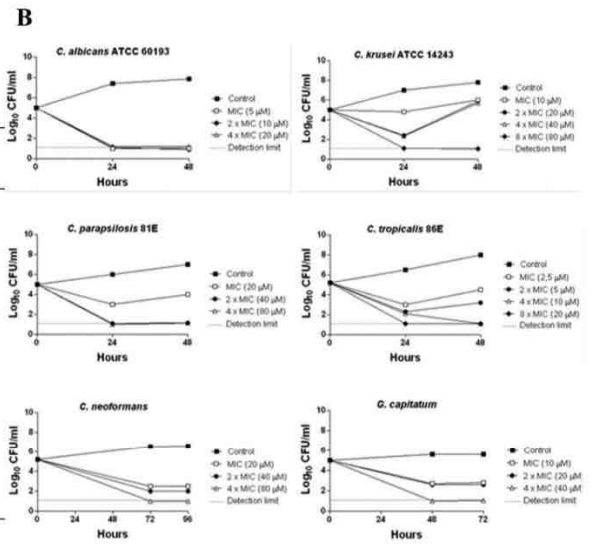


Figure 1. In vitro assays on compound **1**. (A) Table of antifungal activity of **1** against 14 yeast strains. (B) Time-kill activity of **1** on representative fungal species. Concentrations of **1** are expressed as multiples of the minimum inhibitory concentration (MIC). Fungal viability was determined by plate counts after different times of exposure to the drug.

treated with fluconazole and amphotericin B might be unsuccessful.[8] This situation is further aggravated by the emergence of drug-resistant *Candida* strains of both *albicans* and non-*albicans* species, whose treatment represents an increasingly worrisome challenge to clinicians.[9,10] We recently reported data on the in vitro efficacy of a novel class of macrocyclic amidinoureas,[11] which proved highly active against various *Candida* species, including drug-resistant strains, and showed a low cytotoxicity in vitro.[12−15] Further efforts to optimize the antifungal activity of the compounds allowed us to develop a new generation of compounds showing extremely promising biological activities and pharmacological properties.[16,17] One representative of such optimized molecules is compound **1** (Figure 1), whose synthesis and chemical characterization have been described elsewhere.[13,18,19] In this work, we provide an in-depth biological characterization of this compound, its impact on the fungal cell physiology, and a clear assessment of its in vivo efficacy against drug-resistant *Candida* clinical strains, using a murine model of systemic infection.

■ RESULTS AND DISCUSSION

In Vitro Antifungal Activity. In vitro experiments were conducted to investigate the biological activity of compound **1**. A total of 14 clinical isolates belonging to clinically relevant yeast species were tested, including strains of *Candida albicans*, *Candida parapsilosis*, *Candida krusei*, *Candida glabrata*, *Candida tropicalis*, *Cryptococcus neoformans*, *Geotrichum silvicola*, and *Geotrichum capitatum*. Strains were previously identified by standard morphological, cultural, and biochemical tests.[20] The minimum inhibitory concentration (MIC) of **1** ranged between 20 and 2.5 μM (Figure 1A). At 24 h, **1** was very active in inhibiting the growth of all strains of *C. albicans*, *C. krusei*, *C. parapsilosis*, and *C. tropicalis* (MIC = 2.5−20 μM). It also showed activity against isolates of *Cry. neoformans* (MIC = 20 μM) and *Geotrichum* spp. (MIC = 10 μM), though MICs for these species could only be determined at 72 and 48 h, respectively, when growth in the control row could be seen. The lowest activity was observed toward *C. glabrata* (MIC > 80 μM). We also investigated the effect of incubation time on antifungal activity. Compound **1** showed a species-dependent increase of MICs during time, most strikingly for *C. krusei* and *C. parapsilosis*. At 48 h, the MIC increased 8-fold for *C. krusei* (ATCC 14243 and 193^T) and *C. parapsilosis* ATCC 34136, and 2-fold for *C. parapsilosis* (64E and 81E) and *C. tropicalis* 86E. No significant effect of the incubation time was observed for *C. albicans*, *Cry. neoformans*, and *Geotrichum* spp., with unchanged MICs for up to 96 h. It is possible that extended incubation times allow resistant organisms to overgrow the initially susceptible subpopulation, or that the whole population requires an adaptation phase after exposure to the drug, resulting in higher MICs after prolonged incubation. On the other hand, during prolonged incubation, degradation of the antifungal compound may also occur, possibly by enzymes expressed only by certain species or strains. Indeed, the stability of **1** in RMPI-1640 medium was assessed in order to exclude spontaneous degradation of the compound during the antifungal susceptibility assay. These experiments confirmed the stability of **1** in RMPI-1640 for up to 72 h of incubation at 30 °C (Figure S1, Supporting Information).

To evaluate the fungicidal activity of **1**, yeast viability was assessed after 24 h of incubation at 37 °C by colony-forming unit (CFU) counts on Sabouraud dextrose agar. It was considered to have fungicidal activity if a decrease greater than or equal to 3·Log10 CFU/mL (99.9%) of the initial inoculum was observed at 24 h (Experimental Section). Figure 1B shows that **1** exerted the strongest fungicidal activity on *C. albicans*, resulting in 99.9% killing within 24 h at 5 μM (corresponding to the MIC value) of the drug. Compound **1** was fungistatic for *C. parapsilosis* at the MIC value and fungicidal at 2 times the MIC. The killing activity was also

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

observed on *Cry. neoformans* and *G. capitatum* at 2-fold and 4-fold the MICs determined at 72 and 48 h, respectively. *C. krusei* and *C. tropicalis* exhibited 99.9% CFU reduction for 8-fold the MICs of **1** determined at 24 h. The overall fungicidal effect observed has relevant clinical implications, since in vivo killing of fungi would greatly facilitate the immune-mediated eradication of the infection.

Transcriptional Analysis of *C. albicans* Exposed to Compound 1. To better understand the antifungal activity of **1** on *C. albicans*, whole genome transcriptional profile experiments were performed in the presence and absence of sub-inhibitory concentrations (3 μM) of this compound at two different time points (15 and 45 min). These conditions were chosen to identify the primary effects of the drug without inducing extensive cell damage. Transcriptional profiles were obtained with high-density microarrays and single-labeled cRNA. Data were analyzed with biological triplicates. Figure 2

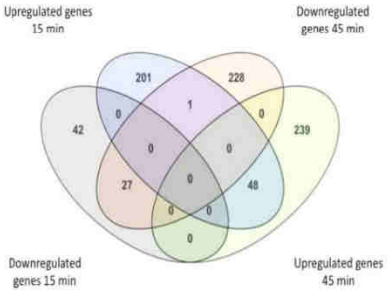


Figure 2. Venn diagram showing the results of the comparative transcriptome analysis and the impact of **1** on gene expression. Genes were at least 1.5-fold commonly up- and down-regulated between 15 and 45 min exposure to **1**. The diagram was obtained using Genespring Software Version 12 (Agilent).

shows Venn diagrams of up- and down-regulated genes (by at least 1.5-fold) at both time points. The diagram reveals that 48 and 27 genes were commonly up- and down-regulated between the two time points.

Details on these commonly expressed genes are given in Tables 1 (up-regulated genes) and S2 (down-regulated genes, Supporting Information). Strikingly, up-regulated genes exhibited a typical signature for *TAC1*-regulated genes (Table 1). *TAC1* is a transcription factor regulating several genes involved in drug resistance and particularly azole resistance. As summarized in Table 1, *TAC1* is itself up-regulated as well as its target genes such as *CDR1*, *CDR2*, *RTA3*, *LCB4*, and *IFU5*. *CDR1* and *CDR2* up-regulation was verified by separate real-time quantitative polymerase chain reaction (RT-qPCR) analysis. Results are given in Figure 3, and show that genes identified to be up-regulated by microarrays are regulated in a similar way by RT-qPCR analysis. *CDR1* and *CDR2* are ATP binding cassette (ABC) transporters and contribute to drug efflux and azole resistance.[21]

Susceptibility of *C. albicans CDR1/CDR2* Mutants to Compound 1. We showed above that *CDR1* and *CDR2* were up-regulated by **1**. These genes have been reported to be up-regulated by a large number of drugs, including antifungal agents.[22] In the majority of cases, inducers themselves serve as substrates for transporters. In order to test this hypothesis, we used *C. albicans* mutants lacking both *CDR1* and *CDR2* (DSY448, DSY654)[23] and addressed their susceptibility to **1** (Figure 4). Hypersusceptibility to **1** of the mutants as compared to a wild-type control (CAF2-1) would strongly suggest that **1** is a substrate for these ABC transporters. Surprisingly, as shown in Figure 4, the mutants became unexpectedly more resistant to **1** as compared to the wild type. The validity of the assay was confirmed, since the *C. albicans CDR1/CDR2* mutants were susceptible to fluphenazine, which is consistent with previously published results. The resistance of the *CDR1/CDR2* mutants to **1** indicates a unique feature of this antifungal substance, which probably requires the presence of ABC transporters in order to be active. A similar behavior has been recently observed by Sun and co-workers, in the case of the antifungal compound BMQ,[24] which proved to be more active in *C. albicans* isolates overexpressing multi-drug-resistant transporter Mdrp1. Those authors demonstrated that the transporter facilitates the uptake and increases the intracellular accumulation of the compound, rather than acting as an efflux pump. It is thus rational to assume that, in our case, the intracellular accumulation of **1** can be favored by the expression of ABC transporters. The reversed biological profile observed for this compound, compared to current antifungal drugs, enhances the potential of our class of macrocyclic amidinoureas for development as new antifungal agents to treat drug-resistant infections.

Fluorescence Microscopy Experiments. To test the aforementioned hypothesis of an intracellular accumulation of the compound, we performed fluorescence microscopy assays. The aim of our study is to investigate the interaction of macrocyclic amidinoureas with the fungal cell and to verify if the compounds are accumulated in the cytoplasm and/or they interact with other structures (membrane, nucleus). In order to perform fluorescence experiments, three different fluorescent probes were designed and synthesized. Two of them, namely compounds **2** and **3** (Figure 5), derived from **1**, with the fluorophore (dansyl group) directly attached on the side chain. In particular, **2** is linked to the fluorophore directly on the terminal amino group, while **3** has the dansyl group attached to the guanidine moiety. We used the dansyl group as first choice for two reasons: it is easily introducted by nucleophilic substitution of commercial dansyl chloride, and the small dimensions give good chances to retain the biological activity. Unfortunately, the introduction of the fluorophore on the guanidinium moiety in **2** and **3** resulted in a loss of antifungal activity in vitro (MIC > 125 μM), probably because the terminal guanidine does not allow for a bulky substituent. Fluorescence microscopy experiments were conducted on samples of *C. albicans* and *Cry. neoformans* cultures, but, although both compounds showed a clear cell-association, the resolution used did not allow for the discrimination between cell wall association and internalization (Figure S2, Supporting Information). In order to obtain better results, a new fluorescent probe was thus synthesized (compound **4**, Figure 5), bearing the fluorophore group far away from the guanidinium moiety. In this case an aromatic derivative (from Sanguinetti et al.[16]) was chosen as the parent compound, since it possess antifungal activity comparable to that of **1**, but the presence of a benzene ring offers a useful attachment point for fluorophores (fluorescein). Despite its chemical structure being bulkier than the dansyl group, we chose fluorescein as fluorophore because it has better performance in confocal microscopy and the distance from the active guanidine as well

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

Table 1. Selection of Genes Commonly Up-Regulated by Compound 1 in *C. albicans SC5314* at 15 and 45 min Time Points[a]

orf19 name	gene name	15 min	45 min	description/annotation
orf19.5958	CDR2	71.84	134.38	multi-drug transporter, ATP-binding cassette (ABC) superfamily
orf19.4531		17.22	21.54	putative PDR subfamily ABC transporter
orf19.1438		6.13	17.28	protein with homology to NADH dehydrogenase
orf19.2726		5.86	6.56	putative plasma membrane protein; Plc1p-regulated
orf19.3378		5.32	3.52	predicted ORF in assemblies 19, 20, and 21
orf19.1027	PDR16	4.43	6.89	phosphatidylinositol transfer protein; increased transcription correlates with CDR1 and CDR2 overexpression and azole resistance
orf19.4907		3.90	5.30	putative protein of unknown function; Hap43p-repressed gene
orf19.24	RTA2	3.07	2.93	putative flippase required for sphingolipid long-chain base
orf19.6000	CDR1	3.07	3.75	multi-drug transporter of ABC superfamily
orf19.6817	FCR1	2.86	5.88	zinc cluster transcription factor; negative regulator of fluconazole resistance
orf19.7336		2.80	3.49	predicted membrane transporter; member of the drug:proton antiporter (14 spanner) (DHA2) family and major facilitator superfamily (MFS)
orf19.3188	TAC1	2.79	3.38	transcriptional activator of drug-responsive genes, including CDR1 and CDR2
orf19.6026	ERG2	2.61	2.83	C-8 sterol isomerase; enzyme of ergosterol biosynthesis pathway
orf19.3089		2.39	2.63	orthologue(s) have role in cardiolipin metabolic process and cristae formation
orf19.2356	CRZ2	2.28	2.66	zinc finger transcription factor; crz1, not crz2, null mutation suppresses fluconazole resistance of homozygous cka2 null (CK2 kinase defective)
orf19.1887		2.19	3.44	orthologue(s) have sterol esterase activity and role in sterol metabolic process, and are integral to membrane lipid particle localization
orf19.6873.1		2.16	1.90	orthologue(s) have electron carrier activity and iron and zinc ion-binding activity
orf19.2248	ARE2	2.11	2.88	acyl CoA:sterol acyltransferase (ASAT)
orf19.5257	LCB4	2.04	3.69	putative sphingosine kinase; expression is Tac1p-regulated
orf19.2568	IFU5	1.63	2.58	predicted membrane protein; estradiol-induced; increased transcription associated with CDR1 and CDR2 overexpression or fluphenazine treatment

[a]Complete data are given in Table S1 (Supporting Information).

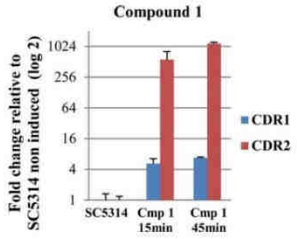


Figure 3. Validation of microarray results with RT-qPCR on *C. albicans* SC5314.

as the C3 spacer should avoid a pronounced loss of activity. As a result, **4** showed a retention of antifungal activity, even though with a reduced potency (MIC = 80 μM), and it was used in confocal microscopy experiments.

Confocal microscopy examination of *C. albicans* ATCC 60193 and *Cry. neoformans* exposed for 48 h to **4** showed strong cell-associated fluorescence which was absent in untreated cells (Figure 6). Fluorescent images were visually compared, and no significant difference was observed between the two fungal species treated with **4**. Fluorescence was mostly associated with the cytoplasm, confirming the intracellular accumulation of the compound, in accordance with the hypothesis of an ABC transporters-mediated accumulation. Moreover, fluorescence showed a patchy distribution in some cells, suggesting that the compound can interact with multiple and/or differently located targets within the fungal cell. No fluorescent signal was observed in cells presenting a damaged cell wall, suggesting release of the fluorescent compound due to cell lysis, in accordance with the previously demonstrated fungicidal effect of this class of compounds.

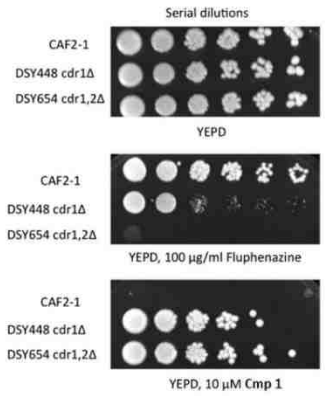


Figure 4. Serial dilution assay of *C. albicans* with drug inhibitory substances. *C. albicans* isolates were 10-fold serially diluted starting from a density of 2×10^7 cells/mL. An aliquot (5 μL) was spotted onto YEPD agar containing the different drugs as indicated.

Chemistry. The synthesis of compounds **2** and **3** is depicted in Scheme 1, starting from primary amine **5**, which was prepared according to a procedure already described in a previous paper.[19] For the synthesis of **2**, amine **5** was reacted with dansyl chloride (**6**) in CH_3CN, furnishing the sulfonamide derivative **7**, which was deprotected with freshly distilled 10% trifluoroacetic acid (TFA) in anhydrous dichloromethane (DCM) solution, to give the desired fluorescent probe **2**. For the preparation of fluorescent probe **3**, it was necessary to synthesize the opportune guanylating agent **9**, which was obtained by reacting dansyl chloride (**6**) with *N*-Boc-1*H*-

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

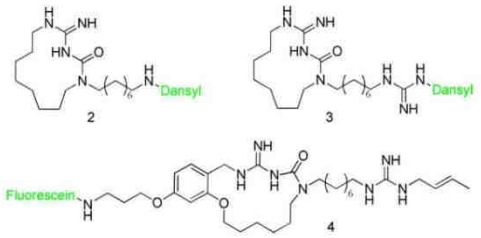


Figure 5. Structures of the fluorescent probes synthesized. Fluorophores are represented in green.

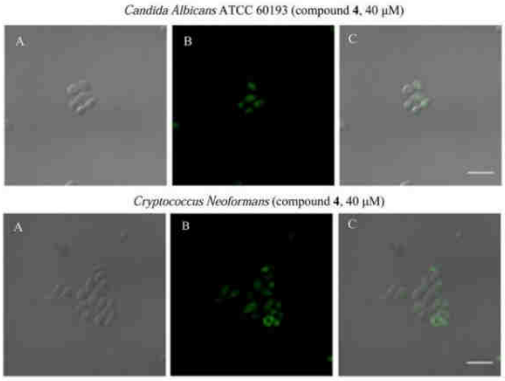


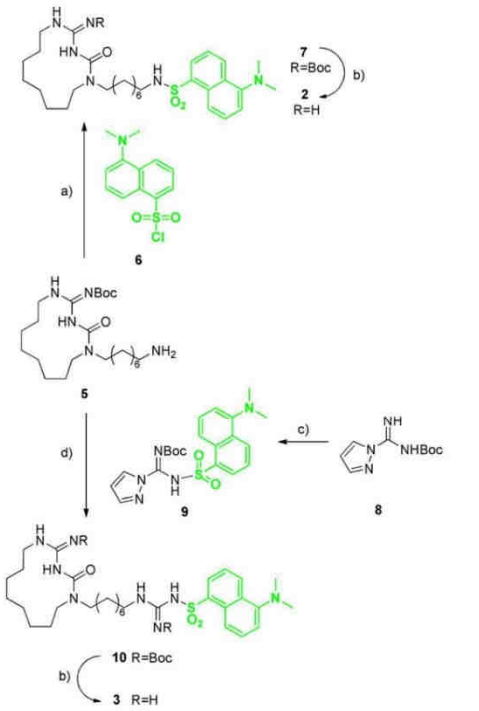
Figure 6. Confocal microscope images of *C. albicans* ATCC 60193 and *Cry. neoformans* cells treated with compound **4** at 40 μM concentration. Panels A, B, and C show the visible, fluorescent, and merged images, respectively. Scale bar = 10 μm.

pyrazolecarboxamidine (**8**) in the presence of NaH as a base. The final guanylation reaction of amine **5** with **9** furnished, after Boc-cleavage, the desired fluorescent probe **3** (Scheme 1).

The synthesis of **4** is described herein. Starting from 2,4-dihydroxybenzaldehyde (**11**), we first functionalized the 4-OH position with *N*-(3-bromopropyl)phthalimide to afford intermediate **12**, which was then etherified with 5-bromo-1-pentene to give **13**. The aromatic aldehyde was converted to aldoxime **14** and then reduced to benzyl amine **15**. During this reaction one carbonyl group of the phthalimide moiety was reduced, giving a 3-hydroxyisoindolinone moiety. Intermediate **16** was obtained by reacting **15** with 1,3-di-Boc-2-(trifluoromethylsulfonyl)guanidine. The coupling reaction between the secondary amine **17**[16] and **16**, followed by ring-closing metathesis reaction, furnished the amidinoureic macrocycle **19** in good yields (Scheme 2).

While trying to remove the carboxybenzyl (Cbz) protecting group by hydrogenation over Pd/C, we observed the simultaneous reduction of the 3-hydroxyisoindolinone ring to isoindolin-1-ol; despite several attemps, we were not able to obtain selective Cbz removal. For this reason we decided to change the protecting group on the C3 lateral chain of **19**. To do so, we first oxidized the alcohol group of the 3-hydroxyisoindolinone to a carbonyl group with MnO_2, and then cleaved the phthalimide moiety with hydrazine monohydrate to afford the corresponding primary amine **20**, which was then protected using methyl trifluoroacetate, furnishing **21**.

Scheme 1[a]



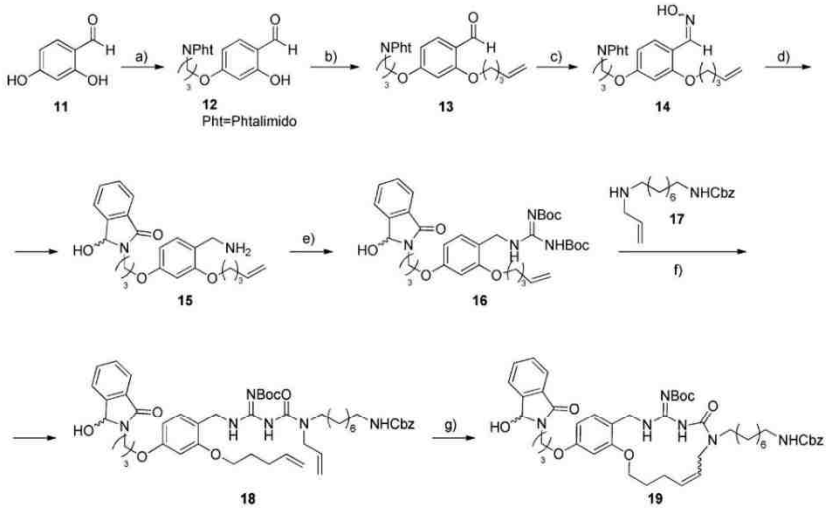
[a]Reagents and conditions: (a) TEA, CH_3CN, rt, 10 min; (b) TFA, DCM, rt, 3 h; (c) **6**, NaH, THF, rt, 3 h; (d) DIPEA, CH_3CN/MeOH, 50 °C, 24 h.

Hydrogenation over Pd/C allowed simultaneous reduction of the endocyclic double bond and removal of the Cbz group, affording derivative **22**. The following guanylation with *N*-crotylpyrazoleguanidine (**23**)[16] furnished **24**. The C3 spacer of **24** was deprotected to give amine **25**, which was reacted with fluorescein isothiocyanate (FITC) in dimethylformamide (DMF) to form the desired conjugate **26**. Finally, butyloxycarbonyl (Boc) deprotection performed in 20% TFA in dry DCM afforded the final product **4** in quantitative yield as a yellow powder (Scheme 3).

In Vivo Animal Studies. In order to test the potential antifungal therapeutic effect of **1** in vivo, experimental treatments of invasive candidiasis were carried out in immunocompetent mice. Two different drug regimens, i.e., 20 and 40 mg/kg/day, were applied, whereas fluconazole was used as a control at a dosage of 8 mg/kg/day. Figure 7A shows results of tissue burdens obtained after 7 days of experiments in spleen and kidneys with both 20 and 40 mg/kg/day regimens of **1**. Experiments showed that **1** significantly reduced the CFU counts of the organs (kidneys and spleen) of mice infected intravenously with *C. albicans* ATCC 90028 (a fluconazole-susceptible control strain). A similar effect was noticed in mice treated with fluconazole, as expected. Furthermore, experimental mouse infections were also done with two isogenic strains of *C. albicans*: DSY294 that is susceptible to azole drugs (fluconazole MIC = 0.25 μg/mL) and DSY296 that is azole-

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

Scheme 2[a]



[a]Reagents and conditions: (a) *N*-(3-bromopropyl)phthalimide, K_2CO_3, CH_3CN, reflux, 6 h; (b) 5-bromo-1-pentene, K_2CO_3, CH_3CN, reflux, 6 h; (c) NH_2OH–HCl, pyridine, EtOH, reflux, 3.5 h; (d) Zn, HCl, THF, reflux, 2 h; (e) 1,3-di-Boc-2-(trifluoromethylsulfonyl)guanidine, TEA, DCM, rt, 18 h; (f) TEA, THF, reflux, 12 h; (g) Grubb's second generation catalyst, degassed DCM, reflux, 3 h.

resistant (fluconazole MIC = 64 μg/mL).[25] In these experiments, only the 40 mg/kg/day regimen of **1** was applied, whereas fluconazole was used at the same dosage (8 mg/kg/day) (Figure 7B). By contrast to fluconazole treatment, which was effective only against DSY294, **1** significantly reduced CFU counts of the selected organs of mice infected with both DSY294 and DSY296, and it is worth noting that the in vivo efficacy of **1** was thus achieved also with the azole-resistant isolate strain. With regard to DSY296, the reductions in CFU counts by **1** treatment were markedly higher than those observed in mice treated with fluconazole. Remarkably, the resistance to fluconazole of the isolate DSY296 is due to the overexpression of multi-drug transporters, triggered by an up-regulation of *CDR1* and *CDR2* genes.[25] The in vivo efficacy of **1** against DSY296 infection thus corroborates the assumption of the ABC transporters-mediated accumulation of the molecule inside the fungal cell.

Conclusions. To date, only three classes of antifungal agents are used for the treatment of systemic candidiasis, and the market calls for new drugs that would address a growing medical need, determined by the emergence of drug-resistant clinical isolates. In this work, we present a set of in vitro and in vivo data supporting the promising potential of the herein described novel antifungal agent, a macrocyclic amidinourea derivative. Our research highlights that the molecular target is new and not shared by any other antifungal compound on the market, since our compound preserves a potent activity also against drug-resistant strains. Indeed, compound **1** was proved particularly active on resistant isolates overexpressing multi-drug transporters of the ABC superfamily, a mutation usually associated with resistance to therapeutic antifungals. The involvement of ABC transporters in the mode of action of the compound was demonstrated with in vitro experiments on mutant strains and confirmed with fluorescence microscopy and in vivo assays. Our findings could represent the first step toward the development of a new therapeutic class of antifungal agents, potentially useful to treat acquired infections, and might represent a suitable strategy to address the growing emergence of antifungal-resistant isolates in the clinical setting.

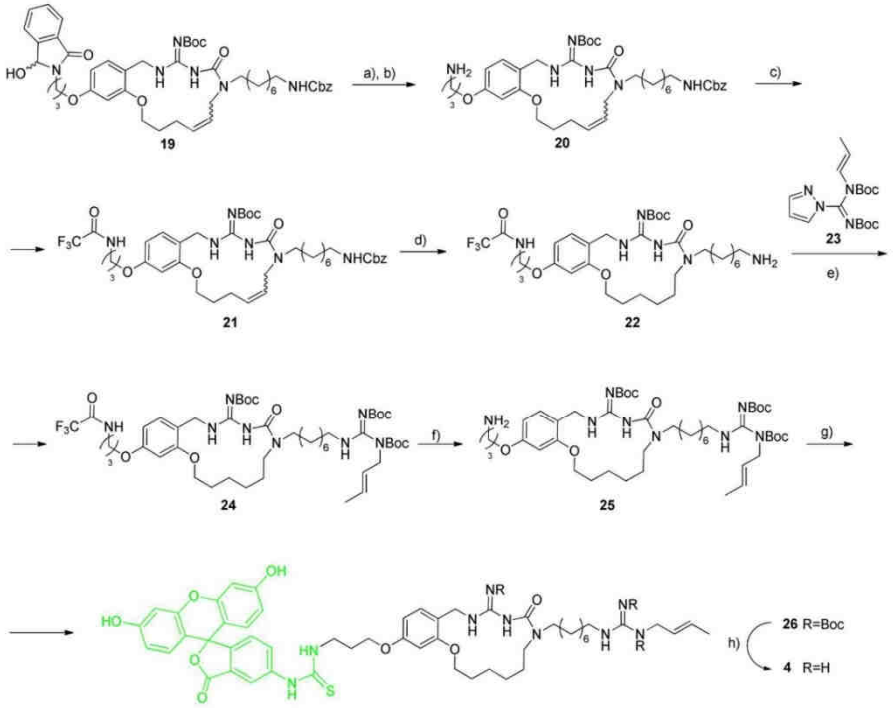
EXPERIMENTAL SECTION

Antifungal Activity in Vitro. *C. albicans* ATCC 60193, *C. krusei* ATCC 14243, and *C. parapsilosis* ATCC 34136 were purchased from the American Type Culture Collection (Manassas, VA, USA). The EUCAST susceptibility testing protocol, including the update for testing of non-fermenting yeasts (*Cryptococcus* spp. and *Geotrichum* spp.), was used for the determination of the minimal inhibitory concentrations (MICs) of antifungal agents.[26,27] Results were read after 24, 48, and 72 h of incubation at 37 °C. The MIC was defined as the lowest compound concentration which prevented visible growth. Aliquots of 10 μL were removed from the wells corresponding to the MIC, 2MIC, 4MIC, and 8MIC and serially 10-fold diluted in sterile saline, to minimize the drug carry-over. An aliquot of 10 μL of each dilution was plated on Sabouraud dextrose agar plates. The plates were incubated at 37 °C for 24/48 h, and the number of CFUs was counted. For *Cryptococcus* spp. and *Geotrichum* spp., incubation was for 48 h to facilitate colony counting.[28] When less than 100 CFU/mL was expected, 10 μL of sample was plated without dilution. The experiment was performed in duplicate. The minimum detection limit of this method was 10 CFU/mL. The Log10 CFU/mL was plotted on a graph as a function of time and used to compare the rate and extent of antifungal activity in the presence various concentrations of **1** and in its absence. Activity was considered fungicidal when there was a decrease greater than or equal to 3-Log10 CFU/mL (99.9%) of the initial inoculum in 24 h. Activity lower than 3-Log10 reduction in the number of CFU/mL of the initial inoculum was considered fungistatic.[29] *C. glabrata* was not tested since it is poorly susceptible to **1**.

RPMI Medium Stability Assay. To verify the chemical stability of **1** in RPMI-1640 medium, 0.4 mL of a stock solution in MeOH (10 mM) of the compound and 1.6 mL of RPMI-1640 (final concentration 2 mM) were mixed in a test tube. The solution was filtered through a 0.45 μm nylon filter (Acrodisc) before LC-UV-MS analysis. The tube

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

Scheme 3[a]



[a]Reagents and conditions: (a) MnO_2, DCM, rt, 18 h; (b) hydrazine monohydrate, MeOH sol., rt, 12 h; (c) methyl trifluoroacetate, TEA, THF, 0 °C to rt, 16 h; (d) H_2, Pd/C, cat. HCl, i-PrOH, rt, 3 h; (e) *N*-crotylpyrazoleguanidine (**23**), DIPEA, CH_3CN/MeOH, 60 °C, 16 h; (f) K_2CO_3, MeOH, H_2O, 70 °C, 2 h; (g) FITC, DIPEA, DMF, rt, 24 h; (h) TFA/DCM 20%, rt, 8 h.

was maintained at 30 °C, and four aliquots (20 μL each) were removed and analyzed by HPLC at predetermined time points (1, 24, 48, and 72 h). Each aliquot was analyzed in triplicate. The stability was followed by HPLC with an UV-MS detection method (Supporting Information). Quantitative analysis was performed using an appropriate calibration curve obtained by solubilizing **1** in MeOH.

Transcriptional Analysis. Sample preparation was performed on three biological triplicates. Total RNA was extracted from log phase cultures in liquid yeast extract peptone dextrose (YEPD) as previously described.[30] Compound **1** was added to an end concentration of 3 μM, and incubation points of 15 and 45 min were chosen. Control cells were taken at the same time points but without **1**. Briefly, after centrifugation of 5 mL of culture (corresponding to 10^8 cells), the yeast cell pellets were mixed with 0.3 g of glass beads, 300 μL of RNA extraction buffer (0.1 M Tris-HCl at pH 7.5, 0.1 M LiCl, 10 mM EDTA, 0.5% SDS), and 300 μL of phenol−chloroform−isoamyl alcohol (24:24:1). After 1 min of vortexing in a bead beater (Fastprep-24 Instrument, MP Biomedicals Switzerland, Zürich), the aqueous phase was re-extracted with phenol−chloroform−isoamyl alcohol, and RNA was precipitated with 600 μL of ethanol at −20 °C for 1 h. The RNA pellet was resuspended in 50 μL of diethyl pyrocarbonate-treated H_2O. The integrity of the input template RNA was determined prior to labeling/amplification, using an Agilent RNA 6000 Nano LabChip kit and 2100 bioanalyzer (Agilent Technologies). Agilent's One-Color Quick Amp Labeling Kit (Agilent Technologies) was used to generate fluorescent cRNA according to the manufacturer's instructions. Briefly, 1 μg of total RNA from each sample was used, to which a spike mix and T7 promoter primers were added, both of which are provided by the manufacturer. cDNA synthesis was promoted by MMLV-RT (Moloney Murine Leukemia Virus Reverse Transcriptase) in the presence of dNTPs and RNaseOUT. Next, cRNA was produced from this first reaction with T7 RNA polymerase, which simultaneously amplifies target material and incorporates cyanine 3-labeled CTP. The labeled cRNAs were purified with RNeasy Mini Kit (Qiagen) and quantified using a NanoDrop ND-1000 UV−vis spectrophotometer. Next, 600 ng of Cy3-labeled cRNAs was fragmented and hybridized for 17 h at 65 °C to each array using the Gene Expression Hybridization Kit (Agilent Technologies) and a gasket slide with an 8 microarrays/slide format for sample hybridization to separate each sample in specific sub-arrays of the 8 × 15K format. The *C. albicans* microarray format was published by Synnott et al. (design ID 017942).[31]

Slides were washed and processed according to the Agilent 60-mer Oligo Microarray Processing protocol and scanned on a Agilent microarray scanner G2565BA (Agilent Technologies). Data were extracted from the images with Feature Extraction (FE) software (Agilent Technologies). FE software flags outlier features, and detects and removes spatial gradients and local backgrounds. Data were normalized using a combined rank consistency filtering with LOWESS intensity normalization. The gene expression values obtained from FE software were imported into GeneSpring 12 software (Agilent Technologies) for preprocessing and data analysis. For inter-array comparisons, a linear scaling of the data was performed using the 75th percentile signal value of all of non-control probes on the microarray to normalize one-color signal values. Probe sets with a signal intensity value below the 20th percentile were considered as absent and discarded from subsequent analysis. The expression of each gene was normalized by its median expression across all samples. Genes were

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

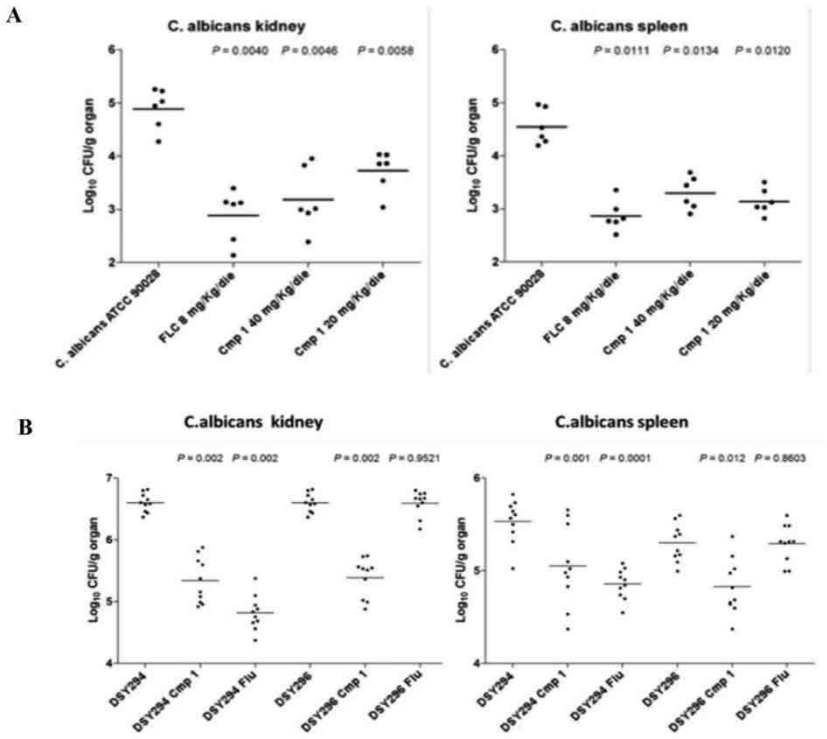


Figure 7. In vivo animal studies. The regimens for **1** and fluconazole were 40 and 8 mg/kg/day, respectively. Each data point corresponds to one animal, and the origin of tissue (kidney or spleen) is shown for each diagram. (A) Fungal tissue burdens of mice infected with *C. albicans* and treated with fluconazole (FLC) and **1**. (B) Fungal tissue burdens of mice infected with *C. albicans* DSY294 and DSY296, treated with fluconazole (Flu) and **1**.

included in the final data set if their expression changed by at least 1.5-fold as compared to **1**-unexposed cells. Corrected *P*-value <0.05 was chosen as the cutoff for significance. Microarray data have been uploaded to the NCBI GEO microarray repository.

Real-Time Quantitative Polymerase Chain Reaction (RT-qPCR). Total RNA was extracted from log phase cultures with an RNeasy Protect Mini kit (Qiagen) by a process involving mechanical disruption of the cells with glass beads and an RNase-free DNase treatment step as previously described.[32] To remove contaminating DNA, 10 μg of each RNA preparation was treated with DNase using a DNA-Free DNA removal kit (Ambion). For the reverse transcription reactions, a Transcriptor High Fidelity cDNA Synthesis Kit (Roche) was used on 1 μg of DNA-free RNA of each sample, and the manufacturer's instructions were followed. RNA samples were stored at −80 °C, and cDNA samples were stored at −20 °C. Quantitative expression of *CDR1* and *CDR2* was performed with the StepOne Real-time PCR System (Life Technologies). RT-qPCR was carried out in a 10 μL volume containing the following reagents: 5 μL of iTaq Supermix with Rox (BioRad), each primer pair, the Taqman probe at a final concentration of 200 nM for the primers (CDR2-ORF-F: TAGATATTTGAGCCACATG and CDR2-ORF-R: TTGGCA-TTGAAATTTTCG; CDR1-ORF-F: ATGACTCGAGATATTTTG-ATA; and CDR1-ORF-R: TTAACAGCAATGGTCTTTA) and 100 nM for the probes (CDR2-P2: TTAGTCCATTCAACGGCA-ACATTAG; CDR1-P2: CATTATGAGACCTGGTGAACTTACT), and 1% of the total cDNA sample produced as described above. Each reaction was run in triplicate, and data were analyzed with the StepOne software. For relative quantification of the target genes, each set of primer pairs and the Taqman probe were used in combination with the *ACT1* primers (ACT1-RT-F: ATAACGGTTCTGGTATGT; ACT1-RT-R: CCTTGATGTCTTGGTCTA) and an *ACT1* probe (ACT1-RT-P: CGGTGACGACGCTCCAAG).

For data analysis and fold change calculations, the comparative CT method was used. For each experiment, a standard curve for the reference gene and the studied gene was included, and the amplification efficiency was determined for all genes. The CT values of the target genes were normalized to the endogenous reference. Fold changes were obtained from the mean normalized expression of the samples relative to the mean normalized expression of a selected control.

Confocal Fluorescence Microscopy Assays. Aliquots of 100 μL of *C. albicans* and *Cry. neoformans* treated for 48 h with a 40 μM DMSO solution of **4** were washed twice in phosphate-buffered saline (PBS) and fixed in 4% formaldehyde for 30 min at 4 °C. After a second PBS wash, the cells were suspended in 100 μL of PBS, and 3 μL of the suspension was spotted onto a glass slide, dried, and mounted with 6 μL of Vectashield mounting medium. Images were acquired with a Leica TCS SP5 inverted confocal microscope equipped with a 63×/1.40 OIL objective (Zeiss), set for fluorescein (FITC) (excitation 488 nm, emission window from 500 to 540 nm).

General Chemistry Directions. Reagents were obtained from commercial suppliers and used without further purification. DCM and CH_3CN were dried over sodium hydride. THF was dried over Na/benzophenone prior to use. Anhydrous reactions were run under a positive pressure of dry N_2 or argon. Degassed DCM was prepared by using the freeze−pump−thaw method. Silica gel 60 was used for flash

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

chromatography (23−400 mesh). ^{1}H NMR and ^{13}C NMR are reported in parts per million (δ scale) and internally referenced to the $CDCl_3$ or CD_3OD signal, respectively at δ 7.24 and 3.31 ppm. Chemical shifts for carbon are reported in parts per million (δ scale) and referenced to the carbon resonances of the solvent ($CDCl_3$ at δ 77.00 and CD_3OD at δ 49.00 ppm). Data are presented as follows: chemical shift, multiplicity (s = singlet, d = doublet, t = triplet, m = multiplet and/or multiplet resonances, br = broad), coupling constant in hertz (Hz), and integration. Mass spectrometry (MS) data were acquired on an Agilent 1100 LC/MSD VL system (G1946C) with a 0.4 mL/min flow rate using a binary solvent system of 95:5 methanol/water. UV detection was monitored at 254 nm. Mass spectra were acquired in positive mode, scanning over the mass range. Purity of final products was assessed by HPLC/MS analysis, conducted using a Polaris C18 column (150−4.6 mm, 5 μm particle size) at a flow rate of 0.8 mL min^{-1}, with a mobile phase composed of 50% CH_3CN/50% H_2O−formic acid 0.1%. The purity of all final compounds was above 95%.

5-(Dimethylamino)-N-(8-(2-oxo-4-(tert-butoxycarbonylimino)-1,3,5-triazacyclotridecan-1-yl)octyl)naphthalene-1-sulfonamide (7). A solution of macrocyclic amine **5** (20 mg, 1 equiv) and TEA (0.01 mL, 1.5 equiv) in CH_3CN at room temperature under argon atmosphere was treated with a solution of dansyl chloride **6** (17 mg, 1.5 equiv) in CH_3CN. The mixture was stirred at room temperature for 10 min. Solvent was then evaporated, and the crude residue was purified with flash chromatography on silica gel, eluting with 10% AcOEt/Hex to give **7** (isolated yield 79%). ^{1}H NMR (400 MHz $CDCl_3$) 12.00 (1H, s); 8.50−8.48 (d, J = 8.4 Hz, 1H), 8.24−8.22 (d, J = 8.2 Hz, 1H), 8.19−8.17 (d, J = 8.1 Hz, 1H), 8.02 (t, J = 4.0 Hz, 1H), 7.52−7.44 (m, 2H), 7.14−7.12 (d, J = 8.2 Hz, 1H), 4.50 (s, 1H), 3.36 (m, 2H), 3.30 (m, 2H), 3.13−3.11(m, 2H), 2.81(s, 6H), 2.78 (m, 2H), 1.56 (m, 8H), 1.40 (s, 9H), 1.25−1.19 (m, 16H) ppm. ^{13}C NMR (100 MHz CD_3OD) 163.9; 154.0; 153.5;134.7; 130.2, 129.6; 128.2; 123.2; 118.8; 115.1; 81.9; 46.9; 46.2; 45.4; 43.2; 39.5; 31.8; 29.6; 29.3; 29.1; 28.8; 28.3, 28.0; 27.7; 26.1; 25.8; 25.2; 23.6; 23.4 ppm. LRMS m/z (ESI) = 695.0 [M + Na]$^+$, 673.0 [M + H]$^+$.

tert-Butyl ((5-(Dimethylamino)naphthalene-1-sulfonamido)(1H-pyrazol-1-yl)methylene)carbamate (9). To a suspension of NaH (27 mg, 3 equiv) in 2 mL of THF was added dropwise a solution of *N*-Boc-1*H*-pyrazole-1-carboxamidine **8** (78 mg, 1 equiv) in 0.5 mL of THF at 0 °C. The mixture was stirred at room temperature for 30 min, and then a solution of dansyl chloride **6** (300 mg, 3 equiv) in 1 mL of THF was added. After further stirring for 2 h at room temperature, water (5 mL) and DCM (10 mL) were added, and the layers were separated. The aqueous layer was extracted with DCM (2 × 10 mL), and the combined organic phases were washed with $NaHCO_3$, water, and brine, dried over Na_2SO_4, filtered, and concentrated under reduced pressure. The crude residue was purified with flash chromatography on silica gel, eluting with 20% AcOEt/Hex to give **9** (isolated yield 90%). ^{1}H NMR (400 MHz $CDCl_3$): δ 8.87 (s, 1H), 8.50−8.48 (d, J = 8.4 Hz, 1H), 8.37−8.35 (d, J = 8.4 Hz, 1H), 8.27−8.25 (d, J = 8.4 Hz, 1H), 7.85 (m, 1H), 7.62 (m, 1H), 7.52−7.44 (m, 2H), 7.13−7.11 (d, J = 8.3 Hz, 1H), 6.27 (m, 1H), 2.84 (s, 6H), 1.40 (s, 9H) ppm. LRMS m/z (ESI) = 466.2 [M + Na]$^+$, 443.2 [M + H]$^+$.

5-(Dimethylamino)-N-(N′-tert-butoxycarbonyl-N-(8-(4-(tert-butoxycarbonylimino)-2-oxo-1,3,5-triazacyclotridecan-1-yl)octyl)-carbamimidoyl)naphthalene-1-sulfonamide (10). To a stirred solution of guanylating agent **9** (21 mg, 1.1 equiv) in 5% MeOH/CH_3CN (1 mL) was added a solution of the macrocyclic amine **5** (20 mg, 1 equiv) in 5% MeOH/CH_3CN (1 mL) dropwise, followed by the addition of catalytic diisopropylethylamine (DIPEA) (2 drops). The mixture was stirred at 50 °C overnight. After cooling, solvent was evaporated under reduced pressure, and the crude residue was purified with flash chromatography on silica gel, eluting with 10% AcOEt/Hex to give **10** (isolated yield 93%). ^{1}H NMR (400 MHz $CDCl_3$): δ 12.02 (s, 1H), 9.79 (s, 1H), 8.50−8.48 (d, J = 8.4 Hz, 1H), 8.37−8.35 (d, J = 8.4 Hz, 1H), 8.27−8.25 (d, J = 8.4 Hz, 1H), 7.49−7.41 (m, 2H), 7.13−7.11 (d, J = 8.2 Hz, 1H), 3.38 (m, 2H), 3.29 (m, 2H), 3.18 (m, 2H), 2.82 (s, 6H), 1.48(m, 8H), 1.40 (s, 9H), 1.37 (s, 9H), 1.25−1.19 (m, 16H) ppm. ^{13}C NMR (100 MHz CD_3OD) 163.9; 154.0; 153.5; 152.5; 150.6; 138.9; 129.7; 129.1; 127.2; 126.2; 123.3; 115.1; 84.1; 81.9; 46.9; 46.2; 45.5; 41.3; 39.5; 31.8; 30.7; 29.5; 29.3; 29.0; 28.7; 28.3, 28.0; 27.7; 26.1; 25.8; 25.2; 23.6; 23.4 ppm. LRMS m/z (ESI) = 837.5 [M + Na]$^+$, 814.5 [M + H]$^+$.

General Procedure for Boc Deprotection. The appropriate protected compound **7** or **10** (0.025 mmol) was dissolved in a 10% solution of freshly distilled TFA in dry DCM (1 mL). The resulting solution was stirred at room temperature under argon for 24 h. The reaction mixture was then concentrated under reduced pressure, affording the desired compound as the trifluoroacetate salt in quantitative yield.

5-(Dimethylamino)-N-(8-(4-imino-2-oxo-1,3,5-triazacyclotridecan-1-yl)octyl)naphthalene-1-sulfonamide (2). ^{1}H NMR (400 MHz $CDCl_3$): δ 9.79 (s, 1H), 8.50−8.48 (d, J = 8.4 Hz, 1H), 8.37−8.35 (d, J = 8.2 Hz, 1H), 8.27−8.25 (d, J = 8.4 Hz, 1H), 7.49−7.41 (m, 2H), 7.13−7.11 (d, J = 8.2 Hz, 1H), 3.56 (m, 2H), 3.46 (m, 2H), 3.23 (m, 2H), 3.05 (m, 2H), 2.80 (s, 6H), 1.48(m, 8H), 1.25−1.19 (m, 16H) ppm. ^{13}C NMR (100 MHz $CDCl_3$): δ 163.6, 153.7, 153.2, 135.1, 130.2, 129.7, 129.2, 124.0, 121.4, 81.6, 46.7, 46.0, 41.9, 39.3, 33.5, 29.2, 28.2, 27.8, 26.8, 25.8, 25.5, 24.9, 23.3 ppm. LRMS m/z (ESI) = 595.0 [M + Na]$^+$, 573.0 [M + H]$^+$.

5-(Dimethylamino)-N-(N-(8-(4-imino-2-oxo-1,3,5-triazacyclotridecan-1-yl)octyl)carbamimidoyl)naphthalene-1-sulfonamide (3). ^{1}H NMR (400 MHz CD_3OD): δ 12.00 (s, 1H); 8.44−8.42 (d, J = 8.4 Hz, 1H), 8.22−8.20 (d, J = 8.4 Hz, 1H), 8.19−8.17 (d, J = 8.4 Hz, 1H), 8.03 (m, 1H), 7.49−7.43 (m, 2H), 7.13−7.11 (d, J = 8.4 Hz, 1H), 3.38 (m, 2H), 3.31 (m, 2H), 3.12 (m, 2H), 2.80 (s, 6H), 2.77 (m, 2H), 1.56 (m, 8H), 1.25−1.19 (m, 16H) ppm. ^{13}C NMR (100 MHz $CDCl_3$): δ 206.8, 163.9, 154.05, 153.0, 152.5, 150.6, 138.9, 129.8, 129.1, 127.2, 126.2, 123.3, 116.2, 84.1, 82.0, 46.9, 46.2, 45.7, 41.3, 39.6, 31.8, 30.8, 29.5, 29.2, 29.0, 28.4, 27.8, 25.8, 25.2, 23.5, 22.6, 14.0 ppm. LRMS m/z (ESI) = 637.5 [M + Na]$^+$, 614.5 [M + H]$^+$.

4-(3-(Phthalimido)propoxy)-2-hydroxybenzaldehyde (12). A solution of 2,4-dihydroxybenzaldehyde **11** (0.500 g, 1 equiv), *N*-(3-bromopropyl) phthalimide (0.970 g, 1 equiv), and K_2CO_3 (0.500 g, 1 equiv) was refluxed in CH_3CN (20 mL) for 6 h. The reaction mixture was concentrated and extracted with DCM and water. The organic phase was dried over Na_2SO_4, filtered, and concentrated. After the purification of the crude by flash chromatography (SiO_2) using DCM as eluent, product **12** was obtained a white solid (isolated yield 34%). ^{1}H NMR (400 MHz $CDCl_3$): δ 11.42 (s, 1H, 2-OH); 9.69 (s, 1H); 7.84−7.81 (m, 2H); 7.74−7.71 (m, 2H); 7.37 (d, J = 8.8 Hz, 1H); 6.41 (d, J = 8.5 Hz, 1H); 6.32 (d, J = 1.2 Hz, 1H); 4.07 (t, J = 6.8 Hz, 2H); 3.90 (t, J = 6.8 Hz, 2H); 2.22 (m, 2H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): 194.3; 168.2; 165.8; 164.3; 135.1; 133.9; 132.0; 123.2; 115.2; 108.5; 101.0; 66.1; 35.1; 27.9 ppm. LRMS m/z (ESI) = 324 [M + H]$^+$.

4-(3-(Phthalimido)propoxy)-2-(pent-4-en-1-yloxy)benzaldehyde (13). A solution of **12** (0.200 g, 1 equiv), K_2CO_3 (0.255 g, 3 equiv), and 5-bromo-1-pentene (0.160 mL, 2.2 equiv) was refluxed in CH_3CN (10 mL) for 6 h. The reaction mixture was concentrated under vacuum and extracted with DCM. The organic phase was dried over Na_2SO_4, filtered, and concentrated. The reaction crude was purified by flash chromatography (SiO_2) using DCM as eluent to obtain **13** as a white solid (isolated yield 78%). ^{1}H NMR (400 MHz $CDCl_3$): δ 10.27 (s, 1H); 7.81−7.79 (m, 2H); 7.72−7.68 (m, 4H); 6.38 (d, J = 8.4 Hz, 1H); 6.27 (s, 1H); 5.87−5.76 (m, 1H); 5.04 (d, J = 17.2 Hz, 1H); 4.99 (d, J = 10 Hz, 1H); 4.06 (t, J = 6 Hz, 2H); 3.95 (t, J = 6.4 Hz, 2H); 3.88 (t, J = 6.8 Hz, 2H); 2.25−2.14 (m, 4H); 1.89 (m, 2H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): 188.1; 183.7; 177.5; 168.2; 165.1; 163.0; 137.3; 133.9; 132.0; 130.1; 123.2; 119.1; 115.4; 106.0; 98.9; 67.5; 66.0; 35.2; 29.9; 28.05; 28.00 ppm. LRMS m/z (ESI) = 394.1 [M + H]$^+$; 416.1 [M + Na]$^+$.

4-(3-(Phthalimido)propoxy)-2-(pent-4-en-1-yloxy)benzaldehyde oxime (14). A stirred solution of aldehyde **13** (0.120 g, 1 equiv) in EtOH (10 mL), hydroxylamine hydrochloride (0.053 g, 2.5 equiv), and pyridine (0.028 g, 1.2 equiv) was heated to 80 °C for 3.5 h. The reaction mixture was concentrated in vacuo and used with no further purification. ^{1}H NMR (400 MHz $CDCl_3$): δ 8.86 (br, 1H); 8.40 (s, 1H); 7.83−7.81 (m, 2H); 7.71 (m, 2H); 7.57 (d, J = 8.4 Hz, 1H); 6.36

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

(d, J = 8.6 Hz, 1H); 5.88−5.77 (m, 1H); 5.05 (d, J = 17.6 Hz, 1H); 4.99 (d, J = 10.4 Hz, 1H); 4.03 (t, J = 5.6 Hz, 2H); 3.91−3.87 (m, 4H); 2.24−2.14 (m, 4H); 1.86 (quint, J = 6.8 Hz, 2H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): 165.1; 163.2; 146.2; 137.3; 133.8; 132.1; 127.3; 123.2; 119.1; 115.3; 105.8; 98.9; 67.5; 66.0; 44.0; 35.4; 29.9; 28.1 ppm. LRMS m/z (ESI) = 409.0 $[M + H]^+$;431.0 $[M + Na]^+$.

*2-(3-(4-(Aminomethyl)-3-(pent-4-en-1-yloxy)phenoxy)propyl)-3-hydroxyisoindolin-1-one (**15**).* To a stirred solution of aldoxime **14** (0.124 g, 1 equiv) in THF (10 mL) were added Zn^0 granules (0.198 g, 10 equiv) and aqueous 2 M HCl solution (1.51 mL, 10 equiv), and the resulting mixture was heated to 80 °C for 2 h. After the solution was cooled to room temperature, it was filtered to remove the residual Zn and extracted in DCM and 1 M NaOH solution. The organic layer was washed with brine, dried over Na_2SO_4, filtered, and concentrated to yield **15** as a colorless oil that was used with no further purification (isolated yield 66%). ^{1}H NMR (400 MHz $CDCl_3$): δ 7.65 (d, J = 7.2 Hz, 1H); 7.53−7.48 (m, 2H); 7.42−7.39 (m, 2H); 6.88 (d, J = 8.4 Hz, 1H); 6.29 (s, 1H); 5.85−5.73 (m, 1H); 5.74 (s, 1H); 5.03 (d, J = 17.6 Hz, 1H); 4.98 (d, J = 11.2 Hz, 1H); 4.00−3.80 (m, 6H); 3.72−3.67 (m, 1H); 3.62−3.55 (m, 1H); 3.48 (s, 1H); 2.17−2.10 (m, 4H); 1.81 (m, 2H) ppm. LRMS m/z (ESI) = 419.1 $[M + Na]^+$.

*1-(4-(3-(1-Hydroxy-3-oxoisoindolin-2-yl)propoxy)-2-(pent-4-en-1-yloxy)benzyl)-2,3-bis(tert-butoxycarbonyl)guanidine (**16**).* To a stirred solution of **15** (0.277 g, 1 equiv) in DCM (15 mL) Et_3N (0.486 mL, 5 equiv) was added 1,3-di-Boc-2-(trifluoromethylsulfonyl)-guanidine (0.300 g, 1.1 equiv), and the resulting mixture was stirred for 18 h at room temperature. Water was added, and after 30 min of stirring, the reaction mixture was extracted with DCM. The organic layer was washed with brine, dried over Na_2SO_4, filtered, and concentrated in vacuo. The crude was purified by flash chromatography (SiO_2) (petroleum ether:EtOAc:MeOH = 7:2:0.5) to give the desired product **16** as an oil (isolated yield 46%). ^{1}H NMR (400 MHz $CDCl_3$): δ 11.44 (s, 1H); 8.64 (m, 1H); 7.52−7.44 (m, 3H); 7.34 (t, J = 7.2 Hz, 1H); 7.08 (d, J = 8.0 Hz, 1H); 6.28 (s, 1H); 6.26 (s, 1H); 5.82−5.76 (m, 1H); 5.73 (s,1H); 5.01 (d, J = 17.6 Hz, 1H); 4.94 (d, J = 10.0 Hz, 1H); 4.67 (br, 1H); 4.47 (d, J = 4.8 Hz, 2H); 3.93−3.87 (m, 2H); 3.84 (t, J = 6.4 Hz, 2H); 3.57−3.44 (m, 2H); 2.18 (q, J = 7.2 Hz, 2H); 2.03 (m, 2H); 1.85 (m, 2H); 1.47 (s, 9H); 1.42 (s, 9H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): 167.6; 163.5; 159.6; 158.0; 155.7; 152.8; 144.0; 137.7; 132.0; 131.4; 130.4; 129.4; 123.2; 122.9; 117.9; 115.1; 104.3; 99.6; 82.6; 81.9; 79.0; 67.1; 65.8; 40.6; 36.7; 30.0; 28.2; 28.1; 27.9 ppm. LRMS m/z (ESI) = 639.2 $[M + H]^+$; 661.2 $[M + Na]^+$.

*N-(N′-(4-(3-(1-Hydroxy-3-oxoisoindolin-2-yl)propoxy)-2-(pent-4-en-1-yloxy)benzyl)-tert-butoxycarbonylcarbamimidoyl)-N′-prop-2-enyl-N′-(8-(benzyloxycarbonylamino)octyl)urea (**18**).* To a stirring solution of **16** (0.276 g, 1 equiv) in dry THF (20 mL) was added amine **17** (0.207 g, 1.5 equiv)[19] as a dry THF solution (5 mL). To the resulting mixture was added Et_3N (0.060 mL, 1 equiv). After 12 h of heating at 80 °C, the reaction mixture was cooled to room temperature and concentrated in vacuo. The reaction crude was purified by flash chromatography (SiO_2) (petroleum ether:EtOAc MeOH = 7:3:1) to yield the desired compound **18** as an oil (isolated yield 60%). ^{1}H NMR (400 MHz $CDCl_3$): δ 12.21 (d, J = 4 Hz, 1H); 8.42 (m, 1H); 7.58 (t, J = 6.4 Hz, 1H); 7.53 (d, J = 7.6 Hz, 1H); 7.48 (t, J = 7.2 Hz, 1H); 7.37 (t, J = 7.6 Hz, 1H); 7.28 (m, 5H); 7.05 (d, J = 8.8 Hz, 1H); 6.31 (d, J = 7.6 Hz, 2H); 5.83−5.78 (m, 1H); 5.75 (s, 1H); 5.12−4.96 (m, 6H); 4.42 (dd, J1 = 5.6 Hz, J2 = 16 Hz, 2H); 4.12 (d, J = 5.2 Hz, 1H); 3.95−3.68 (m, 6H); 3.67−3.54 (m, 2H), 3.42 (t, J = 7.6 Hz, 1H); 3.21 (t, J = 7.2 Hz, 1H); 3.10−3.06 (m, 2H); 2.22 (q, J = 6.8 Hz, 2H); 2.09 (m, 2H); 1.89 m, 2H); 1.48 (m, 2H); 1.41 (s, 9H); 1.24 (m, 8H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): 167.5; 163.8; 163.7; 159.4; 157.9; 157.8; 156.4; 153.8; 153.6; 153.1; 144.1; 137.1; 136.5; 135.3; 134.7; 131.9; 131.6; 130.1; 129.6; 129.4; 128.4; 127.9; 123.1; 122.9; 118.9; 115.5; 115.1; 104.3; 99.6; 82.0; 81.9; 81.8; 67.1; 66.4; 65.8; 50.44; 48.4; 47.5; 45.6; 41.0; 36.9; 30.0; 29.8; 29.6; 29.4; 29.3; 29.1; 28.5; 28.2 ppm. LRMS m/z (ESI) = 883.3 $[M + H]^+$.

*tert-Butyl (10-(8-(Benzyloxycarbonylamino)octyl)-11-oxo-18-(3-(1-Hydroxy-3-oxoisoindolin-2-yl)propoxy)-2,3,4,5,8,9,10,11,12,15-decahydrobenzo[b][1,5,7,9]oxatriazacycloheptadecin-13-yl)-carbamate (**19**).* In a 250 mL round-bottomed flask equipped with a stirring bar and a condenser, **18** (0.231 g, 1 equiv) was dissolved in 175 mL of dry DCM, and the resulting mixture was degassed three times. After the third degassing cycle, Grubbs's second-generation (0.024 g, 0.1 equiv) catalyst was added, and the mixture was degassed another time. The reaction mixture was refluxed for 3 h. The solvent was removed, and the reaction crude was purified by flash chromatography (SiO_2) (petroleum ether:EtOAc: = 7:3) to yield the desired compound as a mixture of two isomers (E/Z) (isolated yield 75%). ^{1}H NMR (400 MHz $CDCl_3$): δ 12.21 (s, 1H), 12.03 (s, 1H), 8.21 (s, 1H), 7.61 (d, J = 7.3 Hz, 3H), 7.57−7.47 (m, 6H), 7.39 (t, J = 7.1 Hz, 3H), 7.31 (s, 7H), 7.06 (d, J = 8.1 Hz, 3H), 6.33 (t, J = 7.6 Hz, 3H), 6.27 (s, 2H), 6.05−5.95 (m, 1H), 5.76 (s, 3H), 5.63−5.48 (m, 2H), 5.41−5.31 (m, 2H), 5.03 (s, 4H), 4.85 (s, 2H), 4.56 (s, 4H), 4.22 (d, J = 5.8 Hz, 2H), 3.95 (s, 9H), 3.87 (d, J = 3.3 Hz, 6H), 3.64 (m, 6H), 3.32−3.22 (m, 4H), 3.12 (s, 6H), 2.40 (d, J = 6.7 Hz, 2H), 2.13 (m, 10H), 1.52 (s, 8H), 1.41 (m, 14H), 1.38 (m, 12H), 1.24 (m, 28H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): δ 167.48, 164.05, 163.39, 159.81, 159.26, 158.29, 157.38, 156.36, 153.41, 153.21, 152.42, 144.01, 134.30, 132.13, 131.96, 131.57, 131.13, 129.51, 128.38, 127.93, 125.98, 125.03, 123.15, 123.03, 119.52, 118.06, 104.80, 104.29, 99.80, 99.43, 82.00, 81.82, 69.70, 66.42, 65.85, 60.29, 50.36, 47.11, 44.58, 40.98, 39.61, 36.85, 32.25, 29.77, 29.56, 29.31, 29.11, 28.66, 27.99, 27.68, 26.98, 26.90, 26.55, 14.06 ppm. LRMS m/z (ESI) = 855.3 $[M + H]^+$.

*tert-Butyl (10-(8-(Benzyloxycarbonylamino)octyl)-11-oxo-18-(3-(amino)propoxy)-2,3,4,5,8,9,10,11,12,15-decahydrobenzo[b]-[1,5,7,9]oxatriazacycloheptadecin-13-yl)carbamate (**20**).* To a stirred solution of **19** (0.160 g, 1 equiv) in 30 mL of DCM was added MnO_2 (0.244 g, 15 equiv) in one portion, and the resulting mixture was stirred for 18 h at room temperature. The reaction mixture was filtered through a pad of Celite and concentrated. The crude was purified by flash chromatography (SiO_2) (petroleum ether:EtOAc:MeOH = 7:2:1) to yield the desired phthalimido derivative (*tert*-butyl (10-(8-(benzyloxycarbonylamino)octyl)-11-oxo-18-(3-(1,3-dioxoisoindolin-2-yl)propoxy)-2,3,4,5,8,9,10,11,12,15-decahydrobenzo[b][1,5,7,9]oxatriazacycloheptadecin-13-yl)-carbamate) as a mixture of two isomers (E/Z) (isolated yield 85%). ^{1}H NMR (400 MHz $CDCl_3$): δ 12.28 (s, 1H), 12.11 (s, 1H), 8.21 (t, J = 5.3 Hz, 1H), 7.83−7.77 (m, 4H), 7.75 (t, J = 4.4 Hz, 1H), 7.67 (m, 4H), 7.31 (d, J = 3.8 Hz, 5H), 7.07 (d, J = 8.2 Hz, 2H), 6.32 (t, J = 7.8 Hz, 3H), 6.26 (s, 2H), 6.01 (m,1H), 5.64−5.48 (m, 2H), 5.06 (s, 4H), 4.86 (s, 2H), 4.57 (s, 4H), 4.22 (d, J = 6.6 Hz, 2H), 3.97 (m, 8H), 3.88 (dd, J = 8.1, 5.1 Hz, 10H), 3.35−3.22 (m, 4H), 3.15 (d, J = 5.7 Hz, 5H), 2.41 (dd, J = 14.3, 7.1 Hz, 2H), 2.15 (dd, J = 5.6, 3.5 Hz, 8H), 1.54 (s, 6H), 1.47−1.44 (m, 4H), 1.41 (s, 9H), 1.39 (s, 9H), 1.27 (m, 18H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): δ 168.22, 164.07, 163.40, 159.83, 159.29, 158.26, 157.35, 156.31, 153.43, 153.23, 152.47, 136.65, 134.26, 133.79, 132.08, 131.11, 129.46, 128.37, 127.95, 127.90, 126.03, 125.08, 123.12, 119.54, 118.08, 104.67, 104.18, 99.85, 99.47, 81.88, 81.73, 69.68, 66.58, 66.38, 65.68, 50.33, 47.09, 44.58, 42.49, 40.99, 39.61, 38.23, 35.40, 32.27, 29.81, 29.58, 29.32, 29.29, 29.12, 28.68, 28.14, 28.00, 27.70, 27.00, 26.91, 26.56, 24.30 ppm. LRMS m/z (ESI) = 853.3 $[M + H]^+$; 875.3 $[M + Na]^+$.

The protected intermediate (0.137 g, 1 equiv) was dissolved in 50 mL of 0.2 M hydrazine monohydrate methanol solution. The resulting mixture was stirred overnight at room temperature. The solvent was removed and the crude purified by flash chromatography (SiO_2) (petroleum ether:EtOAc:MeOH:Et_3N = 7:2:1:0.25) to yield the primary amine **20** as a mixture of two isomers (E/Z) as an oil (isolated yield 80%). ^{1}H NMR (400 MHz, $CDCl_3$): δ 12.36 (s, 1H), 12.28 (s, 1H), 8.21 (s, 1H), 7.98 (t, J = 5.1 Hz, 2H), 7.75 (s, 1H), 7.35−7.31 (m, 6H), 7.08 (dd, J = 8.2, 3.7 Hz, 2H), 6.39 (s, 4H), 6.36 (s, 1H), 6.01 (dd, J = 14.8, 7.4 Hz, 1H), 5.64−5.48 (m, 1H), 5.42−5.33 (m, 1H), 5.07 (s, 4H), 4.83 (s, 4H), 4.77 (s, 4H), 4.57 (d, J = 5.0 Hz, 4H), 4.23 (d, J = 6.4 Hz, 1H), 3.99 (d, J = 5.3 Hz, 4H), 3.95 (d, J = 5.2 Hz, 2H), 3.90 (s, 4H), 3.65 (t, J = 6.2 Hz, 2H), 3.35−3.28 (m, 1H), 3.28−3.20 (m, 4H), 3.15 (d, J = 6.0 Hz, 4H), 3.02 (s, 4H), 2.07−1.99 (m, 4H), 1.65 (s, 4H), 1.59−1.49 (m, J = 4.8 Hz, 6H), 1.47−1.42 (m, 6H), 1.41 (s, 5H), 1.40 (s, 9H), 1.38 (s, 4H), 1.27 (s, 18H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): δ 164.21, 163.40, 159.66, 158.09, 153.44, 153.39,

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

153.26, 138.00, 136.62, 134.27, 130.37, 128.38, 127.96, 127.93, 118.90, 104.55, 99.94, 81.93, 81.83, 67.99, 66.42, 65.59, 44.43, 40.99, 39.03, 38.44, 30.36, 29.80, 29.58, 29.31, 29.13, 28.29, 28.01, 27.85, 27.03, 26.56, 26.17, 24.49, 24.24 ppm. LRMS m/z (ESI) = 723.4 $[M + H]^+$.

*tert-Butyl (10-(8-(Benzyloxycarbonylamino)octyl)-11-oxo-18-(3-(trifluoroacetylamino)propoxy)-2,3,4,5,8,9,10,11,12,15-decahydrobenzo[b][1,5,7,9]oxatriazacycloheptadecin-13-yl)-carbamate (**21**).* Primary amine **20** (0.100 g, 1 equiv) was dissolved in dry THF (15 mL), and Et_3N was added (20 μL). To the resulting mixture, cooled to 0 °C, was added methyl trifluoroacetate dropwise, and the ice bath was removed. The reaction mixture was stirred for 16 h at room temperature. The reaction was diluted with MeOH, concentrated in vacuo, and purified by flash chromatography (SiO_2) (petroleum ether:EtOAc:MeOH = 7:2:1) to yield the desired product **21** as a mixture of two isomers (E/Z) as an amorphous white solid (isolated yield 90%). ^{1}H NMR (400 MHz, $CDCl_3$): δ 12.35−12.31 (m, 1H), 12.27−12.22 (m, 1H), 12.09−12.05 (m, 1H), 8.23 (t, J = 5.3 Hz, 1H), 7.99 (t, J = 5.1 Hz, 1H), 7.77 (t, J = 4.3 Hz, 1H), 7.40 (s, 2H), 7.32 (d, J = 3.9 Hz, 4H), 7.12 (dd, J = 7.9, 4.3 Hz, 2H), 6.42−6.33 (m, 4H), 6.07−5.97 (m, J = 14.7, 7.2 Hz, 1H), 5.64−5.48 (m, 1H), 5.42−5.33 (m, 1H), 5.06 (s, 3H), 4.86 (s, 1H), 4.62−4.57 (m, J = 5.1 Hz, 4H), 4.23 (d, J = 6.5 Hz, 1H), 4.04 (t, J = 5.2 Hz, 4H), 3.98−3.88 (m, J = 18.4, 8.2 Hz, 4H), 3.66 (t, J = 6.5 Hz, 2H), 3.54 (q, J = 5.7 Hz, 4H), 3.35−3.20 (m, 4H), 3.14 (d, J = 6.2 Hz, 4H), 2.07−1.99 (m, 4H), 1.67 (s, 4H), 1.54 (s, 6H), 1.48−1.43 (m, 5H), 1.42 (s, 4H), 1.40 (s, 9H), 1.39 (s, 3H), 1.30−1.23 (m, 17H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): δ 164.20, 164.06, 163.39, 159.39, 159.21, 158.82, 158.47, 158.44, 158.17, 157.54, 157.31, 156.94, 156.35, 153.45, 153.40, 153.26, 153.23, 152.42, 136.61, 134.23, 132.06, 131.27, 130.47, 129.63, 128.38, 127.92, 126.02, 125.11, 120.11, 119.33, 118.64, 117.30, 114.44, 104.55, 104.33, 104.05, 99.71, 99.56, 99.25, 82.02, 81.93, 81.88, 69.76, 67.95, 66.43, 60.28, 50.35, 47.10, 44.46, 42.52, 40.98, 39.03, 38.22, 32.23, 29.79, 29.57, 29.30, 29.11, 28.66, 28.26, 27.98, 27.84, 27.68, 27.01, 26.90, 26.55, 26.18, 24.48, 24.23 ppm. LRMS m/z (ESI) = 819.4 $[M + H]^+$.

*tert-Butyl (10-(8-(Amino)octyl)-11-oxo-18-(3-(trifluoroacetylamino)propoxy)-2,3,4,5,6,7,8,9,10,11,12,15-dodecahydrobenzo[b][1,5,7,9]oxatriazacycloheptadecin-13-yl)-carbamate (**22**).* To a solution of **21** (0.100 g, 1 equiv) in 2-propanol (20 mL) were added 10 μL of 36% HCl and a catalytic amount of 10% Pd/C (0.032 g, 0.2 equiv). The resulting mixture was stirred under H_2 atmosphere for 3 h. The reaction mixture was filtered through a pad of Celite and the filtrate concentrated in vacuo. The crude was used for the next reaction without any further purification. ^{1}H NMR (400 MHz, CD_3OD): δ 9.27 (s, 1H), 7.20 (s, 1H), 6.51 (s, 2H), 4.57 (s, 2H), 4.01 (m, 4H), 3.65 (s, 2H), 3.47 (s, 2H), 2.91 (s, 2H), 2.02 (s, 2H), 1.77−1.52 (m, 14H), 1.46 (s, 9H), 1.43−1.25 (m, 12H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$/ CD_3OD): δ 160.47, 158.37, 131.40, 104.92, 99.78, 68.03, 66.28, 45.22, 37.73, 28.58, 28.46, 28.42, 28.13, 27.72, 27.02, 26.01, 25.87, 25.78, 24.39 ppm. LRMS m/z (ESI) = 687.2 $[M + H]^+$; 344.1 $[M+2H]^{2+}$.

*tert-Butyl (10-(8-(((N′-tert-Butoxycarbonyl)-N-tert-butoxycarbonyl-N-crotylcarbamimidoyl)amino)octyl)-11-oxo-18-(3-(trifluoroacetylamino)propoxy)-2,3,4,5,6,7,8,9,10,11,12,15-dodecahydrobenzo[b][1,5,7,9]oxatriazacycloheptadecin-13-yl)-carbamate (**24**).* To a stirred solution of amine **22** (0.082 g, 1 equiv) in 20 mL of CH_3CN and MeOH (9:1) were added DIPEA (0.068 mL, 3 equiv) and *N*,*N*′-di-Boc-*N*-crotyl-1*H*-pyrazole-1-carboxamidine (**23**) (0.071 g, 1.5 equiv).[16] The reaction mixture was heated to 60 °C for 16 h. The solvent was removed in vacuo and the crude purified by flash chromatography (SiO_2) (petroleum ether:EtOAc:MeOH = 7:2:1) to yield the desired product **24** as an oil (isolated yield 70%). ^{1}H NMR (400 MHz, $CDCl_3$): δ 12.34 (s, 1H), 7.99 (t, J = 5.3 Hz, 1H), 7.25 (s, 1H), 7.12 (d, J = 8.1 Hz, 1H), 6.39 (d, J = 1.9 Hz, 1H), 6.36 (s, 2H), 5.67−5.41 (m, 2H), 4.59 (d, J = 5.3 Hz, 2H), 4.13 (d, J = 6.3 Hz, 2H), 4.05 (t, J = 5.5 Hz, 2H), 3.91 (s, 2H), 3.65 (t, J = 6.6 Hz, 2H), 3.55 (q, J = 6.0 Hz, 2H), 3.26−3.21 (m, 2H), 3.18 (t, J = 7.1 Hz, 2H), 2.09−2.02 (m, 2H), 1.71−1.62 (m, 8H), 1.58−1.50 (m, J = 5.5 Hz, 6H), 1.47 (s, 9H), 1.43 (s, 9H), 1.40 (s, 9H), 1.29 (s, 8H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): δ 164.20, 159.12, 158.18, 153.40, 153.27, 130.49, 126.08, 119.42, 104.30, 99.68, 81.89, 78.97, 67.94, 66.62, 44.46, 43.72, 39.01, 38.37, 29.28, 29.23, 29.12, 28.25, 28.13, 28.08, 27.99, 27.91, 27.88, 27.04, 26.75, 26.17, 24.46.

*tert-Butyl (10-(8-(((N′-tert-Butoxycarbonyl)-N-tert-butoxycarbonyl-N-crotylcarbamimidoyl)amino)octyl)-11-oxo-18-(3-(amino)-propoxy)-2,3,4,5,6,7,8,9,10,11,12,15-dodecahydrobenzo[b][1,5,7,9]-oxatriazacycloheptadecin-13-yl)carbamate (**25**).* K_2CO_3 (0.045 g, 5 equiv) was added to a solution (10 mL) of **24** (0.064 g, 1 equiv) in MeOH:H_2O (9:1). The resulting mixture was heated to 70 °C for 2 h, concentrated, extracted with DCM (3 × 10 mL), dried over Na_2SO_4, and concentrated. The crude was used without any further purification. ^{1}H NMR (400 MHz, $CDCl_3$): δ 12.36 (s, 1H), 7.99 (t, J = 4.9 Hz, 1H), 7.11 (d, J = 8.8 Hz, 1H), 6.40 (d, J = 6.3 Hz, 2H), 5.72−5.41 (m, 2H), 4.59 (d, J = 5.3 Hz, 2H), 4.14 (d, J = 6.3 Hz, 2H), 4.02 (t, J = 6.0 Hz, 2H), 3.93 (s, 2H), 3.66 (t, J = 6.5 Hz, 2H), 3.27−3.21 (m, 2H), 3.18 (t, J = 7.1 Hz, 2H), 2.91 (t, J = 6.2 Hz, 2H), 1.96−1.89 (m, 2H), 1.71−1.62 (m, 8H), 1.55 (d, J = 5.7 Hz, 8H), 1.47 (s, 9H), 1.44 (s, 9H), 1.41 (s, 9H), 1.29 (s, 12H), 1.23 (s, 6H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$): δ 164.23, 159.90, 158.11, 153.38, 153.28, 130.40, 126.13, 118.71, 104.56, 99.90, 81.81, 67.96, 65.85, 44.43, 43.74, 39.05, 32.48, 29.57, 29.28, 29.13, 28.39, 28.15, 28.10, 28.01, 27.89, 27.06, 26.77, 26.18, 24.50, 24.24, 17.59 ppm. LRMS m/z (ESI) = 887.5 $[M + H]^+$; 909.5 $[M + Na]^+$.

*tert-Butyl (10-(8-(((N′-tert-Butoxycarbonyl)-N-tert-butoxycarbonyl-N-crotylcarbamimidoyl)amino)octyl)-11-oxo-18-(3-(5-fluorescein-amino)thiocarbonylaminopropoxy)-2,3,4,5,6,7,8,9,10,11,12,15-dodecahydrobenzo[b][1,5,7,9]-oxatriazacycloheptadecin-13-yl)carbamate (**26**).* To a stirred solution of **25** (0.035 g, 1 equiv) in 1.5 mL of dry DMF were added sequentially DIPEA (0.047 mL, 7 equiv) and FITC isomer I (0.022 g, 1.5 equiv). The resulting mixture was stirred for 24 h at room temperature. The solvent was removed and the crude purified by flash chromatography (SiO_2) (DCM:MeOH = 9:1) to yield the desired product a yellow powder (isolated yield 52%). ^{1}H NMR (400 MHz, $CDCl_3$/ CD_3OD): δ 7.83 (d, J = 11.3 Hz, 1H), 7.03 (t, J = 7.1 Hz, 2H), 6.61 (s, 2H), 6.54 (d, J = 8.3 Hz, 2H), 6.43 (t, J = 10.1 Hz, 1H), 5.64−5.35 (m, 1H), 4.50 (s, 2H), 4.10−4.02 (m, 2H), 4.02−3.94 (m, 2H), 3.85 (s, 2H), 3.77 (s, 2H), 3.58 (s, 2H), 3.46−3.37 (m, 4H), 3.31 (d, J = 15.0 Hz, 4H), 3.13 (m, 6H), 2.13−2.04 (m, 2H), 1.58 (d, J = 4.7 Hz, 6H), 1.47 (d, J = 4.2 Hz, 6H), 1.40 (s, 9H), 1.38 (s, 9H), 1.34 (s, 9H), 1.23 (s, 10H) ppm. LRMS m/z (ESI) = 1276.6 $[M + H]^+$.

*(10-(8-((N-Crotylcarbamimidoyl)amino)octyl)-11-oxo-18-(3-(5-fluorescein-amino)thiocarbonylaminopropoxy)-2,3,4,5,6,7,8,9,10,11,12,15-dodecahydrobenzo[b][1,5,7,9]-oxatriazacycloheptadecin-13-yl)amine (**4**).* Boc-protected **26** (0.012 g, 1 equiv) was treated with a 20% CF_3COOH solution in DCM. The reaction mixture was stirred at room temperature for 8 h to yield the final product **4** as a yellow solid in quantitative yield. ^{1}H NMR (400 MHz, CD_3OD): δ 7.95 (s, 1H), 7.10−7.01 (m, 2H), 6.64 (m, 4H), 6.51 (m, 3H), 5.69 (s, 1H), 5.46 (s, 1H), 4.14−3.75 (m, 5H), 3.70 (s, 3H), 3.20 (s, 4H), 3.11 (s, 3H), 2.97 (s, 2H), 2.83 (s, 2H), 1.67 (s, 4H), 1.51 (s, 9H), 1.34 (s, 7H), 1.26 (s,15H) ppm. ^{13}C NMR (100 MHz, $CDCl_3$/ CD_3OD): δ 180.77, 164.09, 159.54, 158.00, 153.12, 152.82, 152.74, 140.00, 130.27, 128.98, 125.69, 124.76, 118.90, 112.58, 110.03, 105.07, 102.75, 99.80, 82.34, 82.22, 79.34, 67.99, 66.37, 44.69, 44.52, 43.67, 38.96, 31.77, 29.54, 29.21, 29.08, 28.02, 27.88, 26.96, 26.72, 22.52, 17.50, 13.91 ppm. LRMS m/z (ESI) = 976.5 $[M + H]^+$; 488.5 $[M+2H]^{2+}$; 326.4 $[M+3H]^{3+}$.

Animal Experiments. Mouse experiments were performed under the approval of the Institutional Animal Use and Care Committee at the "Università Cattolica del Sacro Cuore", Rome, Italy, and authorized by the Italian Ministry of Health, according to the Legislative Decree 116/92, which implemented the European Directive 86/609/EEC on laboratory animal protection in Italy. Animal welfare was routinely checked by veterinarians of the Service for Animal Welfare.

Experimental animal infections were carried out according to protocols described previously.[33] Briefly, cell suspensions of *C. albicans* strains, namely the reference ATCC 90028 and the clinical isolates DSY294 (azole-susceptible) and DSY296 (azole-resistant), were prepared in sterile saline and used separately (each in a volume of 200 μL) to inject immunocompetent BALB/c mice into their lateral

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

vein. Groups of 10 mice were established for each strain and used in tissue burden experiments. Seven days after injection with 7×10^4 CFU of yeast cells, mice were sacrificed, and their target organs (spleen and kidneys) were excised aseptically, weighted individually, and homogenized in sterile saline. Organ homogenates were then diluted and plated onto yeast peptone dextrose agar, and plates were incubated for 2 days at 30 °C. After growth, yeast colonies were counted, and the numbers of CFU per gram of organ were calculated. CFU counts were analyzed with nonparametric Wilcoxon rank-sum tests. Statistical significance was set at a P-value of <0.05. An injectable solution of **1** was prepared by adding 1 volume of PEG 400 to 2 volumes of a DMSO solution of **1** (0.2 mg/mL) in order to obtain a 0.1 mg/mL solution with DMSO/PEG 400 ratio of 2:1. The DMSO/PEG solution was injected in a 100 μL volume intraperitoneally each day. Fluconazole was injected intraperitoneally at dosages of 8 mg/kg/day.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge on the ACS Publications website at DOI: 10.1021/acs.jmedchem.6b00018.

Molecular formula strings (CSV)

Figure S1, showing the result of RMPI stability assays; Tables S1 and S2, showing common up- and down-regulated genes resulting from transcriptional analysis; Figure S2, showing fluorescence microscopy images of **2** and **3**; and ^{1}H NMR and ^{13}C NMR spectra of selected compounds (PDF)

■ AUTHOR INFORMATION

Corresponding Author

*Tel.: +39 0577 234306. E-mail: botta.maurizio@gmail.com.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

This work was partially supported by Bakker Medical s.r.l. D.S. was supported by a Swiss Research National Foundation grant (31003A 146936/1).

■ ABBREVIATIONS USED

ABC, ATP binding cassette; ATCC, American type culture collection; CFU, colony-forming unit; cRNA, complementary RNA; DCM, dichloromethane; DIPEA, diisopropylethylamine; DMF, dimethylformamide; FITC, fluorescein isothiocyanate; LRMS, low-resolution mass spectrometry; Mdrp1, multi-drug-resistant protein; RT-qPCR, real-time quantitative polymerase chain reaction; TEA, triethylamine; TFA, trifluoroacetic acid; YEPD, yeast extract peptone dextrose

■ REFERENCES

(1) Arendrup, M. C. Candida and Candidaemia. Susceptibility and Epidemiology. *Dan. Med. J.* **2013**, *60*, 4698.

(2) Morschhäuser, J. Regulation of Multidrug Resistance in Pathogenic Fungi. *Fungal Genet. Biol.* **2010**, *47*, 94−106.

(3) Pfaller, M. A.; Messer, S. A.; Moet, G. J.; Jones, R. N.; Castanheira, M. Candida Bloodstream Infections: Comparison of Species Distribution and Resistance to Echinocandin and Azole Antifungal Agents in Intensive Care Unit (ICU) and Non-ICU Settings in the SENTRY Antimicrobial Surveillance Program (2008−2009). *Int. J. Antimicrob. Agents* **2011**, *38*, 65−69.

(4) Groll, A. H.; Lumb, J. New Developments in Invasive Fungal Disease. *Future Microbiol.* **2012**, *7*, 179−184.

(5) Alexander, B. D.; Johnson, M. D.; Pfeiffer, C. D.; Jiménez-Ortigosa, C.; Catania, J.; Booker, R.; Castanheira, M.; Messer, S. A.; Perlin, D. S.; Pfaller, M. A. Increasing Echinocandin Resistance in Candida Glabrata: Clinical Failure Correlates with Presence of FKS Mutations and Elevated Minimum Inhibitory Concentrations. *Clin. Infect. Dis.* **2013**, *56*, 1724−1732.

(6) Bizerra, F. C.; Jimenez-Ortigosa, C.; Souza, A. C.; Breda, G.; Queiroz-Telles, F.; Perlin, D. S.; Colombo, A. L. Breakthrough Candidemia Due to Multidrug-Resistant Candida Glabrata during Prophylaxis with a Low Dose of Micafungin. *Antimicrob. Agents Chemother.* **2014**, *58*, 2438−2440.

(7) Sternberg, S. The Emerging Fungal Threat. *Science* **1994**, *266*, 1632−1634.

(8) Pappas, P. G.; Rex, J. H.; Lee, J.; Hamill, R. J.; Larsen, R. A.; Powderly, W.; Kauffman, C. A.; Hyslop, N.; Mangino, J. E.; Chapman, S.; Horowitz, H. W.; Edwards, J. E.; Dismukes, W. E. NIAID Mycoses Study Group. A Prospective Observational Study of Candidemia: Epidemiology, Therapy, and Influences on Mortality in Hospitalized Adult and Pediatric Patients. *Clin. Infect. Dis.* **2003**, *37*, 634−643.

(9) Klepser, M. E. Candida Resistance and its Clinical Relevance. *Pharmacotherapy* **2006**, *26*, 68−75.

(10) Ahmad, A.; Khan, A.; Manzoor, N. Reversal of Efflux Mediated Antifungal Resistance Underlies Synergistic Activity of two Monoterpenes with Fluconazole. *Eur. J. Pharm. Sci.* **2013**, *48*, 80−86.

(11) Castagnolo, D.; Raffi, F.; Giorgi, G.; Botta, M. Macrocyclization of Di-Boc-guanidino-alkylamines Related to Guazatine Components: Discovery and Synthesis of Innovative Macrocyclic Amidinoureas. *Eur. J. Org. Chem.* **2009**, *3*, 334−337.

(12) Dreassi, E.; Zizzari, A. T.; D'Arezzo, S.; Visca, P.; Botta, M. Analysis of Guazatine Mixture by LC and LC-MS and Antimycotic Activity Determination of Principal Components. *J. Pharm. Biomed. Anal.* **2007**, *43*, 1499−1506.

(13) Raffi, F.; Corelli, F.; Botta, M. Efficient Synthesis of Iminoctadine, a Potent Antifungal Agent and Polyamine Oxidase Inhibitor (PAO). *Synthesis* **2007**, *19*, 3013−3016.

(14) Manetti, F.; Castagnolo, D.; Raffi, F.; Zizzari, A. T.; Rajamäki, S.; D'Arezzo, S.; Visca, P.; Cona, A.; Fracasso, M. A.; Doria, D.; Posteraro, B.; Sanguinetti, M.; Fadda, G.; Botta, M. Synthesis of New Linear Guanidines and Macrocyclic Amidinourea Derivatives Endowed with High Antifungal Activity against Candida spp. and Aspergillus spp. *J. Med. Chem.* **2009**, *52*, 7376−7379.

(15) Castagnolo, D.; Schenone, S.; Botta, M. Guanylated Diamines, Triamines, and Polyamines: Chemistry and Biological Properties. *Chem. Rev.* **2011**, *111*, 5247−5300.

(16) Sanguinetti, M.; Sanfilippo, S.; Castagnolo, D.; Sanglard, D.; Posteraro, B.; Donzellini, G.; Botta, M. Novel Macrocyclic Amidinoureas: Potent Non-Azole Antifungals Active against Wild-Type and Resistant Candida Species. *ACS Med. Chem. Lett.* **2013**, *4*, 852−857.

(17) Sanfilippo, S.; Posteraro, B.; Sanguinetti, M.; Botta, M.; Maccari, G.; De Luca, F.; Docquier, J. D.; Deodato, D. New Macrocyclic Amidinourea, Methods of Preparation and Uses Thereof as Chitinase Inhibitors. Int. Patent Appl. WO/2014/202697A1, 2014.

(18) Botta, M.; Raffi, F.; Visca, P. Linear and Cyclic Guanidine Derivatives, Method of Preparation and Uses Thereof. Int. Patent Appl. WO/2009/113033A2, 2009.

(19) Maccari, G.; Sanfilippo, S.; De Luca, F.; Deodato, D.; Casian, A.; Dasso Lang, M. C.; Zamperini, C.; Dreassi, E.; Rossolini, G. M.; Docquier, J. D.; Botta, M. Synthesis of Linear and Cyclic Guazatine Derivatives Endowed with Antibacterial Activity. *Bioorg. Med. Chem. Lett.* **2014**, *24*, 5525−5529.

(20) Putignani, L.; Paglia, M. G.; Bordi, E.; Nebuloso, E.; Pucillo, L. P.; Visca, P. Identification of Clinically Relevant Yeast Species by DNA Sequence Analysis of the D2 Variable Region of the 25−28S rRNA Gene. *Mycoses* **2008**, *51*, 209−227.

(21) Coste, A. T.; Karababa, M.; Ischer, F.; Bille, J.; Sanglard, D. TAC1, Transcriptional Activator of CDR Genes, is a New Transcription Factor Involved in the Regulation of Candida Albicans ABC Transporters CDR1 and CDR2. *Eukaryotic Cell* **2004**, *3*, 1639−1652.

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

(22) De Micheli, M.; Bille, J.; Schueller, C.; Sanglard, D. A Common Drug-Responsive Element Mediates the Upregulation of the Candida Albicans ABC Transporters CDR1 and CDR2, Two Genes Involved in Antifungal Drug Resistance. *Mol. Microbiol.* **2002**, *43*, 1197−1214.
(23) Sanglard, D.; Ischer, F.; Monod, M.; Bille, J. Cloning of Candida Albicans Genes Conferring Resistance to Azole Antifungal Agents: Characterization of CDR2, a New Multidrug ABC Transporter Gene. *Microbiology* **1997**, *143*, 405−416.
(24) Sun, N.; Li, D.; Fonzi, W.; Li, X.; Zhang, L.; Calderone, L. Multidrug-Resistant Transporter Mdr1p-Mediated Uptake of a Novel Antifungal Compound. *Antimicrob. Agents Chemother.* **2013**, *57*, 5931−5939.
(25) MacCallum, D. M.; Coste, A.; Ischer, F.; Jacobsen, M. D.; Odds, F. C.; Sanglard, D. Genetic Dissection of Azole Resistance Mechanisms in Candida Albicans and their Validation in a Mouse Model of Disseminated Infection. *Antimicrob. Agents Chemother.* **2010**, *54*, 1476−1483.
(26) Rodriguez-Tudela, J. L.; Arendrup, M. C.; Barchiesi, F.; Bille, J.; Chryssanthou, E.; Cuenca-Estrella, M.; Dannaoui, E.; Denning, D. W.; Donnelly, J. P.; Dromer, F.; Fegeler, W.; Lass-Flörl, C.; Moore, C.; Richardson, M.; Sandven, P.; Velegraki, A.; Verweij, P. EUCAST Definitive Document EDef 7.1: Method for the Determination of Broth Dilution MICs of Antifungal Agents for Fermentative Yeasts: Subcommittee on Antifungal Susceptibility Testing (AFST) of the ESCMID European Committee for Antimicrobial Susceptibility Testing (EUCAST). *Clin. Microbiol. Infect.* **2008**, *14*, 398−405.
(27) Arendrup, M. C.; Cuenca-Estrella, M.; Lass-Flörl, C.; Hope, W. EUCAST Technical Note on the EUCAST Definitive Document EDef 7.2: Method for the Determination of Broth Dilution Minimum Inhibitory Concentrations of Antifungal Agents for Yeasts EDef 7.2 (EUCAST-AFST). *Clin. Microbiol. Infect.* **2012**, *18*, 246−247.
(28) National Committee for Clinical Laboratory Standards. *Methods for Determining Bactericidal Activity of Antimicrobial Agents, Approved Guideline*, Document M26-A; National Committee for Clinical Laboratory Standards: Wayne, PA, 1999.
(29) Pfaller, M. A.; Sheehan, D. J.; Rex, J. H. Determination of Fungicidal Activities against Yeasts and Molds: Lessons Learned from Bactericidal Testing and the Need for Standardization. *Clin. Microbiol. Rev.* **2004**, *17*, 268−280.
(30) Sanglard, D.; Ischer, F.; Bille, J. Role of ATP-Binding-Cassette Transporter Genes in High-Frequency Acquisition of Resistance to Azole Antifungals in Candida Glabrata. *Antimicrob. Agents Chemother.* **2001**, *45*, 1174−1183.
(31) Synnott, J. M.; Guida, A.; Mulhern-Haughey, S.; Higgins, D. G.; Butler, G. Regulation of the Hypoxic Response in Candida Albicans. *Eukaryotic Cell* **2010**, *9*, 1734−1746.
(32) Sanguinetti, M.; Posteraro, B.; Fiori, B.; Ranno, S.; Torelli, R.; Fadda, G. Mechanisms of Azole Resistance in Clinical Isolates of Candida Glabrata Collected During a Hospital Survey of Antifungal Resistance. *Antimicrob. Agents Chemother.* **2005**, *49*, 668−679.
(33) Silva, L. V.; Sanguinetti, M.; Vandeputte, P.; Torelli, R.; Rochat, B.; Sanglard, D. Milbemycins: More than Efflux Inhibitors for Fungal Pathogens. *Antimicrob. Agents Chemother.* **2013**, *57*, 873−886.

DOI: 10.1021/acs.jmedchem.6b00018
J. Med. Chem. 2016, 59, 3854−3866

2.4.3 INSIGHT THE UPTAKE OF BM01, RATIONALIZING THE STUDIES ON CANDIDA ALBICANS MUTANTS

During the study of the biological profile, we obtained very interesting results in particular from the *in vitro* experiments. The transcriptional analysis conducted with BM01 on *Candida albicans* strain showed several down- and up-regulated genes. Among these last, a typical signature for TAC1-regulated genes was present. TAC1 transcription factor, CDR1 and CDR2: involved in drug resistance (in particular in azole resistance), was in fact upregulated. The results of the transcriptional experiment initially suggested us a possible resistance mechanism similar to what happens for azoles, with the involvement of efflux pumps which extrude the molecule outside the cells. To confirm this, we performed an inhibition test on *Candida albicans* mutants lacking CDR1 (strain DSY448) or both CDR1 and CDR2 (DSY654), [54] expecting an improved activity versus the mutants respect to the wild type. Surprisingly, the outcome of the test was exactly the opposite. Mutants were much more resistant to BM01 compared to the wild type (Figure 2-7).

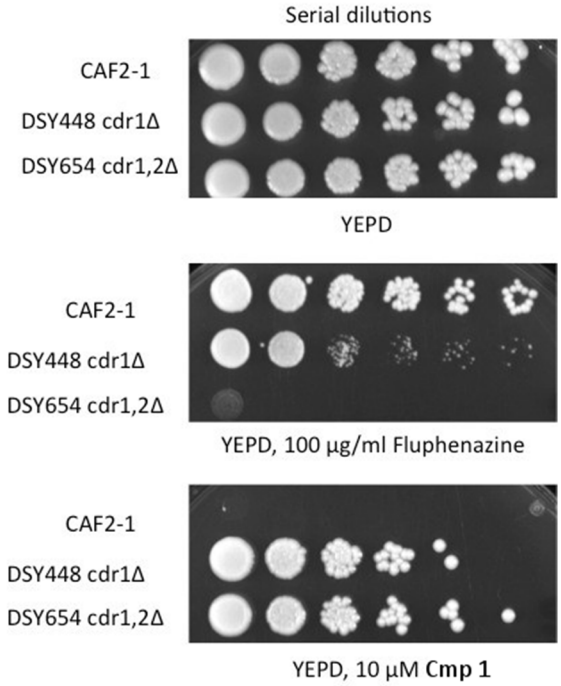


Figure 2-7: Serial dilution assay of Candida albicans with inhibitory substances (Cmp1 is BM01).

This unexpected result prompted us to suppose that CDR1 or CDR2 or another transporter regulated by TAC1 were probably required to the ingress of BM01 into the cell. As already mentioned, a similar behaviour is reported in the literature: in that paper authors demonstrates that the Mdrp1 transporter facilitates the uptake and increases the intracellular accumulation of the antifungal compound BQM. [25] Thus, we assumed a similar outcome for BM01, of which accumulation could be favoured by the expression of ABC transporter.

2.4.3.1 ABC TRANSPORTERS INVOLVEMENT: TESTS ON CANDIDA GLABRATA

To further corroborate our hypothesis, we performed a specific test on *Candida glabrata* comparing the activity of BM01 on the wild type respect to the over expressing, or the missing, CDR1 and CDR2 genes: BPY55 and SFY95 strains respectively. We used *Candida glabrata* because all the three strains needed were readily available. Results are reported in Table 2-2.

Table 2-2: Results of the inhibition tests of BM01. BPY55: Over expresses CDR1 and CDR2. SFY95 (Pdr1 Δ): Not expresses CDR1 and CDR2. Samples tested in triplicate, experiment conducted in double.

	ACTIVITY
Candida glabrata (CI)[a]	++
Candida glabrata (BPY55)	+++
Candida glabrata (SFY95)	-

[a] Clinical Isolate.
- Not active
+ Active

From the experiment, we obtained an activity value for the BPY55 smaller than the clinical isolate (compare the firsts two rows in Table 2-2). This was in perfect agreement with our initial hypothesis. In fact, a better activity value in the overexpressing strain strongly supports a possible involvement of the CDR genes in the uptake of the macrocycle. Moreover, the activity on SFY95 was, as expected, better than on the wild type, and

also in this case strongly suggesting an involvement of the ABC transporters in the uptake of BM01.

2.4.3.2 SPERMIDINE ASSAY

CDRs belong to the same family of other transporters involved in the uptake and efflux of amino acids, peptides, sugar molecules and polyamines as well as polyamine based drugs. [55] Thus, we hypothesized that besides CDR1 and CDR2, polyamine transporters that catalyse the uptake of polycationic molecules might also be involved in the translocation of BM01 as polycationic substrate.

The intracellular concentration of polyamine (such as putrescine, spermidine and spermine) is tightly regulated by fungal cells by balancing biosynthesis and degradation with cellular polyamine transport (both uptake and efflux). [56] In fact, polyamine uptake is repressed by high intracellular levels of polyamines. Thus, we designed an experiment in order to evaluate if the polyamine transporters were implied in the uptake of BM01, by growing different strains of *Candida glabrata* in presence or absence of spermidine. Results are reported in Table 2-3.

Table 2-3: Results of the inhibition tests in presence of 2mM of spermidine of BM01. BPY55: Over expresses CDR1 and CDR2. SFY95 (Pdr1 Δ): Not expresses CDR1 and CDR2. Samples tested in triplicate, experiment conducted in double.

	ACTIVITY
Candida glabrata (CI)[a] + 2mM spermidine	++
Candida glabrata (BPY55) + 2mM spermidine	+
Candida glabrata (SFY95) + 2mM spermidine	-

[a] Clinical Isolate.
- Not active
+ Active

Results show a protective effect of the polyamine: compare data on BPY55 without (Table 2-2) and with (Table 2-3) spermidine. This result confirms that polyamine transporters are involved in the uptake of BM01,

because both (spermidine and BM01) seems competing for the same transporter. Further experiments are planned in order to better clarify if the uptake is due to a specific transporter, or if BM01 can be the substrate of different members of the ABC superfamily.

2.5 Conclusions

In conclusion, a very particular mechanism of uptake for our hit compound has been proposed. Although further experiments must still be done, this first preliminary results are extremely interesting, because they can be the first proof-of-concept for a new way to address antifungal infections. A molecule that become more active with the increasing of efflux pumps expression can be easily associated with the current antifungal therapy (which indeed promotes that expression). If further experiments will proof this, we might have created a new trend for the pharmacological treatment of fungal infections, with a molecule which become more and more active in parallel to the increase of resistance to current azole drugs.

2.6 References

1 Stuart, H. Essential Microbiology (2nd edition). Wiley-Blackwell, 2013.

2 Adams, DJ. Fungal cell wall chitinases and glucanases. *Microbiology* (Jul 2004), 2029-2035.

3 Bowman, SM and Free, SJ. The structure and synthesis of the fungal cell wall. *Bioessays*, 28, 8. (Aug 2006), 799-808.

4 Colan, DE; Tashjian, AH; Armstrong, EJ; Armstrong, AW. Section 5, Principles of Chemotherapy. In *Principles of Pharmacology: The Pathophysiologic Basis of Drug Therapy.* 2011.

5 Bowman, SM and Free, SJ. The structure and synthesis of the fungal cell wall. *Bioessays*, 28, 8 (Aug 2006), 799-808.

6 Passos, XS; Costa, CR; Araújo, CR et al. Species distribution and antifungal susceptibility patterns of Candida spp. bloodstream isolates from a Brazilian tertiary care hospital. *Mycopathologia*, 163, 3 (Mar 2007), 145-151.

7 Walsh, TJ and Dixon, DM. Spectrum of Mycoses. In *Medical Microbiology*. Baron, S, 1996.

8 Koga, T; Matsuda, T; Matsumoto, T; Furue; M. Therapeutic approaches to subcutaneous mycoses. *Am J Clin Dermatol*, 4, 8 (2003), 537-543.

9 Arendrup, MC. Candida and candidaemia. Susceptibility and epidemiology. *Dan Med J*, 60, 11 (Nov 2013), B4698.

10 Carrasco-Zuber, JE; Navarrete-Dechent, C; Bonifaz, A; Fich, F; Vial-Letelier, V; Berroeta-Mauriziano, D. Cutaneous involvement in the deep mycoses: A review. Part II - Systemic mycoses. *Actas Dermosifiliogr*, 16 (Aug 2016), 30183.

11 Ruhnke, M. Antifungal stewardship in invasive Candida infections. *Clin Microbiol Infect*, 6 (Jun 2014), 11-18.

12 Pfaller, MA; Pappas, PG; Wingard, JR. Invasive Fungal Pathogens: Current Epidemiological Trends. *Clinical Infectious Diseases*, 43, Supplement 1 (2006), S3-S14.

13 Montagna, MT; Lovero, G; Borghi, E et al. Candidemia in intensive care unit: a nationwide prospective observational survey (GISIA-3 study) and review of the European literature from 2000 through 2013. *Eur Rev Med Pharmacol Sci*, 15, 5 (2014), 661-674.

14 Pfaller, MA; Boyken, L; Hollis, RJ; Messer, SA; Tendolkar, S; Diekema, DJ. In vitro activities of anidulafungin against more than 2,500 clinical isolates of Candida spp., including 315 isolates resistant to fluconazole. *J Clin Microbiol*, 43, 11 (Nov 2005), 5425-5427.

15 Yapar, N. Epidemiology and risk factors for invasive candidiasis. *Ther Clin Risk Manag*, 10 (Feb 2014), 95-105.

16 da Silva Dantas, A; Lee, KK; Raziunaite, I; Schaefer, K; Wagener, J; Yadav, B; Gow, NA. Cell biology of Candida albicans-host interactions. *Curr Opin Microbiol*, 34 (Spe 2016), 111-118.

17 Berman, J and Sudbery, PE. Candida Albicans: a molecular revolution built on lessons from budding yeast. *Nat Rev Genet*, 3, 12 (Dec 2002), 918-930.

18 Garcia, MC; Lee, JT; Ramsook, CB; Alsteens, D; Dufrêne, YF; Lipke, PN. A role for amyloid in cell aggregation and biofilm formation. *PLoS One*, 6, 3 (Mar 2011), e17632.

19 Du, H and Huang, G. Environmental pH adaption and morphological transitions in Candida albicans. *Curr Genet*, 62, 2 (May 2016), 283-286.

20 Naglik, JR; Challacombe, SJ; Hube, B. Candida albicans secreted aspartyl proteinases in virulence and pathogenesis. *Microbiol Mol Biol Rev*, 67, 3 (Sep 2003), 400-428.

21 Donlan, RM. Biofilms: Microbial Life on Surfaces. *Emerg Infect Dis*, 8, 9 (Sep 2002), 881-890.

22 Zonios, DI and Bennett, JE. Update on azole antifungals. *Semin Respir Crit Care Med*, 29, 2 (Apr 2008), 198-210.

23 Sanguinetti, M; Posteraro, B; Fiori, B; Ranno, S; Torelli, R; Fadda, G. Mechanisms of azole resistance in clinical isolates of Candida glabrata collected during a hospital survey of antifungal resistance. *Antimicrob Agents Chemother*, 49, 2 (Feb 2005), 668-679.

24 Coste, AT; Karababa, M; Ischer, F; Bille, J; Sanglard, D. TAC1, transcriptional activator of CDR genes, is a new transcription factor involved in the regulation of Candida albicans ABC transporters CDR1 and CDR2. *Eukaryot Cell*, 3, 6 (Dec 2004), 1639-1652.

25 Sun, N; Li, D; Fonzi, W; Li, X; Zhang, L; Calderone, R. Multidrug-resistant transporter mdr1p-mediated uptake of a novel antifungal compound. *Antimicrob Agents Chemother*, 57, 12 (Dec 2013), 5931-5939.

26 Löffler, J; Kelly, SL; Hebart, H; Schumacher, U; Lass-Flörl, C; Einsele, H. Molecular analysis of cyp51 from fluconazole-resistant Candida albicans strains. *FEMS Microbiol Lett*, 151, 2 (Jun 1997), 263-268.

27 Sanglard, D; Ischer, F; Koymans, L; Bille, J. Amino acid substitutions in the cytochrome P-450 lanosterol 14alpha-demethylase (CYP51A1) from azole-resistant Candida albicans clinical isolates contribute to resistance to azole antifungal agents. *Antimicrob Agents Chemother*, 42, 2 (Feb 1998), 241-253.

28 Lopez-Ribot, JL; McAtee, RK; Lee, LN; Kirkpatrick, WR; White, TC; Sanglard, D; Patterson, TF. Distinct patterns of gene expression associated with development of fluconazole resistance in serial candida albicans isolates from human immunodeficiency virus-infected patients with oropharyngeal candidiasis. *Antimicrob Agents Chemother*, 42, 11 (Nov 1998), 2932-2937.

29 Volmer, AA; Szpilman, AM; Carreira, EM. Synthesis and biological evaluation of amphotericin B derivatives. *Nat Prod Rep*, 27, 9 (Sep 2010), 1329-1349.

30 Holz, RW. The effects of the polyene antibiotics nystatin and amphotericin B on thin lipid membranes. *Ann N Y Acad Sci*, 235 (May 1974), 469-479.

31 Herve, M; Debouzy, JC; Borowski, E; Cybulska, B; Garybobo, CM. The role of the carboxyl and amino-groups of polyene macrolides in their interactions with sterols and their selective toxicity - a P-31-NMR study. *Biochim. Biophys. Acta Biomembr*, 980 (1989), 261-272.

32 Dick, JD; Merz, WG; Saral, R. Incidence of polyene-resistant yeasts recovered from clinical specimens. *Antimicrob Agents Chemother*, 18, 1 (Jul 1980), 158-163.

33 Ryder, NS; Wagner, S; Leitner, I. In vitro activities of terbinafine against cutaneous isolates of Candida albicans and other pathogenic yeasts. *Antimicrob Agents Chemother*, 42, 5 (May 1998), 1057-1061.

34 Lanyi, JK; Plachy, WZ; Kates, M. Lipid interactions in membranes of extremely halophilic bacteria. II. Modification of the bilayer structure by squalene. *Biochemistry*, 13, 24 (Nov 1974), 4914-4920.

35 Denning, DW. Echinocandin antifungal drugs. *Lancet*, 362, 9390 (Oct 2003), 1142-1151.

36 Ramage, G; Robertson, SN; Williams, C. Strength in numbers: antifungal strategies against fungal biofilms. *Int J Antimicrob Agents*, 43, 2 (Feb 2014), 114-120.

37 Flemming, HC; Wingender, J; Griegbe; Mayer, C. Physico-chemical properties of biofilms. In Publishers, Harwood Academic, ed., *Biofilms: recent advances in theirstudy and control*. Evans LV, Amsterdam, 2000.

38 Taff, HT; Nett, JE; Zarnowski, R et al. A Candida biofilm-induced pathway for matrix glucan delivery: implications for drug resistance. *PLoS Pathog*, 8, 8 (2012), e1002848.

39 Nobile, CJ; Nett, JE; Hernday, AD et al. Biofilm matrix regulation by Candida albicans Zap1. *PLoS Biol*, 7, 6 (Jun 2009), e1000133.

40 Rajendran, R; Sherry, L; Lappin, DF et al. Extracellular DNA release confers heterogeneity in Candida albicans biofilm formation. *BMC Microbiol*, 14, 1 (Dec 2014), 303.

41 Kollef, M; Micek, S; Hampton, N; Doherty, JA; Kumar, A. Septic shock attributed to Candida infection: importance of empiric therapy and source control. *Clin Infect Dis*, 54, 12 (Jun 2012), 1739-1746.

42 Sherry, L; Rajendran, R; Lappin et al. Biofilms formed by Candida albicans bloodstream isolates display phenotypic and transcriptional heterogeneity that are associated with resistance and pathogenicity. *BMC Microbiol*, 14, 1 (Jul 2014), 182.

43 Tobudic, S; Kratzer, C; Lassnigg, A; Graninger, W; Presterl, E. In vitro activity of antifungal combinations against Candida albicans biofilms. *J Antimicrob Chemother*, 65, 2 (Feb 2009), 271-274.

44 Cannon, RD; Lamping, E; Holmes, AR et al. Efflux-mediated antifungal drug resistance. *Clin Microbiol Rev*, 22, 2 (Apr 2009), 291-321.

45 Mukherjee, PK; Chandra, J; Kuhn, DM; Ghannoum, MA. Mechanism of fluconazole resistance in Candida albicans biofilms: phase-specific role of efflux pumps and membrane sterols. *Infect Immun*, 71, 8 (Aug 2003), 4333-4340.

46 Prasad, R and Goffeau, A. Yeast ATP-binding cassette transporters conferring multidrug resistance. *Annu Rev Microbiol*, 66 (2012), 39-63.

47 Maurya, IK; Thota, CK; Verma, SD et al. Rationally designed transmembrane peptide mimics of the multidrug transporter protein Cdr1 act as antagonists to selectively block drug efflux and chemosensitize azole-resistant clinical isolates of Candida albicans. *J Biol Chem*, 288, 23 (Jun 2013), 16775-16787.

48 Morschhäuser, J; Barker, KS; Liu, TT; BlaB-Warmuth, J; Homayouni, R; Rogers; PD. The transcription factor Mrr1p controls expression of the MDR1 efflux pump and mediates multidrug resistance in Candida albicans. *PLoS Pathog*, 3, 11 (Nov 2007), e164.

49 Hiller, D; Sanglard, D; Morschhäuser, J. Overexpression of the MDR1 gene is sufficient to confer increased resistance to toxic compounds in Candida albicans. *Antimicrob Agents Chemother*, 50, 4 (Apr 2006), 1365-1371.

50 Dreassi, E; Zizzari, AT; D'Arezzo, S; Visca, P; Botta, M. Analysis of guazatine mixture by LC and LC-MS and antimycotic activity determination of principal components. *J Pharm Biomed Anal*, 43, 4 (Mar 2007), 1499-1506.

51 Castagnolo, D; Raffi, F; Giorgi, G; Botta, M. Macrocyclization of Di-Boc-guanidino-alkylamines Related to Guazatine Components: Discovery and Synthesis of Innovative Macrocyclic Amidinoureas. *European J. Org. Chem*, 3 (Jan 2009), 334-337.

52 Manetti, F; Castagnolo, D; Raffi, F et al. Synthesis of new linear guanidines and macrocyclic amidinourea derivatives endowed with high antifungal activity against Candida spp. and Aspergillus spp. *J Med Chem*, 52, 23 (2009), 7376-7379.

53 Sanguinetti, M; Sanfilippo, S; Castagnolo, D; Sanglard, D; Posteraro, B; Donzellini, G; Botta, M. Novel Macrocyclic Amidinoureas: Potent Non-Azole Antifungals Active against Wild-Type and Resistant Candida Species. *ACS Med Chem Lett*, 4, 9 (Jul 2013), 852-857.

54 Sanglard, D; Ischer, F; Monod, M; Bille, J. Cloning of Candida albicans genes conferring resistance to azole antifungal agents: characterization of CDR2, a new multidrug ABC transporter gene. *Microbiology*, 143 (Feb 1997), 405-416.

55 Sanglard, D; Ischer, F; Calabrese, D; Majcherczyk, PA; Bille, J. The ATP binding cassette transporter gene CgCDR1 from Candida glabrata is involved in the resistance of clinical isolates to azole antifungal agents. *Antimicrob Agents Chemother*, 43, 11 (Nov 1999), 2753-2765.

56 Kumar, R; Chadha, S; Saraswat, D; Bajwa, JS; Li, RA; Conti, HR; Edgerton, M. Histatin 5 uptake by Candida albicans utilizes polyamine transporters Dur3 and Dur31 proteins. *J Biol Chem*, 286, 51 (Dec 2011), 43748-43757.

CHAPTER 3

Developing KNIME nodes: From properties calculation to the aid in molecules selection

Inte:Ligand ILKE. Release: ***August 2016****.*
Implementation: Standard Properties.
New nodes: Chirality Enumerator, Fingerprint Calculator, Fingerprint Similarity Calculator.

3.1 Introduction

The drug discovery and development process is becoming extremely expensive in term of money invested and in term of time to gain the marketing approval. In a report realized by the Tufts Center for the Study of Drug Development the average cost for a new molecule to reach the market was estimated to be $2.558 million. A similar study, conducted in 2003, estimated the cost to be $802 million, indicating that the cost to develop and win marketing approval for a new drug has increased by 145% between the two periods. [1]

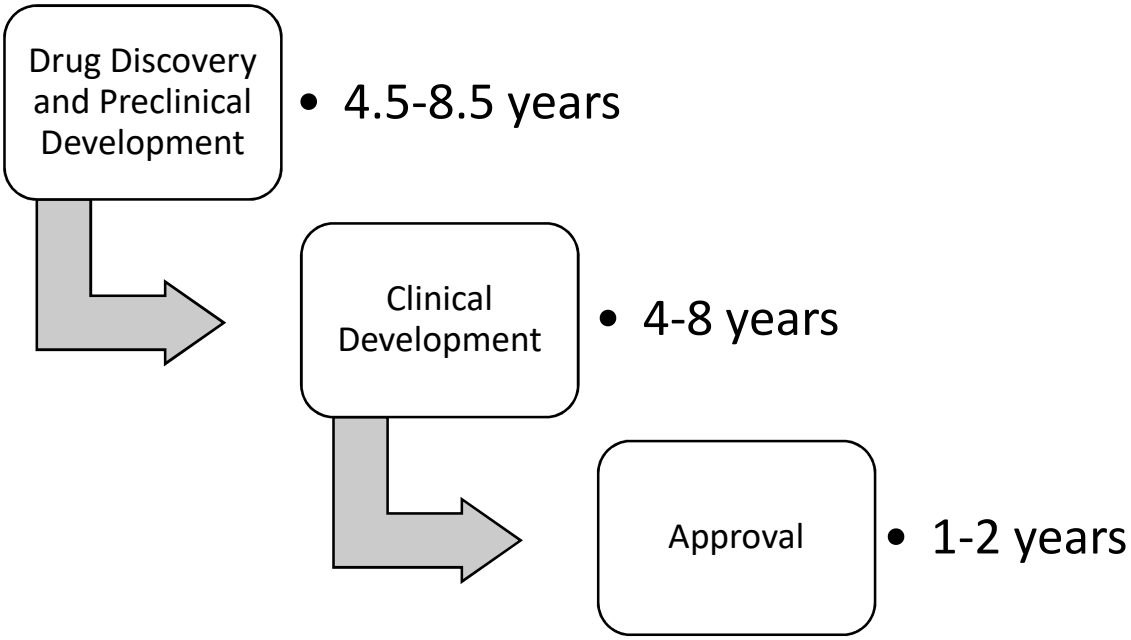


Figure 3-1: Timeline for the discovery of a new drug.

Considering this, and the fact that the whole process takes no less than 10 years, being able to save even just few months could make the difference in term of revenue guaranteed by the remaining time of the patents rights. A good point to do this is at the very early stages of drug discovery, when, computers can be used to save time in almost every step of the Drug Discovery and Preclinical Development step (Figure 3-2).

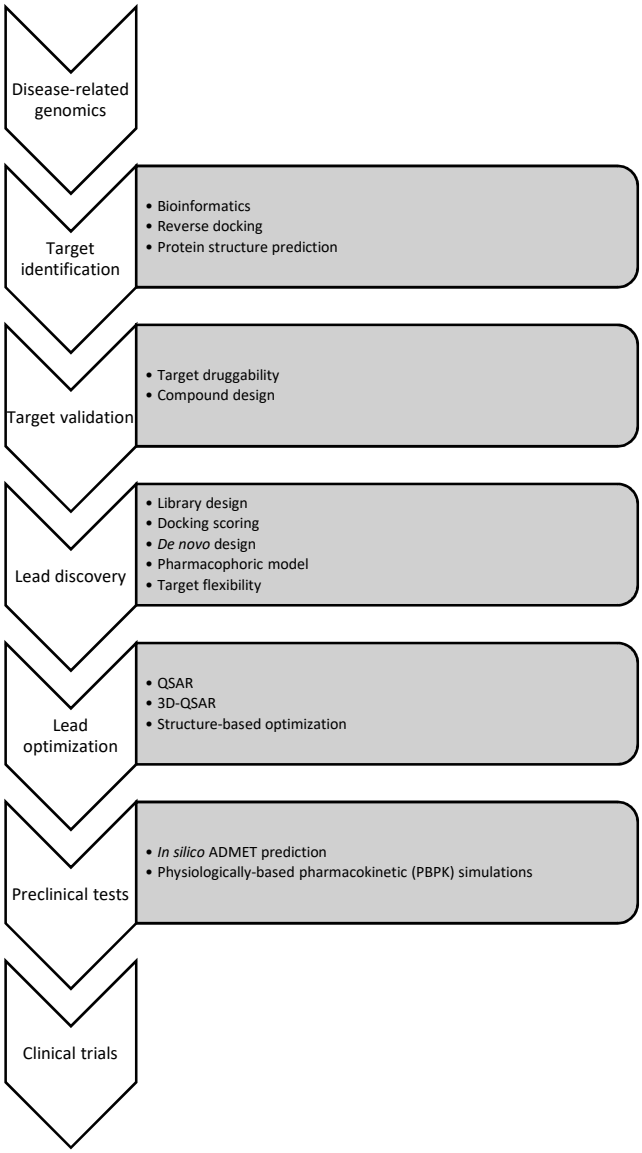


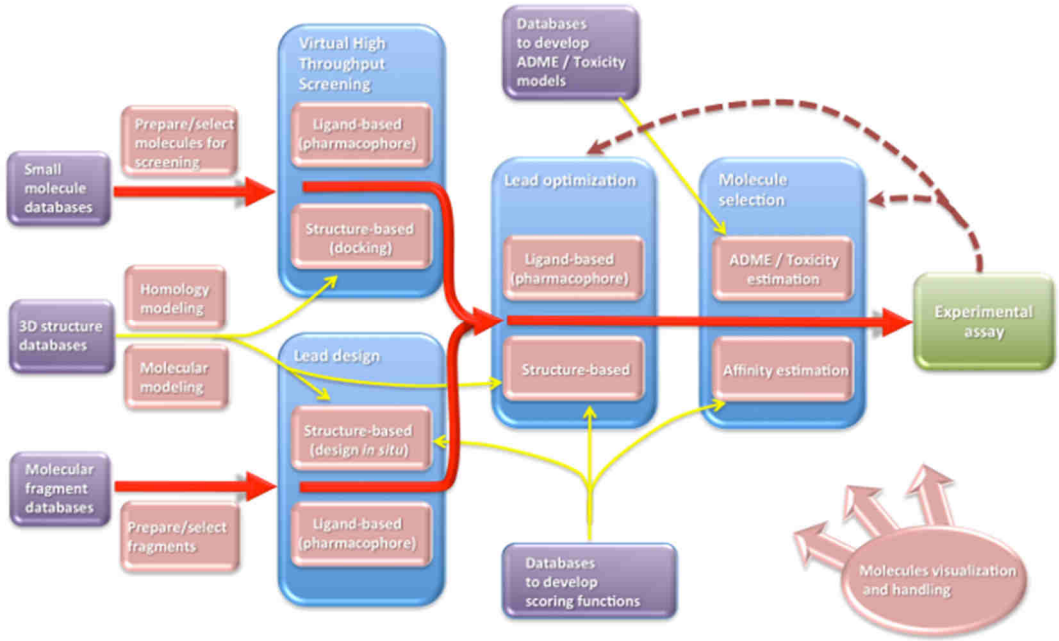
Figure 3-2: Computational approaches cover the drug-discovery pipeline, spanning from target identification to lead discovery to preclinical test. [2]

The role of computational models is in fact to increase prediction accuracy based on existing knowledge, with the final aim to increase the hit rate.

Computational methods gain every day a more important role in the drug discovery and development process, inasmuch offer a means of improved efficiency for the industry. [3] They let focus chemical synthesis and biological testing, generating a decrease in the traditional resource requirements. [4] Approximately 10 years ago, was estimated that computer

modelling and simulations accounted for ~10% of pharmaceutical R&D expenditure and was predicted a rise to 20% until now. [5]

With the availability of increasingly powerful machines, new computational techniques were developed and new software were written to help the molecular modeller in the drug design process, the so called CADDD. In Figure 3-3, an *in silico* drug discovery pipeline is represented. It displays only a part of the available techniques, showing the importance reached by the molecular modelling.



Credit: Swiss Institute of Bioinformatics.

Figure 3-3: in-silico *drug design pipeline.* [6]

The main goal of pharmaceutical companies is to identify new molecules that can be transformed into drugs, and the most widely used techniques to do this is undoubtedly the HTVS. Under this acronym, several different methodologies are collected; the most famous are Docking and Pharmacophores. However, despite the technique behind, the HTVS software have all in common the same main feature: identify drug candidates from large collection of compound libraries.

A successful example was at Pharmacia (acquired by Pfizer in 2003), where the researchers used computational tools to perform a virtual screening on an enzyme implicated in diabetes. They found 127 effective inhibitors out of 365 selected molecules, with a hit rate of nearly 35%. Simultaneously, this group performed a traditional HTS against the same target. Of the 400000 compounds tested, 81 showed inhibition, producing a hit rate of only 0.021%. [7]

The advent of the HTVS has also favoured the foundation of several chemical companies, with the precise aim to provide to the customer a wide spectrum of new chemical compounds. Therefore, for both companies and universities, huge virtual databases of commercial compounds are today available.

A virtual commercial database is basically a file containing the 2D structure of molecules together with an unique identification code. Usually, during for example a HTVS process, the db has to be filtered, to remove all the unwanted molecules. Moreover, the remaining structures have to be prepared in order to be used to the various programs: 3D coordinates, right protonation state, tautomers, salt removal, and chirality enumeration are, for example, some of the feature required to perform a good screening. [8]

The preparation requires, as seen, several steps and the best way to automate them is using a data-pipelining software.

3.1.1 DATA-PIPELINING SOFTWARE

Managing computational workflows has always been an important topic in computer science in general, and, in computational chemistry in particular.

The main goal of any data pipeline is to execute multiple subprocesses in a specific order. These subprocesses need to be able to independently modify the data to be processed and pass their specific output on to the next subprocess. Data-pipelining software aims to use this concept to enable its users to create complex workflows by stringing several

independent subprograms together. An illustration of this basic concept is given in Figure 3-4.

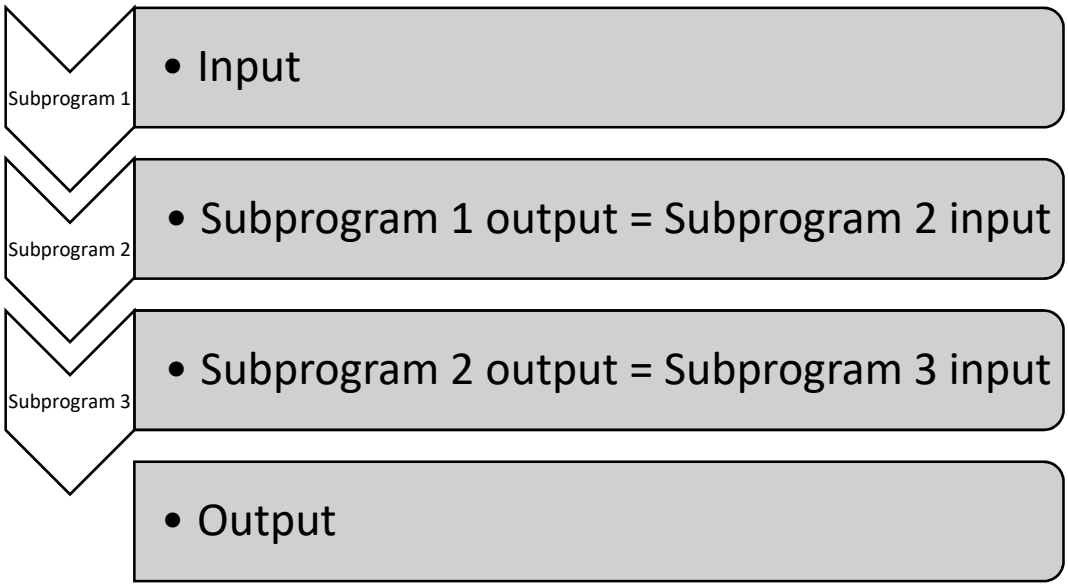


Figure 3-4: General illustration of the data-pipelining concept.

One of the program which has had an aggressively expanding in its market share is undoubtedly KNIME. [9] This program, written in Java language, [10] had an increase of 156% from 2013 to 2014 among users of software of this kind. [11]

3.1.2 THE JAVA LANGUAGE

The Java computer language is a high level programming language. It was developed by Sun Microsystems in 1995 and in 2010 was acquired by Oracle, the current owner of the trade mark. The main characteristic of Java is to run over the Java Virtual Machine, which makes it independent from the type of machine and the operative system in which the code is executed.

Java is an object oriented programming language. This means that it uses objects (that have states and behaviours) which communicate one to

each other to perform operations. To create objects Java uses classes as blueprint.

3.1.3 WHAT IS KNIME?

KNIME is an open solution for data science software. The development started 2004 at the University of Konstanz, Germany, with the goal of creating a tool capable of dealing with huge amounts of data. [12] The program provides a graphical user interface that relies on drag and drop mechanics, in order to easily allow to create complex data pipelines. The graphical workflow editor is implemented as an Eclipse IDE [13] plug-in and the whole software is written, as already mentioned, in Java. Figure 3-5 shows a screenshot of the program.

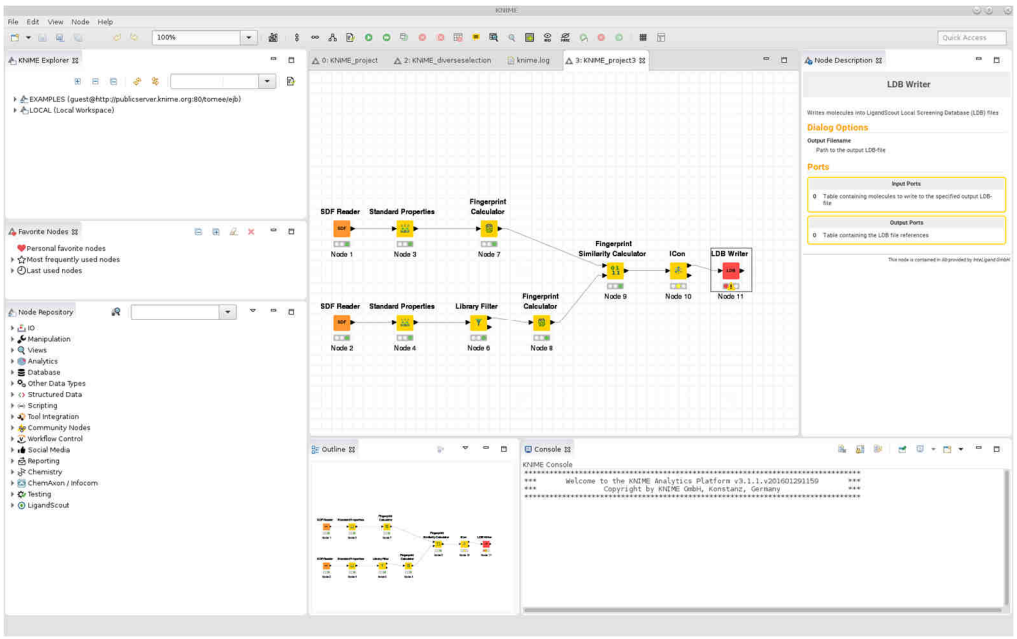


Figure 3-5: Screenshot of the KNIME program showing an exemplary workflow.

KNIME uses three main architectural principles:

- Visual, interactive framework
- Modularity
- Easy expandability

KNIME main window allows a simple drag and drop construction of data pipelines. The so-called nodes are connected via edges; every node separately processes the data arriving at its input port and forwards the results to its output ports. KNIME allows third parties to add new processing nodes and distribute them. [14]

3.1.4 DEVELOPING NEW NODES

To be able to include new functionality in KNIME nodes, the specific SDK version was used. This is an Eclipse IDE (Integrated Development Environment) with several preinstalled plugins, which offers the node creation wizard, making the development process as simple as possible. Essential (or important) classes are below listed:

- `NodeModel` has to be extended and implemented. It requires methods to validate input data, to configure and execute the node.
- `NodeDialog` has to be extended if it is necessary for the user to make any kind of choices or select parameters. For doing this KNIME provides standard dialog components.
- `NodeView` has to be extended if the developer wants to offers different views onto the node's underlying models and data

These concepts are illustrated in Figure 3-6.

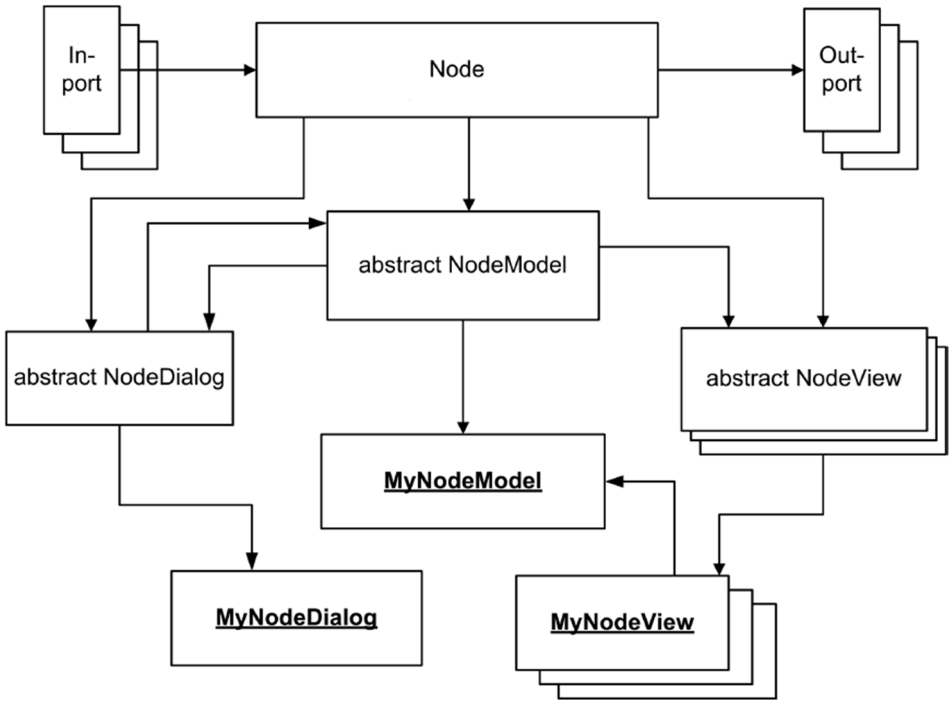


Figure 3-6: Diagram of the Node and the main classes it relies on. [14]

The `DataTable` class is recommended to manage data flows between single nodes, providing information about the data types. Moreover, in order to process the table row by row, information about iterators are also contained in this class. Connections of `DataTable` class to other classes are shown in Figure 3-7.

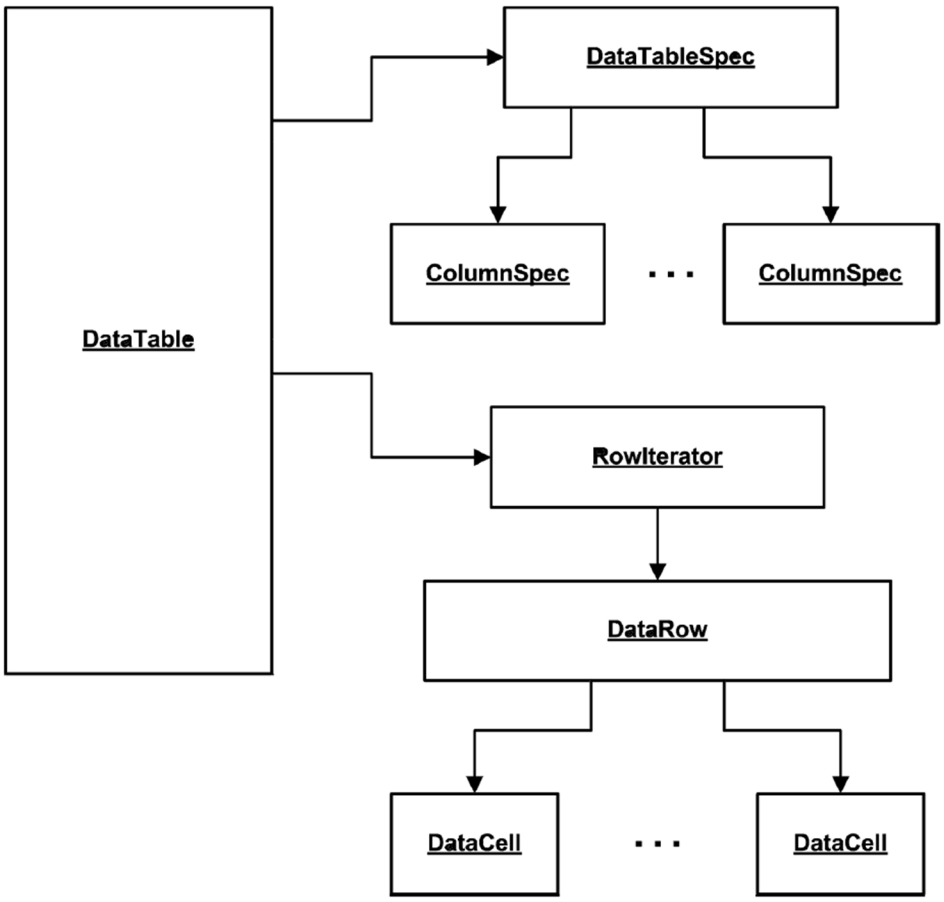


Figure 3-7: Diagram of the data structure and the main classes it relies on. [14]

3.1.5 LIGANDSCOUT

LigandScout is the flagship software developed at Inte:Ligand. [15] Originally created to realize a pharmacophoric models starting from known 3D structure in a fully automated way, the program have been improved during years offering a lot more functionality. [16] An example is the incorporation of an hit list analysis tool able to visualise ROC curves [17] and enrichment factor calculations. A very interesting ligand-based pharmacophore modelling feature is also present. Together with the friendly graphical user interface (Figure 3-8) and the nice 3D rendering, LigandScout is nowadays a complete pharmacophore-based drug development platform. [18]

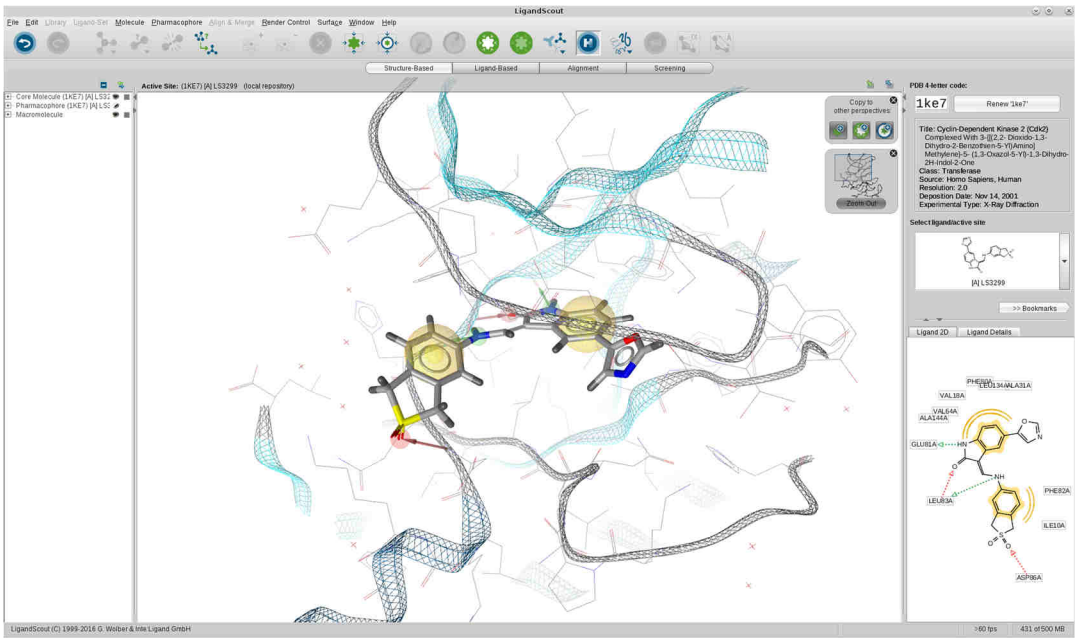


Figure 3-8: Screenshot of the LigandScout program showing a 3D rendering of a ligand embedded in a corresponding binding site.

3.1.6 LIGANDSCOUT IN KNIME

Inte:Ligand has recently joined the KNIME program to make available its technology and know-how in molecular design and virtual screening to the KNIME community. The ILKE package combines KNIME's pipelining technology with LigandScout, extending the capabilities of molecular designers and medicinal chemists. Moreover, Inte:Ligand's algorithms and knowledge based tools for pharmacophore creation and clustering, molecular dynamics analysis, ligand-activity profiling and molecular structure evaluation are accessible thanks to the ILKE package.

The ILKE package is becoming every release more complete. This is mainly due to the possibility to transfer some of the algorithm used in LigandScout directly into the nodes of the data-pipeline software. Despite the compatibility offered by the characteristic to be both written in Java, some useful features were missing in LigandScout, and the implementation of specific KNIME noses is extremely important to complete the parent code and offer to the final user the total control on its drug discovery and development process. Analysing the list of available nodes, specific tools to manage chiral property of molecules were missing,

together to a fast algorithm to evaluate the structural similarity of a set of ligands.

3.1.7 CHIRALITY

Determination of chiral centres and labelling by the R S system is important in computational chemistry for several reasons. Aside the usual uses of nomenclature, a proper labelling of defined and undefined chiral centres results to be extremely useful as parameter to use as filter in a HTVS procedure.

Moreover, the individuation and the determination of the number of undefined chiral centres results to be crucial in the correct preparation of the input file for every molecular modelling software which needs tridirectionally ligands. In fact, although almost every program use 3D molecules, the commercial compounds sold as raceme mixture are often reported in the structure file provided by the vendor as unique entry, creating problems during the 3D coordinates generation.

3.1.8 FINGERPRINTS

Molecular fingerprints [19] are representations of chemical structures using binary format. They encode structural information in a very simple way, and for this reason are a daily-used tool of machine learning and cheminformatics in drug discovery applications. They were originally designed to assist in chemical database substructure searching, [20] but later they were used also for similarity searching, [21] clustering, [22] and classification. [23]

Several types of fingerprint exist nowadays. Some of the mayor representative elements are:

- FP2: [24] a path-based fingerprint which indexes molecular fragments.

- MACCS: [25] a key-based fingerprint which uses 166 predefined keys.
- ECFP: [26] a circular topological fingerprint which is derived using a variant of the Morgan algorithm.

Among these different types, the ECFP is the best performing while ranking diverse structure by similarity. [27]

3.1.8.1 ECFPs

Extended-connectivity fingerprints (ECFPs) [26] are a type of circular topological fingerprints designed to hold molecular features relevant to molecular activity. They are derived using a variant of the Morgan algorithm, [28] to which make several changes. For instance, instead of waiting for the attainment of a unique identifier, a predetermined number of iterations has to be reached before terminating the ECFP generation. A set containing the initial atom identifiers, together with all identifiers after each iteration (intermediate atom identifiers), are collected. These ones define the ECFPs.

The generation process can be summarized in Figure 3-9:

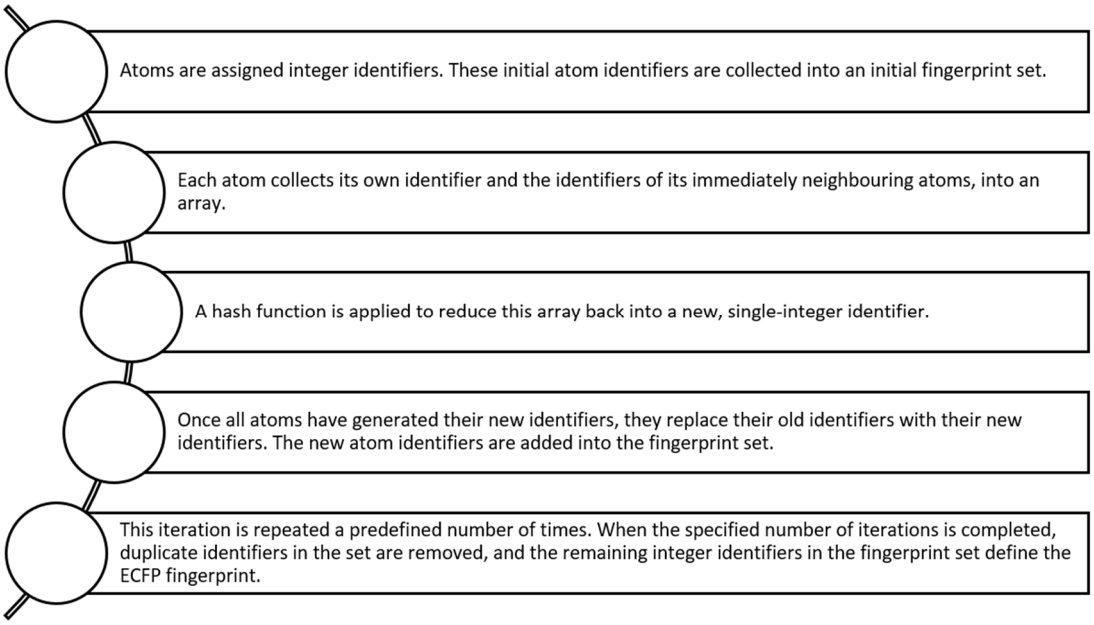


Figure 3-9: ECFP generation process.

The effect of iteratively updating the information is shown in Figure 3-10.

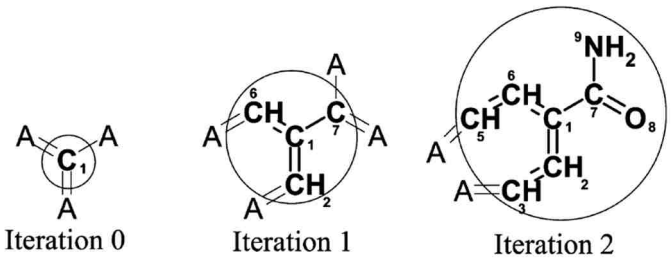


Figure 3-10: Illustration of the effect of iterative updating on the information represented by an atom identifier. Each iteration has the effect of creating an identifier that represents larger and larger circular substructures around the central atom, as shown at the top of the figure. [26]

3.1.8.2 K-MEANS ALGORITHM

As mentioned, one of the main use of the fingerprints is the clustering. Among the clustering algorithm, because of low memory space requirement and high computational efficiency, the K-means [29] is the most widely used partitional clustering algorithm. This one represents each cluster with its mean vector and assign each pattern to the closest cluster

iteratively. It terminates when the cluster labels do not change. [30] Basically the algorithm performs the steps reported in Figure 3-11. [31]

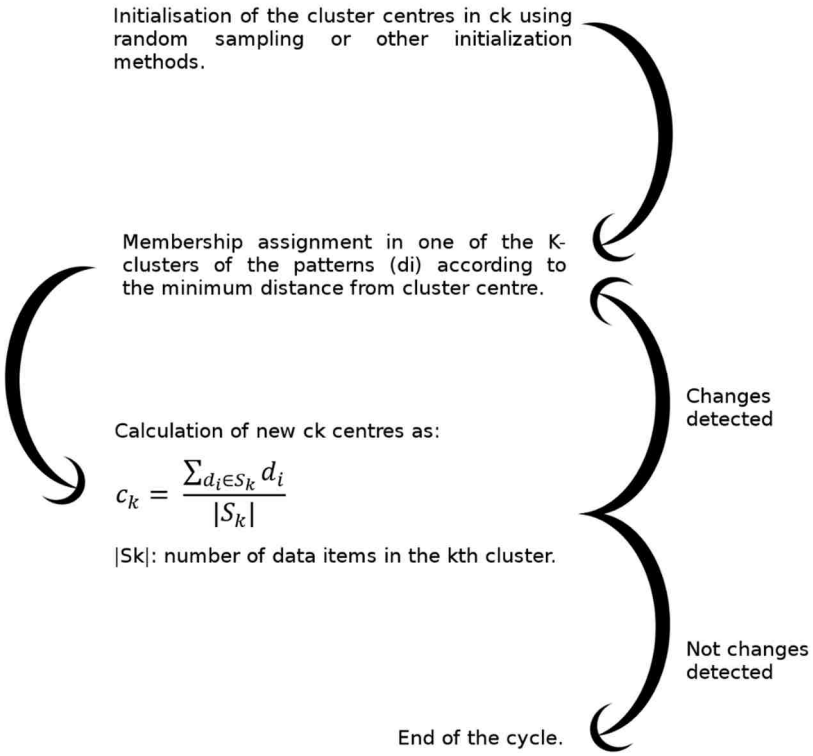


Figure 3-11: Steps for K-Mean Algorithm.

The K-Means is a heuristic algorithm and for this reason there is no guarantee that it will converge to the global optimum. Considering its non-deterministic nature, the result may depend on the initial clusters created at the first point.

3.2 Aim of the work

With the purpose of improving the tools present in the LigandScout suite for KNIME, an implementation of existing node, and the creation of several new ones, was carried out.

Thus, the first part of this chapter is dedicated to the development to chirality-related nodes. This provide a simple way to filter and prepare commercial databases, making the user able to have a total control on the chiral properties of the compounds composing them.

The second part instead is dedicated to the development of nodes based on the fingerprints. The fingerprint calculation is extremely fast and results to be the perfect technique to perform ligand-based operation conducted on big data. In fact, thanks to them, I've created specific nodes for i) molecule clustering, ii) similarity scoring and iii) diversity picking. These nodes result to be a very useful addiction for the ILKE package, because before them there wasn't the possibility to have a very fast guidance on how to identify or measure whether two or more molecules are (or aren't) structurally similar.

All the nodes were developed keeping in mind the basic LigandScout concept of creating user friendly software, in fact the easy usage, even for non-experienced users, was seen as mandatory.

3.3 Results and discussion

3.3.1 IMPLEMENTATION OF EXISTING KNIME NODES AND DEVELOPMENT OF NEW ONES

The first challenge faced was the management of the chiral molecules. For this problem, new code was integrated in the Standard Property, and a complete new node was written: Chirality Enumerator.

The second step was the development of new algorithm which could be used by the user to have a compete and fast way to identify the structure similarity (or the diversity) of a given collection of molecules. Using the concept of the Extended Connectivity Fingerprint, I've programmed: Fingerprint Calculator Node, Fingerprint Similarity Node, Fingerprint Clustering Node and the Diversity Picker Node. Figure 3-12 shows some of my nodes (Chirality Enumerator, Fingerprint Calculator and Fingerprint Similarity Calculator) present in the ILKE package in August 2016 Release.

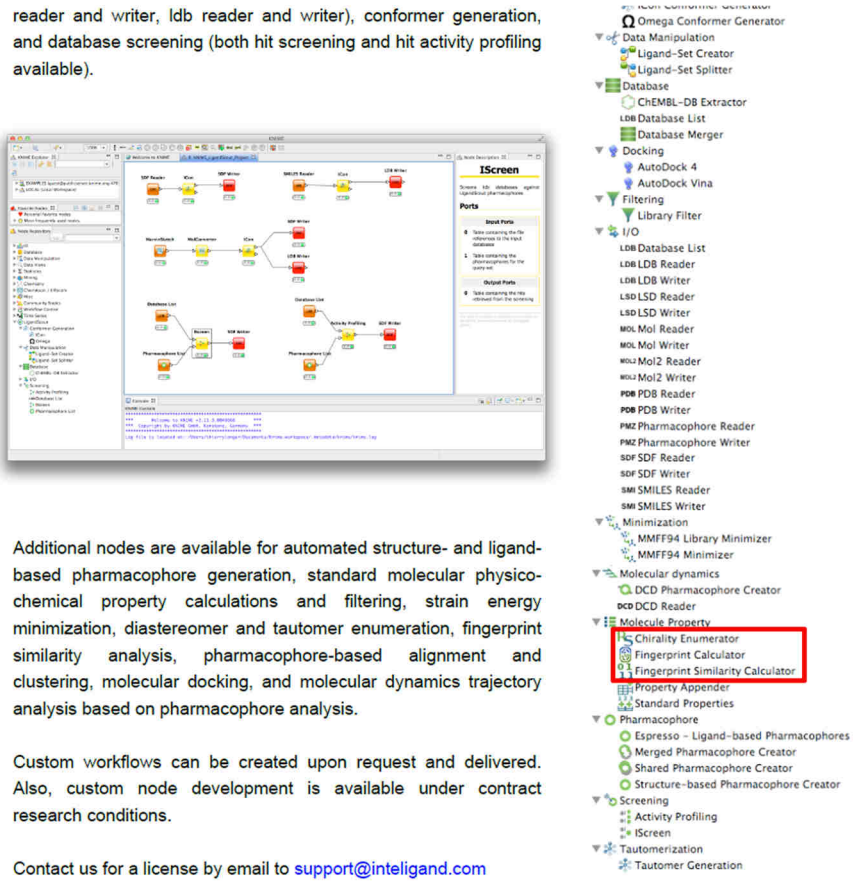
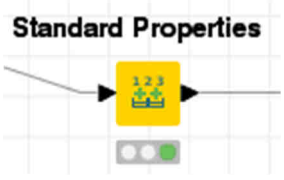


Figure 3-12: Inte:Ligand ILKE Informational Flyer. Release: August 2016. Some of my nodes are highlighted inside the red rectangle.

Each node will be treated in the further paragraphs of this chapter by explaining first the functional requirements (functions of the specific software defined by a set of inputs, the behaviour and the resulting outputs) and then the usage with a brief description.

3.3.1.1 Implementing the Standard Properties node: Number of defined and undefined chiral centres calculation



3.3.1.1.1 Functional requirements:

Name:	**Calculate standard properties for databases**
Primary actor:	End user
Brief description:	The Chiral Centres checkbox of the Standard Properties node is able to calculate the number of Undefined, R-defined and S-defined chiral centres (carbon atoms) present in a molecule. To do this, it uses proprietary code implemented at Inte:Ligand to evaluate the type of stereochemistry of each chiral carbon. The node proceeds then with the count of each type of chirality label: R, S or Undefined.
Trigger:	User drags the Chirality Enumerator node into the KNIME workspace window.
Preconditions:	A preceding node that outputs table containing structural information of the ligands (2D or 3D) has been executed successfully.
Postconditions:	Table containing a number of additional columns equal to the number of properties to calculate selected by the user in the Dialog box. If the Chiral Centres checkbox is selected three additional columns will be added, reporting the number of R-defined, S-defined and Undefined chiral centres of the input molecules.
Normal Flow: 1. User connects the output port of an adequate preceding node with the input port of the Standard Properties node. 2. User opens the configuration dialog. 3. System generates and shows the dialog for configuring the node. 4. User selects which properties wants to calculate through a series of checkboxes 5. User starts the execution of the node. 6. System indicates the finished execution.	
Alternative Flows: 7. User inspects the generated molecular properties after successful execution.	

Priority:	High. It is a primary use case.
Frequency of Occurrence:	It has to be expected that some users require this functionality multiple times a day.
Assumptions:	It is assumed that the LigandScout code is able to determine which carbon atoms are stereocenters in a molecule, and is also able to mark them as R-defined, S-defined or Undefined.

Table 3-1: Functional requirements of the Standard Properties node, in which the Chiral Centres checkbox was added.

3.3.1.1.2 Usage and description:

To use the Standard Properties node, a suitable preceding node has to be connected to its input port. To start using the Chirality Centres property calculation present in this node the user has to check the corresponding checkbox present in the node Dialog box (Figure 3-13).

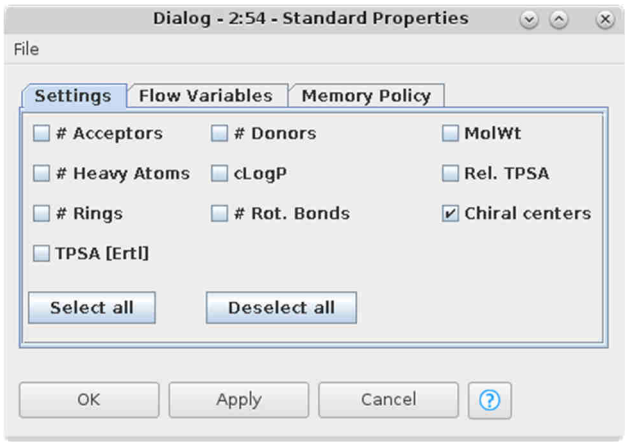


Figure 3-13: Dialog box of the Standard Properties node. The "Chiral centers" property is checked.

After the selection of the desired properties, which can be performed also by pressing the button *Select All*, the user executes the node and the calculation starts. The code for the Chirality Centres calculation goes through each atom of the molecule and, using the private Inte:Ligand method `getTetrahedralChirality`, retrieves the chirality label.

According to their type (`TetrahedralChirality.R`, `TetrahedralChirality.S` or `TetrahedralChirality.UNDEFINED`) the corresponding counter increases. The Code 1 is an example of the method written to get the number of undefined chiral centres for a molecule. The same code is valid also for R-defined and S-defined.

```
public int getNumberOfChiralCentersUndefined()
{
  int count = 0;
  for (Atom atom : orderedAtomList)
  {
    if (atom.getTetrahedralChirality() == TetrahedralChirality.UNDEFINED)
      count++;
  }
  return count;
}
```

Code 1: Code reporting the `getNumberOfChiralCentersUndefined` *method.*

The output displays a table which lists all the input molecule together with the new calculated properties.

For a more complete overview of the code see the APPENDIX 1.

3.3.1.2 CHIRALITY ENUMERATOR NODE (CHIRALATOR NODE)

3.3.1.2.1 Functional requirements:

Name:	**Enumerate all undefined chiral centres of a series of input molecules**
Primary actor:	End user

Brief description:	The Chirality Enumerator node can handle enumeration of R/S stereochemistry for carbon atoms. It uses proprietary code implemented at Inte:Ligand to determine which atoms are stereocenters and if they are marked as `UNDEFINED`. If a stereocenter has a designated stereochemistry, the program does not change it. However, if a stereocenter does not have a specified stereochemistry, the node will enumerate both stereochemistry states, creating two different molecules for each undefined chiral centre.
Trigger:	User drags the Chirality Enumerator node into the KNIME workspace window.
Preconditions:	A preceding node that outputs table containing structural information of the ligands (2D or 3D) has been executed successfully.
Postconditions:	Table containing two molecules for each undefined chiral centre present in the input molecule. The output contains also the updated chirality properties if the user has specified to recalculate them.

Normal Flow:

1. User connects the output port of an adequate preceding node with the input port of the Chirality Enumerator node.
2. User opens the configuration dialog.
3. System generates and shows the dialog for configuring the node.
4. User selects if wants to recalculate the chirality properties or not through a checkbox.
5. User starts the execution of the node.
6. System indicates the finished execution.

Exceptions:

1a. The input table doesn't contain any column with structural information (SDF, MOL, MOL2, SMILES or CCT). The node displays an error message notifying the user of the problem.

Alternative Flows:

7. User inspects the generated molecules after successful execution.

Priority:	High. It is a primary use case.

Frequency of Occurrence:	It has to be expected that some users require this functionality multiple times a day.
Assumptions:	It is assumed that the LigandScout code is able to determine which carbon atoms are stereocenters in a molecule, and is also able to mark them as R-defined, S-defined or Undefined.

Table 3-2:Functional requirements of Chirality Enumerator node.

3.3.1.2.2 Usage and description:

To use the Standard Properties node, a suitable preceding node has to be connected to its input port. Apart from this condition, enumerate all the stereoisomers of a molecule is actually fully automated and does not require any further action from the user, besides choosing if the chirality properties have to be updated (checkbox in the Dialog box) and starting the execution of the KNIME workflow.

The core of the node is the Code 2. This class contains the `calculate` method, which is composed by two `for` cycles and an `if` statement. With the first for cycle it calculates the tetrahedral chirality centres for the molecule using private Inte:Ligand code, then, with the second `for` loop inside the first one, it goes through each atom of the molecule and check if it is labelled as `UNDEFINED`. If it is the case, the program generates a copy of the molecule, sets the chirality of that atom to R, copies all the properties from the original molecule, adds the copy to the list of output molecules. In the same `if` statement performs this operation also to generate the S stereoisomer. The chirality check is done on each atom of the molecule, generating all the possible combinations if more than one chiral centre is present.

```java
package com.inteligand.ilib.base.stereo;

import java.util.ArrayList;
import java.util.List;

import com.inteligand.ilib.base.Atom;
import com.inteligand.ilib.base.Compound;
import com.inteligand.ilib.base.Molecule;

public class ChiralatorCalculation
{
  public List<Molecule> calculate(Molecule molecule)
  {
    List<Molecule> list = new ArrayList<Molecule>();
    for (Compound comp : molecule.getCompoundsCopy())
    {
      comp.calcTetrahedralChiralCenters();
      int index = 0;
      boolean allIsDefined = true;
      for (Atom atom : comp.getOrderedAtomList())
      {
        if (atom.getTetrahedralChirality() == TetrahedralChirality.UNDEFINED)
        {
          allIsDefined = false;
          Compound compR = comp.safeCopy();
          compR.getOrderedAtomList().get(index)
              .setTetrahedralChirality(TetrahedralChirality.R);
          Molecule newMolR = new Molecule(compR);
          newMolR.setProperties(molecule.getProperties());
          list.addAll(calculate(newMolR));

          Compound compS = comp.safeCopy();
          compS.getOrderedAtomList().get(index)
              .setTetrahedralChirality(TetrahedralChirality.S);
          Molecule newMolS = new Molecule(compS);
          newMolS.setProperties(molecule.getProperties());
          list.addAll(calculate(newMolS));

          break;
        }
        index++;
      }
      if (allIsDefined)
        list.add(molecule);
    }
    return list;
  }
}
```

Code 2: Core of the Chirality Enumerator node. The `ChiralatorCalculation` *class.*

Once generated all the stereoisomers, in order to distinguish them, a simple progressive index is added at the end of the name of the parent compound. The Code 3 reports the steps for this operation.

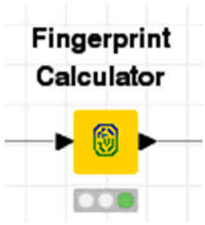
```
int index = 1;
for (Molecule outputMolecule : outputMolecules)
{
  if (outputMolecules.size() > 1) //
    outputMolecule.setName(molecule.getName() + "_" + (index++));
  calculateChiralityProperty(outputMolecule);
  MoleculeResult outputDataHolder = new MoleculeResult(outputMolecule);
  outputList.add(outputDataHolder);
}
return outputList;
```

Code 3: Index addition to the name of the new generate chiral molecules.

Moreover, if the checkbox is checked in the Dialog box, the code updates the chiral properties for each molecule. This step is needed because otherwise old chiral properties of the parent compound, if present, would be copied into the new generated molecules. For the code, see the APPENDIX 1.

The output displays a table which lists all the input molecules together with all their possible stereoisomers.

3.3.1.3 FINGERPRINT CALCULATOR NODE



3.3.1.3.1 Functional requirements:

Name:	**Calculate the Extended Connectivity Fingerprints of a series of input molecules**
Primary actor:	End user
Brief description:	The user connects the output port of an appropriate node (usually a molecule file reader) to the input port of the Fingerprint Calculator node. The Fingerprint Calculator node

	generates the opportune bitset for each molecule present in the input table. The fingerprints calculated with this node, which uses proprietary code implemented at Inte:Ligand, belong to the Extended Connectivity Fingerprints.
Trigger:	User drags the Fingerprint Calculator node into the KNIME workspace window.
Preconditions:	A preceding node that outputs table containing structural information of the ligands (2D or 3D) has been executed successfully.
Postconditions:	Table containing a copy of the input table with an additional column reporting the ECFP. The length of the fingerprint is depending on the options selected by the user in the initial Dialog box.

Normal Flow:

1. User connects the output port of an adequate preceding node with the input port of the Fingerprint Calculator node.
2. User opens the configuration dialog.
3. System generates and shows the dialog for configuring the node.
4. User selects the number of bits for the output fingerprint and specifies the number of iteration for the ECFP calculation.
5. User starts the execution of the node.
6. System indicates the finished execution.

Exceptions:

1a. The input table doesn't contain any column with structural information (SDF, MOL, MOL2, SMILES or CCT). The node displays an error message notifying the user of the problem.

4a. The number of bits for the output is negative or bigger than `MAXBITVALUE` (2048). The node displays an error message notifying the user of the problem.

4b. The number of iteration for the ECFP calculation is negative or bigger than 10. The node displays an error message notifying the user of the problem.

Priority:	High. It is a primary use case.
Frequency of Occurrence:	It has to be expected that some users require this functionality multiple times a day.

Assumptions:	It is assumed that the LigandScout code is able to generate the appropriate ECFP for a molecule.

Table 3-3: Functional requirements of Fingerprint Calculator node.

3.3.1.3.2 Usage and description:

To use the Fingerprint Calculator node, a suitable preceding node has to be connected to its input port. Mandatory is the presence, in the input table, of the structural information of at least one molecule (2D or 3D).

Once successfully connected to the previous node, the user opens the Dialog box (Figure 3-14) and chooses the number of bits composing the output fingerprint and the number of iteration used by the code to realise the ECFPs of the molecules.

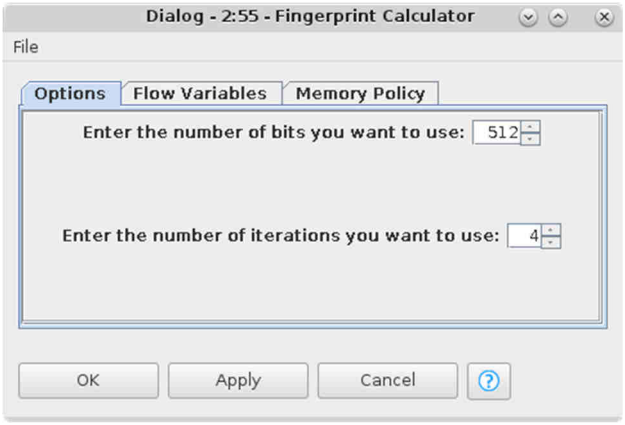


Figure 3-14: Dialog box of the Fingerprint Calculator node. Default values are reported.

While the lines to perform the calculation were already present in the LigandScout code, the interface to let the user to recall it was programmed. In Code 4 is reported the part of the `NodeModel` in which it reads the structural data from each single row of the input table and performs the calculation thanks to the newly programmed `calculateFingerprints` method (of which the code is reported in Code 5). After the calculation, the program continues creating and setting a bit vector as the fingerprint. Then,

it copies the input properties to the output row and adds a cell at the end of the row containing the bit vector.

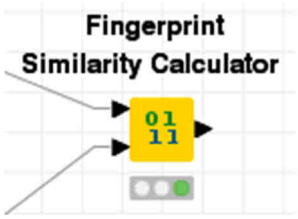
```
for (DataRow row : table)
    {
      exec.checkCanceled();
      exec.setProgress((double) currentRow / rowCount,
          " processing row " + currentRow);

      MoleculeDataHolder molDataHolder = molDataHolderFactory
          .getMolecule(row);

      BitSet bitSet = wrapper.calculateFingerprints(molDataHolder,
          settingsNumBits.getIntValue(), settingsNumIterations.getIntValue());
      DenseBitVectorCellFactory bitVectorCellFactory = new
          DenseBitVectorCellFactory(bitSet.length());

      for (int i = 0; i < bitSet.length(); i++)
        if (bitSet.get(i))
          bitVectorCellFactory.set(i);

      List<DataCell> dataCells = new ArrayList<DataCell>();

      for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
      {
        DataCell cell = row.getCell(i);
        dataCells.add(cell);
      }

      dataCells.add(bitVectorCellFactory.createDataCell());
      container.addRowToTable(
          new DefaultRow(RowKey.createRowKey(rowIndex++), dataCells));
      dataCells.clear();
      currentRow++;
    }
```

Code 4: Part of the code of the `FingerprintsNodeModel` *class.*

```
package com.inteligand.knime.fingerprints;

import java.util.BitSet;
import com.inteligand.ilib.base.fingerprint.SciTegicECFingerprint;
import com.inteligand.knime.wrapper.data.molecule.MoleculeDataHolder;

public class FingerprintsWrapper
{
  public BitSet calculateFingerprints(MoleculeDataHolder dataHolder,
      int numBits, int numIterations) throws Exception
  {
    SciTegicECFingerprint fingerprint = new SciTegicECFingerprint();

    fingerprint.includeMMFF94AtomType(false);

    fingerprint.setNumIterations(numIterations);
    fingerprint.setNumBits(numBits + 1);

    fingerprint.calculate(dataHolder.getMolecule());
    return fingerprint.getBits();
  }
}
```

Code 5: `FingerprintsWrapper` *class. This code invokes the proprietary LigandScout code to perform the ECFPs calculation.*

3.3.1.4 FINGERPRINT SIMILARITY CALCULATOR NODE



3.3.1.4.1 Functional requirements:

Name:	**Calculate the similarity, expressed as Tanimoto similarity score, of a set of molecules to a query compound**
Primary actor:	End user
Brief description:	The user uses a table containing one (or more) fingerprint query molecule(s) as input for port number 0. They proceed connecting to input port number 1 the database of fingerprint molecules in which to search. The Fingerprint Similarity node compares each molecule of the database with the query molecule selected by the user in the Dialog box among the molecules present in the input port number 0. The comparison of the fingerprints results in a score (Tanimoto similarity score) which is added as new column in the output table.
Trigger:	User drags the Fingerprint Similarity Calculator node into the KNIME workspace window.
Preconditions:	A preceding node that outputs table containing fingerprints has been executed successfully. The use of ECFPs is not mandatory, inasmuch each type of fingerprint can be used by the node to calculate the similarity score.
Postconditions:	Table containing a copy of the input table used for the port number 1, with an additional column reporting the Fingerprint Similarity Score to

	the query molecule used as input through port 0.
Normal Flow: 1. User connects the output port of an adequate preceding node with the input port number 0 of the Fingerprint Similarity Calculator node. This input contains the fingerprint for the query molecule. 2. User connects the output port of an adequate preceding node with the input port number 1 of the Fingerprint Similarity Calculator node. This input contains the fingerprint of each molecule in the database. 3. User opens the configuration dialog. 4. System generates and shows the dialog for configuring the node. 5. User selects the number of the row corresponding to the query molecule of the input table for the port number 0. 6. User selects the fingerprint to use for the calculation both for the query (input port 0) and the database (input port 1). 7. User starts the execution of the node. 8. System indicates the finished execution.	
Alternative Flows: 9. User inspects the similarity value calculated for the query molecule respect to itself after successful execution.	
Exceptions: 3a. The output port connected to one or both the input ports of the Fingerprint Similarity Calculator does not contain any fingerprint. This is checked through the presence of at least one `BitVectorValue` cell type. The node displays an error message notifying the user of the problem. 5a. The number of the specified row for the query molecule is negative or greater than `MAXBITVALUE` (2147483647). The node displays an error message notifying the user of the problem. 5b. The number of the specified row for the query molecule exceeds the number of molecules present in the input table for the port number 0 but is less than `MAXBITVALUE` (2147483647). The node takes as query the last row in the list.	
Priority:	High. It is a primary use case.
Frequency of Occurrence:	It has to be expected that some users require this functionality multiple times a day.
Assumptions:	It is assumed that the LigandScout code is able to calculate the Tanimoto similarity score between two molecules.

Table 3-4: Functional requirements of Fingerprint Similarity Calculator node.

3.3.1.4.2 Usage and description:

Basic requirement to use the Fingerprint Similarity Calculator node is the connection of its input ports to output of previous nodes containing at least one bit vector representing the fingerprint. The first port requires as input the query fingerprint, while the second input port has to be connected with the fingerprints database. It is not necessary for the query molecule to be in a table with a single row, inasmuch, as showed in Figure 3-15, in the Dialog box there is the possibility to choose which molecule use as query among the ones present at the port 0. Moreover, the Dialog box is used also to leave to the user the possibility to choose different types of fingerprint present in the input for both ports 0 and 1. Thanks to this feature, the use of the ECPSs calculated with the Fingerprint Calculator node is recommended but not mandatory, and multiple fingerprint properties present in the input can be easily managed.

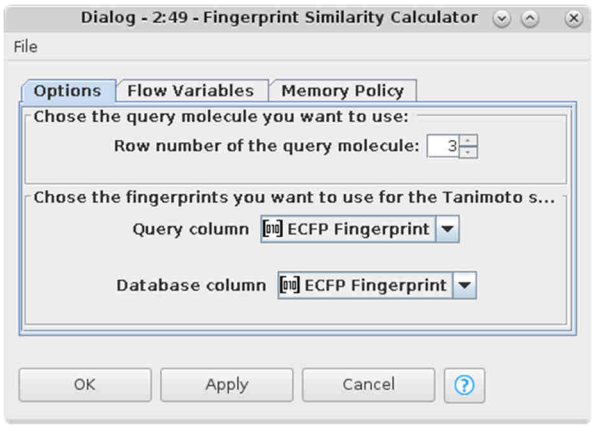


Figure 3-15: Dialog box of the Fingerprint Similarity Calculator node.

Once opportunely applied the Dialog box settings, the execution of the node can be started. It retrieves the query fingerprint from the port 0 from the row set by the user and then performs the calculation of the Tanimoto Similarity. To do this, it uses the `calcTanimotoSimilarity` method reported in Code 6.

```
public float calcTanimotoSimilarity(Fingerprint other)
{
  int thisNumSetBits = numSetBits();
  int otherNumSetBits = other.numSetBits();

  BitSet commonBits = (BitSet) bits.clone();
  commonBits.and(other.bits);

  int numCommonBits = commonBits.cardinality();

  return ((float) numCommonBits
      / (float) (thisNumSetBits + otherNumSetBits - numCommonBits));
}
```

Code 6: `calcTanimotoSoimilarity` *method from the LigandScout code.* `numSetBits` *refers to the query molecule and* `other.numSetBits` *refers to the molecule from the database. This method returns the value of the Tanimoto Similarity score.*

The Code 6 is invoked for each row present in the input 1, allowing the code to add the "Similarity Fingerprint Score" property to the output table and fill it with the similarity of that molecule to the query.

For a more complete overview of the code see the APPENDIX 1.

3.3.1.5 FINGERPRINT CLUSTERING NODE

3.3.1.5.1 Functional requirements:

Name:	**Cluster a set of molecules using fingerprints**
Primary actor:	End user
Brief description:	The user connects the output port of an appropriate node (usually the Fingerprint Calculator node) to the input port of the Fingerprint Clustering node. This node, exploiting the K-Mean algorithm implemented into the LigandScout

	code, clusters the input molecules in a specific number of clusters set by the user in the Dialog box. After successfully execution of the calculation, an additional column in the output table will be added, reporting the Cluster ID of each molecule.
Trigger:	User drags the Fingerprint Clustering node into the KNIME workspace window.
Preconditions:	A preceding node that outputs table containing fingerprints has been executed successfully. The use of ECFPs is not mandatory, inasmuch each type of fingerprint can be used by the node to cluster molecules.
Postconditions:	Table containing a copy of the input table with an additional column reporting the Cluster ID for each molecule.

Normal Flow:

1. User connects the output port of an adequate preceding node with the input port of the Fingerprint Clustering node.
2. User opens the configuration dialog.
3. System generates and shows the dialog for configuring the node.
4. User selects the fingerprint to use for the clustering calculation.
5. User selects the number of cluster to obtain.
6. User starts the execution of the node.
7. System indicates the finished execution

Exceptions:

3a. The output port connected to the input port of the Fingerprint Clustering node does not contain any fingerprint. This is checked through the presence of at least one `BitVectorValue` cell type. The node displays an error message notifying the user of the problem.

5a. The number of the clusters to obtain is negative or is greater than the molecules in the input table. The node displays an error message notifying the user of the problem.

Priority:	High. It is a primary use case.
Frequency of Occurrence:	It has to be expected that some users require this functionality multiple times a day.

Assumptions:	It is assumed that the K-Means algorithm in the LigandScout code is well implemented.

Table 3-5: Functional requirements of Fingerprint Clustering node.

3.3.1.5.2 Usage and description:

To use the Fingerprint Clustering node, a suitable preceding node has to be connected to its input port. Also in this case the presence in the input table of at least one Bit Vector column containing the fingerprints must be present. This column, with which the calculation is performed, is selected by the user in the Dialog box, together with the final number of clusters to obtain. Figure 3-16 shows the Fingerprint Clustering Node Dialog box.

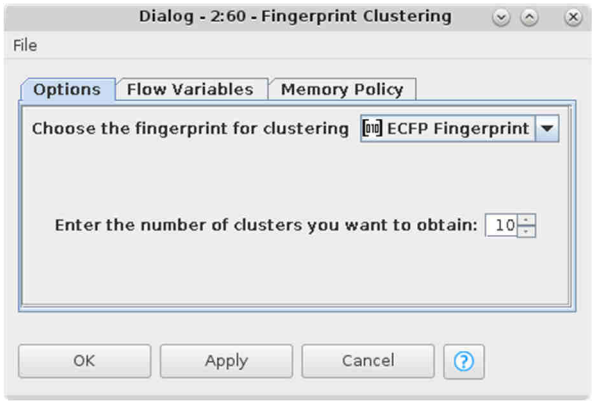


Figure 3-16: Dialog box of the Fingerprint Clustering node. Default values are reported.

The core of this node is the K-Means algorithm implemented in the proprietary Inte:Ligand code with the method `cluster` of the `KMeansClusterGenerator` class. The steps of the K-Means cluster generation are described in the dedicated paragraph.

The node invokes this method several times, inasmuch it is a non-deterministic algorithm. That brings two consequences:

- Changing the order of the starting data points the final result can change.
- It will “choose” by itself the final number of clusters.

While the first issue is an intrinsic limitation of the algorithm and cannot be addressed without exponentially increasing the calculation time to determine each time the best starting conditions, the second one is treated using a series of `while` and `for` loops one inside another.

Going into details of the code (Code 7), the first check is the number of clusters set by the user, which has to be smaller than the number of molecules in the input table. Afterwards, an *n* number of empty clusters equal to the value `settingNumCluster` are created, and the first *n* `fingerprintDataPoints` are added one per cluster. Then, the `cluster` method is invoked. Considering what mentioned above regarding the algorithm itself, some of the clusters sometimes can remain empty, and they are removed.

```
if (inputRowIndex < settingNumCluster
    .getIntValue()) { throw new InvalidSettingsException(
        "Please enter a number of maximum clusters less or equal to the
        molecules you have in the input table"); }

for (int i = 0; i < settingNumCluster.getIntValue(); i++)
{
  initClusters.add(new KMeansCluster(fingerprintDataPoints.get(i)));
}
KMeansClusterGenerator kmeansClusterGen = new KMeansClusterGenerator();
Collection<KMeansCluster> kmeansClusters = kmeansClusterGen
    .cluster(initClusters, fingerprintDataPoints);
for (KMeansCluster cluster : new ArrayList<>(kmeansClusters))
{
  if (cluster.getDataPoints().isEmpty())
  {
    kmeansClusters.remove(cluster);
  }
}
```

Code 7: `FingerprintClusteringNodeModel` *class. Frame of the code performing the initial clustering.*

In order to force the algorithm to obtain the desired number of final clusters, if this is not reached during the first process, the K-Means algorithm is recalled. But this time only to cluster the points belonging to the biggest cluster obtained during the first attempt, in order to have specifically two new clusters. This is made creating a sorted collection of clusters at the beginning of the `while` loop. The collection is sorted every

time the loop is executed as far as the `settingNumCluster` is reached, as reported in Code 8.

```
while (kmeansClusters.size() < settingNumCluster.getIntValue())
{
  List<KMeansCluster> biggestClusters = new ArrayList<>();
  for (KMeansCluster cluster : kmeansClusters)
    biggestClusters.add(cluster);
  Collections.sort(biggestClusters, new Comparator<KMeansCluster>()
  {
    public int compare(KMeansCluster cluster1, KMeansCluster cluster2)
    {
      return cluster2.getDataPoints().size() - cluster1.getDataPoints().size();
    }
  });
   Collection<KMeansCluster> newClusters = new ArrayList<>();
  for (KMeansCluster biggestCluster : biggestClusters)
  {
…
```

Code 8: Frame of the `FingerprintClusteringNodeModel` *in which the collection of clusters sorted by size is created.*

For the biggest cluster the same procedure seen in Code 7 is applied. However, to obtain specifically two new clusters from the biggest, all the three possibilities have to be considered:

- `newClusters`: 2. The initial biggest cluster is removed from the cluster collection and the two news were added.
- `newClusters` > 2. The two smallest clusters among the new clusters are merged together until two clusters are obtained. Than the initial biggest cluster is removed from the cluster collection and the new two were added.
- `newClusters` < 2. If this appends means that the K-Means algorithm is not able to further split in two the first element in the cluster collection. Thus, it passes to the second biggest cluster and repeats the process.

If none of the cluster in the collection can be further splitted in two, a manual procedure to divide the biggest cluster is applied as reported in Code 9.

```
if (newClusters.size() < 2)
{
  newClusters.clear();
  KMeansCluster biggestCluster = biggestClusters.iterator().next();

  KMeansCluster manuallySplittedBiggestCluster = new KMeansCluster(null);
  int halfDataOfBiggestCluster = biggestCluster.getDataPoints().size() / 2;
  int index = 0;
  for (KMeansDataPoint dataPoint : new ArrayList<>(
       biggestCluster.getDataPoints()))
  {
    KMeansCluster.swap(biggestCluster, manuallySplittedBiggestCluster,
        dataPoint);
    index++;
    if (index == halfDataOfBiggestCluster) //
      break;
    }

  newClusters.add(manuallySplittedBiggestCluster);
…
```

Code 9: Frame of the `FingerprintClusteringNodeModel` *in which the biggest cluster is manually splitted in two clusters because the K-Means algorithm is not able to do it.*

Once obtained the desired number of clusters, the software assigns the corresponding cluster ID to each data point using an `HashMap`, as reported in the Code 10.

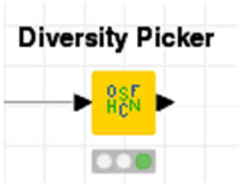
```
Map<Integer, Integer> indexToClusterIdMap = new HashMap<>();
Set<Integer> outputRowIndices = new HashSet<>();
int clusterIDIndex = 1;
for (KMeansCluster cluster : kmeansClusters)
{
  Collection<KMeansDataPoint> dataPoints = cluster.getDataPoints();
  for (KMeansDataPoint dataPoint : dataPoints)
  {
    outputRowIndices.add(dataPoint.getID());
    indexToClusterIdMap.put(dataPoint.getID(), clusterIDIndex);
  }
  clusterIDIndex++;
}
```

Code 10: Frame of the `FingerprintClusteringNodeModel` *in which the corresponding cluster ID is assigned to each data point.*

The output displays a table which lists all the input molecules together with the new “Cluster ID” property.

For a more complete overview of the code see the APPENDIX 1.

3.3.1.6 DIVERSITY PICKER NODE



3.3.1.6.1 Functional requirements:

Name:	**Choose the most diverse compounds using K-Means together with an ad-hoc custom algorithm for best picking molecules**
Primary actor:	End user
Brief description:	The user connects the output port of an appropriate node (usually the Fingerprint Calculator node) to the input port of the Diversity Picker. This node, exploiting the K-Mean algorithm implemented into the LigandScout code, performs an initial cluster analysis on the input data set and generates a number of cluster equal to the number of molecules set by the user in the Dialog box. Then, for each cluster, a dummy molecule is created. The dummy molecule is generated setting the bit to 1 if in the corresponding fingerprint position of at least 85% of molecules in that cluster is set to 1. The subsequent comparison, using the Tanimoto similarity score, between the dummy molecule and each molecule of its cluster makes possible the choice of the “most diverse” compound in that cluster. Collecting the most diverse molecule in each cluster results in the most diverse molecules present in the initial dataset.
Trigger:	User drags the Fingerprint Clustering node into the KNIME workspace window.
Preconditions:	A preceding node that outputs table containing fingerprints has been

	executed successfully. The use of ECFPs is not mandatory, inasmuch each type of fingerprint can be used by the node to cluster molecules.
Postconditions:	Table containing the most diverse molecules present in the initial dataset.
Normal Flow: 1. User connects the output port of an adequate preceding node with the input port of the Diversity Picker node. 2. User opens the configuration dialog. 3. System generates and shows the dialog for configuring the node. 4. User selects the fingerprint to use for the clustering calculation. 5. User selects the number of diverse molecule to obtain. 6. User starts the execution of the node. 7. System indicates the finished execution	
Exceptions: 3a. The output port connected to the input port of the Diversity Picker node does not contain any fingerprint. This is checked through the presence of at least one `BitVectorValue` cell type. The node displays an error message notifying the user of the problem. 5a. The number of diverse molecules to obtain is negative or is greater than the molecules in the input table. The node displays an error message notifying the user of the problem.	
Priority:	High. It is a primary use case.
Frequency of Occurrence:	It has to be expected that some users require this functionality multiple times a day.
Assumptions:	It is assumed that the K-Means algorithm in the LigandScout code is well implemented.

Table 3-6: Functional requirements of Diversity Picker node.

3.3.1.6.2 Usage and description:

To use the Diversity Picker node, a suitable preceding node has to be connected to its input port. This column containing the fingerprint with which perform the calculation is selected by the user in the Dialog box, together with the final number of diverse molecule to obtain.

This node is created with the aim to choose, between a given data set of molecule, the most diverse. Tacking in the account the need for speed, the fingerprints are used also here in this node. Considering this, the basic idea behind this node can be summarized in two passages:

- Cluster the molecules.
- Choose the “most diverse” molecule in each cluster.

While to accomplish the first step the code for the Clustering Fingerprint node can be used, to address the second one new software is necessary. The concept of “diverse” is something that can be translated as “most distant” in terms of similarity. But in a cluster of molecules the main problem to choose the most diverse is that something “to be distant from” is missing. To solve this issue, a “dummy” molecule for each cluster is created by the program. This one is basically a fingerprint with the same length of the ones of the molecules in the cluster, and is created setting the bit to 1 only if in that position a 1 is present in more than 85% of the fingerprints of the molecules composing that cluster. Once this fingerprint is created, the similarity of each molecule of the cluster to its dummy is also calculated. The program, at the end, returns a molecule for each cluster with the greater distance to the dummy molecule created for that cluster. The most important part of the code of the Diversity Picker node is the `pickBestDataPoint` method, which is reported in Code 11.

```
private KMeansDataPoint pickBestDataPoint(
    Collection<KMeansDataPoint> dataPoints, KMeansCluster cluster,
    long fingerprintLength)
{
  Fingerprint dummyFingerprint = new SciTegicECFingerprint();
  for (int i = 0; i < fingerprintLength; i++)
  {
    int numSet = 0;
    int numUnset = 0;
    for (KMeansDataPoint dataPoint : dataPoints)
    {
      Fingerprint fingerprint = ((FingerprintDataPoint) dataPoint)
          .getFingerprint();
      if (fingerprint.getBits().get(i))
        numSet++;
      else
        numUnset++;
    }
    float percentage = numSet / (numSet + numUnset);
    if (percentage > 0.85) dummyFingerprint.setBit(i);
  }

  float minSimilarityToDummy = 1;
  KMeansDataPoint lessSimilarDataPoint = null;
  for (KMeansDataPoint dataPoint : dataPoints)
  {
    Fingerprint fingerprint = ((FingerprintDataPoint) dataPoint)
        .getFingerprint();
    float similarityToDummy = dummyFingerprint
        .calcTanimotoSimilarity(fingerprint);
    if (similarityToDummy < minSimilarityToDummy)
    {
      minSimilarityToDummy = similarityToDummy;
      lessSimilarDataPoint = dataPoint;
    }
  }
  return lessSimilarDataPoint;
}
```

Code 11: `pickBestDataPoint` *method of the Diversity Picker node.*

The output displays a table which lists only the most diverse molecules present in the input data set, with a number equal to the one set by the user in the Dialog box.

For a more complete overview of the code see the APPENDIX 1.

3.4 References

1 TUFTS CENTER FOR THE STUDY OF DRUG DEVELOPMENT. *http://csdd.tufts.edu/news/complete_story/pr_tufts_csdd_2014_cost_study*. (Nov 2014).

2 Li, H; Zheng, M; Luo, X; Zhu, W; Jiang, H. Computational Approaches in Drug Discovery and Development. In *Wiley Encyclopedia of Chemical Biology*. Wiley, 2008.

3 Kumar, N; Hendriks, BS; Janes, KA; de Graaf, D; Lauffenburger, DA. Applying computational modeling to drug discovery and development. *Drug Discov Today*, 11, 17-18 (Nov 2006), 806-811.

4 Kapetanovic, IM. COMPUTER-AIDED DRUG DISCOVERY AND DEVELOPMENT (CADDD): in silico-chemico-biological approach. *Chem Biol Interact*, 171, 2 (Jan 2009), 165-176.

5 van de Waterbeemd, H and Gifford, E. ADMET in silico modelling: towards prediction paradise? *Nat Rev Drug Discov*, 2, 3 (Mar 2003), 192-204.

6 SWISS INSTITUTE OF BIOINFORMATICS. *http://www.click2drug.org/*. (Aug 2016).

7 Sliwoski, G; Kothiwale, S; Meiler, J; Lowe, EW. Computational Methods in Drug Discovery. *Pharmacol Rev*, 66, 1 (Jan 2014), 334–395.

8 Sastry, GM; Adzhigirey, M; Day, T; Annabhimoju, R; Sherman, W. Protein and ligand preparation: parameters, protocols, and influence on virtual screening enrichments. *J Comput Aided Mol Des*, 27, 3 (Mar 2013), 221-234.

9 KNIME. *https://www.knime.org*.

10 ORACLE. *https://www.java.com*. Accessed Sep 2016.

11 DATA MINING, DATA SCIENCE SOFTWARE POLL: KDNUGGETS 15TH ANNUAL ANALYTICS. *http://www.kdnuggets.com/2014/06/kdnuggets-annual-software-poll-rapidminer-continues-lead.html*. Accessed on Sep 2016.

12 Jovic, A; Brkic, K; Bogunovic, N. An overview of free software tools for general data. In *37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (2014), 1112-1117.

13 ECLIPSE. *https://eclipse.org*. Accessed Sep 2016.

14 Berthold, MR; Cebron, N; Dill, F et al. KNIME: The Konstanz Information Miner. In Preisach, C et al., eds., *Data Analysis, Machine Learning and Applications*. Springer Berlin Heidelberg, 2008.

15 Inte:Ligand. *http://www.inteligand.com*. Accessed Sep 2016.

16 Wolber, G and Langer, T. LigandScout: 3-D pharmacophores derived from protein-bound ligands and their use as virtual screening filters. *J Chem Inf Model*, 45, 1 (Jan-Feb 2005), 160-169.

17 Triballeau, N; Acher, F; Brabet, I; Pin, JP; Bertrand, HO. Virtual screening workflow development guided by the "receiver operating characteristic" curve approach. Application to high-throughput docking on metabotropic glutamate receptor subtype 4. *J. Med. Chem*, 48, 7 (Apr 2005), 2534-2547.

18 Swain, C. *http://www.macinchem.org/reviews/ligandscout-update.php*. LigandScout Review. 2015.

19 Todeschini, R and Consonni, V. *Handbook of Molecular Descriptors*. 2000.

20 Christie, BD; Leland, BA; Nourse, JG. Structure Searching in Chemical Databases by Direct Lookup Methods. *J. Chem. Inf. Comput. Sci.*, 33 (1993), 545-547.

21 Johnson, M and Maggiora, G, eds. *Concepts and Applications of Molecular Similarity*. Wiley: New York, 1990.

22 McGregor, MJ and Pallai, PV. Clustering of Large Databases of Compounds: Using the MDL 'keys' as Structural Descriptors. *J. Chem. Inf. Comput. Sci.*, 37 (1997), 443-448.

23 Breiman, L; Friedman, JH; Olshen, RA; Stone, CJ. *Classification and Regression Trees*. Wadsworth and Brooks/Cole: Monterey, 1984.

24 O'Boyle, N; Banck, M; James, C; Morley, C; Vandermeersch, T; Hutchison, G. Open Babel: An open chemical toolbox. *J Cheminform*, 3, 33 (Oct 2011).

25 Durant, JL; Leland, BA; Henry, DR; Nourse, JG. Reoptimization of MDL keys for use in drug discovery. *J Chem Inf Comput Sci*, 42, 6 (Nov 2002), 1273-1280.

26 Rogers, D and Hahn, M. Extended-connectivity fingerprints. *J Chem Inf Model*, 50, 5 (May 2010), 742-754.

27 O'Boyle, NM and Sayle, RA. Comparing structural fingerprints using a literature-based similarity benchmark. *J Cheminform*, 8, 36 (Jul 2016).

28 Morgan, HL. The Generation of a Unique Machine Description for Chemical Structures - A Technique Developed at Chemical Abstracts Service. *J Chem Doc*, 5 (1965), 107-112.

29 MacQueen, J. *Some methods for classification and analysis of multivariate observations*. Proc. Fifth Berkeley Symp. on Math. Statist. and Prob., Vol. 1. 1967.

30 Liu, M; Jiang, X; Kot, AC. Efficient fingerprint search based on database clustering. *Pattern Recognition*, 40 (2007), 1793-1803.

31 Khan, SS and Ahmad, A. Cluster center initialization algorithm for K-means clustering. *Pattern Recognition Letters*, 25, 11 (Aug 2004), 1293-1302.

3.5 Appendix 1: KNIME Code

3.5.1 STANDARD PROPERTY NODE, CHIRAL PROPERTIES CALCULATION

3.5.1.1 NUMBEROFCHIRALCENTERSR.JAVA

```
package com.inteligand.ilib.base.property;
import java.text.DecimalFormat;
import com.inteligand.ilib.base.Compound;
import com.inteligand.ilib.base.Molecule;
import com.inteligand.utils.RangeFloat;

public class NumberOfChiralCentersR extends NumericMolecularProperty
{
  public NumberOfChiralCentersR(Molecule mol)
  {
    super(mol);
    numberFormat = new DecimalFormat("0");
  }
  public NumberOfChiralCentersR(NumberOfChiralCentersR other)
  {
    super(other);
  }
  @Override
  public NumberOfChiralCentersR safeCopy()
  {
    return new NumberOfChiralCentersR(this);
  }
  @Override
  public void recalculate()
  {
    int count = 0;
    for (Compound comp : getMolecule().getCompoundsCopy())
    {
      comp.calcTetrahedralChiralCenters();
      count += comp.getNumberOfChiralCentersR();
    }
    value = count;
  }
  @Override
  public String getVerboseDescription()
  {
    return "Number of defined R chiral centers";
  }
  @Override
  public String getShortDescription()
  {
    return "# R Defined Chiral Centers";
  }
  @Override
  public String[] getOldDescriptions()
  {
    return null;
  }
  @Override
  public String getCategory()
  {
    return "Chirality";
  }
  @Override
  public RangeFloat getRecommendedRange()
  {
    return new RangeFloat(0, 2);
  }
  @Override
  public int getComputationalCostIndex()
  {
    return 6;
  }
}
```

To obtain NumberOfChiralCentersS.java change in the previous code R with S.

To obtain NumberOfChiralCentersUndefined.java change in the previous code R with Undefined.

3.5.2 CHIRALITY ENUMERATOR NODE

3.5.2.1 CHIRALATORNODEFACTORY.XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE knimeNode PUBLIC "-//UNIKN//DTD KNIME Node 2.0//EN"
"http://www.knime.org/Node.dtd">
<knimeNode icon="/icons/chiralator.png" type="Manipulator">
    <name>Chiralator</name>

    <shortDescription>Chiralator Calculator</shortDescription>

    <fullDescription>
        <intro>Generate different molecules for each undefined
stereocenter</intro>
    </fullDescription>

    <ports>
       <inPort index="0" name="Input molecule table">Table containing the
       molecules</inPort>
       <outPort index="0" name="Output molecule table">Table containing the
molecules
       with opportune chirality</outPort>
    </ports>

</knimeNode>
```

3.5.2.2 CHIRALATORNODEFACTORY.JAVA

```
package com.inteligand.knime.nodes.chiralator;

import org.knime.core.node.NodeDialogPane;
import org.knime.core.node.NodeFactory;
import org.knime.core.node.NodeView;

public class ChiralatorNodeFactory
        extends NodeFactory<ChiralatorNodeModel> {

    /**
     * {@inheritDoc}
     */
    @Override
    public ChiralatorNodeModel createNodeModel() {
        return new ChiralatorNodeModel();
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public int getNrNodeViews() {
        return 1;
    }

    /**
     * {@inheritDoc}
     */
```

```
    @Override
    public NodeView<ChiralatorNodeModel> createNodeView(final int viewIndex,
            final ChiralatorNodeModel nodeModel) {
        return new ChiralatorNodeView(nodeModel);
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public boolean hasDialog() {
        return true;
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public NodeDialogPane createNodeDialogPane() {
        return new ChiralatorNodeDialog();
    }
}
```

3.5.2.3 CHIRALATORNODEDIALOG.JAVA

```
package com.inteligand.knime.nodes.chiralator;

import org.knime.core.node.defaultnodesettings.DefaultNodeSettingsPane;
import org.knime.core.node.defaultnodesettings.DialogComponentBoolean;
import org.knime.core.node.defaultnodesettings.SettingsModelBoolean;

public class ChiralatorNodeDialog extends DefaultNodeSettingsPane
{
  protected ChiralatorNodeDialog()
  {
    super();

    createNewGroup("");
    addDialogComponent(
        new DialogComponentBoolean(
            new SettingsModelBoolean(ChiralatorNodeModel.SETTINGS_CHRIAL_PROPS,
true),
            "Calculate also chiral properties"));
  }
}
```

3.5.2.4 CHIRALATORNODEMODEL.JAVA

```
package com.inteligand.knime.nodes.chiralator;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import org.knime.core.data.DataCell;
import org.knime.core.data.DataColumnSpec;
import org.knime.core.data.DataColumnSpecCreator;
import org.knime.core.data.DataRow;
import org.knime.core.data.DataTableSpec;
import org.knime.core.data.RowKey;
import org.knime.core.data.def.DefaultRow;
import org.knime.core.data.def.StringCell;
import org.knime.core.node.BufferedDataContainer;
import org.knime.core.node.BufferedDataTable;
import org.knime.core.node.CanceledExecutionException;
import org.knime.core.node.ExecutionContext;
import org.knime.core.node.ExecutionMonitor;
import org.knime.core.node.InvalidSettingsException;
import org.knime.core.node.NodeLogger;
```

```
import org.knime.core.node.NodeModel;
import org.knime.core.node.NodeSettingsRO;
import org.knime.core.node.NodeSettingsWO;
import org.knime.core.node.defaultnodesettings.SettingsModelBoolean;
import com.inteligand.knime.chiralator.ChiralatorWrapper;
import com.inteligand.knime.data.cellvalues.CCTMoleculeDataCell;
import com.inteligand.knime.data.cellvalues.MoleculeDataHolderFactory;
import com.inteligand.knime.data.cellvalues.MoleculeDataTableDefaults;
import com.inteligand.knime.data.molecule.MoleculeDataHolder;
import com.inteligand.knime.data.molecule.MoleculeResult;
import com.inteligand.knime.utils.DataTableUtils;
public class ChiralatorNodeModel extends NodeModel
{
  static final String SETTINGS_CHRIAL_PROPS = "chiralityProps";
  private final SettingsModelBoolean settingChiralityProps = new
SettingsModelBoolean(
      SETTINGS_CHRIAL_PROPS, true);
  protected List<MoleculeResult> moleculeData = new ArrayList<MoleculeResult>();
  protected List<String> properties = new ArrayList<String>();
  private MoleculeDataHolderFactory molDataHolderFactory;
  private static final NodeLogger logger = NodeLogger
      .getLogger(ChiralatorNodeModel.class);
  protected ChiralatorNodeModel()
  {
    super(1, 1);
  }

  @Override
  protected BufferedDataTable[] execute(final BufferedDataTable[] inData,
      final ExecutionContext exec) throws Exception
  {
    logger.info("Node Model Stub... this is not yet implemented !");
    DataTableSpec inputTableSpec = inData[0].getDataTableSpec();
    properties = new ArrayList<>();
    for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
    {
      DataColumnSpec columnSpec = inputTableSpec.getColumnSpec(i);
      if (i != molDataHolderFactory.getDataColumnIndex() && !columnSpec
          .getName().equals(MoleculeDataTableDefaults.MOLECULE_NAME_COLUMN))
        properties.add(columnSpec.getName());
    }
    ChiralatorWrapper wrapper = new ChiralatorWrapper();
    wrapper.prepareProperties(properties,
        settingChiralityProps.getBooleanValue());
    int outputColNum = properties.size() + 2;
    DataColumnSpec[] outputColumnSpecs = new DataColumnSpec[outputColNum];
    int outputIndex = 0;
    outputColumnSpecs[outputIndex++] = new DataColumnSpecCreator(
        MoleculeDataTableDefaults.MOLECULE_NAME_COLUMN, StringCell.TYPE)
            .createSpec();
    outputColumnSpecs[outputIndex++] = new DataColumnSpecCreator(
        MoleculeDataTableDefaults.MOLECULE_DATA_COLUMN,
        CCTMoleculeDataCell.TYPE).createSpec();
    List<String> remainingProps = new ArrayList<>(properties);
    for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
    {
      DataColumnSpec columnSpec = inputTableSpec.getColumnSpec(i);
      if (properties.contains(columnSpec.getName()))
      {
        outputColumnSpecs[outputIndex++] = columnSpec;
        remainingProps.remove(columnSpec.getName());
      }
    }
    for (String property : remainingProps)
    {
      outputColumnSpecs[outputIndex++] = new DataColumnSpecCreator(property,
          DataTableUtils.getCellTypeByJavaType(wrapper.getType(property)))
              .createSpec();
    }
    DataTableSpec outputTableSpec = new DataTableSpec(outputColumnSpecs);
    BufferedDataContainer container = exec.createDataContainer(outputTableSpec);
    BufferedDataTable table = inData[0];
    int rowCount = table.getRowCount();
    int currentRow = 0;
    int rowIndex = 0;
    for (DataRow row : table)
```

```
    {
      exec.checkCanceled();
      exec.setProgress((double) currentRow / rowCount,
          " processing row " + currentRow);
      MoleculeDataHolder molDataHolder = molDataHolderFactory
          .createDataHolder(row);
      List<MoleculeResult> results = wrapper.calculate(molDataHolder);
      List<DataCell> dataCells = new ArrayList<DataCell>();
      for (int i = 0; i < results.size(); i++)
      {
        MoleculeResult result = results.get(i);
        dataCells.add(new StringCell(result.getName()));
        dataCells.add(new CCTMoleculeDataCell(result.getMoleculeData()));
        int currentColumnIndex = dataCells.size();
        for (String key : properties)
        {
          String property = result.getProperties().get(key);
          int spaceIndex = property.indexOf(' ');
          if (spaceIndex > 0)
          {
            property = property.substring(0, spaceIndex);
          }
          dataCells.add(DataTableUtils.getDataCellByTypeAndValue(property,
              container.getTableSpec().getColumnSpec(currentColumnIndex)));
          currentColumnIndex++;
        }
        container.addRowToTable(
            new DefaultRow(RowKey.createRowKey(rowIndex++), dataCells));
        dataCells.clear();
      }
    }
    container.close();
    return new BufferedDataTable[] { container.getTable()};
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void reset()
  {}

  /**
   * {@inheritDoc}
   */
  @Override
  protected DataTableSpec[] configure(final DataTableSpec[] inSpecs)
      throws InvalidSettingsException
  {
    molDataHolderFactory = DataTableUtils
        .getMoleculeDataHolderFactory(inSpecs[0], null);
    return new DataTableSpec[] { null};
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void saveSettingsTo(final NodeSettingsWO settings)
  {
    settingChiralityProps.saveSettingsTo(settings);
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void loadValidatedSettingsFrom(final NodeSettingsRO settings)
      throws InvalidSettingsException
  {
    settingChiralityProps.loadSettingsFrom(settings);
  }

  /**
   * {@inheritDoc}
   */
```

```
  @Override
  protected void validateSettings(final NodeSettingsRO settings)
      throws InvalidSettingsException
  {
    settingChiralityProps.validateSettings(settings);
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void loadInternals(final File internDir,
      final ExecutionMonitor exec)
          throws IOException, CanceledExecutionException
  {}

  /**
   * {@inheritDoc}
   */
  @Override
  protected void saveInternals(final File internDir,
      final ExecutionMonitor exec)
          throws IOException, CanceledExecutionException
  {}

}
```

3.5.2.5 CHIRALATORWRAPPER.JAVA

```
package com.inteligand.knime.chiralator;

import java.util.ArrayList;
import java.util.List;
import com.inteligand.ilib.base.Compound;
import com.inteligand.ilib.base.Molecule;
import com.inteligand.ilib.base.property.NumberOfChiralCentersR;
import com.inteligand.ilib.base.property.NumberOfChiralCentersS;
import com.inteligand.ilib.base.property.NumberOfChiralCentersUndefined;
import com.inteligand.ilib.base.stereo.ChiralatorCalculation;
import com.inteligand.knime.data.molecule.MoleculeDataHolder;
import com.inteligand.knime.data.molecule.MoleculeResult;
import
com.inteligand.knime.nodes.numericMolecularProperties.IlibKnimeStandardProperties

public class ChiralatorWrapper
{
  private final ChiralatorCalculation calculator;

  public ChiralatorWrapper()
  {
    calculator = new ChiralatorCalculation();
  }

  public List<MoleculeResult> calculate(MoleculeDataHolder molHolder)
      throws Exception
  {
    List<MoleculeResult> outputList = new ArrayList<>();
    Molecule molecule = molHolder.getMolecule();
    List<Molecule> outputMolecules = calculator.calculate(molecule);
    int index = 1;
    for (Molecule outputMolecule : outputMolecules)
    {

      if (outputMolecules.size() > 1) //
        outputMolecule.setName(molecule.getName() + "_" + (index++));
      calculateChiralityProperty(outputMolecule);
      MoleculeResult outputDataHolder = new MoleculeResult(outputMolecule);
      outputList.add(outputDataHolder);
    }

    return outputList;
  }
```

```
  private void calculateChiralityProperty(Molecule molecule)
  {
    int countR = 0;
    for (Compound comp : molecule.getCompoundsCopy())
      countR += comp.getNumberOfChiralCentersR();
    molecule.getProperties().put(
        new NumberOfChiralCentersR(molecule).getShortDescription(),
        Integer.toString(countR));

    int countS = 0;
    for (Compound comp : molecule.getCompoundsCopy())
      countS += comp.getNumberOfChiralCentersS();
    molecule.getProperties().put(
        new NumberOfChiralCentersS(molecule).getShortDescription(),
        Integer.toString(countS));

    int countU = 0;
    for (Compound comp : molecule.getCompoundsCopy())
      countU += comp.getNumberOfChiralCentersUndefined();
    molecule.getProperties().put(
        new NumberOfChiralCentersUndefined(molecule).getShortDescription(),
        Integer.toString(countU));
  }

  public void prepareProperties(List<String> properties,
      boolean considerChiralityProperties)
  {
    String chiralityR = new NumberOfChiralCentersR((Molecule) null)
        .getShortDescription();
    String chiralityS = new NumberOfChiralCentersS((Molecule) null)
        .getShortDescription();
    String chiralityUndefined = new NumberOfChiralCentersUndefined(
        (Molecule) null).getShortDescription();

    for (String property : new ArrayList<>(properties))
    {
      if (property.equals(chiralityR) || property.equals(chiralityS)
          || property.equals(chiralityUndefined))
        properties.remove(property);
    }
    if (considerChiralityProperties)
    {
      properties.add(chiralityR);
      properties.add(chiralityS);
      properties.add(chiralityUndefined);
    }
  }

  public Class<?> getType(String propertyName)
  {
    return IlibKnimeStandardProperties.getValueType(propertyName);
  }
}
```

3.5.2.6 CHIRALATORCALCULATION.JAVA

```
package com.inteligand.ilib.base.stereo;

import java.util.ArrayList;
import java.util.List;
import com.inteligand.ilib.base.Atom;
import com.inteligand.ilib.base.Compound;
import com.inteligand.ilib.base.Molecule;

public class ChiralatorCalculation
{
  public List<Molecule> calculate(Molecule molecule)
  {
    List<Molecule> list = new ArrayList<Molecule>();
    for (Compound comp : molecule.getCompoundsCopy())
    {
      comp.calcTetrahedralChiralCenters();
      int index = 0;
```

```
      boolean allIsDefined = true;

      for (Atom atom : comp.getOrderedAtomList())
      {
        if (atom.getTetrahedralChirality() == TetrahedralChirality.UNDEFINED)
        {
          allIsDefined = false;
          Compound compR = comp.safeCopy();
          compR.getOrderedAtomList().get(index)
              .setTetrahedralChirality(TetrahedralChirality.R);
          Molecule newMolR = new Molecule(compR);
          newMolR.setProperties(molecule.getProperties());
          list.addAll(calculate(newMolR));

          Compound compS = comp.safeCopy();
          compS.getOrderedAtomList().get(index)
              .setTetrahedralChirality(TetrahedralChirality.S);
          Molecule newMolS = new Molecule(compS);
          newMolS.setProperties(molecule.getProperties());
          list.addAll(calculate(newMolS));

          break;
        }
        index++;
      }
      if (allIsDefined)
        list.add(molecule);
    }
    return list;
  }
}
```

3.5.3 FINGERPRINT CALCULATOR NODE

3.5.3.1 FINGERPRINTSNODEFACTORY.XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE knimeNode PUBLIC "-//UNIKN//DTD KNIME Node 2.0//EN"
"http://www.knime.org/Node.dtd">
<knimeNode icon="/icons/fingerprints.png" type="Manipulator">
    <name>Fingerprint Calculator</name>

    <shortDescription>
        Extended Connectivity Fingerprint Calculator
    </shortDescription>

    <fullDescription>
        <intro>This nodes calculates "Extended Connectivity Fingerprint"s (ECFP)
for each input molecule </intro>
    </fullDescription>

    <ports>
       <inPort index="0" name="Input molecule table">Table containing the
       molecules</inPort>
       <outPort index="0" name="Output molecule table">Table containing the
molecules
       with calculated fingerprints</outPort>
    </ports>

</knimeNode>
```

3.5.3.2 FINGERPRINTSNODEFACTORY.JAVA

```
package com.inteligand.knime.nodes.fingerprints;

import org.knime.core.node.NodeDialogPane;
import org.knime.core.node.NodeFactory;
import org.knime.core.node.NodeView;
```

```
public class FingerprintsNodeFactory extends NodeFactory<FingerprintsNodeModel>
{

  /**
   * {@inheritDoc}
   */
  @Override
  public FingerprintsNodeModel createNodeModel()
  {
    return new FingerprintsNodeModel();
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public int getNrNodeViews()
  {
    return 0;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public NodeView<FingerprintsNodeModel> createNodeView(final int viewIndex,
      final FingerprintsNodeModel nodeModel)
  {
    return null;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public boolean hasDialog()
  {
    return true;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public NodeDialogPane createNodeDialogPane()
  {
    return new FingerprintsNodeDialog();
  }

}
```

3.5.3.3 FINGERPRINTSNODEDIALOG.JAVA

```
package com.inteligand.knime.nodes.fingerprints;

import org.knime.core.node.defaultnodesettings.DefaultNodeSettingsPane;
import org.knime.core.node.defaultnodesettings.DialogComponentBoolean;
import org.knime.core.node.defaultnodesettings.DialogComponentNumber;
import org.knime.core.node.defaultnodesettings.SettingsModelBoolean;
import org.knime.core.node.defaultnodesettings.SettingsModelInteger;

public class FingerprintsNodeDialog extends DefaultNodeSettingsPane
{
  protected FingerprintsNodeDialog()
  {
    super();

    createNewGroup("");
    addDialogComponent(new DialogComponentNumber(
        new SettingsModelInteger(FingerprintsNodeModel.SETTINGS_NUM_BITS, 512),
        "Enter the number of bits you want to use:", 1));

    addDialogComponent(
```

```
        new DialogComponentNumber(
            new SettingsModelInteger(
                FingerprintsNodeModel.SETTINGS_NUM_ITERATIONS, 4),
            "Enter the number of iterations you want to use:", 1));

  }
}
```

3.5.3.4 FINGERPRINTSNODEMODEL.JAVA

```
package com.inteligand.knime.nodes.fingerprints;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.BitSet;
import java.util.List;
import org.knime.core.data.DataCell;
import org.knime.core.data.DataColumnSpec;
import org.knime.core.data.DataColumnSpecCreator;
import org.knime.core.data.DataRow;
import org.knime.core.data.DataTableSpec;
import org.knime.core.data.RowKey;
import org.knime.core.data.def.DefaultRow;
import org.knime.core.data.vector.bitvector.DenseBitVectorCell;
import org.knime.core.data.vector.bitvector.DenseBitVectorCellFactory;
import org.knime.core.node.BufferedDataContainer;
import org.knime.core.node.BufferedDataTable;
import org.knime.core.node.CanceledExecutionException;
import org.knime.core.node.ExecutionContext;
import org.knime.core.node.ExecutionMonitor;
import org.knime.core.node.InvalidSettingsException;
import org.knime.core.node.NodeLogger;
import org.knime.core.node.NodeModel;
import org.knime.core.node.NodeSettingsRO;
import org.knime.core.node.NodeSettingsWO;
import org.knime.core.node.defaultnodesettings.SettingsModelIntegerBounded;
import com.inteligand.knime.data.cellvalues.molecule.MoleculeDataTableDefaults;
import com.inteligand.knime.data.cellvalues.molecule.MoleculeFromCellFactory;
import com.inteligand.knime.fingerprints.FingerprintsWrapper;
import com.inteligand.knime.utils.DataTableUtils;
import com.inteligand.knime.wrapper.data.molecule.MoleculeDataHolder;
import com.inteligand.knime.wrapper.data.molecule.MoleculeResult;

public class FingerprintsNodeModel extends NodeModel
{
  static final String SETTINGS_NUM_BITS = "numBits";
  static final String SETTINGS_NUM_ITERATIONS = "numIterations";
  private final SettingsModelIntegerBounded settingsNumBits = new
SettingsModelIntegerBounded(
      SETTINGS_NUM_BITS, 512, 1, 2048);
  private final SettingsModelIntegerBounded settingsNumIterations = new
SettingsModelIntegerBounded(
      SETTINGS_NUM_ITERATIONS, 4, 1, 10);
  protected List<MoleculeResult> moleculeData = new ArrayList<MoleculeResult>();
  protected List<String> properties = new ArrayList<String>();
  private MoleculeFromCellFactory molDataHolderFactory;

  private static final NodeLogger logger = NodeLogger
      .getLogger(FingerprintsNodeModel.class);
  protected FingerprintsNodeModel()
  {
    super(1, 1);
  }

public BitSet calculateFingerprints(MoleculeDataHolder dataHolder,
    int numBits, int numIterations) throws Exception
{
  SciTegicECFingerprint fingerprint = new SciTegicECFingerprint();
  fingerprint.includeMMFF94AtomType(false);
  fingerprint.setNumIterations(numIterations);
  fingerprint.setNumBits(numBits + 1);
  fingerprint.calculate(dataHolder.getMolecule());
```

```
  return fingerprint.getBits();
}

  @Override
  protected BufferedDataTable[] execute(final BufferedDataTable[] inData,
      final ExecutionContext exec) throws Exception
  {
    DataTableSpec inputTableSpec = inData[0].getDataTableSpec();
    FingerprintsWrapper wrapper = new FingerprintsWrapper();
    DataColumnSpec[] outputColumnSpecs = new DataColumnSpec[inputTableSpec
        .getNumColumns() + 1];
    for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
    {
      DataColumnSpec columnSpec = inputTableSpec.getColumnSpec(i);
      outputColumnSpecs[i] = columnSpec;
    }
    int currentCol = inputTableSpec.getNumColumns();
    outputColumnSpecs[currentCol] = new DataColumnSpecCreator(
        MoleculeDataTableDefaults.FINGERPRINTS_COLUMN, DenseBitVectorCell.TYPE)
            .createSpec();
    DataTableSpec outputTableSpec = new DataTableSpec(outputColumnSpecs);
    BufferedDataContainer container = exec.createDataContainer(outputTableSpec);
    BufferedDataTable table = inData[0];
    int rowCount = table.getRowCount();
    int currentRow = 0;
    int rowIndex = 1;
    for (DataRow row : table)
    {
      exec.checkCanceled();
      exec.setProgress((double) currentRow / rowCount,
          " processing row " + currentRow);
      MoleculeDataHolder molDataHolder = molDataHolderFactory
          .getMolecule(row);
      BitSet bitSet = wrapper.calculateFingerprints(molDataHolder,
          settingsNumBits.getIntValue(), settingsNumIterations.getIntValue());
      DenseBitVectorCellFactory bitVectorCellFactory = new
DenseBitVectorCellFactory(
          bitSet.length());
      for (int i = 0; i < bitSet.length(); i++)
        if (bitSet.get(i)) //
          bitVectorCellFactory.set(i);
      List<DataCell> dataCells = new ArrayList<DataCell>();
      for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
      {
        DataCell cell = row.getCell(i);
        dataCells.add(cell);
      }
      dataCells.add(bitVectorCellFactory.createDataCell());
      container.addRowToTable(
          new DefaultRow(RowKey.createRowKey(rowIndex++), dataCells));
      dataCells.clear();
      currentRow++;
    }
    container.close();
    return new BufferedDataTable[] { container.getTable()};
  }
  @Override
  protected void reset()
  {}
  @Override
  protected DataTableSpec[] configure(final DataTableSpec[] inSpecs)
      throws InvalidSettingsException
  {
    molDataHolderFactory = DataTableUtils
        .getMoleculeDataHolderFactory(inSpecs[0], null);
    return new DataTableSpec[] { null};
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void saveSettingsTo(final NodeSettingsWO settings)
  {
    settingsNumBits.saveSettingsTo(settings);
    settingsNumIterations.saveSettingsTo(settings);
```

```
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void loadValidatedSettingsFrom(final NodeSettingsRO settings)
      throws InvalidSettingsException
  {
    settingsNumBits.loadSettingsFrom(settings);
    settingsNumIterations.loadSettingsFrom(settings);
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void validateSettings(final NodeSettingsRO settings)
      throws InvalidSettingsException
  {
    settingsNumBits.validateSettings(settings);
    settingsNumIterations.validateSettings(settings);
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void loadInternals(final File internDir,
      final ExecutionMonitor exec)
          throws IOException, CanceledExecutionException
  {}

  /**
   * {@inheritDoc}
   */
  @Override
  protected void saveInternals(final File internDir,
      final ExecutionMonitor exec)
          throws IOException, CanceledExecutionException{}
}
```

3.5.4 FINGERPRINT SIMILARITY CALCULATION NODE

3.5.4.1 FINGERPRINTSIMILARITYNODEFACTORY.XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE knimeNode PUBLIC "-//UNIKN//DTD KNIME Node 2.0//EN"
"http://www.knime.org/Node.dtd">
<knimeNode icon="/icons/fingerprintsimilarity.png" type="Manipulator">
    <name>FingerprintSimilarity</name>

    <shortDescription>
        Fingerprint Similarity Calculator
    </shortDescription>

    <fullDescription>
        <intro>Calculate Similarity Fingerprints for a database respect to a
query
        molecule</intro>
    </fullDescription>

    <ports>
        <inPort index="0" name="Input molecule table">Table containing the query
        molecules</inPort>
        <inPort index="1" name="Input database table">Database containing the
molecules
        to be compared</inPort>
        <outPort index="0" name="Output molecule table">Table containing the
database
        with Tanimoto similarity score</outPort>
    </ports>
```

```
</knimeNode>
```

3.5.4.2 FINGERPRINTSIMILARITYNODEFACTORY.JAVA

```
package com.inteligand.knime.nodes.fingerprintsimilarity;

import org.knime.core.node.NodeDialogPane;
import org.knime.core.node.NodeFactory;
import org.knime.core.node.NodeView;

public class FingerprintSimilarityNodeFactory
    extends
    NodeFactory<FingerprintSimilarityNodeModel>
{

  /**
   * {@inheritDoc}
   */
  @Override
  public FingerprintSimilarityNodeModel createNodeModel()
  {
    return new FingerprintSimilarityNodeModel();
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public int getNrNodeViews()
  {
    return 0;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public NodeView<FingerprintSimilarityNodeModel> createNodeView(
      final int viewIndex, final FingerprintSimilarityNodeModel nodeModel)
  {
    return null;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public boolean hasDialog()
  {
    return true;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public NodeDialogPane createNodeDialogPane()
  {
    return new FingerprintSimilarityNodeDialog();
  }

}
```

3.5.4.3 FINGERPRINTCHIRALITYNODEMODEL.JAVA

```
package com.inteligand.knime.nodes.fingerprintsimilarity;

import java.io.File;
import java.io.IOException;
```

```
import java.util.ArrayList;
import java.util.List;
import org.knime.core.data.DataCell;
import org.knime.core.data.DataColumnSpec;
import org.knime.core.data.DataColumnSpecCreator;
import org.knime.core.data.DataRow;
import org.knime.core.data.DataTableSpec;
import org.knime.core.data.RowKey;
import org.knime.core.data.container.CloseableRowIterator;
import org.knime.core.data.def.DefaultRow;
import org.knime.core.data.def.DoubleCell;
import org.knime.core.data.vector.bitvector.BitVectorValue;
import org.knime.core.node.BufferedDataContainer;
import org.knime.core.node.BufferedDataTable;
import org.knime.core.node.CanceledExecutionException;
import org.knime.core.node.ExecutionContext;
import org.knime.core.node.ExecutionMonitor;
import org.knime.core.node.InvalidSettingsException;
import org.knime.core.node.NodeModel;
import org.knime.core.node.NodeSettingsRO;
import org.knime.core.node.NodeSettingsWO;
import org.knime.core.node.defaultnodesettings.SettingsModelIntegerBounded;
import org.knime.core.node.defaultnodesettings.SettingsModelString;

import com.inteligand.ilib.base.fingerprint.Fingerprint;
import com.inteligand.ilib.base.fingerprint.SciTegicECFingerprint;
import com.inteligand.knime.data.cellvalues.molecule.MoleculeDataTableDefaults;
import com.inteligand.knime.utils.DataTableUtils;

public class FingerprintSimilarityNodeModel extends NodeModel
{
  static final String QUERYCOLSEL = "selColQuery";
  static final String MOLCOLSEL = "selColMol";
  static final String QUERYROWSEL = "selRowForQuery";
  private final SettingsModelString colSelectionQuery = new SettingsModelString(
      QUERYCOLSEL, MoleculeDataTableDefaults.FINGERPRINTS_COLUMN);
  private final SettingsModelString colSelectionMolecule = new
SettingsModelString(
      MOLCOLSEL, MoleculeDataTableDefaults.FINGERPRINTS_COLUMN);

  private final SettingsModelIntegerBounded rowSelectionQuery = new
SettingsModelIntegerBounded(
      QUERYROWSEL, 1, 1, Integer.MAX_VALUE);

  protected FingerprintSimilarityNodeModel()
  {
    super(2, 1);
  }

  @Override
  protected BufferedDataTable[] execute(final BufferedDataTable[] inData,
      final ExecutionContext exec) throws Exception
  {
    String fingerprintColNameQuery = colSelectionQuery.getStringValue();
    int indexFingerprintsColQuery = inData[0].getDataTableSpec()
        .findColumnIndex(fingerprintColNameQuery);
    int queryRow = rowSelectionQuery.getIntValue();
    DataRow rowQuery = null;
    CloseableRowIterator iterator = inData[0].iterator();
    for (int i = 1; iterator.hasNext(); i++)
    {
      rowQuery = iterator.next();
      if (i == queryRow) break;
    }

    BitVectorValue queryFingerprintValue = (BitVectorValue) rowQuery
        .getCell(indexFingerprintsColQuery);
    SciTegicECFingerprint queryFingerprint = new SciTegicECFingerprint();
    for (int i = 0; i < queryFingerprintValue.length(); i++)
    {
      if (queryFingerprintValue.get(i)) queryFingerprint.setBit(i);
    }

    DataTableSpec inputTableSpec = inData[1].getDataTableSpec();
    DataColumnSpec[] outputColumnSpecs = new DataColumnSpec[inputTableSpec
        .getNumColumns() + 1];
```

```
  for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
  {
    DataColumnSpec columnSpec = inputTableSpec.getColumnSpec(i);
    outputColumnSpecs[i] = columnSpec;
  }
  int currentCol = inputTableSpec.getNumColumns();
  outputColumnSpecs[currentCol] = new DataColumnSpecCreator(
      MoleculeDataTableDefaults.FINGERPRINTS_SIMILARITY, DoubleCell.TYPE)
          .createSpec();
  DataTableSpec outputTableSpec = new DataTableSpec(outputColumnSpecs);
  BufferedDataContainer container = exec.createDataContainer(outputTableSpec);
  BufferedDataTable table = inData[1];
  String fingerprintColNameMol = colSelectionMolecule.getStringValue();;
  int indexFingerprintsColMol = inData[1].getDataTableSpec()
      .findColumnIndex(fingerprintColNameMol);
  long rowCount = table.size();
  int currentRow = 0;
  long rowIndex = 1;
  for (DataRow rowMol : table)
  {
    exec.checkCanceled();
    exec.setProgress((double) currentRow / rowCount,
        " processing row " + currentRow);
    BitVectorValue molFingerprintValue = (BitVectorValue) rowMol
        .getCell(indexFingerprintsColMol);
    Fingerprint molFingerprint = new SciTegicECFingerprint();
    for (int i = 0; i < molFingerprintValue.length(); i++)
    {
      if (molFingerprintValue.get(i))
      {
        molFingerprint.setBit(i);
      }
    }

    Float fingerprinSimilarityScore = queryFingerprint
        .calcTanimotoSimilarity(molFingerprint);
    List<DataCell> dataCells = new ArrayList<DataCell>();
    for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
    {
      DataCell cell = rowMol.getCell(i);
      dataCells.add(cell);
    }
    dataCells.add(new DoubleCell(fingerprinSimilarityScore));
    container.addRowToTable(
        new DefaultRow(RowKey.createRowKey(rowIndex), dataCells));
    dataCells.clear();
    currentRow++;
    rowIndex++;
  }
  container.close();
  return new BufferedDataTable[] { container.getTable()};

}

/**
 * {@inheritDoc}
 */
@Override
protected void reset()
{}

/**
 * {@inheritDoc}
 */
@Override
protected DataTableSpec[] configure(final DataTableSpec[] inSpecs)
    throws InvalidSettingsException
{
  String fingerprintColNameQuery = colSelectionQuery.getStringValue();
  if (inSpecs[0].findColumnIndex(
      fingerprintColNameQuery) < 0) { throw new InvalidSettingsException(
          "Could not find fingerprint column in query molecule table!"); } ;

  String fingerprintColNameMol = colSelectionMolecule.getStringValue();;
  if (inSpecs[1].findColumnIndex(
      fingerprintColNameMol) < 0) { throw new InvalidSettingsException(
```

```
            "Could not find fingerprint column in database molecule table!"); } ;

    DataTableUtils.getMoleculeDataHolderFactory(inSpecs[0], null);
    return new DataTableSpec[] { null};
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void saveSettingsTo(final NodeSettingsWO settings)
  {
    colSelectionQuery.saveSettingsTo(settings);
    colSelectionMolecule.saveSettingsTo(settings);
    rowSelectionQuery.saveSettingsTo(settings);
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void loadValidatedSettingsFrom(final NodeSettingsRO settings)
      throws InvalidSettingsException
  {
    colSelectionQuery.loadSettingsFrom(settings);
    colSelectionMolecule.loadSettingsFrom(settings);
    rowSelectionQuery.loadSettingsFrom(settings);
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void validateSettings(final NodeSettingsRO settings)
      throws InvalidSettingsException
  {
    colSelectionQuery.validateSettings(settings);
    colSelectionMolecule.validateSettings(settings);
    rowSelectionQuery.validateSettings(settings);
  }

  /**
   * {@inheritDoc}
   */
  @Override
  protected void loadInternals(final File internDir,
      final ExecutionMonitor exec)
      throws IOException, CanceledExecutionException
  {}

  /**
   * {@inheritDoc}
   */
  @Override
  protected void saveInternals(final File internDir,
      final ExecutionMonitor exec)
      throws IOException, CanceledExecutionException
  {}

}
```

3.5.5 FINGERPRINT CLUSTERING NODE

3.5.5.1 FINGERPRINTCLUSTERINGNODEFACTORY.XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE knimeNode PUBLIC "-//UNIKN//DTD KNIME Node 2.0//EN"
"http://www.knime.org/Node.dtd">
<knimeNode icon="/icons/fingerprintclustering.png" type="Manipulator">
       <name>Fingerprint Clustering</name>

       <shortDescription>
```

```
                Fingerprint Clustering
        </shortDescription>

        <fullDescription>
                <intro>The node clusters molecules from input table using kmeans
                        algorithm. It uses a not deterministic algorithm, so
                        results could be slightly different when performing
                        different running processes.
                </intro>
        </fullDescription>

        <ports>
                <inPort index="0" name="Input molecule table">Table containing the
input
                molecules
                </inPort>
                <outPort index="0" name="Output molecule table">Table containing
the
                input molecules
                        with an additional column containing the Cluster ID.
                </outPort>
        </ports>

</knimeNode>
```

3.5.5.2 FINGERPRINTCLUSTERINGNODEFACTORY.JAVA

```
package com.inteligand.knime.nodes.fingerprintclustering;

import org.knime.core.node.NodeDialogPane;
import org.knime.core.node.NodeFactory;
import org.knime.core.node.NodeView;

public class FingerprintClusteringNodeFactory extends
    NodeFactory<FingerprintClusteringNodeModel>
{

  /**
   * {@inheritDoc}
   */
  @Override
  public FingerprintClusteringNodeModel createNodeModel()
  {
    return new FingerprintClusteringNodeModel();
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public int getNrNodeViews()
  {
    return 0;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public NodeView<FingerprintClusteringNodeModel> createNodeView(
      final int viewIndex, final FingerprintClusteringNodeModel nodeModel)
  {
    return null;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public boolean hasDialog()
  {
    return true;
  }
```

```
  /**
   * {@inheritDoc}
   */
  @Override
  public NodeDialogPane createNodeDialogPane()
  {
    return new FingerprintClusteringNodeDialog();
  }
}
```

3.5.5.3 FINGERPRINTCLUSTERINGNODEDIALOG.JAVA

```
package com.inteligand.knime.nodes.fingerprintclustering;

import org.knime.core.data.vector.bitvector.BitVectorValue;
import org.knime.core.node.defaultnodesettings.DefaultNodeSettingsPane;
import
org.knime.core.node.defaultnodesettings.DialogComponentColumnNameSelection;
import org.knime.core.node.defaultnodesettings.DialogComponentNumber;
import org.knime.core.node.defaultnodesettings.SettingsModelInteger;
import org.knime.core.node.defaultnodesettings.SettingsModelString;

public class FingerprintClusteringNodeDialog extends DefaultNodeSettingsPane
{

  protected FingerprintClusteringNodeDialog()
  {
    super();
    createNewGroup("");

    addDialogComponent(new DialogComponentColumnNameSelection(
        new SettingsModelString(FingerprintClusteringNodeModel.COLSEL, ""),
        "Choose the fingerprint for clustering ", 0, true,
        BitVectorValue.class));

    addDialogComponent(new DialogComponentNumber(
        new
SettingsModelInteger(FingerprintClusteringNodeModel.SETTINGSNUMBERCLUSTER,
            10),
        "Enter the number of clusters you want to obtain:", 1));
  }
}
```

3.5.5.4 FINGERPRINTCLUSTERINGNODEMODEL.JAVA

```
package com.inteligand.knime.nodes.fingerprintclustering;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Collections;
import java.util.Comparator;
import java.util.HashMap;
import java.util.HashSet;
import java.util.List;
import java.util.Map;
import java.util.Set;
import org.knime.core.data.DataCell;
import org.knime.core.data.DataColumnSpec;
import org.knime.core.data.DataColumnSpecCreator;
import org.knime.core.data.DataRow;
import org.knime.core.data.DataTableSpec;
import org.knime.core.data.RowKey;
import org.knime.core.data.def.DefaultRow;
import org.knime.core.data.def.IntCell;
import org.knime.core.data.vector.bitvector.BitVectorValue;
import org.knime.core.node.BufferedDataContainer;
import org.knime.core.node.BufferedDataTable;
```

```
import org.knime.core.node.CanceledExecutionException;
import org.knime.core.node.ExecutionContext;
import org.knime.core.node.ExecutionMonitor;
import org.knime.core.node.InvalidSettingsException;
import org.knime.core.node.NodeModel;
import org.knime.core.node.NodeSettingsRO;
import org.knime.core.node.NodeSettingsWO;
import org.knime.core.node.defaultnodesettings.SettingsModelIntegerBounded;
import org.knime.core.node.defaultnodesettings.SettingsModelString;
import com.inteligand.ilib.base.fingerprint.Fingerprint;
import com.inteligand.ilib.base.fingerprint.SciTegicECFingerprint;
import com.inteligand.ilib.clustering.KMeansCluster;
import com.inteligand.ilib.clustering.KMeansClusterGenerator;
import com.inteligand.ilib.clustering.KMeansDataPoint;
import com.inteligand.knime.data.cellvalues.molecule.MoleculeDataTableDefaults;
import com.inteligand.knime.data.cellvalues.molecule.MoleculeFromCellFactory;
import com.inteligand.knime.utils.DataTableUtils;

public class FingerprintClusteringNodeModel extends NodeModel
{
  static final String COLSEL = "selCol";
  static final String SETTINGSNUMBERCLUSTER = "numberOfClusters";
  private final SettingsModelString colSelection = new SettingsModelString(
      COLSEL, MoleculeDataTableDefaults.FINGERPRINTS_COLUMN);
  private final SettingsModelIntegerBounded settingNumCluster = new
SettingsModelIntegerBounded(
      SETTINGSNUMBERCLUSTER, 10, 1, Integer.MAX_VALUE);
  private MoleculeFromCellFactory molDataHolderFactory;

  protected FingerprintClusteringNodeModel()
  {
    super(1, 1);
  }

  @Override
  protected BufferedDataTable[] execute(final BufferedDataTable[] inData,
      final ExecutionContext exec) throws Exception
  {
    DataTableSpec inputTableSpec = inData[0].getDataTableSpec();
    DataColumnSpec[] outputColumnSpecs = new DataColumnSpec[inputTableSpec
        .getNumColumns() + 1];
    for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
    {
      DataColumnSpec columnSpec = inputTableSpec.getColumnSpec(i);
      outputColumnSpecs[i] = columnSpec;
    }
    int currentCol = inputTableSpec.getNumColumns();
    outputColumnSpecs[currentCol] = new DataColumnSpecCreator("Cluster ID",
        IntCell.TYPE).createSpec();
    DataTableSpec outputTableSpec = new DataTableSpec(outputColumnSpecs);
    BufferedDataContainer container = exec.createDataContainer(outputTableSpec);
    BufferedDataTable table = inData[0];
    String fingerprintColName = colSelection.getStringValue();
    int indexFingerprintsCol = inData[0].getDataTableSpec()
        .findColumnIndex(fingerprintColName);
    int rowCount = table.getRowCount();
    int currentRow = 0;
    int inputRowIndex = 0;
    long fingerprintLength = 0;

    List<KMeansDataPoint> fingerprintDataPoints = new
ArrayList<KMeansDataPoint>();
    List<KMeansCluster> initClusters = new ArrayList<KMeansCluster>();
    List<Fingerprint> molFingerprints = new ArrayList<Fingerprint>();
    for (DataRow row : table)
    {
      exec.checkCanceled();
      exec.setProgress((double) currentRow / (rowCount * 2),
          " processing row " + currentRow);
      BitVectorValue molFingerprintValue = (BitVectorValue) row
          .getCell(indexFingerprintsCol);
      fingerprintLength = molFingerprintValue.length();
      Fingerprint molFingerprint = new SciTegicECFingerprint();
      for (int i = 0; i < fingerprintLength; i++)
      {
        if (molFingerprintValue.get(i)) //
```

```
          molFingerprint.setBit(i);
        }
        fingerprintDataPoints
            .add(new FingerprintDataPoint(inputRowIndex, molFingerprint));
        molFingerprints.add(molFingerprint);
        inputRowIndex++;
      }
      if (inputRowIndex < settingNumCluster
          .getIntValue()) { throw new InvalidSettingsException(
              "Please enter a number of maximum clusters less or equal to the
molecules
                  you have in the input table"); }

      for (int i = 0; i < settingNumCluster.getIntValue(); i++)
      {
        initClusters.add(new KMeansCluster(fingerprintDataPoints.get(i)));
      }
      KMeansClusterGenerator kmeansClusterGen = new KMeansClusterGenerator();
      Collection<KMeansCluster> kmeansClusters = kmeansClusterGen
          .cluster(initClusters, fingerprintDataPoints);
      for (KMeansCluster cluster : new ArrayList<>(kmeansClusters))
      {
        if (cluster.getDataPoints().isEmpty())
        {
          kmeansClusters.remove(cluster);
        }
      }
      while (kmeansClusters.size() < settingNumCluster.getIntValue())
      {
        List<KMeansCluster> biggestClusters = new ArrayList<>();
        for (KMeansCluster cluster : kmeansClusters)
          biggestClusters.add(cluster);
        Collections.sort(biggestClusters, new Comparator<KMeansCluster>()
        {
          public int compare(KMeansCluster cluster1, KMeansCluster cluster2)
          {
            return cluster2.getDataPoints().size()
                - cluster1.getDataPoints().size();
          }
        });
        Collection<KMeansCluster> newClusters = new ArrayList<>();
        for (KMeansCluster biggestCluster : biggestClusters)
        {
          if (biggestCluster.getDataPoints().size() == 1) //
            break;

          int numInitialClusters = 1;
          while (newClusters.size() < 2
              && numInitialClusters < fingerprintDataPoints.size()
              && numInitialClusters < 10)
          {
            newClusters.clear();
            numInitialClusters++;
            initClusters = new ArrayList<>();
            fingerprintDataPoints = new ArrayList<>(
                biggestCluster.getDataPoints());
            for (KMeansDataPoint dataPoint : fingerprintDataPoints)
              dataPoint.setCluster(null);
            for (int i = 0; i < numInitialClusters; i++)
            {
              initClusters.add(new KMeansCluster(fingerprintDataPoints.get(i)));
            }
            kmeansClusterGen = new KMeansClusterGenerator();
            newClusters = kmeansClusterGen.cluster(initClusters,
                fingerprintDataPoints);
            for (KMeansCluster cluster : new ArrayList<>(newClusters))
            {
              if (cluster.getDataPoints().isEmpty()) //
                newClusters.remove(cluster);
            }
            while (newClusters.size() > 2)
            {
              KMeansCluster smallestCLuster1 = null;
              KMeansCluster smallestCLuster2 = null;
              int minNumberDataPoints = Integer.MAX_VALUE;
              for (KMeansCluster cluster : newClusters)
```

```
          {
            int numDataPointsInCluster = cluster.getDataPoints().size();
            if (numDataPointsInCluster < minNumberDataPoints)
            {
              minNumberDataPoints = numDataPointsInCluster;
              smallestCLuster1 = cluster;
            }
          }
          newClusters.remove(smallestCLuster1);
          minNumberDataPoints = Integer.MAX_VALUE;
          for (KMeansCluster cluster : newClusters)
          {
            int numDataPointsInCluster = cluster.getDataPoints().size();
            if (numDataPointsInCluster < minNumberDataPoints)
            {
              minNumberDataPoints = numDataPointsInCluster;
              smallestCLuster2 = cluster;
            }
          }
          newClusters.remove(smallestCLuster2);
          KMeansCluster mergedCluster = new KMeansCluster(null);
          for (KMeansDataPoint dataPoint : new ArrayList<>(
              smallestCLuster1.getDataPoints()))
          {
            KMeansCluster.swap(smallestCLuster1, mergedCluster, dataPoint);
          }
          for (KMeansDataPoint dataPoint : new ArrayList<>(
              smallestCLuster2.getDataPoints()))
          {
            KMeansCluster.swap(smallestCLuster2, mergedCluster, dataPoint);
          }
          newClusters.add(mergedCluster);
        }
      }
      if (newClusters.size() == 2) //
      {
        kmeansClusters.remove(biggestCluster);
        break;
      }
    }
    if (newClusters.size() < 2)
    {
      newClusters.clear();
      KMeansCluster biggestCluster = biggestClusters.iterator().next();
      KMeansCluster manuallySplittedBiggestCluster = new KMeansCluster(null);
      int halfDataOfBiggestCluster = biggestCluster.getDataPoints().size()
          / 2;
      int index = 0;
      for (KMeansDataPoint dataPoint : new ArrayList<>(
          biggestCluster.getDataPoints()))
      {
        KMeansCluster.swap(biggestCluster, manuallySplittedBiggestCluster,
            dataPoint);
        index++;
        if (index == halfDataOfBiggestCluster) //
          break;
      }
      newClusters.add(manuallySplittedBiggestCluster);
    }
    kmeansClusters.addAll(newClusters);
  }
  Map<Integer, Integer> indexToClusterIdMap = new HashMap<>();
  Set<Integer> outputRowIndices = new HashSet<>();
  int clusterIDIndex = 1;
  for (KMeansCluster cluster : kmeansClusters)
  {
    Collection<KMeansDataPoint> dataPoints = cluster.getDataPoints();
    for (KMeansDataPoint dataPoint : dataPoints)
    {
      outputRowIndices.add(dataPoint.getID());
      indexToClusterIdMap.put(dataPoint.getID(), clusterIDIndex);
    }
    clusterIDIndex++;
  }
  inputRowIndex = 0;
  int outputRowIndex = 1;
```

```
  for (DataRow row : table)
  {
    exec.checkCanceled();
    exec.setProgress((double) currentRow / (rowCount * 2),
        " processing row " + currentRow);
    if (outputRowIndices.contains(inputRowIndex))
    {
      List<DataCell> dataCells = new ArrayList<DataCell>();
      for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
      {
        DataCell cell = row.getCell(i);
        dataCells.add(cell);
      }
      dataCells.add(new IntCell(indexToClusterIdMap.get(inputRowIndex)));
      container.addRowToTable(
          new DefaultRow(RowKey.createRowKey(outputRowIndex++), dataCells));
      dataCells.clear();
      currentRow++;
    }
    inputRowIndex++;
  }
  container.close();
  return new BufferedDataTable[] { container.getTable()};
}

/**
 * {@inheritDoc}
 */
@Override
protected void reset()
{}

/**
 * {@inheritDoc}
 */
@Override
protected DataTableSpec[] configure(final DataTableSpec[] inSpecs)
    throws InvalidSettingsException
{
  molDataHolderFactory = DataTableUtils
      .getMoleculeDataHolderFactory(inSpecs[0], null);
  return new DataTableSpec[] { null};
}

/**
 * {@inheritDoc}
 */
@Override
protected void saveSettingsTo(final NodeSettingsWO settings)
{
  colSelection.saveSettingsTo(settings);
  settingNumCluster.saveSettingsTo(settings);
}

/**
 * {@inheritDoc}
 */
@Override
protected void loadValidatedSettingsFrom(final NodeSettingsRO settings)
    throws InvalidSettingsException
{
  colSelection.loadSettingsFrom(settings);
  settingNumCluster.loadSettingsFrom(settings);
}

/**
 * {@inheritDoc}
 */
@Override
protected void validateSettings(final NodeSettingsRO settings)
    throws InvalidSettingsException
{
  colSelection.validateSettings(settings);
  settingNumCluster.validateSettings(settings);
}
```

```
  /**
   * {@inheritDoc}
   */
  @Override
  protected void loadInternals(final File internDir,
      final ExecutionMonitor exec)
      throws IOException, CanceledExecutionException
  {}

  /**
   * {@inheritDoc}
   */
  @Override
  protected void saveInternals(final File internDir,
      final ExecutionMonitor exec)
      throws IOException, CanceledExecutionException
  {}

}
```

3.5.5.5 FINGERPRINTDATAPOINT.JAVA

```
package com.inteligand.knime.nodes.fingerprintclustering;

import com.inteligand.ilib.base.fingerprint.Fingerprint;
import com.inteligand.ilib.clustering.KMeansDataPoint;

public class FingerprintDataPoint extends KMeansDataPoint
{
  private final int id;
  private final Fingerprint fingerprint;
  public FingerprintDataPoint(int id, Fingerprint fingerprint)
  {
    this.id = id;
    this.fingerprint = fingerprint;
  }

  @Override
  public int getID()
  {
    return id;
  }

  @Override
  public double calculateDistanceTo(KMeansDataPoint dataPoint)
  {
    // First part of the if is needed for kmeans initialization
    if (this != dataPoint && dataPoint instanceof FingerprintDataPoint)
    {
      FingerprintDataPoint other = (FingerprintDataPoint) dataPoint;
      return 1 - fingerprint.calcTanimotoSimilarity(other.fingerprint);
    }
    return 1;
  }

  public Fingerprint getFingerprint()
  {
    return fingerprint;
  }

  @Override
  public String toString()
  {
    return Integer.toString(id);
  }
}
```

3.5.6 DIVERSITY PICKER NODE

3.5.6.1 DIVERSITYPICKERNODEFACTORY.XML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE knimeNode PUBLIC "-//UNIKN//DTD KNIME Node 2.0//EN"
"http://www.knime.org/Node.dtd">
<knimeNode icon="/icons/diversitypicker.png" type="Manipulator">
        <name>Diversity Picker</name>
        <shortDescription>
                Diversity Picker
        </shortDescription>
        <fullDescription>
                <intro>The node clusters the input table molecules using
fingerprints
                        and chooses the most diverse for each cluster. A non-
deterministic
                        algorithm is used, which means that results can be slightly
                        different for multiple node executions.
                </intro>
        </fullDescription>
        <ports>
                <inPort index="0" name="Input molecule table">Table containing the
input
                molecules.
                </inPort>
                <outPort index="0" name="Output molecule table">Table containing
the most
                diverse
                        molecules. Number for output molecules is set by the user.
                </outPort>
        </ports>

</knimeNode>
```

3.5.6.2 DIVERSITYPICKERNODEFACTORY.JAVA

```
package com.inteligand.knime.nodes.diversitypicker;

import org.knime.core.node.NodeDialogPane;
import org.knime.core.node.NodeFactory;
import org.knime.core.node.NodeView;

public class DiversityPickerNodeFactory extends
    NodeFactory<DiversityPickerNodeModel>
{

  /**
   * {@inheritDoc}
   */
  @Override
  public DiversityPickerNodeModel createNodeModel()
  {
    return new DiversityPickerNodeModel();
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public int getNrNodeViews()
  {
    return 0;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public NodeView<DiversityPickerNodeModel> createNodeView(
      final int viewIndex, final DiversityPickerNodeModel nodeModel)
```

```
  {
    return null;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public boolean hasDialog()
  {
    return true;
  }

  /**
   * {@inheritDoc}
   */
  @Override
  public NodeDialogPane createNodeDialogPane()
  {
    return new DiversityPickerNodeDialog();
  }

}
```

3.5.6.3 DIVERSITYPICKERNODEDIALOG.JAVA

```
package com.inteligand.knime.nodes.diversitypicker;

import org.knime.core.data.vector.bitvector.BitVectorValue;
import org.knime.core.node.defaultnodesettings.DefaultNodeSettingsPane;
import
org.knime.core.node.defaultnodesettings.DialogComponentColumnNameSelection;
import org.knime.core.node.defaultnodesettings.DialogComponentNumber;
import org.knime.core.node.defaultnodesettings.SettingsModelInteger;
import org.knime.core.node.defaultnodesettings.SettingsModelString;

public class DiversityPickerNodeDialog extends DefaultNodeSettingsPane
{

  @SuppressWarnings("unchecked")
  protected DiversityPickerNodeDialog()
  {
    super();

    createNewGroup("");
    addDialogComponent(new DialogComponentColumnNameSelection(
        new SettingsModelString(DiversityPickerNodeModel.COLSEL, ""),
        "Choose the fingerprint to use: ", 0, true,
        BitVectorValue.class));

    addDialogComponent(new DialogComponentNumber(
        new SettingsModelInteger(DiversityPickerNodeModel.SETTINGSNUMBERMOL,
            10),
        "Enter the number of diverse molecules you want to obtain:", 1));
  }
}
```

3.5.6.4 DIVERSITYPICKERNODEMODEL.JAVA

```
package com.inteligand.knime.nodes.diversitypicker;

import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Collections;
import java.util.Comparator;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
```

```
import org.knime.core.data.DataCell;
import org.knime.core.data.DataColumnSpec;
import org.knime.core.data.DataRow;
import org.knime.core.data.DataTableSpec;
import org.knime.core.data.RowKey;
import org.knime.core.data.def.DefaultRow;
import org.knime.core.data.vector.bitvector.BitVectorValue;
import org.knime.core.node.BufferedDataContainer;
import org.knime.core.node.BufferedDataTable;
import org.knime.core.node.CanceledExecutionException;
import org.knime.core.node.ExecutionContext;
import org.knime.core.node.ExecutionMonitor;
import org.knime.core.node.InvalidSettingsException;
import org.knime.core.node.NodeModel;
import org.knime.core.node.NodeSettingsRO;
import org.knime.core.node.NodeSettingsWO;
import org.knime.core.node.defaultnodesettings.SettingsModelIntegerBounded;
import org.knime.core.node.defaultnodesettings.SettingsModelString;
import com.inteligand.ilib.base.fingerprint.Fingerprint;
import com.inteligand.ilib.base.fingerprint.SciTegicECFingerprint;
import com.inteligand.ilib.clustering.KMeansCluster;
import com.inteligand.ilib.clustering.KMeansClusterGenerator;
import com.inteligand.ilib.clustering.KMeansDataPoint;
import com.inteligand.knime.data.cellvalues.molecule.MoleculeDataTableDefaults;
import com.inteligand.knime.utils.DataTableUtils;

public class DiversityPickerNodeModel extends NodeModel
{
  static final String COLSEL = "selCol";
  static final String SETTINGSNUMBERMOL = "numberOfMoleules";
  private final SettingsModelString colSelection = new SettingsModelString(
      COLSEL, MoleculeDataTableDefaults.FINGERPRINTS_COLUMN);
  private final SettingsModelIntegerBounded settingNumMol = new
SettingsModelIntegerBounded(
      SETTINGSNUMBERMOL, 10, 1, Integer.MAX_VALUE);
  protected DiversityPickerNodeModel()
  {
    super(1, 1);
  }

  @Override
  protected BufferedDataTable[] execute(final BufferedDataTable[] inData,
      final ExecutionContext exec) throws Exception
  {
    DataTableSpec inputTableSpec = inData[0].getDataTableSpec();

    DataColumnSpec[] outputColumnSpecs = new DataColumnSpec[inputTableSpec
        .getNumColumns()];
    for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
    {
      DataColumnSpec columnSpec = inputTableSpec.getColumnSpec(i);
      outputColumnSpecs[i] = columnSpec;
    }

    DataTableSpec outputTableSpec = new DataTableSpec(outputColumnSpecs);
    BufferedDataContainer container = exec.createDataContainer(outputTableSpec);
    BufferedDataTable table = inData[0];
    String fingerprintColName = colSelection.getStringValue();
    int indexFingerprintsCol = inData[0].getDataTableSpec()
        .findColumnIndex(fingerprintColName);
    long rowCount = table.size();
    int currentRow = 0;
    int inputRowIndex = 0;
    long fingerprintLength = 0;

    List<KMeansDataPoint> fingerprintDataPoints = new
ArrayList<KMeansDataPoint>();
    List<KMeansCluster> initClusters = new ArrayList<KMeansCluster>();
    List<Fingerprint> molFingerprints = new ArrayList<Fingerprint>();
    for (DataRow row : table)
    {
      exec.checkCanceled();
      exec.setProgress((double) currentRow / (rowCount * 2),
          " processing row " + currentRow);
      BitVectorValue molFingerprintValue = (BitVectorValue) row
          .getCell(indexFingerprintsCol);
```

```
      fingerprintLength = molFingerprintValue.length();
      Fingerprint molFingerprint = new SciTegicECFingerprint();
      for (int i = 0; i < fingerprintLength; i++)
      {
        if (molFingerprintValue.get(i))
        {
          molFingerprint.setBit(i);
        }
      }
      fingerprintDataPoints
          .add(new FingerprintDataPoint(inputRowIndex, molFingerprint));
      molFingerprints.add(molFingerprint);
      inputRowIndex++;
    }
    if (inputRowIndex < settingNumMol
        .getIntValue()) { throw new InvalidSettingsException(
            "Please enter a number of maximum molecules less or equal to the
molecules
              you have in the input table"); }
    for (int i = 0; i < settingNumMol.getIntValue(); i++)
    {
      initClusters.add(new KMeansCluster(fingerprintDataPoints.get(i)));
    }
    KMeansClusterGenerator kmeansClusterGen = new KMeansClusterGenerator();
    Collection<KMeansCluster> kmeansClusters = kmeansClusterGen
        .cluster(initClusters, fingerprintDataPoints);
    for (KMeansCluster cluster : new ArrayList<>(kmeansClusters))
    {
      if (cluster.getDataPoints().isEmpty())
      {
        kmeansClusters.remove(cluster);
      }
    }
    while (kmeansClusters.size() < settingNumMol.getIntValue())
    {
      List<KMeansCluster> biggestClusters = new ArrayList<>();
      for (KMeansCluster cluster : kmeansClusters)
      {
        biggestClusters.add(cluster);
      }
      Collections.sort(biggestClusters, new Comparator<KMeansCluster>()
      {
        @Override
        public int compare(KMeansCluster cluster1, KMeansCluster cluster2)
        {
          return cluster2.getDataPoints().size()
              - cluster1.getDataPoints().size();
        }
      });

      Collection<KMeansCluster> newClusters = new ArrayList<>();
      for (KMeansCluster biggestCluster : biggestClusters)
      {
        if (biggestCluster.getDataPoints().size() == 1)
        {
          break;
        }
        int numInitialClusters = 1;
        while (newClusters.size() < 2
            && numInitialClusters < fingerprintDataPoints.size()
            && numInitialClusters < 1000)
        {
          newClusters.clear();
          numInitialClusters++;
          initClusters = new ArrayList<>();
          fingerprintDataPoints = new ArrayList<>(
              biggestCluster.getDataPoints());
          for (KMeansDataPoint dataPoint : fingerprintDataPoints)
          {
            dataPoint.setCluster(null);
          }
          for (int i = 0; i < numInitialClusters; i++)
          {
            initClusters.add(new KMeansCluster(fingerprintDataPoints.get(i)));
          }
          kmeansClusterGen = new KMeansClusterGenerator();
```

```
      newClusters = kmeansClusterGen.cluster(initClusters,
          fingerprintDataPoints);
      for (KMeansCluster cluster : new ArrayList<>(newClusters))
      {
        if (cluster.getDataPoints().isEmpty())
        {
          newClusters.remove(cluster);
        }
      }
      while (newClusters.size() > 2)
      {
        KMeansCluster smallestCLuster1 = null;
        KMeansCluster smallestCLuster2 = null;
        int minNumberDataPoints = Integer.MAX_VALUE;
        for (KMeansCluster cluster : newClusters)
        {
          int numDataPointsInCluster = cluster.getDataPoints().size();
          if (numDataPointsInCluster < minNumberDataPoints)
          {
            minNumberDataPoints = numDataPointsInCluster;
            smallestCLuster1 = cluster;
          }
        }
        newClusters.remove(smallestCLuster1);
        minNumberDataPoints = Integer.MAX_VALUE;
        for (KMeansCluster cluster : newClusters)
        {
          int numDataPointsInCluster = cluster.getDataPoints().size();
          if (numDataPointsInCluster < minNumberDataPoints)
          {
            minNumberDataPoints = numDataPointsInCluster;
            smallestCLuster2 = cluster;
          }
        }
        newClusters.remove(smallestCLuster2);
        KMeansCluster mergedCluster = new KMeansCluster(null);
        for (KMeansDataPoint dataPoint : new ArrayList<>(
            smallestCLuster1.getDataPoints()))
        {
          KMeansCluster.swap(smallestCLuster1, mergedCluster, dataPoint);
        }
        for (KMeansDataPoint dataPoint : new ArrayList<>(
            smallestCLuster2.getDataPoints()))
        {
          KMeansCluster.swap(smallestCLuster2, mergedCluster, dataPoint);
        }
        newClusters.add(mergedCluster);
      }
    }
    if (newClusters.size() == 2) //
    {
      kmeansClusters.remove(biggestCluster);
      break;
    }
  }

  if (newClusters.size() < 2)
  {
    newClusters.clear();
    KMeansCluster biggestCluster = biggestClusters.iterator().next();
    KMeansCluster manuallySplittedBiggestCluster = new KMeansCluster(null);
    int halfDataOfBiggestCluster = biggestCluster.getDataPoints().size() / 2;
    int index = 0;
    for (KMeansDataPoint dataPoint : new ArrayList<>(
        biggestCluster.getDataPoints()))
    {
      KMeansCluster.swap(biggestCluster, manuallySplittedBiggestCluster,
          dataPoint);
      index++;
      if (index == halfDataOfBiggestCluster)
      {
        break;
      }
    }
    newClusters.add(manuallySplittedBiggestCluster);
  }
```

```
      kmeansClusters.addAll(newClusters);
    }
    Set<Integer> outputRowIndices = new HashSet<>();
    for (KMeansCluster cluster : kmeansClusters)
    {
      Collection<KMeansDataPoint> dataPoints = cluster.getDataPoints();
      // pickBestDataPoint
      KMeansDataPoint dataPoint = pickBestDataPoint(dataPoints, cluster,
          fingerprintLength);
      outputRowIndices.add(dataPoint.getID());
    }
    inputRowIndex = 0;
    long outputRowIndex = 1;
    for (DataRow row : table)
    {
      exec.checkCanceled();
      exec.setProgress((double) currentRow / (rowCount * 2),
          " processing row " + currentRow);
      if (outputRowIndices.contains(inputRowIndex))
      {
        List<DataCell> dataCells = new ArrayList<DataCell>();
        for (int i = 0; i < inputTableSpec.getNumColumns(); i++)
        {
          DataCell cell = row.getCell(i);
          dataCells.add(cell);
        }
        container.addRowToTable(
            new DefaultRow(RowKey.createRowKey(outputRowIndex++), dataCells));
        dataCells.clear();
        currentRow++;
      }
      inputRowIndex++;
    }
    container.close();
    return new BufferedDataTable[] { container.getTable()};
  }
  private KMeansDataPoint pickBestDataPoint(
      Collection<KMeansDataPoint> dataPoints, KMeansCluster cluster,
      long fingerprintLength)
  {
    Fingerprint dummyFingerprint = new SciTegicECFingerprint();
    for (int i = 0; i < fingerprintLength; i++)
    {
      int numSet = 0;
      int numUnset = 0;
      for (KMeansDataPoint dataPoint : dataPoints)
      {
        Fingerprint fingerprint = ((FingerprintDataPoint) dataPoint)
            .getFingerprint();
        if (fingerprint.getBits().get(i))
        {
          numSet++;
        }
        else
        {
          numUnset++;
        }
      }
      float percentage = numSet / (numSet + numUnset);
      if (percentage > 0.85)
      {
        dummyFingerprint.setBit(i);
      }
    }
    float minSimilarityToDummy = 1.1F;
    KMeansDataPoint lessSimilarDataPoint = null;
    for (KMeansDataPoint dataPoint : dataPoints)
    {
      Fingerprint fingerprint = ((FingerprintDataPoint) dataPoint)
          .getFingerprint();
      float similarityToDummy = dummyFingerprint
          .calcTanimotoSimilarity(fingerprint);
      if (similarityToDummy < minSimilarityToDummy)
      {
        minSimilarityToDummy = similarityToDummy;
        lessSimilarDataPoint = dataPoint;
```

```
    }
  }
  return lessSimilarDataPoint;
}

/**
 * {@inheritDoc}
 */
@Override
protected void reset()
{}

/**
 * {@inheritDoc}
 */
@Override
protected DataTableSpec[] configure(final DataTableSpec[] inSpecs)
    throws InvalidSettingsException
{
  String fingerprintColName = colSelection.getStringValue();
  if (inSpecs[0].findColumnIndex(
      fingerprintColName) < 0) { throw new InvalidSettingsException(
          "Could not find fingerprint column in query molecule table!"); } ;

  DataTableUtils.getMoleculeDataHolderFactory(inSpecs[0], null);
  return new DataTableSpec[] { null};
}

/**
 * {@inheritDoc}
 */
@Override
protected void saveSettingsTo(final NodeSettingsWO settings)
{
  colSelection.saveSettingsTo(settings);
  settingNumMol.saveSettingsTo(settings);
}

/**
 * {@inheritDoc}
 */
@Override
protected void loadValidatedSettingsFrom(final NodeSettingsRO settings)
    throws InvalidSettingsException
{
  colSelection.loadSettingsFrom(settings);
  settingNumMol.loadSettingsFrom(settings);
}

/**
 * {@inheritDoc}
 */
@Override
protected void validateSettings(final NodeSettingsRO settings)
    throws InvalidSettingsException
{
  colSelection.validateSettings(settings);
  settingNumMol.validateSettings(settings);
}

/**
 * {@inheritDoc}
 */
@Override
protected void loadInternals(final File internDir,
    final ExecutionMonitor exec)
    throws IOException, CanceledExecutionException
{}

/**
 * {@inheritDoc}
 */
@Override
protected void saveInternals(final File internDir,
    final ExecutionMonitor exec)
    throws IOException, CanceledExecutionException
```

```
  {}
}
```

4.0 Appendix 2: Congress participations

- 03 July – 18 September 2016, University of Vienna, Vienna, Austria. (PhD period of research abroad).
- 26 June - 01 July 2016, European School of Medicinal Chemistry (ESMEC), Urbino, Italy.
- 15 May - 19 May 2016, VI European Workshop in Drug Synthesis (VI EWDSy), Siena, Italy.
- 20 September - 25 September 2015, Vienna Summer School on Drug Design, Vienna, Austria.
- 06 September - 09 September 2015, XXIII National Meeting in Medicinal Chemistry and 9th Young Medicinal Chemistry Symposium, Salerno, Italy
- 17 May - 22 May 2015, X European Workshop in Drug Design (X EWDD), Siena, Italy.
- 28 September - 03 October 2014, 2nd Innovative Approaches for identification of Antiviral Agents Summer School (2nd IAAASS), Cagliari, Italy.
- 21 June - 26 June 2014, EUROSCIENCE OPEN FORUM (ESOF 2014), Copenhagen, Denmark.
- 18 May - 23 May 2014, V European Workshop in Drug Synthesis (V EWDSy), Siena, Italy.